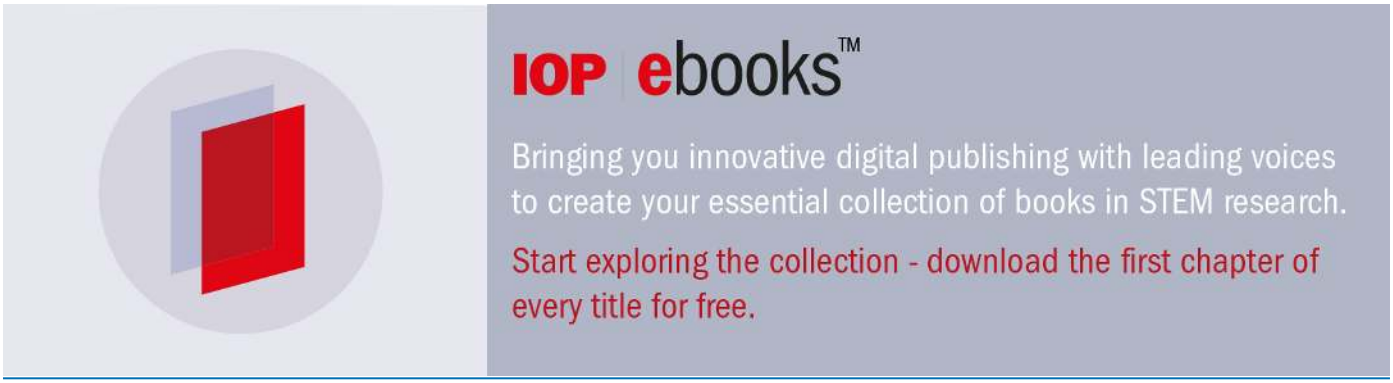
Many-body perturbation theory calculations using the yambo code

Journal of Physics: Condensed Matter

PAPER • **OPEN ACCESS**

Many-body perturbation theory calculations using the yambo code

To cite this article: D Sangalli *et al* 2019 *J. Phys.: Condens. Matter* **31** 325902



OPEN ACCESS
IOP Publishing
Journal of Physics: Condensed Matter
J. Phys.: Condens. Matter **31** (2019) 325902 (31pp)
https://doi.org/10.1088/1361-648X/ab15d0

Many-body perturbation theory calculations using the yambo code

D Sangalli[1,16], A Ferretti[2,16], H Miranda[3], C Attaccalite[4,16], I Marri[2], E Cannuccia[5,6], P Melo[7,16], M Marsili[8], F Paleari[9], A Marrazzo[10], G Prandini[10], P Bonfà[11], M O Atambo[2,12], F Affinito[11], M Palummo[5,16], A Molina-Sánchez[13], C Hogan[5,14,16], M Grüning[15,16], D Varsano[2,16] and A Marini[1,16]

[1] Istituto di Struttura della Materia—Consiglio Nazionale delle Ricerche (CNR-ISM), Division of Ultrafast Processes in Materials (FLASHit), Via Salaria Km 29.5, CP 10, I-00016 Monterotondo Stazione, Italy
[2] Centro S3, Istituto Nanoscienze—Consiglio Nazionale delle Ricerche (CNR-NANO), via Campi 213/A, 41125 Modena, Italy
[3] Institute of Condensed Matter and Nanoscience (IMCN), Université catholique de Louvain, B-1348, Louvain-la-Neuve, Belgium
[4] Aix Marseille Université, CNRS, CINaM UMR 7325, Campus de Luminy Case 913, 13288 Marseille, France
[5] Dipartimento di Fisica, Università di Roma 'Tor Vergata', Via della Ricerca Scientifica 1, I-00133 Roma, Italy
[6] Aix-Marseille Université, Laboratoire de Physique des Interactions Ioniques et Moléculaires (PIIM), UMR CNRS 7345, F-13397 Marseille, France
[7] Nanomat/Q-mat/CESAM, Université de Liège, Institut de Physique, B-4000 Sart Tilman, Liège, Belgium
[8] Dipartimento di Scienze Chimiche, University of Padova, I-35131, Padova, Italy
[9] Physics and Materials Science Research Unit, University of Luxembourg, 162a avenue de la Faïencerie, L-1511 Luxembourg, Luxembourg
[10] Theory and Simulations of Materials (THEOS) and National Centre for Computational Design and Discovery of Novel Materials (MARVEL), École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[11] CINECA National Supercomputing Center, Casalecchio di Reno, I-40033 Bologna, Italy
[12] Università di Modena e Reggio Emilia, Via Campi 213/A, 41125 Modena, Italy
[13] Institute of Materials Science (ICMUV), University of Valencia, Catedrático Beltrán 2, E-46980, Valencia, Spain
[14] Istituto di Struttura della Materia—Consiglio Nazionale delle Ricerche (CNR-ISM), via Fosso del Cavaliere 100, 00133 Rome, Italy
[15] School of Mathematics and Physics, Queen's University Belfast, Belfast BT7 1NN, Northern Ireland, United Kingdom
[16] European Theoretical Spectroscopy Facility (ETSF)

E-mail: andrea.marini@cnr.it

Received 6 February 2019, revised 18 March 2019
Accepted for publication 3 April 2019
Published 29 May 2019

Abstract

`yambo` is an open source project aimed at studying excited state properties of condensed matter systems from first principles using many-body methods. As input, `yambo` requires ground state electronic structure data as computed by density functional theory codes such as `Quantum ESPRESSO` and `Abinit`. `yambo`'s capabilities include the calculation of linear response

quantities (both independent-particle and including electron–hole interactions), quasi-particle corrections based on the GW formalism, optical absorption, and other spectroscopic quantities. Here we describe recent developments ranging from the inclusion of important but oft-neglected physical effects such as electron–phonon interactions to the implementation of a real-time propagation scheme for simulating linear and non-linear optical properties. Improvements to numerical algorithms and the user interface are outlined. Particular emphasis is given to the new and efficient parallel structure that makes it possible to exploit modern high performance computing architectures. Finally, we demonstrate the possibility to automate workflows by interfacing with the `yambopy` and AiiDA software tools.

Keywords: electronic structure, optical properties, real-time dynamics, electron–phonon, spin and spinors, Kerr effect, parallelism

(Some figures may appear in colour only in the online journal)

Contents

1. The Yambo project

Computational materials science based on first principles atomistic methods plays a key role in the discovery, characterization, and engineering of novel and complex materials. While density functional theory (DFT) is the established workhorse for ground state properties of a wide range of systems ranging from atoms and molecules to solids and nanostructures containing thousands of atoms, there is an increasing demand for an *accurate* description of *excited state* properties in even the most challenging materials. Within the framework of solid state physics, the Green's function formulation of many-body perturbation theory (MBPT)—specifically the GW approach to quasiparticles (QP) for charged excitations and the Bethe–Salpeter equation (BSE) for neutral excitations—offers a quantitatively accurate solution [1]. The

GW-BSE approach has been implemented in a number of free and commercially available codes, both in plane-waves [2–8] and with other basis-sets [9–15], and applied to a wide range of materials (for a recent and more comprehensive review, see [16]). Nonetheless, the complexity and relatively poor scaling of the GW-BSE method, and often of its implementation, constitutes a barrier towards its application to realistic systems of large size or to physical phenomena that lie outside the scope of most state-of-the-art approaches.

Tackling these challenges in a software environment requires a fourfold strategy:

- First, the description of underlying physical phenomena must be regularly advanced, both in terms of extensions of existing tools and by devising new methods. Oft-neglected terms such as electron–phonon and spin–orbit coupling play a crucial role in several physical phenomena. Examples are the finite temperature properties (dictated by the electron–phonon interaction) or the study of novel materials like topological insulators, perovskites and layered transition metal dichalcogenides. In addition to extensions of existing tools `yambo` implements brand new methods like real-time tools to tackle the calculation of nonlinear optical properties.
- Second, algorithms must be refined and augmented in order to improve technical precision and numerical efficiency. This includes tricks for accelerating convergence as well as implementing alternatives to standard GW-BSE approximations such as plasmon-pole models of electronic screening and the Tamm–Dancoff approximation to exciton coupling.
- Third, codes must be designed to follow current trends in high-performance computing towards massively parallel, distributed memory architectures, while allowing for flexibility and control over tasks, memory, and disk usage in order to keep simulations efficient.
- Fourth, as the codes themselves become more complex and harder to maintain, modern software practices must be adopted. These include a wide range of aspects including improved documentation, use of modules and standard libraries, and automation of tasks for convergence, benchmarking and reproducibility.

In this paper we describe how the `yambo` project has embraced this broad strategy. `yambo` is an open-source code based on many-body perturbation theory for computing electronic and optical excitations within a high performance environment (figure 1). Since its first public release in 2008, the project has evolved in a dramatic fashion and its development and user base has greatly expanded. Within the following ten years, the original paper was cited more than 500 times—considerable for a pure MBPT code—and the code has been used in many high impact studies spanning a wide range of novel materials and exciting technologies. The highest cited applications cover graphene derivatives [17–20], metal-halide perovskites [21, 22], van der Waals bonded layered compounds [23–26], Li-air and K-ion batteries [27, 28], and TiO_2 photocatalytic surfaces [29, 30], to select just a handful. `yambo` has moreover helped advance fundamental understanding of physical phenomena such as excitonic Bose–Einstein condensation [31], excitonic insulators [32], the influence of zero point motion [33], charge transfer excitations [34], etc. A full list of publications can be found through our website [35], www.yambo-code.org.

Part of `yambo`'s popularity and success may be ascribed to the code's user-friendliness: thanks to an intelligent command line interface, a full GW-BSE calculation on an unfamiliar material can in principle be carried out launching a single command. Extensive user documentation is provided on our website [35]. This includes descriptions of the fundamental theory, command line interface, and input variables, and provides a wide range of tutorials directed at explaining different functionalities of the code across a number of systems with different dimensionalities. Support is given by the developers through a forum. In addition to the website [35][17], the theory and use of `yambo` has been disseminated through a number of international schools and workshops including a dedicated biennial CECAM event run by the developers and aimed at showcasing the latest developments.

The first major release, version 3.2.0, was described in detail in Marini *et al* [4] (henceforth referred to as CPC2009), and therefore the basic methodology, formalism, and code structure will not be repeated here. Instead we describe the main additions made to the code up to and including version 4.4.0. Much development of the code has been driven by its status as a key *ab initio* spectroscopy code of the European Theoretical Spectroscopy Facility (ETSF) [36] and as a flagship code of the *MaX European Centre of Excellence for Materials Design at the Exascale* [37] and of the *Nanoscience Foundries and Fine Analysis—Europe* user infrastructure [38].

With regard to the broad strategy outlined above, `yambo` now includes the possibility to compute the following state-of-the-art physical phenomena discussed later:

- Electron–phonon and exciton-phonon interaction: influence of temperature on electronic structure and optical spectra (section 6);
- Real-time propagation of the density matrix (section 7.1) and Bloch states for nonlinear optics (section 7.2);
- Spin–orbit coupling and Kerr effect within a fully noncollinear BSE framework (section 5.2).

Numerous methodological advances have been incorporated in the code in the last decade. We will discuss in more detail the following key features:

- Alternative approaches for computing dipole matrix elements and commutators (section 3.1);
- Incorporation of empty state terminators in the linear response (section 3.3) and self-energy (section 4.3);
- Full frequency GW, including computation of lifetimes (section 4.1);
- Double grid approach and Krylov algorithm for improved BSE efficiency (section 5.1).

Regarding parallelism, section 8 outlines the code's strategies for exploiting massively-parallel architectures through the use

[17] Content will eventually be updated and ported to a more user-friendly wiki style site.

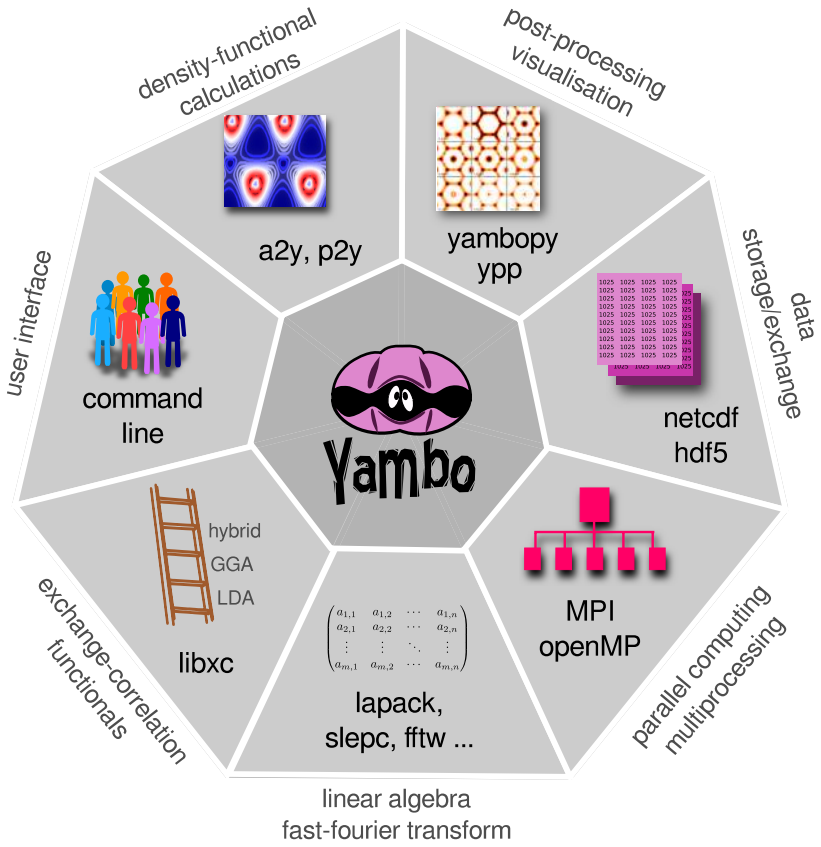


Figure 1. The `yambo` project combines cutting edge computational materials science within a beyond-DFT framework with high performance algorithms, tools, and libraries.

of a highly user-tunable mixed MPI-OpenMP coding paradigm and the use, where possible, of external parallel libraries for linear algebra and I/O tasks. As different quantities (i.e. linear response, GW, BSE) computed by `yambo` have very different behaviours in terms of performance, scalability, and memory distribution, it is important to outline the different approaches—ultimately controlled by the user—adopted by the code in each case.

Last, `yambo` has been almost completely rewritten since the first major release in order to follow modern software design practices such as modularity, reuse of routines and libraries, and so on, and the project as a whole has been expanded to include rigorous self-testing and automation frameworks. Here we highlight a few key features:

- Test-suite and benchmarking scripts (section 9.3);
- The `yambopy` python scripts for automation and analysis (section 9.1);
- Plugin for workflow management via AiiDA (section 9.2);
- Wide use of standard libraries (section 2.1).
- Maintenance and distribution through `GitHub`.

In the following section we recall the structure of the `yambo` software package and outline new features in its installation environment and interface with external codes and libraries. Sections 3–7 outline new features implemented relating to improved algorithms and new capabilities. Section 8 discusses the new parallelism paradigm and performance issues. Section 9 introduces new scripting and automation tools. Following some general conclusions, various technical information is presented in the appendices along with a useful glossary of acronyms.

2. Technical overview

The `yambo` package is released under the GNU GPL (v2) license and is hosted on `GitHub` in a set of public and private repositories at https://github.com/yambo-code. Snapshots of major releases are also available for direct download through the `yambo` website [35].

The general structure of the `yambo` software is laid out in figure 2. The software consists of three kinds of executable that generally reflect the order in which the code is run. First, the output from standard DFT codes are converted into NetCDF 'database' files (`ns.db1` and `ns.wf`) within a `SAVE` directory using the `a2y` and `p2y` routines (see section 2.3 below). Second, the main calculations ('runlevels') of linear response, GW, and BSE are performed using the standard executable `yambo` or the project-specific executables. These include `yambo_rt` for real-time propagation (section 7.1), `yambo_nl` for nonlinear optics (section 7.2), and `yambo_ph` for electron–phonon simulations (section 6). Running these codes results in the reading and writing of further databases (`SAVE/ndb.*`), as well as generation of text files for reading or plotting. Third, post-processing routines (`ypp` and runlevel specific `ypp_nl` or `ypp_rt` executables) are used to manipulate

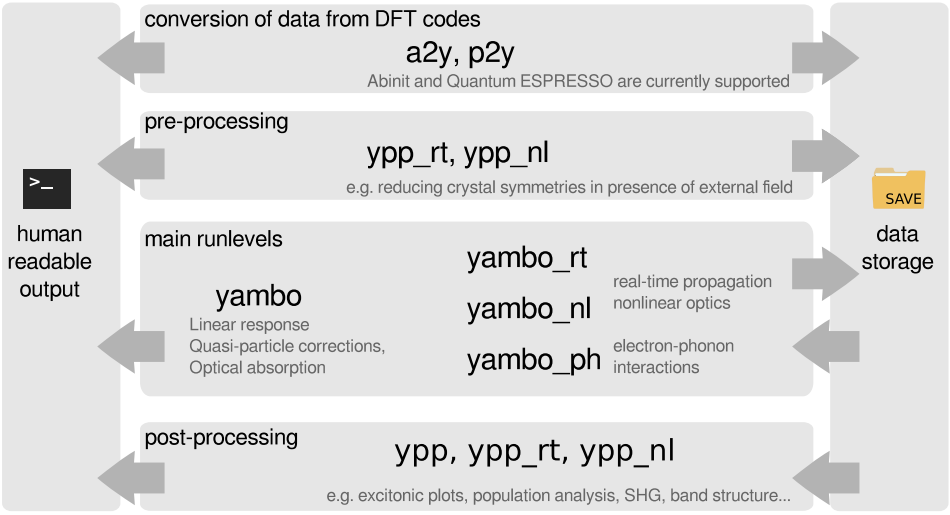


Figure 2. Main software components of the `yambo` suite.

and analyze the computed quantities stored in the databases. In special cases `ypp`, `ypp_nl` or `ypp_rt` executables are needed as pre-processing tools to further manipulate the core databases (i.e. to remove some symmetries before real time simulations) or to create new databases (i.e. an `ndb` containing a mapping between core databases on two different k-grids), before actually running the main calculation.

2.1. Installation & projects

`yambo` is compiled using the standard autotools procedure: `./configure; make all` will generate the main executables listed in figure 2. Since the first release the configure script has been wholly upgraded to reflect the widespread availability of high performance software libraries and to aid portability across a wider range of system architectures.

By running `./configure; make`, the list of possible executables is returned

```
[all projects] all
[project-related suite] project
                (core, rt-project, ...)
[core] yambo
[core] ypp
[core] a2y
[core] p2y

[ph-project] yambo_ph
[ph-project] ypp_ph

[rt-project] yambo_rt
[rt-project] ypp_rt

[nl-project] yambo_nl
[nl-project] ypp_nl

[kerr-project] yambo_kerr
```

While `yambo` and `ypp` are the main code components, a series of projects appear in the form of `yambo_PJ`/`ypp_PJ` with *PJ* being the specific project identifier (`ph`, `rt`, `nl`, `kerr`). These projects correspond to pre-processor flags that, during the compilation, activate lines of code and procedures that are project-specific. In `yambo` several different codes coexist in the same source.

2.2. Configuration

In many cases `configure` will manage to detect the compilation environment and external libraries automatically. For more control, a flexible list of options is available (see `./configure –help`). A wide range of optional features can be activated via `–enable-FEATURE[=ARG]` flags, e.g.

`./configure –enable-open-mp`

including options controlling serial/parallel linear algebra, timing/memory profiling, type of fast Fourier transform (FFT) library, etc. External libraries can be linked to by specifying either the installation directory including the ‘libs’ and ‘include’ folders,

`–with-libname-path = <path>`

or the ‘libs’ and ‘include’ paths directly

`–with-libname-libdir = <path>`
`–with-libname-includedir = <path>`

or finally the libraries and the include command

`–with-libname-libs = <libs>`
`–with-libname-incs = <include command>`

This is an important improvement for allowing installation on machines with non-standard system directories.

Choice of compilers and preprocessors can be overridden via the environmental variables `FC`, `CPP` etc. Finally, the generated `config/setup` file can be tweaked by hand prior to compilation.

`yambo` can make use of several external libraries for improving performance and portability (see table 1). In addition to standard MPI (openmpi, Intel MPI, etc) and OpenMP for parallel computation, these include standard scientific computation libraries such as BLAS and LAPACK (including the Intel MKL and IBM ESSL), scalable versions of these (BLACS,

Table 1. Illustrative list of some of the configuration command line options. More options are available and can be listed by using `./configure –help`.

Library	Flag
Fourier transform	
FFTW (2.0)	Default
Goedecker	`–enable-internal-fftsg`
QE standard	`–enable-internal-fftqe`
QE 3D	`–enable-3d-fft`
MKL, ESSL, FFTW(3.x)	`–with-fft-libs = <libs >`
Linear Algebra	
BLAS, LAPACK	`–with-blas-libs = <libs >`
MKL, ESSL	`–with-lapack-libs = <libs >`
Parallel Linear Algebra	
BLACS &, ScaLAPACK	`–enable-par-linalg +` `–with-blacs-libs = <libs >`
Sparse Linear Algebra	
SLEPC & PETSC	`–enable-slepc-linalg` `–with-slepc-libs = <libs >` `–with-petsc-libs = <libs >`
...	...

ScaLAPACK; `–enable-par-linalg`), as well as advanced parallel numerical libraries (SLEPc, PETSc; `–enable-slepc-linalg`). Use of the latter in `yambo` is discussed in detail in section 8.6. Heavy use is made of FFTs. `yambo` supports many FFT implementations: Goedecker (`–enable-internal-fftsg`), FFTW (internal default) and 3D or standard FFT implementation of `Quantum ESPRESSO` (`–enable-3D-fft` or `–enable-internal-fftqe`) can be compiled while MKL and ESSL can be externally linked. Regarding internal I/O, linking to NetCDF or HDF5 format libraries is a requirement. The exchange-correlation functional library `libxc` is also required. Interfacing with the `yambopy` and AiiDA platforms is explained thoroughly in section 9. Libraries related to porting data from DFT codes are discussed in the following section.

2.2.1. External libraries. An important feature of the new configuration procedure in `yambo` is that all required libraries can be automatically downloaded, configured and compiled at the compilation time.

Indeed, if `configure` does not find a required library (dependency), it will automatically download and compile it. A useful option is the

`–with-extlibs-path = <full_path >`

where one can provide a path of choice where `yambo` will install all the automatically downloaded libraries, once and for all. The content of the folder is never erased. In subsequent compilation the library will be automatically re-used just specifying the same option.

2.3. Interfaces with DFT codes

`yambo` is interfaced with two widely used plane-wave first-principles codes: `pwscf` from the `Quantum ESPRESSO` (QE) distribution [39, 40] and `Abinit` [5, 41, 42]. The two interfaces have been introduced in [4] (sections 5.1 and 5.3). Both work with norm conserving pseudo-potentials and import Kohn–Sham (KS) eigen-energies $\epsilon_{n\mathbf{k}}$ and eigen-functions $\psi_{n\mathbf{k}}$ as well as information needed to compute the non-local part of the pseudo-potential $V^{nl}(\mathbf{x}, \mathbf{x}')$. Since the publication of [4] both interfaces have been largely improved and extended. All interfaces are now able to deal with both collinear and non-collinear spin systems. All interfaces take advantage of the `XC` library [43, 44], thus a very broad class of functionals is supported. A more detailed summary of the changes follows.

2.3.1. Interface with `Quantum ESPRESSO`. `p2y` (`pwscf-2-yambo`) is the `yambo` interface with `Quantum ESPRESSO`. Its development line followed two routes, one related to the developments of QE I/O and one aimed at adding new features to `p2y`.

A wider class of pseudo-potentials (psps) is now supported, including UPF version 2, and multi-projector psps—i.e. with more then one projector per angular momentum channel. In the same direction the `XC` library [43, 44] allows for the support of most of the LDA and GGA functionals as a starting point for the MBPT (quasiparticle or response function) calculations. In addition, hybrid functionals, with fractions of exchange and screened exchange interaction, are also supported within `p2y`. To keep compatibility with all versions of QE within a user-friendly approach, `p2y` has now an automatic detection of the I/O format used in the ground-state calculation and is able to read different xml data-file formats (`qexml` and `qexsd`, in the QE language), also supporting the more recent HDF5 binary files.

Spin is now fully supported both in collinear and non-collinear frameworks. For example, the use of magnetic symmetries allows to take advantage of composite symmetries, i.e. which contain time-reversal, in systems which are not invariant under pure time-reversal. Work is in progress to extend the support of ultra-soft pseudopotentials (USPP).

Other important changes were carried out to optimize the interface, first of all with an improved parallelization (implemented over the writing of wavefunction fragments). Moreover the Kleinman–Bylander (KB) form factors are now converted in a `yambo`-like database, while the calculation of the commutator $[\mathbf{r}, V^{nl}]$, which was previously done at the `p2y` level, is now more efficiently done by `yambo` while computing the dipoles.

2.3.2. Interface with `Abinit`. `a2y` and `e2y` are the `Abinit-2-yambo` interfaces. The original `a2y` implementation reads data in Fortran binary format. `e2y` was developed later and is based on the `ETSF-IO` [45] and `NetCDF` [46, 47][18] libraries. Both interfaces are based on the `Abinit` KSS file and were developed following the evolution of `Abinit`.

[18] The patch works from `Abinit-6.12` to `Abinit-7.4` and it is meant to be used with `a2y`.

However, since the support to the KSS file was dropped by the Abinit team, the development and maintenance of interfaces based on it became difficult. As an example, the support for multi-projector pseudo-potentials was first released via a patch for the Abinit code, which allows the printing of the relevant data into the Abinit KSS file[18]. As a consequence, the development of the KSS-based interfaces was also dropped by the yambo team. The old a2y implementation works up to Abinit version 7, while e2y is supported up to the very recent Abinit 8 releases.

Starting with yambo 4.4, we will release a new version of the a2y interface, which is based on the direct reading of the Abinit wave-function files (WFK files) written in NetCDF format. A preliminary version of e2y based onto the WFK file was also released with yambo 4.0. However, since the support to the ETSF-IO library is not developed anymore, the WFK based e2y interface was never finalized. The new strategy (i) avoids the need for the KSS file, (ii) is numerically more efficient and (iii) reduces the I/O, since wave functions are stored on the smaller **k**-centred spheres in reciprocal space (as opposed to the KSS file which relied on a larger gamma-centred sphere). Finally, since WFK files are fully supported by the Abinit team, the new interface will be compatible with all recent Abinit developments (also including multi-projectors pseudo-potentials) and naturally portable to work with future Abinit releases.

2.4. Data post- (and pre-) processing

ypp is the yambo postprocessing and preprocessing tool. It has several capabilities which can be used to prepare yambo simulations (preprocessing) or subsequently analyse (postprocessing) the outcome.

As one of the preprocessing options, ypp can generate random grids of **k**-points to be used as input for a DFT code to compute the corresponding KS energies. The same ypp can then generate an auxiliary database with a map linking the KS energies on the random grid to the uniform grid used to compute, for example, spectral properties. The approach is useful to speed up convergence as discussed in section 5.1.1. Another preprocessing option is the removal of a specific set of symmetries and thus the expansion of the wave-functions from the IBZ associated to the full set of symmetries to the resulting IBZ. This is needed to perform real-time simulations as described in section 7. Finally preprocessing can also be used to map DFT calculations with and without spin–orbit coupling (SOC) to compute absorption spectra with SOC corrections included in a perturbative way as described in the supplemental material of [48].

Most of the postprocessing features involve data analysis. ypp can prepare readable ascii files to plot several single-particle properties such as wave-functions, charge density, density of states (DOS), magnetization, current and band structures. In particular, it can be used to obtain the QP-DOS and to interpolate QP-energies to plot the resulting band-structure along high-symmetry paths. A mixed feature (i.e. which can be used both for preprocessing and for postprocessing) is the ability of ypp to manipulate QP-databases (ndb.QP). Indeed, this is useful both for QP plots or for using ndb.QP files as input in the BSE calculations. Finally, it can be used as a tool to analyse the excitonic wave-function. As examples of postprocessing, we discuss in detail (i) how to plot the QP band structure starting from calculations on a regular grid in section 4.4 and (ii) how to plot the excitonic wave-function in section 5.3.

2.5. Usage

yambo relies on a powerful and user friendly command line interface for generating and modifying input files as well as for launching the executables. The basic functionality is unchanged from that described in CPC2009; however, some flags have been changed since the initial release. Several new options have been added to aid usage or debugging on parallel clusters or cross-compiled architectures. For instance, yambo -M and yambo -N switch off the MPI and OpenMP functionalities, respectively, yambo -Q stops the text editor from launching, and yambo -W <opt> places an internal wall clock limit on the runtime. Launching yambo -H shows the fully updated list of command options: see table 2.

3. Linear response

In the independent particle (IP) approximation, the density-density response function can be written as:

$$\chi^0_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = \frac{f_s}{N_\mathbf{k}\Omega}\sum_{nm\mathbf{k}} \rho_{nm\mathbf{k}}(\mathbf{q},\mathbf{G})\rho^\star_{nm\mathbf{k}}(\mathbf{q},\mathbf{G}') \times \left[\frac{f_{m\mathbf{k}}(1-f_{n\mathbf{k}-\mathbf{q}})}{\omega-(\epsilon_{m\mathbf{k}}-\epsilon_{n\mathbf{k}-\mathbf{q}})-\mathrm{i}\eta} - \frac{f_{m\mathbf{k}}(1-f_{n\mathbf{k}-\mathbf{q}})}{\omega-(\epsilon_{n\mathbf{k}-\mathbf{q}}-\epsilon_{m\mathbf{k}})+\mathrm{i}\eta}\right], \quad (1)$$

where n, m indexes represent band indexes (which also include the spin index in case of spin collinear calculations and which refer to spinors in case of non-collinear calculations), $f_{n\mathbf{k}}$ and $\epsilon_{n\mathbf{k}}$ are the occupations and the energies of the KS states, $f_s = 1$ for spin-polarized calculations, $f_s = 2$ otherwise. In practice, the sum in equation (1) is split into two terms as described in appendix B. The matrix elements

$$\rho_{nm\mathbf{k}}(\mathbf{q},\mathbf{G}) = \langle n\mathbf{k}|\mathrm{e}^{\mathrm{i}(\mathbf{q}+\mathbf{G})\cdot\hat{\mathbf{r}}}|m\mathbf{k}-\mathbf{q}\rangle, \quad (2)$$

have been already introduced in [4] and constitute one of the core quantities computed by the yambo code. Their evaluation is done via the Fourier transform of the wave-function product in real space, $\psi^*_{n\mathbf{k}}(\mathbf{r})\psi_{m\mathbf{k}-\mathbf{q}}(\mathbf{r})$, and has been strongly optimized being one of the most common operations performed by yambo (see discussion in section 8.3).

3.1. Dipole matrix elements

Despite the computational cost, the numerical algorithm to compute the terms in equation (2) is straightforward. Since absorption is defined as the macroscopic average of the density-density response function, $\chi(\mathbf{q}\to 0)$, the knowledge of $\rho_{nm\mathbf{k}}(\mathbf{q}\to 0, 0)$ is also needed. To this end, the dipole matrix elements $\mathbf{r}_{nm\mathbf{k}} = \langle n\mathbf{k}|\mathbf{r}|m\mathbf{k}\rangle$ are commonly computed [1, 49] within periodic boundary conditions (PBC) using the relation $[\mathbf{r}, H] = \mathbf{p} + [\mathbf{r}, V_{nl}]$. Explicitly, this gives

Table 2. Command line options for the various `yambo` tools.

Common to `yambo`, `ypp`, and `a2y` / `p2y` / `e2y`		Common to `yambo` and `ypp`	
`-h`	Short Help	`-J <opt>`	Job string identifier
`-H`	Long Help	`-V <opt>`	Input file verbosity
`-M`	Switch-off MPI support (serial run)	`-F <opt>`	Input file
`-N`	Switch-off OpenMP support (single thread run)	`-I <opt>`	Core I/O directory
		`-O <opt>`	Additional I/O directory
		`-C <opt>`	Communications I/O directory
`yambo`		`ypp`	
`-D`	DataBases properties	`-q <opt>`	(g)enerate-modify/(m)erge quasi-particle DBs
`-W <opt>`	Wall Time limitation (1d2h30m format)	`-k <opt>`	BZ Grid generator
`-Q`	Do not launch the text editor	`-i`	Wannier 90 interface
`-E <opt>`	Environment Parallel Variables file	`-b`	Read BXSF output generated by Wannier90
		`-s <opt>`	Electrons,[(w)ave,(d)ensity,(m)ag,do(s), (b)ands]
		`-e <opt>`	Excitons, [(s)ort,(sp)in,(a)mplitude,(w)ave]
`-i`	Initialization	`-f`	Free hole position [excitonic plot]
`-r`	Coulomb potential	`-m`	BZ map fine grid to coarse
`-a`	ACFDT total energy	`-w <opt>`	WFs:(p)erturbative SOC map or (c) onvertion to new format
`-s`	ScaLapacK test	`-y`	Remove symmetries not consistent with an external potential
`-o <opt>`	Optics [opt = (c)hi/(b)se]	Common to `a2y` / `p2y` / `e2y`	
`-y <opt>`	BSE solver [opt = h/d/s/(p/f)i] (h)aydock/(d)iagonalization/(i)nversion	`-U`	Do not fragment the DataBases
`-k <opt>`	Kernel [opt = hartree/alda/lrc/hf/sex]	`-O <opt>`	Output directory
		`-F <opt>`	PWscf xml index/Abinit file name
`-d`	Dynamical Inverse Dielectric Matrix		
`-b`	Static Inverse Dielectric Matrix	`-b <int>`	Number of bands for each fragment
		`-a <real>`	Lattice constant rescaling factor
`-x`	Hartree–Fock Self-energy and Vxc	`-t`	Force no TR symmetry
`-g <opt>`	Dyson Equation solver	`-n`	Force no symmetries
	[opt = (n)ewton/(s)ecant/(g)reen]	`-w`	Force no wavefunctions
`-p <opt>`	GW approximations, [opt = (p)PA/(c)OHSEX]		
`-l`	G_0W_0 Quasiparticle lifetimes	`-d`	States duplication [a2y only]
		`-v`	Verbose wfc I/O reporting [p2y only]
`-q <opt>`	Compute dipoles [available from v4.4]		
		`ypp_ph`	
		`-p <opt>`	Phonon [(d)os,(e)lias,(a)mplitude]
		`-g`	gkkp databases
`yambo_rt`		`ypp_rt`	
`-v <opt>`	Self-Consistent Potential	`-t`	TD-polarization [(X)response]
	opt = (h)artree,(f)ock, (coh),(sex),(cohsex),(d)ef,(ip)		
`-q <opt>`	Real-time dynamics [replaced by `-n <opt>` in v4.4]		
`-e`	Evaluate Collisions		
`yambo_nl`		`ypp_nl`	
`-u`	Non-linear spectroscopy	`-u`	Non-linear response analysis

$$\langle n\mathbf{k}|\mathbf{r}|m\mathbf{k}\rangle = \frac{\langle n\mathbf{k}|\,\mathbf{p} + [\mathbf{r}, V_{nl}]\,|m\mathbf{k}\rangle}{\epsilon_{n\mathbf{k}} - \epsilon_{m\mathbf{k}}}. \quad (3)$$

The direct evaluation of equation (3) (*G-space v* approach) is quite demanding due to the $[\mathbf{r}, V_{nl}]$ term, and is evaluated from the KB form factors loaded by the interfaces, see also section 2.3. This implementation has been strongly optimized and extended to account for projectors with angular momentum $l > 2$.

We have also made available alternative strategies for computing the dipoles. The *shifted grids* approach is based on the idea of numerically evaluating $\rho_{nm\mathbf{k}}(\mathbf{q}_\epsilon, 0)$ for a very small $q_\epsilon = |\mathbf{q}_\epsilon|$. Thus the wave-function at $\mathbf{k}$ and the wave-functions at $\mathbf{k} - \mathbf{q}_\epsilon$ are needed. Since the $\mathbf{q} \to 0$ limit may be direction dependent, this is done in practice by means of wave-functions computed on four different grids in $\mathbf{k}$-space, i.e. a starting $\mathbf{k}$-grid plus three grids with $\mathbf{k} + q_\epsilon\,\mathbf{e}_i$ slightly shifted along the three Cartesian directions $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$. Such approach is computationally more efficient, although it requires to generate a larger set of wave-functions. However, there exists a random phase associated to the wave-functions on each of the four $\mathbf{k}$-grids, since they are obtained by independent diagonalizations of the KS Hamiltonian. Because of this, *shifted grids* dipoles have inconsistent phases among different directions and it is not possible to use them when the dipole matrix elements are needed (instead of their square modulus only) as for example in the evaluation of the Kerr effect (see section 5.2.1) or for non-linear optics (see section 7.2).

The *G-space v* approach assumes that the only non-local terms in H are the kinetic energy and the pseudopotentials. There are however cases, for example when the Hamiltonian contains non-local hybrid functionals, Hubbard U terms, or nonlocal self-energies, in which the evaluation of the commutator may become very cumbersome. To solve this issue one could in principle use the *shifted grids* approach. However, this approach may also become impractical because of the calculation of wave-functions on the shifted grids.

For those cases we have implemented in `yambo` two alternative strategies, one for extended and one for isolated systems. For extended systems the *Covariant* approach exploits the definition of the position operator in $\mathbf{k}$ space: $\hat{\mathbf{r}} = \mathrm{i}\partial_{\mathbf{k}}$. The dipole matrix elements are then evaluated as finite differences between the $\mathbf{k}$-points of a single regular grid. A five-point midpoint formula is used, with a truncation error $O(\Delta\mathbf{k}^4)$. The *shifted grids* and the *Covariant* approach are very similar, however in the latter the arbitrary phase of the wave-functions at different $\mathbf{k}$-points is correctly accounted for. To this aim, $i\partial_{\mathbf{k}}$ is implemented as a covariant derivative which cancels the relative phase factor (see appendix D for details). For these reasons the *Covariant* approach overcomes the limitations of the *shifted grids* approach. The main drawbacks of the *Covariant* approach is that the numerical value of the dipoles needs to be converged against the size of the $\mathbf{k}$ grid and the present implementation does not work for metals. However, in practice the convergence of dipole matrix elements is usually faster than that of the absorption spectrum.

For finite systems, finally, the dipole matrix elements can be directly evaluated in real space (*R-space x* approach).

We underline that in the case of a local Hamiltonian all approaches are equivalent. The desired strategy can be selected via the input variable:

```
DipApproach="G-space v"
#[Xd]  [G-space v/R-space x/Shifted grids/Covariant]
```

(*G-space v* being the default value).

3.2. Coulomb interaction

The Coulomb interaction enters in many sections of the `yambo` code, such as linear response, self-energy, and BSE kernel calculation. In reciprocal space, the bare Coulomb interaction for bulk systems is defined as $v(\mathbf{q} + \mathbf{G}) = 4\pi/|\mathbf{q} + \mathbf{G}|^2$. For the calculation of quantities requiring integration over transferred momenta in the Brillouin zone (BZ), such as the self-energy, the integrals are evaluated by summations over regular $\mathbf{q}$-grids. In order to remove divergencies in systems of reduced dimensionality, i.e. in the presence of a 2D or 1D sampling of $\mathbf{k}$-points, or to speed up the convergences in 3D systems, `yambo` offers the possibility to evaluate Coulomb integrals by using the *random integration method* (RIM), which consists of evaluating these integrals by Monte Carlo sampling (as already discussed in detail in section 3.1 of [4]), dividing the full BZ in small regions around each $\mathbf{k}$-point of the chosen uniform grid.

In order to avoid spurious interaction between replicas when dealing with low-dimensional materials such as clusters, slabs, or wires, `yambo` can also use Coulomb cutoff truncation techniques. These consists of truncating the Coulomb interaction beyond a certain region (depending on the chosen geometry):

$$\tilde{v}(\mathbf{r}) = \begin{cases} 1/r & \text{if } \mathbf{r} \in D \\ 0 & \text{if } \mathbf{r} \notin D. \end{cases} \quad (4)$$

Different geometrical choices are available. Spherical and cylindrical shapes, suitable to treat 0D and 1D systems, respectively, have been already described in details in [50]. In addition, a box-like cutoff obtained by performing a numerical Fourier transform of the real space expression is available for 0D systems. By defining only one or two sides of the box, it is possible to treat 2D or 1D systems within the same numerical approach. It is important to stress that, as the construction of such potential requires integration over the BZ, the RIM method discussed above must be activated.

Finally, a Wigner–Seitz truncation scheme, similar to the one discussed in [51] is also available. In this scheme the Coulomb interaction is truncated at the edge of the Wigner–Seitz super-cell compatible with the $\mathbf{k}$-point sampling. This truncated Coulomb potential turns out to be suitable for finite systems as well as for 1D and 2D systems, provided that the supercell size, determined by the adopted $\mathbf{k}$-point sampling, is large enough to get converged results [52].

3.3. Sum-over-states terminators in IP linear response

The independent particle polarizability $\chi^0_{\mathbf{GG}'}(\mathbf{q}, \omega)$, equation (1), and the correlation part of the GW self-energy

$\Sigma_c(\omega)$, equation (7) in section 4.1, are evaluated through sum-over-states (SOS) expressions obtained by applying an energy cutoff to the infinite sum over virtual states. These expressions are, however, slowly convergent and, especially for large systems, require the inclusion of a large number of empty states (N_b). This condition makes GW calculations computationally demanding, both in terms of time-to-solution and memory requirements. In order to overcome this limitation, a number of approaches have been proposed to reduce [53–56] or remove [3, 8, 57] sum over states; among them, we have implemented in `yambo` the extrapolar correction scheme proposed by Bruneval and Gonze (BG) [53].

This scheme, here referred as X-terminator, permits to accelerate GW convergence by reducing of a sensible amount the number of virtual orbitals necessary to calculate both polarizability and self-energy. In this procedure extra terms, whose calculation implies a small computational overhead, are introduced to correct both polarizability and self-energy by approximating the effect of the states not explicitly taken into account. The method consists in replacing the energies of empty states that are above a certain threshold, and that are not explicitly treated, by a single adjustable parameter defined as extrapolar energy. When the method of terminators is applied, the independent-particle polarizability can be written as [53]:

$$\chi^0_{\mathbf{GG'}}(\mathbf{q},\omega) = \chi^{0,\mathrm{trunc}}_{\mathbf{GG'}}(\mathbf{q},\omega) + \Delta\chi_{\mathbf{GG'}}(\mathbf{q},\omega,\bar{\epsilon}_{\chi_0}) \qquad (5)$$

where the first term on the rhs is truncated at the N'_b state (in general $N'_b \ll N_b$) and the second term depends on the extrapolar energy for the polarizability $\bar{\epsilon}_{\chi_0}$. The explicit expression for $\Delta\chi_{\mathbf{GG'}}(\mathbf{q},\omega,\bar{\epsilon}_{\chi_0})$ is provided in appendix C.

In the present implementation of `yambo`, the input parameter governing the use of the terminator corrections on the response function (X-terminator) is

```
XTermKind = "none"  # [X] X terminator
("none","BG")
```

(default: `"none"`). When the variable is set to `none` (default option), the X-terminator is not applied. On the contrary when `XTermKind` assumes the value `BG`, the extrapolar corrective term is calculated. The extrapolar energy $\bar{\epsilon}_{\chi_0}$, see equation (C.3), is defined by the input variable (default: 1.5 Ha)

```
XTermEn = 1.5 Ha  # [X] X terminator
energy
```

The value means $\bar{\epsilon}_{\chi_0} = \epsilon_{N_{b'}\mathbf{k}} + 1.5$ Ha, with $\epsilon_{N_{b'}\mathbf{k}}$ the highest energy state included in the calculation.

For demonstration purposes, in figure 3 we report the calculated QP corrections for the valence band maximum (VBM) and the conduction band minimum (CBM) of a bulk Si described in a supercell (36 Si atoms, 72 occupied states). Results are obtained by increasing the number of bands explicitly included in the calculation of the response function χ and by imposing a very high number of bands in the self-energy, that is therefore converged. Empty circles connected with solid lines denote the results obtained for the VBM and CBM states without applying any correction. Improvements induced by the use of the X-terminator

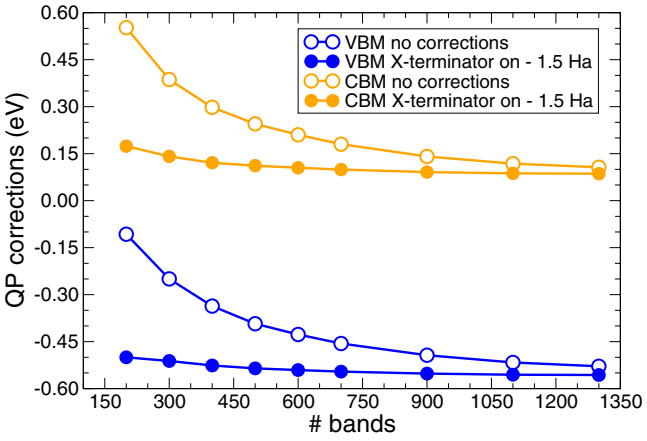


Figure 3. Effect of the X-terminator on the convergence (versus number of bands included in the response function) of the VBM and CBM GW-corrections for a bulk Si described in a supercell.

are depicted by solid circles connected with solid lines that have been obtained imposing `XTermEn` = 1.5 Ha. We can observe that the X-terminator leads to a relevant reduction in the number of bands necessary to converge the polarizability and thus the GW corrections.

4. Quasi-particle corrections

Accurate quasi-particle energies can be obtained by calculating self-energy corrections to KS energies [58]. In general, the non-local, non-Hermitian and frequency dependent electronic self-energy operator can be expressed as the sum of a bare, energy independent exchange term and a screened, dynamic correlation term:

$$\Sigma(\mathbf{r},\mathbf{r}',\omega) = \Sigma_x(\mathbf{r},\mathbf{r}') + \Sigma_c(\mathbf{r},\mathbf{r}',\omega). \qquad (6)$$

In this section we describe features implemented in `yambo` aimed at improving the accuracy of GW calculations by going beyond the commonly used plasmon-pole approximation [59] for the dielectric matrix and in speeding up calculations by reducing the number of empty states needed to get converged results. GW energies on top of KS eigenvalues are commonly calculated by considering one-shot corrections using the G_0W_0 approximation. Nevertheless in `yambo` is also possible to perform partial self consistent calculations (evGW), where eigenvalues entering the Green's function and polarizability are iterated until self-consistency is reached, while wave functions are kept frozen. This approach generally reduces the starting point dependence and it has been shown to provide reliable results for molecular systems [60, 61], wide band-gap materials [62] and perovskites [63, 64]. In the following we will just refer in general to the GW approach and discuss how the GW self-energy is computed.

4.1. Full frequency GW

Within the GW approximation, the matrix elements of the correlation self-energy over the KS basis are expressed as:

$$\langle n\mathbf{k}|\Sigma_c(\omega)|n'\mathbf{k}\rangle = \sum_{m\mathbf{q}} \int \frac{\mathrm{d}\omega'}{2\pi \mathrm{i}} I^{nn'\mathbf{k}}_{m\mathbf{q}}(\omega') \left[\frac{f_{m\mathbf{k}-\mathbf{q}}\theta(\omega')}{\omega - \omega' - \epsilon_{m\mathbf{k}-\mathbf{q}} - \mathrm{i}\eta} + \frac{(1-f_{m\mathbf{k}-\mathbf{q}})\theta(-\omega')}{\omega - \omega' - \epsilon_{m\mathbf{k}-\mathbf{q}} + \mathrm{i}\eta} \right]. \tag{7}$$

I is linked to the self-energy spectral function. From a computational point of view the definition of I is really critical as, in equation (7), it is connected to the self-energy via a complex Hilbert transformation. In `yambo` I is defined as

$$I^{nn'\mathbf{k}}_{m\mathbf{q}}(\omega') = -\frac{1}{N_k \Omega} \sum_{\mathbf{G}\mathbf{G}'} W^{\delta}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega') \times \rho_{nm\mathbf{k}}(\mathbf{q},\mathbf{G}) \rho^{*}_{n'm\mathbf{k}}(\mathbf{q},\mathbf{G}'). \tag{8}$$

In equation (8), W^{δ} is the *delta*-like part of the screened interaction. This is defined by

$$W^{\delta}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = \left[\frac{1}{2} \Im \left(W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) + W_{\mathbf{G}'\mathbf{G}}(\mathbf{q},\omega) \right) - \frac{\mathrm{i}}{2} \Re \left(W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) - W_{\mathbf{G}'\mathbf{G}}(\mathbf{q},\omega) \right) \right]. \tag{9}$$

In equation (9) $W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega)$ is the screened Coulomb potential defined as

$$W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = \epsilon^{-1}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) \frac{4\pi}{|\mathbf{q}+\mathbf{G}||\mathbf{q}+\mathbf{G}'|}. \tag{10}$$

Note that, in the case of systems with both spatial and time reversal symmetry, $W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega) = W_{\mathbf{G}'\mathbf{G}}(\mathbf{q},\omega)$ and $W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega)$ reduces to the imaginary part of W.

In order to take into account the frequency dependence of the self-energy, two different strategies are implemented in `yambo`. As already described in [4], it is possible to adopt the plasmon-pole approximation (PPA) in order to model the dynamic screening matrix. This approximation essentially assumes that all the spectral weight of the dielectric function is concentrated at a plasmon excitation. Among different models present in the literature `yambo` implements the Godby–Needs construction [65] where the parameters of the model are chosen in such a way that $\epsilon^{-1}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega)$ is reproduced at two different frequencies: the static limit $\omega = 0$ and another imaginary frequency $\omega = \mathrm{i}\omega_p$ given in the input file by `PPAPntXp` (default: `1 Ha`). Quasi-particle energy levels calculated within this approximation have been shown to agree to a large extent with numerical integration methods for materials with different characteristics including semiconductors and metal-oxides [59, 66]. Moreover, it has the great advantage to avoid the computation of the inverse of the dielectric matrix for many frequency points and to make the frequency integral of equation (7) expressible in an analytic form. Nevertheless the assumption made for the PPA breaks down in certain situations as when dealing with metals [67–69] or interfaces [70] and the frequency integral needs to be solved numerically. In `yambo` the integral is solved on the real-axis which implies the knowledge of the full frequency dependence of $W_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega)$. In practice, first the inverse dielectric function $\epsilon^{-1}_{\mathbf{G}\mathbf{G}'}(\mathbf{q},\omega)$ is evaluated for a number of frequencies set by the variable `ETStpsXd`, and uniformly distributed in the energy range given by the maximum electron–hole pairs included in the response function defined in equation (1). Next, the summation over $\mathbf{G}$ and $\mathbf{G}'$ is performed computing $I^{nn'\mathbf{k}}_{m\mathbf{q}}(\omega')$ defined in equation (8), and finally the correlation part of the self-energy is computed via a Hilbert transform defined in equation (7).

In this scheme, the evaluation of equation (10) is the most time consuming step due the computation of the inverse dielectric matrix for a large number of frequencies (order of 100) in order to have converged results. Nevertheless as the calculations for each frequency are independent from each other, parallelization over frequencies provides a linear speedup.

Quasi-particle energies calculated by using the real-axis method have been demonstrated to provide the same level of accuracy of other beyond plasmon-pole techniques such as the contour deformation scheme [71].

4.2. Electron-mediated lifetimes

The ability of `yambo` to calculate the real-axis GW self-energy allows direct access to the quasi-particle electron-mediated lifetimes. Indeed if we define $\Gamma^{\mathrm{e-e}}_{n\mathbf{k}}(\omega) \equiv \Im \left(\langle n\mathbf{k}|\Sigma_c(\omega)|n\mathbf{k}\rangle \right)$, from equation (7) it is easy to see that

$$\Gamma^{\mathrm{e-e}}_{n\mathbf{k}}(\omega) = \frac{1}{2} \sum_{m\mathbf{q}} I^{nn\mathbf{k}}_{m\mathbf{q}}(\omega - \epsilon_{m\mathbf{k}-\mathbf{q}}) \times \left[\theta(\omega - \epsilon_{m\mathbf{k}-\mathbf{q}}) f_{m\mathbf{k}-\mathbf{q}} - \theta(\epsilon_{m\mathbf{k}-\mathbf{q}} - \omega)\,(1 - f_{m\mathbf{k}-\mathbf{q}}) \right]. \tag{11}$$

`yambo` can evaluate the quasi-particle lifetimes $\tau_{n\mathbf{k}}(\omega)$, proportional to the inverse of $\Gamma^{\mathrm{e-e}}_{n\mathbf{k}}(\omega)$, either in the on-the-mass-shell approximation (OMS) or in the full GW approximation. The difference between the two is the inclusion of the renormalization factors, $Z_{n\mathbf{k}}$. For details on the theory see, for example, [68] and references therein.

In the OMS we have that $\Gamma^{\mathrm{e-e}}_{n\mathbf{k}}|_{\mathrm{OMS}} = \Gamma^{\mathrm{e-e}}_{n\mathbf{k}}(\epsilon_{n\mathbf{k}})$. The e–e lifetimes of bulk copper are shown in figure 4 using several flavours of GW approximations [68].

An important numerical property of the electron-mediated lifetimes calculation is that they depend only on the $\mathbf{k}$-grid. Indeed, as evident from equation (11) the band summations are limited by the two theta functions that confine the scattering events in reduced regions of the BZ. This is the

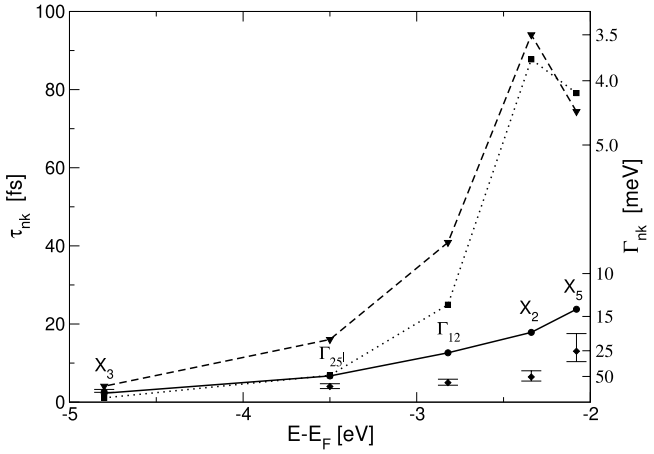


Figure 4. e–e linewidths ($\Gamma_{n\mathbf{k}}$) and lifetimes ($\tau_{n\mathbf{k}}$) of selected *d*-bands of copper. Different level of approximations are shown together with the experimental data (diamonds with error bars). The calculated lifetimes are: full line; G_0W_0. Dotted line: OMS G_0W_0. Dashed line: OMS G_1W_0. (reprinted with permission from [68]. © (2001) by the American Physical Society).

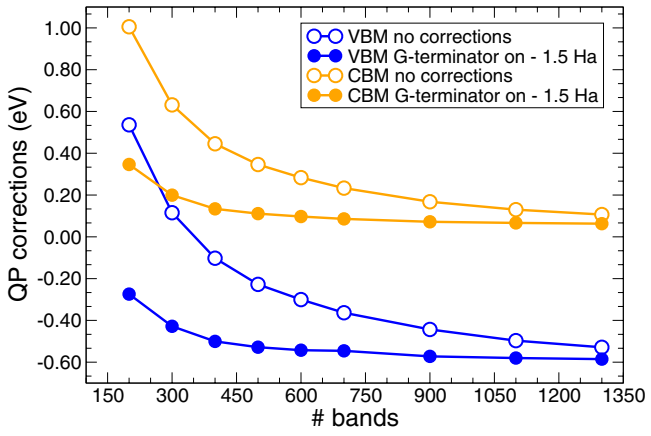


Figure 5. Effect of the G-terminator on the convergence versus number of bands included in the self-energy on the VBM and CBM GW-corrections for a bulk Si described in a supercell.

mechanism that, in simple metals, leads to the well known quadratic scaling of $\Gamma^{e-e}_{n\mathbf{k}}|_{OMS}$ near the Fermi level, as a function of distance of $\epsilon_{n\mathbf{k}}$ from the Fermi level itself.

More physical insight in the electronic lifetimes will be given in section 6.2 where the phonon-mediated case will be described.

4.3. Reducing the number of empty states summation: terminators

In section 3.3 we have discussed the X-terminator procedure. A similar scheme can be adopted to study the correlation part of the GW self-energy, as from equation (16) of [53]. Also in this case the approximation implies the introduction of an extra term that takes into account contributions arising from states not explicitly included in the calculation. The input parameter governing the use of the terminator corrections on the self-energy (G-terminator) is

```
GTermKind = "none"    # GW terminator ("none","BG")
```

(default: `"none"`). When the variable is set to `none`, the G-terminator is not applied. On the contrary when it assumes the value `BG`, the extrapolar corrective term is calculated. The extrapolar energy for the self-energy is defined by the tunable input variable

```
GTermEn = 1.5 Ha    # [X] X terminator energy
```

(default: 1.5 Ha).

Also in this case, the value is referenced to the highest band included in the calculation. In figure 5 we reconsider the system discussed in the example of section 3.3, figure 3. In this case, however, we study the convergence of the self-energy by exploiting the G-terminator procedure. Empty circles connected with solid lines show the usual GW convergence for the VBM and CBM states (no corrections applied). Calculations have been performed by imposing a high number of bands in the polarisability (that is therefore converged) and by increasing the number of bands included in the self-energy. We set `GTermEn` = 1.5 Ha, that represents the best choice for this system. Improvements provided by the use of the G-terminator procedure are represented by solid circles connected with solid lines; it is evident that the application of this scheme accelerates the convergence by leading to a significant reduction in the number of states necessary to converge the GW self-energy and therefore the calculated QP correction.

In order to elucidate the role played by the extrapolar parameter, we report in figure 6 a convergence study of the VBM GW correction for a TiO_2 nanowire (NW). The black line is obtained without applying any correction. Coloured lines are instead obtained by applying both X- and G-terminators, moving the extrapolar energy from 1.0 to 3.0 Ha. Results are reported as a function of the number of states explicitly included in the calculation of both polarisability and self-energy. As pointed out in [53], the extrapolar energy for the self-energy can be safely taken equal to the extrapolar energy introduced in equation (C.3) for the polarizability; for this reason we impose `XTermEn` = `GTermEn`. Consistently with the study of figure 6, the convergence of the VBM without terminators is very slow and requires the inclusion of a large number of bands to be achieved; this condition makes the calculation cumbersome also on modern HPC-machines. When the terminator technique is adopted to correct both polarisability and self-energy, the convergence becomes much faster; especially for some values of the extrapolar energy (about 1.5 Ha), we observe a significant reduction in the number of bands necessary to converge the calculation, with a strong reduction of both the time-to-solution and the allocated memory. Noticeably, the correction is almost independent on the selected extrapolar energy (terminators are convergence accelerators and the extrapolar correction vanished in the limit of infinite bands included); this parameter therefore influences the number of bands necessary to converge the calculation (and thus the computational cost of the simulation) but not the final result.

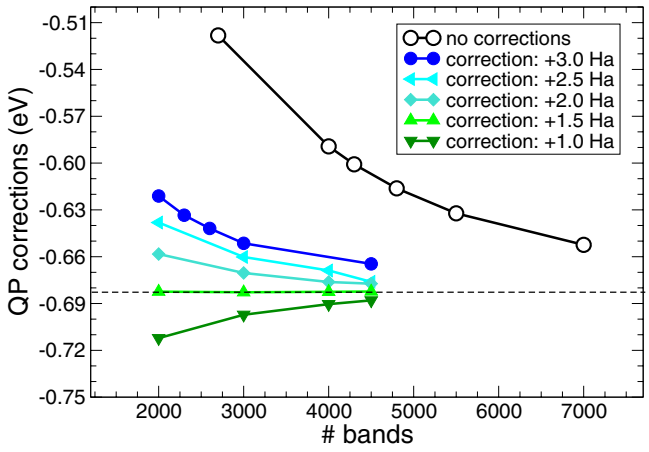


Figure 6. Convergence plots of GW-corrected data for the VBM of a TiO_2 NW (27 atoms, 108 occupied states) as a function of the number of bands included in the calculation. Response and self-energy terminators are simultaneously applied. Calculations have been performed using the same number of bands for the polarizability and the self-energy. The black line show the usual GW convergence with no corrections. Coloured lines are obtained applying the method of [53] with different values of the extrapolar energy, ranging from 1.0 to 3.0 Ha above the last explicitly calculated KS state.

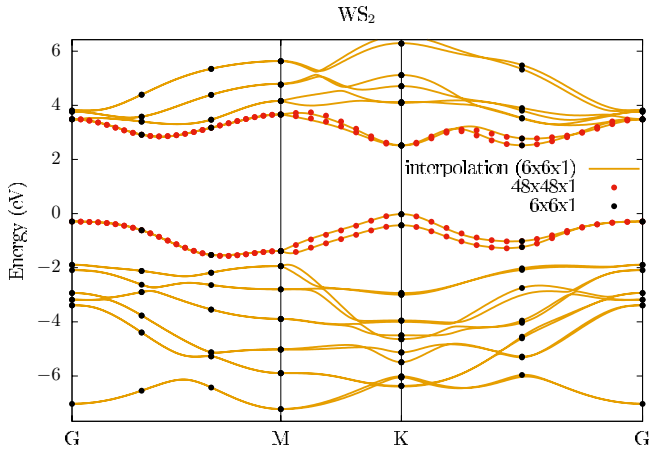


Figure 7. GW band structure of monolayer WS_2 including spin–orbit coupling and using $48 \times 48 \times 1$ **k**-points grid for the self-energy. The orange lines represent Wannier-interpolated bands obtained from 7 QP energies corresponding to a $6 \times 6 \times 1$ grid (black dots), while the red dots shows the QP energies of the full $48 \times 48 \times 1$ grid.

4.4. Interpolation of the QP band structure

In DFT the eigenvalues of the Hamiltonian at every **k**-point can be obtained by the knowledge of the ground-state charge density, allowing one to perform non-self-consistent calculations on an arbitrary set of **k**-points. Instead at the HF or GW level, to obtain QP corrections for a given **k**-point it is necessary to know the KS wave-functions and eigen-energies on all $(\mathbf{k}+\mathbf{q})$-points, having chosen a regular grid of **q**-points as convergence parameter. In practice `yambo` computes QP corrections on a regular grid. As a consequence the evaluation of band structures along high-symmetry lines can be computationally very demanding.

A simple strategy which is implemented in `ypp` is to interpolate the QP corrections from such regular grid to the desired high symmetry lines. The approach implemented is based on a smooth Fourier interpolation [72], which is particularly efficient for 3D grids. The interpolation scheme can also take, as additional input, the KS energies computed along the high symmetry lines to better deal with bands crossing and regions with non analytic behaviour, such as cusp-like features.

A more involved strategy is instead based on the Wannier interpolation scheme as implemented in the `wannier90` [73] and `WanT` [74] codes, where electronic properties computed on a coarse reciprocal-space mesh can be used to interpolate onto much finer meshes at low cost [75][19]. In the context of GW calculations, the Wannier interpolation scheme can be used to interpolate the QP energies and other band structure properties [74] (e.g. effective masses) from QP corrections computed only on selected **k**-points. Wannier interpolation of GW band structures requires two sets of inputs: on one side quantities computed at the DFT level such as KS eigenvalues, overlaps between different KS states, and orbital and spin projections of KS states, that are imported from `Quantum ESPRESSO`, and on the other side the QP corrections computed by `yambo`. In fact, `wannier90` works with uniform coarse meshes on the whole BZ, while `yambo` uses symmetries to compute quantities on the IBZ. In addition, converging the GW self-energy typically requires denser meshes with respect to what is needed for the charge density or Wannier interpolation. To address this issue, `ypp` allows one to unfold the QP corrections from the IBZ to the whole BZ, as required by `wannier90` for interpolation purposes. Finally the `wannier90` code yields a GW-corrected Wannier Hamiltonian and interpolates the GW band structure. A similar procedure is implemented in `WanT`.

For example, in monolayer WS_2 a grid of $48 \times 48 \times 1$ (or denser) is required to converge the GW self-energy. In this case, the band structure can be obtained either by explicitly computing the QP corrections on all **k**-points of the $48 \times 48 \times 1$ grid, or it can be Wannier-interpolated from the QP corrections computed onto coarse subgrids, such as a $6 \times 6 \times 1$ corresponding to seven symmetry-nonequivalent **k**-points only in the IBZ (see figure 7). The second approach requires substantial less CPU time.

5. Optical absorption

The solution of the Bethe–Salpeter equation on top of DFT-GW is the state-of-the-art first principles approach to calculate neutral excitations in solid-state systems [1], with successful applications to, molecules [60, 76], surfaces [77, 78], two-dimensional materials [79, 80], and nanostructures [81, 82], including biomolecules in complex environments [83, 84]. The BSE is a Dyson equation for the four point response function L. It can be rewritten as an eigenproblem for a two-particle effective Hamiltonian H^{2p} in the basis of electron and hole pairs $|eh\rangle$. H^{2p} is the sum of an independent-particle Hamiltonian H^{IP}—i.e. the e–h energy differences

[19] A tutorial on the Wannier interpolation of the GW band structure of silicon is available in the `wannier90` package on `GitHub`.

corresponding to the independent-particle four-point response function L_0—and the exchange V and direct contributions W accounting for the e–h interaction. The original implementation of the BSE in `yambo` (see section 2.2 and 3.2 of CPC2009) has been extended in the past decade to (i) improve its numerical efficiency (section 5.1)—allowing one to treat systems with a large number of electron–hole pairs (i.e. above 10^5)—and (ii) to capture physical effects (section 5.2) that were originally neglected—e.g. allowing for the description of the Kerr effect in magnetic materials (section 5.2.1). Finally, a range of tools have been developed to analyse the exciton localization both in real and reciprocal space (section 5.3).

5.1. Numerical efficiency

The computational cost of the BSE grows as a power of the number of electron–hole pairs. As this number can be as large as 10^5–10^6, it is crucial to devise numerically efficient algorithms for the calculation of the V and W matrix elements and the solution of the BSE. The massive parallelization and memory distribution which contributes in making these calculations possible for very large systems are discussed in section 8. Here we discuss the use of the double grid for the sampling of the BZ [85], where the BSE is solved (section 5.1.1)—which aims at reducing the number of degrees of freedom involved—and the use of Lanczos-based algorithms together with the interface to the SLEPC library (Scalable Library for Eigenvalue Problem Computations) [86] (section 5.1.2)—which aims at avoiding the full diagonalization of H^{2p}.

5.1.1. Double-grid and the inversion solver. The BSE implementation in `yambo` is based on an expansion of the relevant quantities in the basis of electron–hole states. This expansion often requires a very dense **k**-point sampling of the Brillouin zone (BZ). Typically, the number of electron–hole states used in the expansion can be relatively small if one is only interested in the absorption spectra, but the number of **k**-points can easily reach several thousands. Different approaches have been proposed in the literature to solve this problem. A common approach is the use of arbitrarily shifted **k**-point grids, that often yield sufficient sampling of the BZ while keeping the number of **k**-points manageable. Such a shifted grid, indeed, does not use the symmetries of the BZ and guarantees a maximum number of nonequivalent **k**-points thereby accelerating spectrum convergence. However, it may induce artificial splitting of normally degenerate states, thus producing artifacts in the spectrum. In `yambo` we introduced a strategy to solve the BSE equation that alleviates the need for dense **k**-point grids and does not break the BZ symmetries. Such approach takes into account the fast-changing independent-particle contribution [85, 87]. Indeed the independent-particle term of the BSE, L_0, is evaluated on a very dense **k**-grid and then the BSE is solved on a coarse **k**-grid. This means in practice that L_0 remains defined on the coarse grid, but each matrix element of L_0 contains the sum of the nearby poles on the dense grid. The dense grid can be generated by means of DFT and read using `ypp -m`, that creates a mapping between the coarse and the dense grid. Then BSE is solved by inversion setting `BSSmod` = '`i`'. A similar approach can be also used when computing the response function in **G** space, by replacing each transition in the $F_{nmk}(\mathbf{q},\omega)$ term in equation (B.1) with a sum over the transitions in the dense grid.

5.1.2. Spectra and exciton wavefunctions via Krylov subspace methods. Solving the BSE implies the solution of an eigenvalues problem for the two-particle Hamiltonian that in the e–h basis can result in a matrix as large as $10^6 \times 10^6$. The standard dense matrix diagonalization algorithm is available in `yambo` through the interface with the LAPACK and the ScaLAPACK libraries [88] (section 8). Alternatively, when only the spectrum is required, `yambo` provides the Haydock–Lanczos solver [89]. The latter, originally developed for the Hermitian case only (see section 3.2 of CPC2009)—by neglecting the coupling between e–h at positive and negative energies within the Tamm–Dancoff approximation—has been extended to treat the full non-Hermitian two-particle Hamiltonian [90, 91]. Cases in which considering the full non-Hermitian two-particle Hamiltonian turns out to be important have been discussed in [90, 92, 93].

More recently, `yambo` has been interfaced with the SLEPC library [86] which uses objects and methods from the PETSC library [94] to implement Krylov subspace algorithms to iteratively solve eigenvalue problems. These are used in `yambo` to obtain selected eigenpairs of the excitonic Hamiltonian. This allows the user to select a fixed number of excitonic states to be explicitly calculated thus avoiding the full dense diagonalization and saving a great amount of computational time and memory. Two options are available for the SLEPC solver. The first, which is the default, uses the PETSC matrix-vector multiplication scheme; it is faster but duplicates the BSE matrix in memory when using MPI. The second, which is activated by the logical `BSSSlepcShell` in the input file, uses the internal `yambo` subroutines (the same also used for the Haydock solver); it is slower but distributes the BSE matrix among the MPI tasks. To select the part of the spectra of interest, the library allows one to use different extraction methods controlled by the variable `BSSSlepcExtraction`. The standard method, `ritz`, obtains the lowest lying eigenpairs, while the `harmonic` method obtains the eigenpairs closest to a defined energy. The SLEPC solver makes it possible to obtain and plot exciton wave functions (`ypp -e w`) in large systems where the full diagonalization might be computationally too demanding. For example, the spectrum and the wave function of the lowest-lying exciton in monolayer hBN are shown in figure 8. The BSE eigenmodes were extracted only for the two lowest-lying excitonic states, and a $\sqrt{3} \times \sqrt{3} \times 1$ supercell was used in the calculation (the SLEPC spectrum is shown in blue). The full Haydock solution is displayed with a red line for comparison.

5.2. Physical effects

5.2.1. Spin–orbit coupling and Kerr. With the implementation and release of the full support for non-collinear systems it is now possible to account for the effects of spin–orbit coupling (SOC) on the optical properties at the BSE level. A

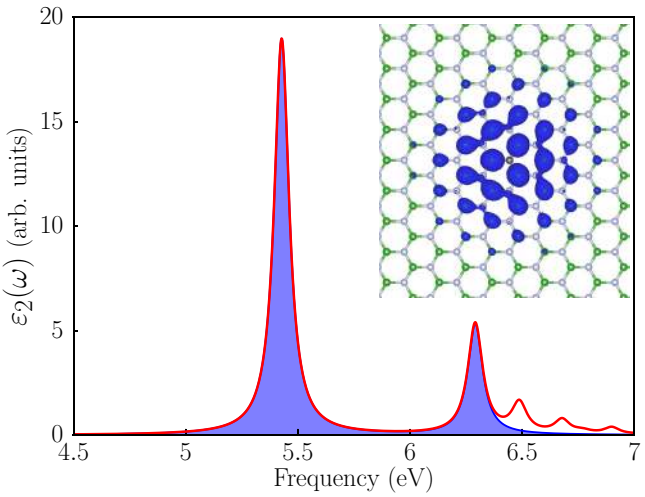


Figure 8. Optical absorption spectrum of monolayer hBN in a $\sqrt{3} \times \sqrt{3} \times 1$ supercell. The red line refers to an iterative solution using the Haydock solver. The blue shaded region corresponds to a SLEPC calculation where only the first two excitons were included. The inset shows the intensity of the exciton wave function corresponding to the main peak, based on the latter calculation (the hole position is fixed above a nitrogen atom and the resulting electron distribution is displayed).

detailed description of the implementation and a comparison with other simplified approaches (like the perturbative SOC) can be found in [96]. Since the BSE is written in transition space, the definition of the excitonic matrix is not different from the collinear cases of both unpolarized and spin-collinear systems. For a given number of bands, the main difference is that in the unpolarized case the matrix can be blocked in two matrices of size $N \times N$, describing singlet and triplet excitations. Already in the spin-collinear case this is not possible and the matrix has twice the size $2N \times 2N$. In the non-collinear case, the z-component of the spin operator, S_z, is not a good quantum number and the size of the matrix becomes $4N \times 4N$. Since SOC is usually a small perturbation, this means in practice that in the non-collinear case there are peaks which are shifted in energy as compared to the collinear cases ($\Delta S_z = 0$ transitions) plus the possible appearance of very low intensity peaks corresponding to spin flip transitions.

The ability of the BSE matrix to capture the interplay between absorption and spin, makes the approach suitable to describe magneto-optical effects. Indeed, starting from the BSE matrix, the off-diagonal matrix elements of the macroscopic dielectric tensor $\varepsilon_{ij}(\omega)$ can be derived, thus describing the magneto-optical Kerr effect [97]. Notice that in the definition of $\varepsilon_{ij}(\omega)$ the product of dipoles $x^*_{nm\mathbf{k}} y_{nm\mathbf{k}}$ enters, thus requiring approaches where the relative phases between different dipoles are correctly accounted for. To this end the `yambo_kerr` executable must be used, activating the `EvalKerr` flag in the input file. The correct off-diagonal matrix elements of the dielectric tensor can be obtained in the velocity gauge (see section 5.2.2), and only for systems with a gap and Chern number equal zero in the length gauge [97].

5.2.2. Fractional occupations, gauges and more. Other extensions have been made available. The implementation has been modified so that the excitonic matrix is now Hermitian (or pseudo-Hermitian if coupling is included) also in the presence of fractional occupations in the ground state. This is done in practice by introducing a slightly modified four-point response function $\tilde{L}$ which is divided by the square root of the occupations as discussed in equations (14)–(16) of [98]. The resulting excitonic Hamiltonian has the form

$$\tilde{H}_{ll'} = \Delta\epsilon_l \delta_{ll'} - \sqrt{\Delta f_l}\,(v_{ll'} - W_{ll'})\,\sqrt{\Delta f_{l'}} \qquad (12)$$

with $l = \{nk\mathbf{k}\}$ a super-index in the transition space, with the square root of the occupation factors appearing on the left and on the right of the BSE kernel $v - W$. This has been used to compute absorption of systems out of equilibrium, but it is also important to describe metallic systems like graphene or carbon nanotubes where excitonic effects can be non-negligible due to the reduced dimensionality of the system.

Further, it is now possible to compute the dielectric tensor starting from the different response functions, as described in [99]. Indeed, starting from the excitonic propagator L, it is possible to construct the density–density response function $\chi_{\rho,\rho}$, or the dipole–dipole response function $\chi_{\mathbf{d},\mathbf{d}}$ at $\mathbf{q} = 0$ (length gauge), and the current–current response function $\chi_{\mathbf{j},\mathbf{j}}$ (velocity gauge). This can be controlled by setting `Gauge="length"` or `Gauge="velocity"` in the input file (the length gauge is the default). In case the velocity gauge is chosen the conductivity sum rule is imposed unless the flag `NoCondSumRule` is activated in the input file. At zero momentum, changing response function is equivalent to change gauge. At finite $\mathbf{q}$ instead the use of $\chi_{\mathbf{j},\mathbf{j}}$ allows for the calculation of both the longitudinal and the transverse components of the dielectric function. The finite-q BSE has been implemented and it is currently under testing before its final release.

Another extension is connected to the output of a BSE run, which also generates a file with the joint density-of-states, at the IP level, and the excitonic density of states, at the BSE level. These can be used for example to visualize dark or very small intensity peaks as shown in figure 9.

5.3. Analysis of excitonic wavefunctions

Once a BSE calculation is performed using an algorithm which explicitly computes the excitonic eigenvectors $A^{\lambda}_{cv\mathbf{k}}$, several properties of the excitons can be analyzed as shown in section 5.2.2 (see figure 8). First of all, the excitonic eigenvalues E_λ can be sorted and plotted. The so-called amplitudes and weights can also be calculated to inspect which are the main contributions in terms of single quasi-particles to a given excitonic state. The weights are defined as the squared modulus of the excitonic wavefunction $|A^{\lambda}_{\mathrm{eh}}|^2$ (by default only electron–hole pairs that contribute to the exciton more than 5% are considered; the threshold can be tuned by modifying the input file '`MinWeight`'). The amplitudes are defined as $\sum_{cv\mathbf{k}} |A^{\lambda}_{cv\mathbf{k}}|^2 \delta(\epsilon_{c\mathbf{k}} - \epsilon_{v\mathbf{k}} - \hbar\omega)$.

Moreover the excitonic wavefunction written in real-space $\Psi_\lambda(\mathbf{r}_e, \mathbf{r}_h) = \sum_{cv\mathbf{k}} A^{\lambda}_{cv\mathbf{k}} \psi^*_{v\mathbf{k}}(\mathbf{r}_h)\psi_{c\mathbf{k}}(\mathbf{r}_e)$ can be computed. $\Psi_\lambda(\mathbf{r}_e, \mathbf{r}_h)$ is a two-body quantity or joint-correlation function. Fixing the position of the hole $\mathbf{r}_h = \bar{\mathbf{r}}_h$, $|\Psi_\lambda(\bar{\mathbf{r}}_h, \mathbf{r})|^2$ provides the conditional probability of finding the electron somewhere

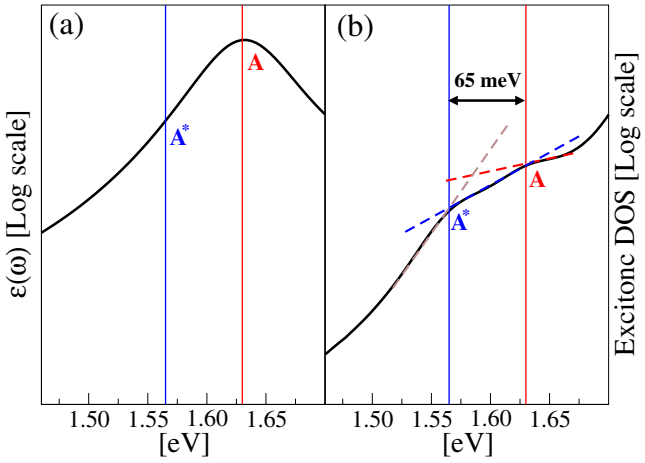


Figure 9. Optical absorption spectrum of a WSe_2 (panel (a)) from [95], compared with excitonic DOS (panel (b)). Calculations are performed including SOC. In the Excitonic DOS both the bright A and the dark A^* exciton are visible as a change in the slope (the dashed lines are a guide for the eyes, while the vertical continuous lines mark the energy positions of the excitons). Only the bright A exciton is instead visible in the absorption.

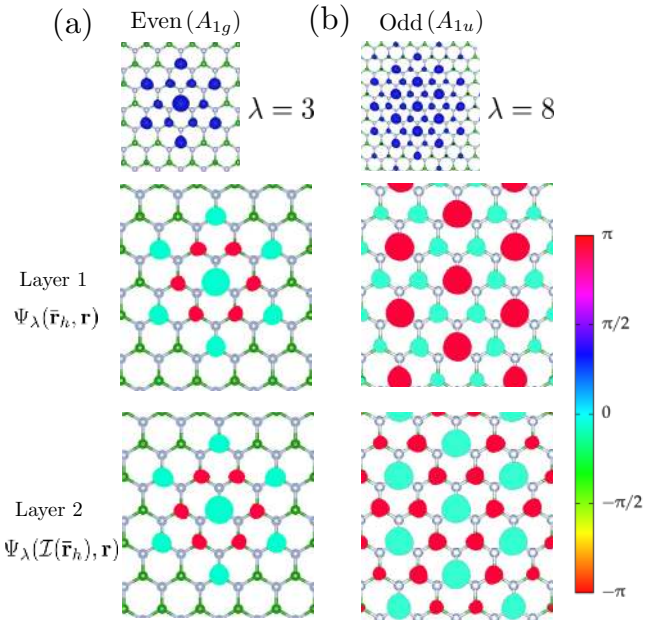


Figure 10. Exciton wave functions Ψ_λ of states $\lambda = 3$ (a) and $\lambda = 8$ (b) of bilayer hBN (only the layer where the electron density is non-negligible is shown). The intensity of Ψ_λ is shown in the top frames. Its phase is displayed in the lower frames for two inversion-symmetrical positions of the hole ($\bar{\mathbf{r}}_h$ and $\mathcal{I}(\bar{\mathbf{r}}_h)$). The hole is always fixed above a nitrogen atom of the layer not shown. Adapted from [100].

in space. This quantity is clearly nonperiodic and its spatial decay can change from material to material, marking the distinction between Frenkel and Wannier excitons. As an alternative it is also possible to plot $|\Psi_\lambda(\mathbf{r}, \mathbf{r})|^2$ which is instead Bloch-like.

In figure 10 we focus on two interlayer excitonic states of bilayer hexagonal boron nitride ($\lambda = 3$ and $\lambda = 8$). We first plot $|\Psi_\lambda(\bar{\mathbf{r}}_h, \mathbf{r})|^2$ (top frames), then we proceed to extract more information by analyzing the phase of $\Psi_\lambda(\bar{\mathbf{r}}_h, \mathbf{r})$ (lower frames). By comparing the phase for two positions of the hole related by inversion symmetry ($\bar{\mathbf{r}}_h$ and $\mathcal{I}(\bar{\mathbf{r}}_h)$), we see that the first exciton (figure 10(a)) is even with respect to inversion symmetry, while the second one is odd (figure 10(a)). Symmetry analysis of the wave function permits us to conclude that $\Psi_3(\mathbf{r}_h, \mathbf{r}_e)$ and $\Psi_8(\mathbf{r}_h, \mathbf{r}_e)$ transform as the A_{1g} and A_{2u} representations of point group D_{3h} of the lattice, respectively.

6. Electron–phonon interaction

The electron–phonon (EP) interaction is related to many materials properties [101] such as the critical temperature of superconductors, the electronic band gap and electronic carrier mobility of semiconductors [102], the temperature dependence of the optical spectra, the Kohn anomalies in metals [103], and the relaxation rates of carriers [104, 105]. `yambo_ph` calculates fully *ab initio* the EP coupling effects on the electronic states, on the excitonic states energies, and on the optical spectra. The approach used is the many-body formulation which is the dynamical extension of the static theory of EP coupling originally proposed by Heine, Allen, and Cardona (HAC) [106, 107]. In this framework, the QP energies are the complex poles of the Green's function written in terms of the EP self-energy, $\Sigma^{\text{el-ph}}_{n\mathbf{k}}(\omega, T)$, composed of two terms, the Fan, $\Sigma^{\text{FAN}}_{n\mathbf{k}}(\omega, T)$, and Debye–Waller $\Sigma^{\text{DW}}_{n\mathbf{k}}(T)$ contributions [108, 109] (for the complete derivation see, for example, [110, 111]):

$$\Sigma^{\text{FAN}}_{n\mathbf{k}}(\mathrm{i}\omega, T) = \sum_{n'\mathbf{q}\lambda} \frac{|g^{\mathbf{q}\lambda}_{nn'\mathbf{k}}|^2}{N_q} \times \left[\frac{N_{\mathbf{q}\lambda}(T) + 1 - f_{n'\mathbf{k}-\mathbf{q}}}{\mathrm{i}\omega - \varepsilon_{n'\mathbf{k}-\mathbf{q}} - \omega_{\mathbf{q}\lambda}} + \frac{N_{\mathbf{q}\lambda}(T) + f_{n'\mathbf{k}-\mathbf{q}}}{\mathrm{i}\omega - \varepsilon_{n'\mathbf{k}-\mathbf{q}} + \omega_{\mathbf{q}\lambda}} \right]. \quad (13)$$

Similarly

$$\Sigma^{\text{DW}}_{n\mathbf{k}}(T) = -\frac{1}{2N_q} \sum_{\mathbf{q}\lambda} \frac{\Lambda^{\mathbf{q}\lambda}_{nn'\mathbf{k}}}{\varepsilon_{n'\mathbf{k}} - \varepsilon_{n\mathbf{k}}} (2N_{\mathbf{q}\lambda}(T) + 1). \quad (14)$$

In equations (13) and (14) $N_{\mathbf{q}\lambda}(T)$ is the Bose function distribution of the phonon mode $(\mathbf{q}, \lambda)$ at temperature T.

The ingredients of $\Sigma^{\text{el-ph}}_{n\mathbf{k}}(\omega, T)$, apart from the electronic states, are the phonon frequencies $\omega_{\mathbf{q}\lambda}$ and the EP matrix elements: $g^{\mathbf{q},\lambda}_{nn'\mathbf{k}}$ (first order derivative of the self-consistent and screened ionic potential) and $\Lambda^{\mathbf{q}\lambda}_{nn'\mathbf{k}}$ (a complicated expression written in terms of the first order derivative [110, 111]).

These quantities are currently calculated with `Quantum ESPRESSO` within the framework of DFPT [112]. They are read and opportunely stored by the post-processing tool `ypp_ph` and then reloaded by `yambo_ph`. The procedure is analogous to the one followed by `Abinit` [113].

The HAC approach corresponds to the limit $\lim_{\omega_{\mathbf{q}\lambda}\to 0} \Sigma^{\text{FAN}}_{n\mathbf{k}}(\epsilon_{n\mathbf{k}}, T)$. In the HAC the Fan correction reduces to a static self-energy [110].

In the next subsections we will give more details about how `yambo_ph` has been used to calculate the temperature

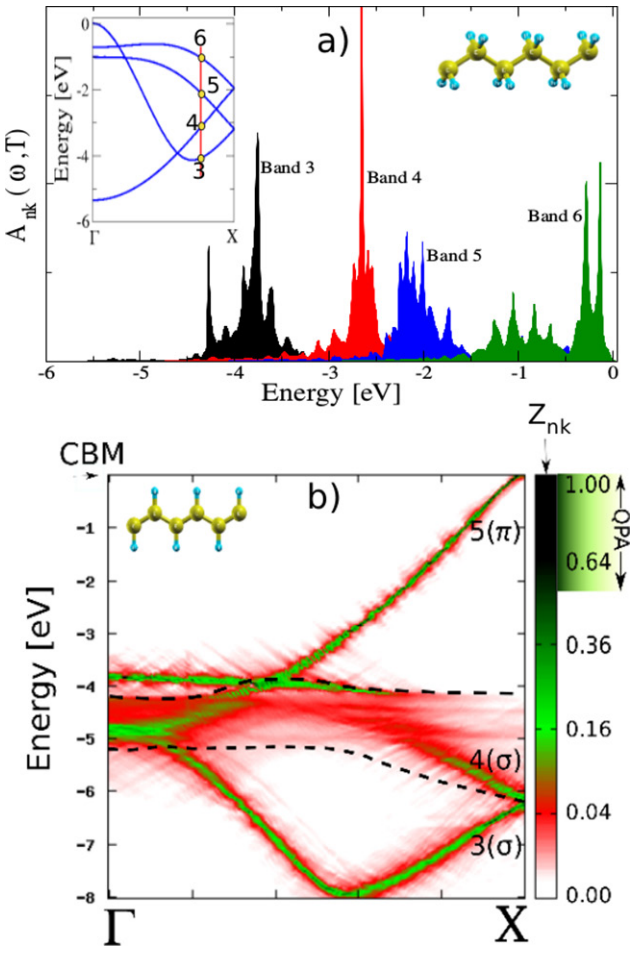


Figure 11. (a) Spectral functions (SFs) of few selected electronic states of trans-polyethylene. In the inset, the last four occupied bands are shown. The red line marks the **k**-point at which the corresponding SFs are presented. The selected states are marked with dots. Reprinted from [110] © EDP Sciences, SIF, Springer-Verlag Berlin Heidelberg 2012. With permission of Springer. (b) Two-dimensional plot of the SFs of trans-polyacetylene. The range of values of A(**k**,ω) are given in terms of dimensionless quantity $Z_{n\mathbf{k}}$. Reprinted with permission from [114], © 2011 American Physical Society.

dependence of the band structure (section 6.1) and of the optical spectrum (section 6.3). Finally, in section 6.4 we will describe the way the $\mathbf{q} \to 0$ divergence of EP matrix elements has been addressed.

6.1. Temperature-dependent electronic structure

The HAC approach is based on the static Rayleigh–Schrödinger perturbation theory, allowing one to calculate the temperature-dependent correction of the bare electronic energies, owing to the phonon field perturbation. In the QP approximation, the bare energy is instead renormalized because of the virtual scatterings described by the real part of the self-energy and it also acquires a finite lifetime due to the imaginary part of the self-energy. The eigenvalues $E_{n\mathbf{k}}(T)$ are then complex and depend on the temperature. The more the QP approximation is valid the more the renormalization factors $Z_{n\mathbf{k}}$ are close to 1, analogously to the GW method.

If the QP approximation holds, the spectral function $A_{n\mathbf{k}}(\omega, T) = \Im\left[G_{n\mathbf{k}}(\omega, T)\right]$ is a single-peak Lorentzian function centered at $\Re[E_{n\mathbf{k}}]$ with width $\Gamma_{n\mathbf{k}} = \Im[E_{n\mathbf{k}}]$. In case of strong EP interaction it has been proven that the spectral function spans a wide energy range [114, 115] and the QP approximation is no longer valid.

Figure 11 shows the spectral functions (SFs) of trans-polyethylene and trans-polyacetylene, calculated at $0K$. In figure 11(a) multiple structures appear in the SFs. SFs are then spread over a large energy range. In figure 11(b) a two-dimensional plot of the SFs reveals a completely different picture with respect to the original electronic band structures. Since SFs are featured by a multiplicity of structures, each carries a fraction of the electronic charge $Z_{n\mathbf{k}}$ depriving the dominant peak of its weight. A crucial aspect is that some SFs overlap, like in the case of trans-polyacetylene, and in the end it is impossible to associate a well defined energy to the electron and to state which band it belongs to.

6.2. Phonon-mediated electronic lifetimes

By following the same strategy used in the electronic case, the phonon-mediated contribution to the electronic lifetimes can be easily calculated from equation (13). Indeed if we define $\Gamma^{\text{e-p}}_{n\mathbf{k}}(\omega, T) \equiv \Im\Sigma^{\text{FAN}}_{n\mathbf{k}}(\omega, T)$ it is easy to see that

$$\Gamma^{\text{e-p}}_{n\mathbf{k}}(\omega, T) = \frac{\pi}{N_{\mathbf{q}}} \sum_{n'\mathbf{q}\lambda} |g^{\mathbf{q}\lambda}_{nn'\mathbf{k}}|^2 \Big[\delta(\omega - \epsilon_{n'\mathbf{k}-\mathbf{q}} - \omega_{\mathbf{q}\lambda})(N_{\mathbf{q}\lambda}(T) + 1 - f_{n'\mathbf{k}-\mathbf{q}}) + \delta(\omega - \epsilon_{n'\mathbf{k}-\mathbf{q}} + \omega_{\mathbf{q}\lambda})(N_{\mathbf{q}\lambda}(T) + f_{n'\mathbf{k}-\mathbf{q}})\Big]. \quad (15)$$

In perfect analogy with the electronic case, within the OMS, we have that $\Gamma^{\text{e-p}}_{n\mathbf{k}}(T)\Big|_{\text{OMS}} = \Gamma^{\text{e-p}}_{n\mathbf{k}}(\epsilon_{n\mathbf{k}}, T)$. Like in the electronic case the most important numerical property of the lifetimes calculation is that they depend only on the **q**-grid.

It is very instructive to compare $\Gamma^{\text{e-p}}_{n\mathbf{k}}(T)\Big|_{\text{OMS}}$ and the $\Gamma^{\text{e-e}}_{n\mathbf{k}}(T)|_{\text{OMS}}$ for a paradigmatic material like bulk silicon. This is done in figure 12 in the zero temperature limit. The very different nature of the two lifetimes appear clearly. By simple energy conservation arguments, the electronic linewidth are zero by definition in the two energy regions $\epsilon_{\text{VBM}} - E_{\text{g}}$ and $\epsilon_{\text{CBM}} + E_{\text{g}}$, with E_{g} the electronic gap (in silicon $E_{\text{g}} \approx 1.1\,\text{eV}$) and ϵ_{CBM} (ϵ_{VBM}) the conduction band minimum (valence band maximum). In these energy regions the e–p contribution is stronger and the corresponding linewidths are larger then the e–e ones. The quadratic energy dependence of the e–e linewidths inverts this trend at higher energies.

While the e–e contributions grow quadratic in the energy dependence, the e–p ones follow the electronic density of states profile. This property is confirmed by figure 13(b) (see also [117]) and remains accurate when the temperature increases. Figure 13(a) shows instead the EP correction in single-layer MoS_2 of the valence and conduction band states for several temperatures, together with the widths and the density of states (DOS). In general, the EP correction tends to close the bandgap. This is visible in figure 13 (panels (a) and (b)), the conduction

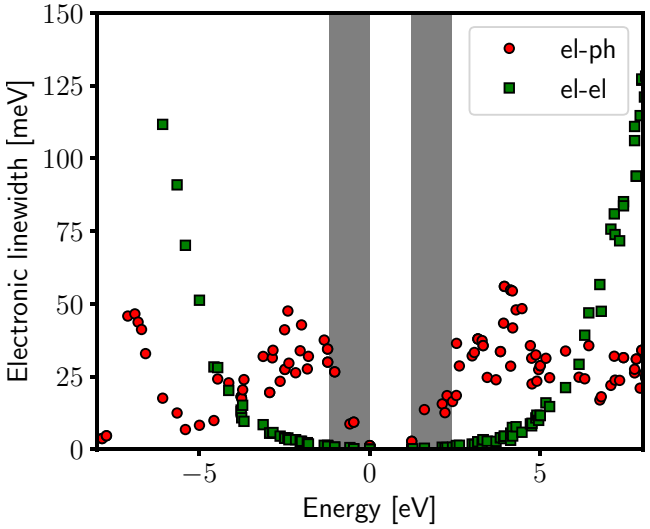


Figure 12. Quasiparticle linewidths of bulk silicon calculated by using the GW approximation for the e–e scattering (green boxes) and the Fan approximation for the e–p scattering (red circles). The two gray areas denote the energy regions $\epsilon_{\mathrm{VBM}} - E_{\mathrm{g}}$ and $\epsilon_{\mathrm{CBM}} + E_{\mathrm{g}}$. Adapted from [116]. © IOP Publishing Ltd. All rights reserved.

state energy decreases with temperature, while the valence one increases. Only in a few cases do we find an opening of the bandgap when temperature increases [118].

6.3. Finite temperature Bethe–Salpeter equation

Once the temperature-dependent corrections to electron and hole states have been calculated, they constitute the key ingredients of the finite temperature excitonic eigenvalue equation. Since the electron and hole eigenvalues are complex numbers the resulting excitonic eigenvalues have a real part (the exciton binding energy) and an imaginary part (the exciton lifetime). The dielectric function then depends explicitly on T, $\epsilon_2(\omega, T) = -(8\pi/V)\sum_\lambda |S_\lambda(T)|^2 \Im[\omega - E_\lambda(T)]^{-1}$, where $S_\lambda(T)$ are the excitonic optical strengths and $E_\lambda(T)$ are the complex excitonic energies. As shown in figure 14 for a single layer MoS_2, the main effect of the temperature on the optical spectra is the renormalization of the energy transitions along with a broadening of the spectrum lineshape related to the finite lifetime of the underlying excitonic states which increases with T [119, 120]. This picture is also valid when $T \to 0$ because of the zero-point vibrations. A remarkable effect of the exciton-phonon coupling has been observed in hexagonal BN. It has been proven that the optical brightness turns out to be strongly temperature-dependent such as to induce bright to dark (and viceversa) transitions [121].

6.4. Double grid in the electron–phonon coupling: a way to deal with the $\mathbf{q} \to \mathbf{0}$ divergence

A technically relevant issue is the slowing down of energy correction convergence at some high-symmetry points. Some EP matrix elements might be zero by symmetry and are not representative of the discretization of an integral. `yambo` deals with this issue by computing the energy shift corrections

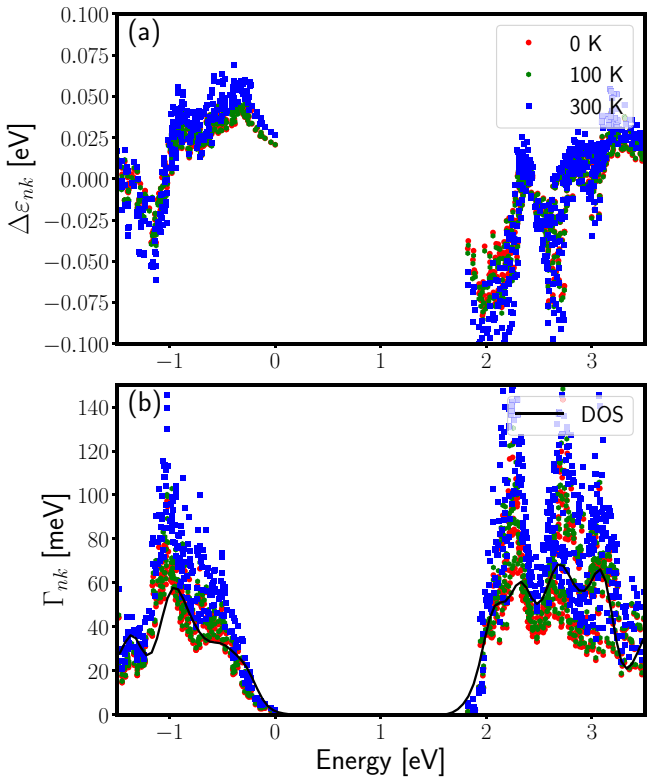


Figure 13. Single-layer MoS_2: (a) Electron–phonon correction of the eigenvalues $\varepsilon_{n\mathbf{k}}$ for several temperatures. (b) Electron–phonon widths of several temperatures and electronic density of states (black line).

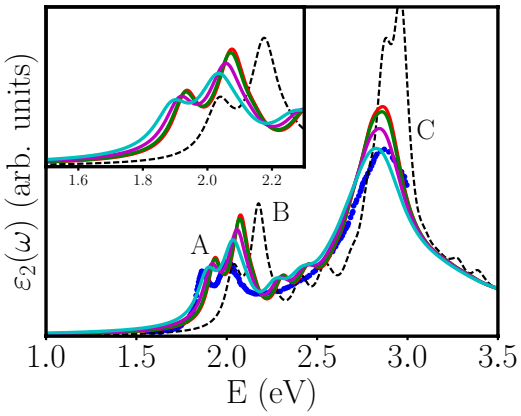


Figure 14. Optical spectra of single-layer MoS_2 for temperatures of 0 (red), 100 (green), 200 (magenta) and 300 K (cyan). Spectrum without electron–phonon effects is also shown (dashed line). Blue dots represent experimental data from [122]. Reprinted figure with permission from [120], © 2016 by the American Physical Society.

on a random $\mathbf{q}$-wavevector grid of transferred momenta. The numerical evaluation of the EP self-energy on a dense $\mathbf{q}$-grid is a formidable task (see equation (5) of [110]). The reason is that such dense grids of transferred momenta are inevitably connected with the use of equally dense grids of $\mathbf{k}$-points. The solution implemented in `yambo` is a double grid approach: matrix elements are calculated for a fixed $\mathbf{k}$-point grid while energies are integrated using a larger grid of random-points in the whole BZ.

To speed-up the convergence with the number of random points, the BZ is divided in small spherical regions $R_{\mathbf{q}}$ centered around each $\mathbf{q}$ point of the regular grid and the integral is calculated using a numerical Monte-Carlo integration technique. Furthermore, the divergence at $\mathbf{q} \to 0$ of the $|g^{\mathbf{q}\lambda}_{n'n\mathbf{k}}|^2$ matrix elements is explicitly taken into account for the 3D case for which the $\mathbf{q}$ integration compensates the $\mathbf{q}^{-2}$ divergence. In the case of 2D materials the divergence of the EP matrix elements would not be lifted by the surface element $2\pi q$. In principle, an analytic functional form for the EP matrix elements can be envisaged as reported by [123].

7. Real time propagation

A new feature in `yambo` is the numerical integration of a time-dependent (TD) equation of motion (EOM), able to describe the evolution of the electronic system under the action of an external laser pulse. Similarly to the equilibrium case, the most diffuse *ab initio* approaches to real-time propagation are based on TD-DFT and there exist a number of GPL codes available to this end [124, 125]. On the contrary the implementation of real time propagation within MBPT is an almost unique feature of the `yambo` code. Two different schemes are available. In one case, the density matrix of the system, $\rho(\mathbf{r}, \mathbf{r}', t)$, is propagated in time, as described in section 7.1. In the second case, the valence bands $u_{n\mathbf{k}}(\mathbf{r}, t)$ are propagated by means of a time dependent Schrodinger equation, as described in section 7.2.

Standard TD-DFT codes often (but not always) implement real time propagation in real space or reciprocal space basis-set. Instead for the two schemes above mentioned, the EOMs in `yambo` are represented in the space of the equilibrium KS wave-functions. Since a direct implementation of MBPT in real space (or in reciprocal space) is very cumbersome, the KS space offers a convenient alternative. The comparison between real-space versus KS-space has been extensively discussed in the literature TD-DFT where both approaches are feasible. Despite strict converge against the number KS states can be hard, very good results are obtained already with very few basis functions. The philosophy is similar to the one used to compute equilibrium QP corrections and BSE spectra, where both the self-energy and the excitonic matrix are written in KS space.

7.1. Time-dependent screened exchange

The EOM for the density matrix projected in the space of the single particle wave-functions, $\underline{\rho}$, is derived from non-equilibrium (NEQ) many-body perturbation theory and reads

$$\mathrm{i}\hbar\partial_t\, \underline{\rho}_{\mathbf{k}}(t) = \left[\underline{h}^{\mathrm{rt}}_{\mathbf{k}}[\underline{\rho}] + \underline{U}^{\mathrm{ext}}_{\mathbf{k}}(t), \underline{\rho}_{\mathbf{k}}(t)\right] - \mathrm{i}\underline{\Gamma}_{\mathbf{k}}\underline{\rho}_{\mathbf{k}}. \qquad (16)$$

Here we underline quantities which are vectors in the transition space (and we will underline twice matrices in transition space). h^{rt} contains the equilibrium eigenvalues $\epsilon_{n\mathbf{k}}$ plus the variation of the self-energy $\Delta\underline{\Sigma}^{\mathrm{Hxc}}[\underline{\rho}]$; $\epsilon_{n\mathbf{k}}$ can be the KS energies or the QP corrected energies. QP corrections can be loaded either from a previous calculation or by adding a scissor operator from input. For $\Delta\underline{\Sigma}^{\mathrm{Hxc}}[\underline{\rho}]$ different levels of approximation can be chosen, setting the `HXC_kind` input variable. $\underline{U}^{\mathrm{ext}} = -e\boldsymbol{\mathcal{E}}\cdot\underline{\mathbf{r}}$ represents the external potential written in the length gauge; shape, polarization, intensity (and eventually frequency) of the field $\boldsymbol{\mathcal{E}}$ can be selected in input. $\underline{\mathbf{r}}$ is the position operator. The coupling to the external field is exact up to first order. From the knowledge of the density matrix, the first order polarization $\mathbf{P}(t) = -e\sum_{i\neq j\mathbf{k}}\mathbf{r}_{ij\mathbf{k}}\rho_{ij\mathbf{k}}$ is computed at each time step. The spectrum of the system can then be obtained by the Fourier transform of the polarization, which can be done as a post-processing step. Absorption is thus obtained via the dipole–dipole response function (equivalent to the length gauge in linear response). A delta-like external field is convenient to obtain the spectrum for all frequencies.

The implementation of the external field in the velocity gauge (equivalent to the velocity gauge in linear response) is currently under testing before its final release. Equation (16) represents a set of equations, one for each $\mathbf{k}$-point in the BZ, coupled via the functional dependency of $\Delta\underline{\Sigma}^{\mathrm{Hxc}}$ on the whole $\underline{\rho}$. Different options of the self-energy are available, by setting the `HXC_kind` variable to: `IP`, `Hartree`, `DFT`, `Fock`, `Hartree + Fock`, `SEX`, or `Hartree + SEX`. For `HXC_kind = IP` one has $\Delta\underline{\Sigma}^{\mathrm{Hxc}} = 0$. For local `HXC_kind` like `Hartree` and `DFT`, $\Delta\underline{\Sigma}^{\mathrm{Hxc}}$ can be computed on the fly from the real-space density $n(\mathbf{r}, t)$ and the approach is in practice equivalent to TD-DFT to linear order in the field. For non-local `HXC_kind` like `Hartree + Fock` (HF) and `Hartree + SEX` (HSEX), the self-energy is written in the form $\Delta\underline{\Sigma}^{\mathrm{Hxc}}[\underline{\rho}] = \underline{\underline{K}}^{\mathrm{Hxc}}\cdot\underline{\rho}$ with $\underline{\underline{K}}^{\mathrm{Hxc}}$ computed before starting the real-time propagation. The calculation of $\underline{\underline{K}}^{\mathrm{Hxc}}$ can be either done in a preliminary run, with the matrix-elements stored on disk and then reloaded, or on-the-fly before starting the real-time propagation. In case the HSEX approximation is used the resulting spectrum is equivalent to a BSE calculation in the limit of small perturbations, as shown both analytically and numerically in [126]. Thus, to linear order, TD-SEX is is able to properly capture excitons, which can be hardly described within TD-DFT. The comparison between the two approaches is reported in figure 15.

When local self-energies are computed directly from the real space density, the numerical cost is mainly due to the projection of the potential on the KS-basis set at each iteration. This step is avoided in real-space TD-DFT where, however, the wave-functions on the real-space grid need to be propagated. The relative computation cost of the two strategies depends critically on the size of the real-space grid versus the number of KS function used. Instead when non local self-energy are used, the computational cost is mostly due to the preliminary calculation of the kernel $\underline{\underline{K}}^{\mathrm{Hxc}}$. This step has, roughly, the same computational cost of a standard BSE run and requires to store $\underline{\underline{K}}^{\mathrm{Hxc}}$ in memory (disk or RAM). The subsequent real-time propagation is instead very fast. In some cases it is convenient to use this scheme also for local self-energies. However, regardless of the self-energy used, only variations of the self-energy which are linear in the density matrix are described when using $\underline{\underline{K}}^{\mathrm{Hxc}}$.

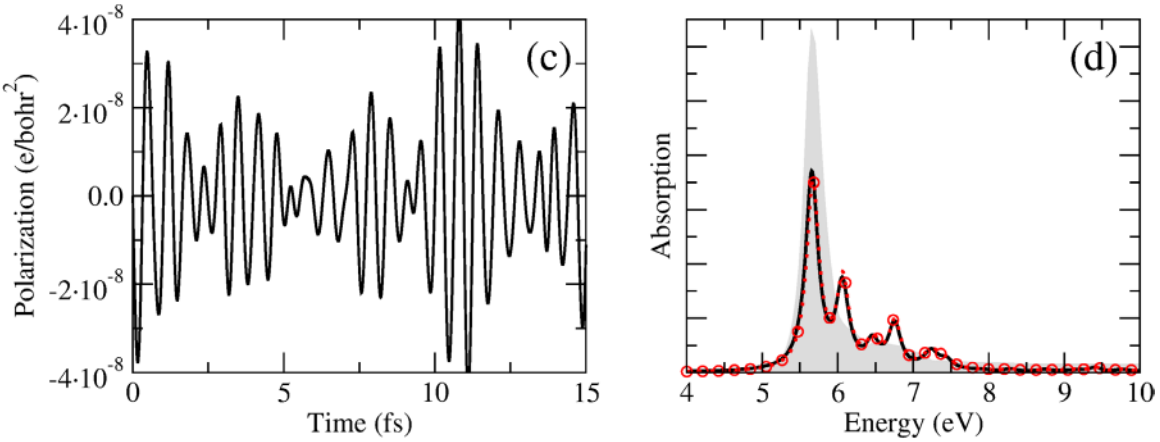


Figure 15. Time dependent polarization in hBN, panel (a), obtained obtained solving equation (16) within the HSEX approximation. In panel (b) its Fourier transform, red circles, matches the absorption computed within BSE, black line. Reprinted figure with permission from [126], © 2011 American Physical Society.

To run simulations and compute the spectra as described in the present section the `yambo_rt` and `ypp_rt` executables need to be used.

7.1.1. Double-grid in real time. As for the BSE case (see section 5.1.1), also the real-time propagation can be done taking advantage of a double-grid in **k**-space. Similarly to the BSE, the matrix elements, i.e. the dipoles and $\underline{\underline{K}}^{\mathrm{Hxc}}$, are computed using the wave-functions on the coarse grid, while energies and occupations are defined on the fine grid. At variance with the BSE implementation however the matrix elements on the coarse grid are then extrapolated onto the fine grid with a nearest-neighbour technique since ρ is then defined and propagated on the double grid. This is different in spirit from the inversion solver. It would be equivalent to define the excitonic matrix (or L propagator) on the double grid. Instead, in the double-grid approach within BSE the excitonic matrix remains defined on the coarse grid, while the fine grid enters only in the definition of L^0, as described in section 5.1.1.

7.2. Nonlinear optics

Alternative to the time-evolution of the density matrix, it is possible to perform the time-evolution of the Schrödinger equation for the periodic part of the Bloch states projected in the eigenstates of the equilibrium Hamiltonian: $|v_{m\mathbf{k}}\rangle$. Here we briefly present the actual implementation in `yambo` and how it can be used to obtain non-linear optics response, for more details see [129]. The EOM for the valence band states reads:

$$\mathrm{i}\hbar\partial_t |v_{m\mathbf{k}}\rangle = \left(h^{\mathrm{rt}}_{\mathbf{k}}[\rho] + \mathrm{i}\boldsymbol{\mathcal{E}}\cdot\tilde{\partial}_{\mathbf{k}}\right)|v_{m\mathbf{k}}\rangle \qquad (17)$$

where the effective Hamiltonian $h^{\mathrm{rt}}_{\mathbf{k}}$ has been introduced in section 7.1 and $\rho(t)$ is constructed starting from $|v_{m\mathbf{k}}\rangle$. The second term in equation (17), $\boldsymbol{\mathcal{E}}\cdot\tilde{\partial}_{\mathbf{k}}$, describes the coupling with the external field $\boldsymbol{\mathcal{E}}$ in the dipole approximation. As we imposed Born–von Kármán periodic boundary conditions, the coupling takes the form of a **k**-derivative operator $\tilde{\partial}_{\mathbf{k}}$. The tilde indicates that the operator is 'gauge covariant' and guarantees that the solutions of equation (17) are invariant under unitary rotations among occupied states at **k** (see [130] for more details).

Propagating the single particle wave-functions presents advantages and disadvantages with respect to the density matrix. The major advantage is that the coupling of electrons with the external field, within the length gauge, is now written in terms of Berry's phase, which is exact to all orders also in extended systems [131]. Moreover, from the evolution of $|v_{m\mathbf{k}}\rangle$ in equation (17) also the time-dependent polarization [132] $\mathbf{P}_{\parallel}$ along the lattice vector **a** can be computed in terms of the Berry phase:

$$\mathbf{P}_{\parallel} = -\frac{ef|\mathbf{a}|}{2\pi\Omega_c}\mathrm{Im}\log\prod_{\mathbf{k}}^{N_{\mathbf{k}}-1}\det S\left(\mathbf{k},\mathbf{k}+\mathbf{q}\right), \qquad (18)$$

where $S(\mathbf{k},\mathbf{k}+\mathbf{q}) = \langle v_{n\mathbf{k}}|v_{m\mathbf{k}+\mathbf{q}}\rangle$ is the overlap matrix between the valence states, Ω_c is the unit cell volume, f is the spin degeneracy, $N_{\mathbf{k}}$ is the number of **k** points along the polarization direction, and $\mathbf{q} = 2\pi/(N_{\mathbf{k}}\mathbf{a})$. The resulting polarization can be expanded in a power series of the field $\boldsymbol{\mathcal{E}}_j$ as:

$$\mathbf{P}_i = \chi^{(1)}_{ij}\boldsymbol{\mathcal{E}}_j + \chi^{(2)}_{ijk}\boldsymbol{\mathcal{E}}_j\boldsymbol{\mathcal{E}}_k + \chi^{(3)}_{ijkl}\boldsymbol{\mathcal{E}}_j\boldsymbol{\mathcal{E}}_k\boldsymbol{\mathcal{E}}_l + O(\boldsymbol{\mathcal{E}}^4)\,, \qquad (19)$$

where the coefficients $\chi^{(i)}$ are functions of the frequency of the perturbing fields and of the outgoing polarization. From the Fourier analysis of the $\mathbf{P}_i$ it is possible to extract all the non-linear coefficients (see [129] for more details). As in section 7.1, the level of approximation of the so-calculated susceptibilities depends on the effective Hamiltonian that appears in the right hand side of equation (17). Different choices are possible, namely, the independent particle approximation (IPA), the time-dependent Hartree, the real-time Bethe–Salpeter equation (RT-BSE) framework, or TD-DFT. This approach has been successfully applied to study second-, third- harmonic generation and two-photon absorption in bulk materials and nanostructures [129, 133, 134]. As before, in the limit of small perturbation equation (17) reproduces the optical absorption calculated with the standard GW + BSE approach [135].

Since the exact polarization is available, the approach based on equation (17) not only reduces to TD-DFT when

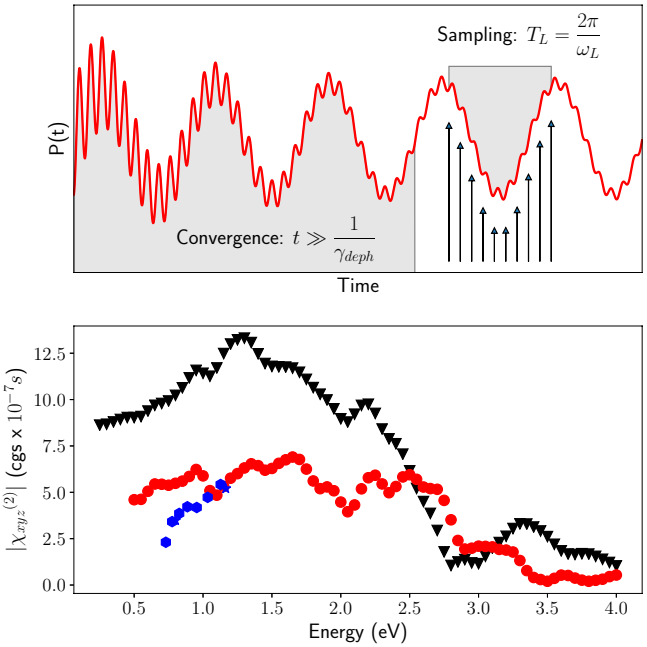


Figure 16. Top panel: schematic representation of real-time simulation for the non-linear response. Bottom panel: magnitude of $\chi^{(2)}(-2\omega, \omega, \omega)$ for bulk CdTe calculated within the QPA (black triangles) and TDH (red circles). Each point corresponds to a real-time simulation at the given laser frequency. Comparison is made with experimental results from [127, 128] (blue stars and hexagons). Reprinted figure with permission from [129], © 2013 American Physical Society.

local functionals are considered, it also includes TD density polarization functional theory (DPFT) as a special case. Thus specific approximations for both the microscopic and the macroscopic part of $\Delta\Sigma^{\mathrm{Hxc}}$ are available, which, within TD-DPFT, are expressed as functionals of ρ for the microscopic part ($v^{\mathrm{Hxc}}[\rho]$) and of $\mathbf{P}$ for the macroscopic part ($\boldsymbol{\mathcal{E}}^{\mathrm{Hxc}}$) as discussed in [136, 137].

A comparison between TD-DPFT in the real-time framework and the solution of the Bethe–Salpeter equation for different zinc-blende compounds has been recently published by Riefer *et al* [138]. While for linear response the different functionals give a satisfactory result [137], for the second harmonic generation the situation is less clear. This is probably due to the fact all exchange-correlation kernels implemented in `yambo` and tested in the previous papers were derived in the linear response regime.

In non-linear optics simulations the system is excited with a laser at given frequency ω and dephasing term λ_{deph} is added to the Hamiltonian. After a time $T \gg \lambda_{\mathrm{deph}}$, sufficient to damp out the eigenmodes of the system, the signal is analyzed to extract the non-linear response functions, see figure 16 and [129]. To run simulations and compute the spectra as described in the present section the `yambo_nl` and `ypp_nl` executables need to be used.

8. Parallelism and performance

During the last years, the evolution of supercomputing technologies pushed towards the adoption of architectural solutions

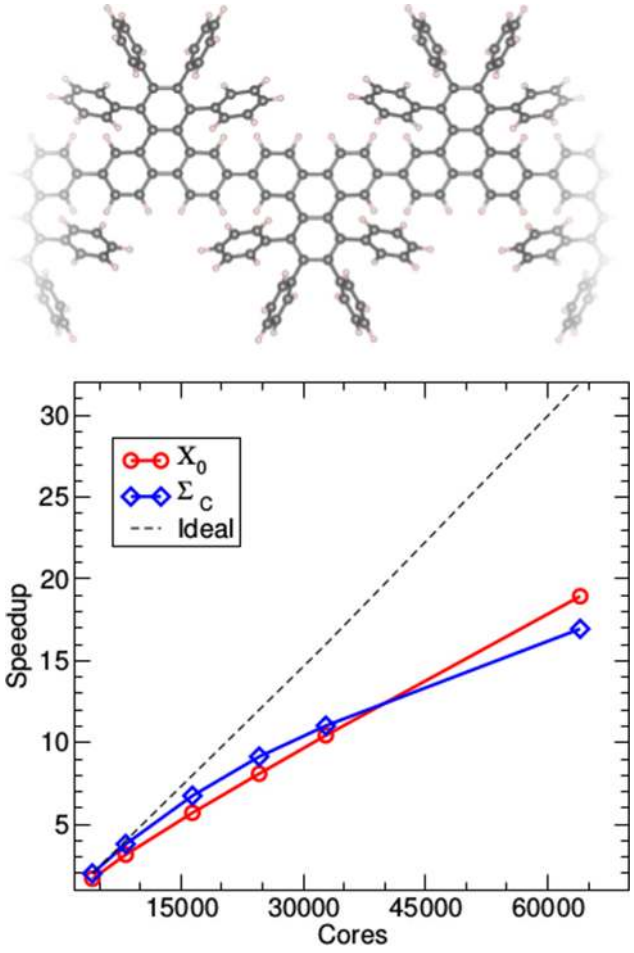


Figure 17. `yambo` parallel performance. Upper panel: chemical structure of the precursor polymer of a chevron-like Graphene nanoribbon. Lower panel: `yambo` speedup of the linear response (χ_0) and self-energy (Σ_c) kernels during a GW run. The scaling (obtained using 16 MPI tasks per node and four OpenMP threads/task) is shown up to 1000 Intel KNL nodes on Marconi at Cineca, A2 partition, corresponding to a computational partition of about three PetaFlops. The dashed line indicates the ideal scaling slope. Adapted from [139] with permission of The Royal Society of Chemistry.

based on many-core platforms. This was due mainly to energetic constraints that did not permit to increase the single core performance, imposing the need for alternative solutions. Two main paradigms arose: on one side, the emergence of hybrid architectures exploiting GPU accelerators. On the other side, homogeneous architectures increased the performance per node, by increasing the number of cores, starting the many-core era. In the latter approach, the main advantage is the possibility to rely on well-known and largely adopted software paradigms, in contrast to the GPU programming model, where the porting required to adopt ad-hoc languages such CUDA or OpenCL, having a deep impact on the sustainability of the software development. However, even if the many-core paradigm can appear easier to adopt, getting a satisfactory performance on such architectures may be very challenging. In fact, in order to exploit as much as possible the features of a many-core node, it is mandatory to use both a shared memory and a distributed memory approach. The first is able to leverage the single node power with an efficient usage of the available memory, while

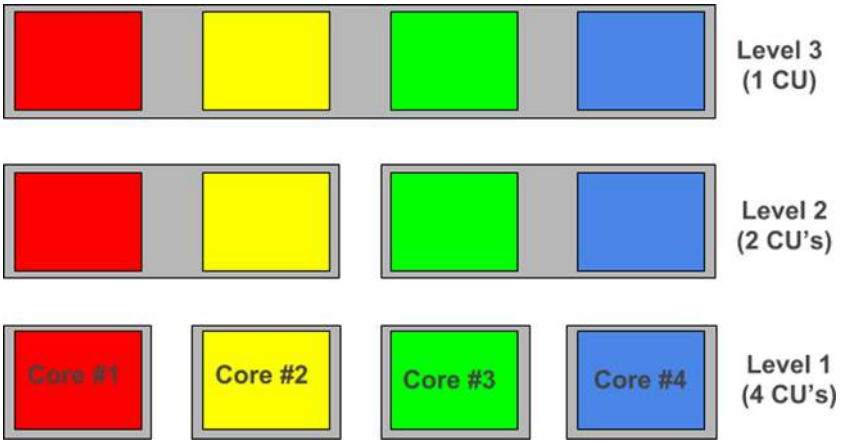


Figure 18. `yambo` parallel structure in the specific case of four cores. Each core is member, at the same time, of three different groups composed of different number of cores. These groups, represented with gray boxes, are the actual computing units of `yambo`. Therefore each core workload is dictated by the computing units directives and changes depending on the group the core belongs to.

the second one can be used to scale-up on the nodes available on a cluster facility or a multi-purpose processor.

From `yambo` version 4.0, a deep refactoring of the parallel structure has been put in place in order to take full advantage of nodes with many-cores and a limited amount of memory per core. In particular, a MPI multi-level (up to 3–5 according to the runlevel) approach has been adopted, together with an OpenMP coarse grain implementation. An example of the measured parallel performance reaching up to the use of 1000 Intel KNL nodes in a single run is shown in figure 17. We refer to the performance page [140] on the `yambo` website for a more complete description and up-to-date data.

This novel multi level distribution of the cores is schematically shown in figure 18 in the case of four cores. Instead of using the core as elemental parallel unit, `yambo` adopts the concept of computing units (CU). A CU is composed of a varying number of cores. The work-load distribution is done among CU's rather than cores. Each core workload is decided by the workload of the CU that encloses it. To be more clear let us take the simple case of four cores shown in figure 18. In this case we have three possible levels of grouping with respectively 4, 2 and 1 CU's. The core workload is assigned to the CU's rather that to the single core. This reduced enormously the inter-core communications and allows the distribution of a very large number of cores. Technical details of the implemented parallelism will be discussed in the next sections.

Finally we also mention that work is in progress to port yambo on GPUs, using CUDA Fortran as a first step. We are currently porting few low-level routines on GPUs taking advantage of CUDA libraries: notably the ones computing the matrix elements of equation (2). This allows to have a preliminary porting on GPUs of dipoles, Hartree–Fock, linear response, GW, and BSE kernels. Based on the results obtained we will decide subsequent strategies.

8.1. General structure

The multilevel MPI structure of `yambo` is reflected in the input file where, for each computational kernel (`runlevel` here) there are two related input variables: the first one, `runlevel_ROLEs`, sets on which parameters the user wants to distribute the MPI workload, while the second `runlevel_CPU` defines how many MPI tasks will be associated to such parameters. As an example

```
X_finite_q_ROLEs="q.k.c.v.g"
X_finite_q_CPU="2.3.5.2.1"
```

is a possible input for running on $2 \times 3 \times 5 \times 2 \times 1 = 60$ MPI tasks, with the **q**-points distributed on two tasks, the **k**-points on three, the conduction and valence bands on five and two respectively. One more level of parallelism (g) is present, acting and distributing the response matrix over space degrees of freedom (plane waves). The order of the parameters in the `runlevel_ROLEs` variables is irrelevant. On top of that, more input variables are available to handle parallel linear algebra (e.g. via scalapack and blacs libs) and to select the number of `OpenMP` threads (on a runlevel basis if needed). Such hierarchical organization makes it possible to have MPI communication only within the subgroups, thus avoiding, whenever possible, to deal with `all2all` communications.

If the user does not wish to deal with the complexity of such multi-level parallelization a default layout is provided. However, the fine-tuning of the MPI/OpenMP related variables can (further) reduce the load imbalance, improve the memory distribution or decrease the total time-to-solution. For this reason, in the sections 8.3–8.6 specific suggestions for best parallel exploitation of each runlevel are provided.

8.2. I/O: parallel and serial

`yambo` stores binary data using the `netcdf` library. Depending on the configuration flags, data can be stored in classic `netcdf` format (file size limit of 2 GB, activated with `–enable-netcdf-classic`), 64-bit `netcdf` format (no file size limit, default) or HDF5 format (requires at least `netcdf` v4 linked to HDF5, activated with `–enable-netcdf-hdf5`). Since version 4.4, in case the HDF5 format is specified, parallel I/O can also be activated

(`–enable-hdf5-par-io`) to store the response function in **G** space or the kernel of the BSE. For the **G**-space response function, parallel I/O avoids extra communication among the MPI tasks and also reduces the amount of memory allocated per task. For the BSE case, parallel I/O makes it possible to load the kernel computed from a previous calculation using a different parallel scheme and/or a different number of MPI tasks. Indeed the calculation of the kernel matrix elements is very time consuming but has a very efficient memory and load distribution. In contrast, the solution of the BSE eigenproblem is less time consuming but also less efficiently distributed. It is thus suggested to first compute the kernel matrix on a large number of cores and then to solve the BSE on fewer tasks as a follow-up step.

8.3. Linear response

According to equation (1), the computation of the response function in G-space can be distributed over five different levels: **q**-points, **k**-points, conduction and valence bands (c, v), and **G**-vectors. The distribution over the **q**-points would be the most natural choice, since the response functions at different **q** are completely independent. However, it may lead to significant memory duplication (multiple sets of wavefunctions are managed at the same time) and load imbalance since the number of possible transitions varies from point to point. It is usually not recommended unless a large number of **q**-points has to be considered. Instead, it is usually more effective to parallelize over **k**, and bands (c,v) indexes. This requires slightly more MPI communication (due to a MPI reduction at the end of the calculation) but is very efficient in terms of speedup (almost linear) and in terms of memory distribution (especially for c, v, since usually wavefunctions are the leading memory contribution). While transitions are evenly distributed (balanced workload), `yambo` sorts and groups transitions which are (almost) degenerate in energy (see appendix B) thus, in practice, small imbalances can still be present.

Further, the distribution over the **G**-vectors is the most internal one, and requires more communication among MPI tasks (unless parallel I/O is activated). It can be useful for systems with a very large number of **G**-vectors (such as low dimensional systems or surfaces) to distribute the response function and ease memory usage. Finally, the computation of χ^0 can also benefit from `OpenMP` parallelism. The distribution over threads has been implemented at the same level of the MPI parallelism (i.e. over transitions), resulting in a very good scaling while reducing the memory usage per core. Note however that some memory duplication (a $M(\mathbf{G}, \mathbf{G}')$ workspace matrix per thread) has to be paid to make the implementation more efficient. The `OpenMP` parallelism of χ^0 (including dipoles) is governed by the input variables:

```
X_Threads = 8
DIP_Threads = 8
```

both defaults being set to 0, i.e. controlled as usual by the `OMP_NUM_THREADS` environment variable. Once the independent-particle response function $\chi^0_{\mathbf{GG}'}(\mathbf{q}, \omega)$ has been computed, a Dyson equation is solved for each frequency to construct the RPA response function. This can be done either by distributing over different frequencies or by using parallel linear algebra (see section 8.6).

8.4. Self-energy: HF-exchange and GW

Following equation (7), the HF and GW correlation self-energies can be parallelized with MPI over three different layers: **q**-points (`q`); bands in the Green's function (`b`) [see m in equation (15)]; and quasiparticle corrections $\Sigma_{nn'\mathbf{k}}$ to be computed (`qp`). `OpenMP` parallelism here acts at the lowest level, dealing with sums over **G** and **G**′, i.e. spatial degrees of freedom. The following variables can be modified to fine-tune the self-energy parallelization (here shown for 60 MPI tasks and eight OpenMP threads):

```
SE_ROLEs="q.qp.b"
SE_CPU="1.4.15"
SE_Threads = 8
```

Since the sum over **q**-points in equation (7) is over the whole BZ, the **q**-parallelism for the self-energy may be even more unbalanced than that for the response function (here every **q**-point needs to be expanded to account by symmetry for its whole star) and is recommended only when a large number of **q**-points is available. Instead, the parallelism over bands b tends to distribute evenly both memory and computation, at the price of a mild MPI communication, thereby resulting a natural choice (when enough bands are included in the calculation). *qp*-parallelism distributes the computation but tends to replicate memory (wavefunctions are not further distributed). In general, the `OpenMP` parallelism is extremely efficient for the GW self-energy without having to pay for any extra memory workspace.

8.5. Bethe Salpeter equation

In the solution of the BSE most of the CPU time is spent in building up the excitonic matrix, or more precisely, its kernel. The input flags which control the parallel distribution of the workload needed to build the kernel are `eh.k.t`. To distribute the workload, first all possible transitions $cv\mathbf{k}$, i.e. from valence band v to conduction band c at the k-point **k**, are split into transition groups (TGs). Then for each pair of TGs a block of the BSE matrix is created $B_{ij} = \{T_i \rightarrow T_j\}$. Defined N_t the total number of TG, then the BSE matrix will be divided into $N_b = N_t^2$ blocks. In the Hermitian case (as in the Tamm–Dancoff approximation), only $N_b = N_t(N_t + 1)/2$ blocks will be computed. The parallelization flags for the BSE define both N_t and N_b, and how the resulting blocks are distributed among the MPI tasks. Indeed $N_t = n_{\text{eh}} n_k^{\text{ibz}}$ where n_{eh} is the number of MPI tasks assigned to the `eh` field and n_k^{ibz} is the number of **k**-points in the IBZ. This means that even setting `eh` $= 1$ a minimum number of **k**-based TGs (k-TGs) is always created, which is eventually split into subgroups when $n_{\text{eh}} > 1$. It is important to note that k-TGs are defined using

the k-sampling in the the IBZ, while the BSE matrix is defined in the whole BZ, resulting in groups of non-uniform size. However, the symmetry operations relating matrix elements within a given k-TGs are taken into account by yambo. As a consequence, in systems where $n_k^{\rm ibz} \neq n_k^{\rm bz}$ the use of $n_{\rm eh} > 1$ is discouraged, as the splitting of k-TGs over different MPI-tasks implies that symmetry-related matrix-elements can be assigned to different MPI-tasks and need to be recomputed. Once N_t and hence N_b are defined, transitions and blocks are distributed among the MPI tasks as explained in the following example.

Suppose we have a system with 18 **k**-points in the IBZ, and we adopt the parallelization strategy 2.3.3 for eh.k.t in the case the BSE is Hermitian. Then $N_t = 2 \times 18 = 36$ and $N_b = 666$. Thus, in our example we are using in total $2 \times 3 \times 3$ MPI-tasks. The eh.k fields define $2 \times 3 = 6$ MPI-groups which split the 36 transition-groups. Thus, each MPI-group has to deal with six transition-groups. For each transition group T_n, there are N_t blocks $B^n_{ij} = T_i \to T_j$ for the Hermitian case, where the T_n appears as initial ($T_i = T_n$) or final ($T_j = T_n$) state. Most of the blocks belongs to two transition-groups (only the blocks B_{ii} belong to one transition-group). This means that each MPI-group builds half of the B_{ij} ($6 \times 35/2$) plus all B_{ii} (6) blocks. These 111 blocks are divided according to the t field and thus each MPI-task will be assigned to 37 blocks.

8.6. Linear algebra

Dense linear algebra is extensively used in yambo. Among the most time-consuming tasks we have identified the inclusion of local field effects [1] in the RPA response function

$$\chi^{RPA}_{\mathbf{GG'}}(\mathbf{q},\omega) = \chi^{0}_{\mathbf{GG'}}(\mathbf{q},\omega) + \chi^{0}_{\mathbf{G\overline{G}}}(\mathbf{q},\omega)\frac{4\pi}{|\mathbf{q}+\overline{\mathbf{G}}|^2}\chi^{RPA}_{\mathbf{\overline{G}G'}}(\mathbf{q},\omega). \quad (20)$$

The solution of equation (20) can be cast in the form of a matrix inversion. Indeed:

$$\chi^{RPA}_{\mathbf{GG'}}(\mathbf{q},\omega) = \left[\delta_{\mathbf{G},\overline{\mathbf{G}}} - \chi^{0}_{\mathbf{G\overline{G}}}(\mathbf{q},\omega)\frac{4\pi}{|\mathbf{q}+\overline{\mathbf{G}}|^2}\right]^{-1} \chi^{0}_{\mathbf{\overline{G}G'}}(\mathbf{q},\omega). \quad (21)$$

Equation (21), and the solution of the BSE (diagonalization), which can be considered prototype kernels.

Once a finite basis set is adopted, the operators involved are represented as (dense) $N \times N$ matrices, with N easily reaching few-to-tens of thousands or more, making standard linear algebra tasks (such as matrix multiplication, inversion, diagonalization) quite intense. We have therefore implemented dense parallel linear algebra by exploiting the ScaLAPACK library [88] within the MPI parallel structure of yambo. Concerning the RPA response, this means that on top of the MPI parallelism over **q**-vectors, multiple instances of parallel linear algebra are run at the same time (one per **q** vector) to compute $\chi^{\rm RPA}$.

The behavior of the yambo parallel linear algebra is governed by the variables:

```
runlevel_nCPU_LinAlg_INV = 64
runlevel_nCPU_LinAlg_DIAGO = 64
```

where (runlevel could be, for example, the RPA response function or the BSE). Given the relevance, the calculation of the IP response function $\chi^0_{\mathbf{GG'}}(\mathbf{q},\omega)$ has also been block-distributed over $\mathbf{G}, \mathbf{G'}$ vectors (g-parallelism in section 8.3), both in terms of computation and memory-usage.

When using the SLEPC diagonalization method to obtain the BSE spectra, the memory distribution of the eigensolver (not to be confused with the memory distribution of the BSE matrix discussed in section 5.1.2) is handled by the SLEPC library itself. For more details, the reader is referred to the SLEPC specific literature [86].

9. Scripting and automation

As a pure many-body code, yambo works as a sort of 'quantum engine' that takes as input DFT calculations and material-specific parameters, producing very large amounts of temporary data (e.g. the response function) and outputs numerical results. Even a single calculation can produce enormous amounts of data. It is therefore necessary to carefully select or extract the relevant information to be stored for future analysis or sharing, possibly ensuring reproducibility.

In addition, the final quantities of interest (e.g. GW band gaps or BSE spectra) are often the results of a complex and tedious sequence of operations, involving transferring data from different codes (e.g. from Quantum ESPRESSO to yambo) or repeating calculations with different parameters (e.g. for convergence tests). The benefit of having platforms to organize, simplify and accelerate many-body perturbation theory calculations is obvious. Two parallel efforts are being developed to facilitate the use of yambo, namely yambopy and the yambo interface with the AiiDA platform [141]. In figure 19 a schematic representation of how such platforms interact with yambo is shown.

9.1. Yambopy

Yambopy is a community project to develop Python classes and scripts to express, automate, and analyze calculations with the yambo code. A typical yambo workflow involves a few steps: generating the KS states with a DFT code, preparing the yambo databases and then running the yambo code. These workflows become more complicated when performing convergence tests or when repeating the same calculations for multiple materials. With yambopy the user can express these yambo workflows on a python script that can then be shared and reproduced among different users. We currently provide python classes to read and write the input files as well as the output of the yambo code. A lightweight python interface for the Quantum ESPRESSO Suite is also provided in the qepy package distributed along with yambopy. For more

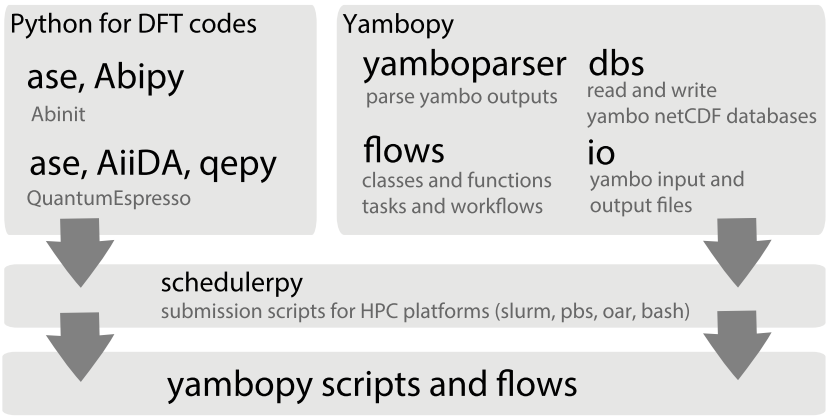


Figure 19. Structure of the `yambopy` package. The `qepy` and `schedulerpy` packages are distributed as part of `yambopy`.

comprehensive python interfaces for the DFT codes see ASE [142], Abipy [143] for the Abinit code, or `AiiDA` [141].

The `YamboIn` class provided by `yambopy` is used to read and write the base `yambo` input file generated by `yambo` and modify it in a programmatic way. With this tool it is possible, for example, to create a set of input files by changing a single variable inside a `for` loop. We also provide classes to read the `yambo` NetCDF databases in Python (for a complete list see the on-line documentation [144]). These classes provide methods to manipulate and represent the data using the matplotlib [145] library giving great flexibility for the interpretation and analysis of the results. Running these workflows on a HPC context requires to write job submission scripts for different job schedulers, this can be done using the `schedulerpy` package also accompanying `yambopy`.

For quick access to some features from the command line, we provide the `yambopy` shell command. The script is automatically installed with `yambopy` in such a way that some functionalities of `yambopy` are directly callable from the command line. This script has features to plot the convergence tests for GW and BSE calculations, excitonic wave functions, and dielectric functions among others.

To ensure software quality and usability we provide `yambopy` as an open-source code along with documentation and automatic testing. A detailed documentation of the classes, features, and a tutorial are available on the `yambopy` website [144]. We keep a public git repository hosted on `Github` where the users can download the latest version of the code as well as contribute with patches, features, and workflows. Sharing the workflows among users allow us to avoid repeated technical work and greatly simplifies the use of the `yambo` code. Continuous integration tests are done using the Travis-CI platform [146] leading the code to be tested at each commit and thus enforcing its reliability. `yambopy` is a project under development, and will be described on its own in a future work.

9.2. `yambo` *within the* `AiiDA` *platform*

The `AiiDA` platform is a materials' informatics infrastructure which implements the so-called ADES model (Automation, Data, Environment and Sharing) for computational science [141]. The `AiiDA` plugins and workflows for `yambo` are publicly available on `GitHub` [147], while online documentation and tutorials are available on Read the Docs [148].

9.2.1. The `yambo-AiiDA` *plugin.* Input parameters and scheduler settings are stored as code-agnostic `AiiDA` data-types in a database, then converted by the `yambo-AiiDA` plugin into `yambo` input files and transferred to a computational unit (e.g. a remote workstation or an HPC cluster). The `AiiDA` daemon submits, monitors and eventually retrieves the output files of the `yambo` calculation, the relevant information is then parsed and stored by the plugin. While the relevant data is properly stored in a suitable database, the raw input and output files are also stored locally in a repository. Therefore inputs, calculations and outputs are all stored as nodes of a database connected by directional links, preserving the full data provenance and ensuring reproducibility.

The `yambo-AiiDA` plugin currently supports calculations of quasi-particle corrections (e.g. at the COHSEX or GW level) and optical properties (e.g. IP-RPA). `Quantum ESPRESSO`, one of the main DFT codes interfaced with `yambo`, is also strongly supported with specific plugins and workflows for `AiiDA` [149]. Some of the parsing functionalities of the plugin are powered by the `yambopy` package [144]. Different types of calculations can be performed, either starting from `Quantum ESPRESSO` or from `p2y` or from a previous (possibly unfinished) `yambo` run.

9.2.2. `AiiDA` *workflows: automated GW.* The `yambo-AiiDA` package provides automated workflows that capture the knowledge of an experienced user in performing e.g. GW calculations within the plasmon-pole approximation, accepting minimal inputs such as a DFT calculation or a crystal structure, and returning as outputs a set of quasi-particle corrections. The `yambo-AiiDA` plugin repository hosts four `AiiDA` workflows [141] of increasing complexity and abstraction: `YamboRestart`, `YamboWf`, `YamboConvergence` and `YamboFullConvergence`, that perform different but mutually interdependent tasks, with the latter depending on the former in the listed order.

`YamboRestart` is a low level `AiiDA` workflow that takes a DFT calculation (or a prior `yambo` calculation) as

input, performs GW or BSE calculations, and returns the results. `YamboRestart` interacts directly with the `yambo` plugin, coping with common failures that may occur during a `yambo` GW run such as insufficient maximum wall-time and out-of-memory issues: the workflow adjusts the scheduler options as well as the parallelization choices accordingly and resubmits the calculations.

`YamboWf` is a higher-level `AiiDA` workflow that uses `YamboRestart` and the `Quantum ESPRESSO-AiiDA` plugin to manage end-to-end a GW calculation from the DFT step to the completion of the `yambo` run. In contrast to `YamboRestart`, which starts from an already existing calculation (either DFT or `yambo`), `YamboWf` does not need to start from any calculation and performs all steps, including all necessary DFT, data interfaces, and `yambo` calculations.

`YamboConvergence` is built on top of `YamboWf` and automates the convergence of QP corrections (by focusing on the quasiparticle gap) with respect to a single parameter. A one-dimensional line search in the parameter space is used. The convergence is determined by comparing a series of the most recent calculations (four of them are used by default), and ensuring the change between all four successive calculations is less than the convergence tolerance. The deviation from convergence is estimated by fitting the gap to a function of the form $f(x) = c + \frac{a}{x+b}$.

`YamboFullConvergence` iterates the above procedure over the main variables governing the convergence of GW calculations, namely the **k**-point grid, the number of **G**-vectors used to represent the response function (χ_0 cutoff), and the number of bands included in the sum-over-states for both the polarizability and the correlation self-energy. Additionally, the possibility to further reduce the FFT grids with respect to the one used at the DFT level is also considered. A beta version of this workflow has been made available on `GitHub` for testing and fine-tuning of the algorithm.

9.3. Test-suite and benchmark-suite

A new important tool introduced to improve and stabilize the development of the `yambo` code is the test-suite. The `yambo` test-suite is stored in a dedicated repository (yambo-tests) on `GitHub` and contains a series of tests which can be run in an automated manner. The repository is freely accessible after registering as a '`yambo` user' on the `GitHub`. While the test-suite is mainly aimed at developers, users can also benefit from accessing its input and reference files and automatically checking if their compiled version of `yambo` works properly.

The test-suite is governed by using a Perl script, `driver.pl`. This script uses internal Perl modules to perform several tasks: it automatically compiles the `yambo` code (a precompiled version can also be used), it runs the code and checks the output against reference files stored in the test-suite repository.

The code can be run in serial, parallel with `OpenMP` threads and checking parallel I/O and/or parallel linear algebra. At least two different groups of tests are available: smaller (and faster) tests which are run on a daily basis and longer tests which are used for a deeper testing of the code before a release.

The same driver can also be used to run `yambo` benchmarks. Benchmarks tests are a particular group of materials that, describing complex nano-structures (a 1D polymer or carbon-based ribbon) or a water cell, require a large number of reciprocal space vectors and/or **k**-points. As a consequence these systems are suitable to be executed using a large number of cores on parallel machines. In this case the test-suite can collect the results and loop on different parallel configurations testing their performances. More importantly the test-suite organizes the results in machine dependent folders that can be, eventually, post-analyzed.

The results of the night runs of the test-suite are publicly available on the web-page [150] and can be inspected without having access to the machines that run the tests. This is very useful in order for any development to reproduce a specific error to be fixes.

10. Conclusions and perspectives

This paper describes the main development lines of the `yambo` project since the 2009 reference paper [4]. `Yambo` is a scientific code supported and continuously developed by a collaborative team of researchers. The long list of authors of this work attests to the involvement of numerous experienced and young developers in addition to the four founders [4].

The `yambo` team currently comprises a balance of renowned scientists, with long-standing experience in *ab initio* approaches, and young researchers. We welcome students and post-docs with new ideas. This combination makes possible the growth of a software suite which is formally rigorous and able to address topics at the frontiers of materials science. By exploiting the power of many body perturbation theory at equilibrium and out-of-equilibrium within a state-of-the-art *ab initio* framework, the code is able to make predictions of the electronic and optical properties of novel materials, and moreover to provide interpretation of cutting-edge experiments ranging from ultrafast electron dynamics to nonlinear optics.

The involvement of parallel computing experts (two members of the Italian National Supercomputing Center CINECA co-authored this paper, for example) ensures that the code is also efficient and portable to the latest supercomputing architectures. As a result, new features added to the code immediately benefit from the native parallelized environment.

The modular structure of the code and the interface to external supporting software (`AiiDA` and `Yambopy`) complete the picture providing the end-user with a wealth of tools that cover the actual preparation, calculation and post-processing of data. The `yambo` suite thus provides all the ingredients for an advanced and computationally powerful approach to theoretical and computational material science.

Indeed, despite being born as a code for MBPT, thus tailored for sophisticated calculations on simple materials, `yambo` can nowadays be used to study complex materials and interfaces as well. This means in practice that, while the first versions of the code were designed to run on unit cells containing very few atoms (like bulk silicon), nowadays `yambo` can be easily used to study unit cells with 10–20 atoms and can be pushed

on HPC centers up to hundreds of atoms [151–153]. The number of atoms which can be dealt is, thus, approximately one order of magnitude less than advanced DFT codes. The exact limit is mainly imposed by the power of HPC facilities. We are also working on a dedicated section on the yambo web-site with detailed information on time scaling of different runlevels across the releases of the code.

What lies in yambo's future? We expect that the future development of yambo will be driven by the need to interpret new experiments. This will be achieved through the implementation of advanced computational algorithms and physical methodologies and will increasingly exploit interoperability with other software. Projects under current development include extension of GW to start from hybrid functionals, the possibility to use ultrasoft pseudopotentials, alternative schemes to avoid empty states, BSE at finite q, and incorporation of exciton-phonon coupling, to name just a few. These new developments will become available to general users in the near future. The code's efficiency will be continuously improved in order to tackle problems that remain computationally cumbersome. We expect that yambo will be further restructured in order to adapt to heterogeneous architectures (GPUs and accelerators) and to fully exploit the computational power of future pre- and 'exascale' machines. Further developments are (and hopefully will be) also driven by the participation in European initiatives and projects. At present yambo is part of a user-based European infrastructure [38] and a member of the suite of codes selected for the exascale transition [37].

In conclusion, yambo is a lively community project characterized by a continuous technical and methodological development. The substantial development between the 2009 reference paper [4] and today demonstrates its enormous potential. The aim is to provide the scientific community with a tool to perform cutting edge simulations in a computationally efficient environment.

Acknowledgments

The yambo team acknowledges financial support from a number of sources, notably including H2020 funding, as follows. This work was in part supported by the project MaX—MAterials at the eXascale—by the European Union H2020-EINFRA-2015-1 and H2020-INFRAEDI-2018-1 programs (Grant No. 676598 and No. 824143, respectively). This work was also supported by the European Union H2020-INFRAIA-2014-2015 initiative (Grant No. 654360, project NFFA—Nanoscience Foundries and Fine Analysis). We also would like to acknowledge support from cost action CA17126. We thank CECAM and Psi-K network for financial and practical support related to the organization of yambo training and tutorial events.

yambo developers have benefit from direct interactions with personnel from INTEL (Hans Pabst) and NVIDIA (Massimiliano Fatica, Everett Phillips, Josh Romero), especially concerning parallel performance and porting. We also acknowledge GitHub for providing free hosting to the yambo project, including several aspects ranging from inner-core developments, GPL releases, and testing.

We acknowledge PRACE for awarding us access to computing resources on Fermi and Marconi machines at CINECA, Italy and on Piz Daint at CSCS, Switzerland. We also acknowledge the access to computational resources obtained by national programs, such as ISCRA by Italian MIUR.

The yambo developers thanks the Abinit team, and in particular Matteo Giantomassi and Xavier Gonze, for: (i) first providing, together with the Abinit source of versions 6 and 7, a patch for supporting multi-channel projectors and (ii) later for coding the new structure of the WFK files in NETCDF format in version 8, which allowed the development of the latest a2y interface.

Appendix A. Glossary

BSE	Bethe–Salpeter equation
CBM	Conduction bands minimum
DFT	Density functional theory
DFPT	Density functional perturbation theory
EOM	Equation of motion
EP	Election-phonon
GGA	Generalized gradient approximation
GW	Green's function (G) / Screened Coulomb interaction (W)
HAC	Heine–Allen–Cardona
HDF	Hierarchical data format
HPC	High performance computing
HF	Hartree–Fock
IPA	Independent particles approximation
KB	Kleinman–Bylander
KS	Kohn–Sham
LDA	Localized density approximation
MBPT	Many-body perturbation theory
MPI	Message passing interface
netCDF	Network common data form
OMS	On-mass-shell
OpenMP	Open multi-processing
PPA	Plasmon-pole approximation
PETSc	Portable, extensible toolkit for scientific computation
QP	Quasiparticle
SF	spectral function
COH	Coulomb Hole
SEX	Screened exchange
SLEPc	Scalable library for eigenvalue problem computations
UPF	Unified pseudopotential format
VBM	Valence bands maximum
XC	Exchange-correlation

Appendix B. Evaluation of the response function

To compute the response function in **G** space in an efficient way, equation (1) is evaluated by splitting the sum in an internal frequency independent term running over all transitions and an external frequency dependent term running over groups of transitions as follows:

$$\chi^0_{\mathbf{GG}'}(\mathbf{q},\omega) = \sum_{\tilde{n}\tilde{m}\tilde{\mathbf{k}}} F_{\tilde{n}\tilde{m}\tilde{\mathbf{k}}}(\omega,\mathbf{q}) \sum_{n'm'\mathbf{k}'\in D_{\tilde{n}\tilde{m}\tilde{\mathbf{k}}}(\mathbf{q})} R^{n'm'\mathbf{k}'}_{\mathbf{GG}'}(\mathbf{q}), \tag{B.1}$$

where

$$F_{nm\mathbf{k}}(\omega,\mathbf{q}) = \left[\frac{1}{\omega-(\epsilon_{m\mathbf{k}}-\epsilon_{n\mathbf{k}-\mathbf{q}})-\mathrm{i}\eta} - \frac{1}{\omega-(\epsilon_{n\mathbf{k}-\mathbf{q}}-\epsilon_{m\mathbf{k}})+\mathrm{i}\eta}\right] \tag{B.2}$$

$$R^{nm\mathbf{k}}_{\mathbf{GG}'}(\mathbf{q}) = \frac{f_s}{N_\mathbf{k}\Omega} f_{m\mathbf{k}}(1-f_{n\mathbf{k}-\mathbf{q}}) \times \rho_{nm\mathbf{k}}(\mathbf{q},\mathbf{G})\rho^\star_{mn\mathbf{k}}(\mathbf{q},\mathbf{G}'). \tag{B.3}$$

The internal sum runs over degenerate poles $\{n'm'\mathbf{k}' \in D_{\tilde{n}\tilde{m}\tilde{\mathbf{k}}}(\mathbf{q})\}$ while the external sum runs over only one member of the degenerate group. Poles are set to be degenerate if

$$(\epsilon_{n\mathbf{k}-\mathbf{q}}-\epsilon_{m\mathbf{k}}) - (\epsilon_{n'\mathbf{k}'-\mathbf{q}}-\epsilon_{m'\mathbf{k}'}) < \epsilon_{\text{thresh}}. \tag{B.4}$$

The degeneracy threshold is controlled via the input variable

```
CGrdSpXd = 100. # [Xd] [o/o] Coarse grid controller
```

The default 100. means the degeneracy threshold is $\epsilon_{\text{thresh}} = 10^{-5}$ Hartree. Reducing the value of the input variable the threshold is increased. Only in case the input value is set to zero the size of the groups is set to 1 and the external sum runs over all transitions.

Appendix C. Sum-over-states terminators

For the sake of completeness, here we report the sum-over-states terminator expressions introduced in [53] and implemented in `yambo`. Introducing

$$\tilde{\rho}_{m\mathbf{k}}(\mathbf{G},\mathbf{G}') = \langle m\mathbf{k}|\mathrm{e}^{\mathrm{i}(\mathbf{G}'+\mathbf{G})\cdot\hat{\mathbf{r}}}|m\mathbf{k}\rangle, \tag{C.1}$$

$$\tilde{F}_{m\mathbf{k}}(\omega,\bar{\epsilon}_{\chi_0}) = \left[\frac{1}{\omega-(\epsilon_{m\mathbf{k}}-\bar{\epsilon}_{\chi_0})-\mathrm{i}\eta} - \frac{1}{\omega-(\bar{\epsilon}_{\chi_0}-\epsilon_{m\mathbf{k}})+\mathrm{i}\eta}\right] \tag{C.2}$$

the correction to the independent particle response function χ, see equation (5), reads:

$$\Delta\chi_{\mathbf{GG}'}(\mathbf{q},\omega) = \sum_{m\mathbf{k}} \tilde{F}_{m\mathbf{k}}(\omega,\bar{\epsilon}_{\chi_0}) \left[\frac{f_s f_{m\mathbf{k}}\,\tilde{\rho}_{m\mathbf{k}}(\mathbf{G},\mathbf{G}')}{N_\mathbf{k}\Omega} - \sum_{n\leqslant N'_b} R^{nm\mathbf{k}}_{\mathbf{GG}'}(\mathbf{q})\right]. \tag{C.3}$$

In equation (C.3), the parameter $\bar{\epsilon}_{\chi_0}$ denotes the extrapolar energy for the polarizability, while N'_b is the number of conduction band states included in the calculation. Finally, as in section 3, f_s is the spin occupation factor, while n and m are band indexes.

Appendix D. Covariant dipoles

In extended system the coupling of electrons with external fields is described in terms of Berry phase [130]. In this formulation the dipole operator is replaced by the derivative in **k**-space, $\mathbf{r} = \mathrm{i}\frac{\partial}{\partial\mathbf{k}}$. In case of a finite **k**-points sampling the **k**-derivative is replaced by a finite-difference representation, described in [129, 130]. In the limit of linear response, it is possible to derive from this representation a new formula for the dipole matrix elements as:

$$\langle m\mathbf{k}|\mathbf{r}|n\mathbf{k}\rangle = \mathrm{w}_{mn\mathbf{k}} + \mathrm{w}^+_{mn\mathbf{k}} + O(\Delta\mathbf{k}^4), \tag{D.1}$$

with

$$\mathrm{w}_{mn\mathbf{k}} = \frac{\mathrm{i}e}{2}\sum_{i=\alpha}^{3}(\mathbf{r}\cdot\mathbf{a}_\alpha)\frac{4D_{mn}(\Delta\mathbf{k}_\alpha) - D_{mn}(2\Delta\mathbf{k}_\alpha)}{3}, \tag{D.2}$$

where $\mathbf{a}_\alpha$ is the crystal lattice versor. The D_{mn} factors are

$$D_{mn}(\Delta\mathbf{k}_\alpha) = \frac{P_{mn}(\mathbf{k}+\Delta\mathbf{k}_\alpha) - P_{mn}(\mathbf{k}-\Delta\mathbf{k}_\alpha)}{2\Delta\mathbf{k}_\alpha}, \tag{D.3}$$

with

$$\Delta\mathbf{k}_\alpha = \frac{2\pi}{|\mathbf{a}_\alpha|N_{\mathbf{k}^\parallel_\alpha}}, \tag{D.4}$$

and

$$P_{mn}(\mathbf{k}+\Delta\mathbf{k}_\alpha) = \sum_{l}^{\text{occ}} [S(\mathbf{k},\mathbf{k}+\Delta\mathbf{k}_\alpha)]_{ml} \times \left[S^{-1}(\mathbf{k},\mathbf{k}+\Delta\mathbf{k}_\alpha)\right]_{ln}. \tag{D.5}$$

In equation (D.4) $N_{\mathbf{k}^\parallel_\alpha}$ is the number of **k**-points along the reciprocal lattice vector $\mathbf{b}_\alpha$, $S(\mathbf{k},\mathbf{k}+\Delta\mathbf{k}_\alpha)_{ml}$ is the overlap matrix between the orbitals m and l at **k** and $\mathbf{k}+\Delta\mathbf{k}_\alpha$ points and $[S^{-1}(\mathbf{k},\mathbf{k}+\Delta\mathbf{k}_\alpha)]_{ln}$ is the inverse of the overlap matrix between the valence bands.

$P_{mn}(\mathbf{k}_i+\Delta\mathbf{k}_\alpha)$ are the matrix elements of the operators projecting the orbitals of the $\mathbf{k}_i+\Delta\mathbf{k}_\alpha$ and $\mathbf{k}_i-\Delta\mathbf{k}_\alpha$ bands on $\mathbf{k}_i$ in such a way to cancel the phase factor and then the derivative is performed.

ORCID iDs

D Sangalli https://orcid.org/0000-0002-4268-9454
A Ferretti https://orcid.org/0000-0003-0855-2590
H Miranda https://orcid.org/0000-0002-2843-0876
C Attaccalite https://orcid.org/0000-0002-7660-261X
I Marri https://orcid.org/0000-0002-1192-8790
E Cannuccia https://orcid.org/0000-0001-8855-0768
P Melo https://orcid.org/0000-0003-4681-1151

M Marsili https://orcid.org/0000-0003-1009-287X
F Paleari https://orcid.org/0000-0001-8038-8575
A Marrazzo https://orcid.org/0000-0003-2053-9962
P Bonfà https://orcid.org/0000-0001-6358-3037
F Affinito https://orcid.org/0000-0003-0716-2849
M Palummo https://orcid.org/0000-0002-3097-8523
A Molina-Sánchez https://orcid.org/0000-0001-5121-4058
C Hogan https://orcid.org/0000-0002-0870-6361
M Grüning https://orcid.org/0000-0002-2549-6351
D Varsano https://orcid.org/0000-0001-7675-7374
A Marini https://orcid.org/0000-0001-9289-5750

References

[1] Onida G, Reining L and Rubio A 2002 *Rev. Mod. Phys.* **74** 601
[2] Deslippe J, Samsonidze G, Strubbe D A, Jain M, Cohen M L and Louie S G 2012 *Comput. Phys. Commun.* **183** 1269
[3] Umari P, Stenuit G and Baroni S 2010 *Phys. Rev.* B **81** 115104
[4] Marini A, Hogan C, Grüning M and Varsano D 2009 *Comput. Phys. Commun.* **180** 1392
[5] Gonze X *et al* 2005 *Z. Kristallogr.* **220** 558 (Special issue on Computational Crystallography)
[6] Shishkin M and Kresse G 2006 *Phys. Rev.* B **74** 035101
[7] Schlipf M, Lambert H, Zibouche N and Giustino F 2018 (arXiv:1812.03717)
[8] Govoni M and Galli G 2015 *J. Chem. Theory Comput.* **11** 2680
[9] Faber C, Duchemin I, Deutsch T, Attaccalite C, Olevano V and Blase X 2012 *J. Mater. Sci.* **47** 7472
[10] Bruneval F, Rangel T, Hamed S M, Shao M, Yang C and Neaton J B 2016 *Comput. Phys. Commun.* **208** 149
[11] Kotani T and van Schilfgaarde M 2002 *Solid State Commun.* **121** 461
[12] Rohlfing M, Krüger P and Pollmann J 1993 *Phys. Rev.* B **48** 17791
[13] Krause K and Klopper W 2017 *J. Comput. Chem.* **38** 383
[14] Gulans A, Kontur S, Meisenbichler C, Nabok D, Pavone P, Rigamonti S, Sagmeister S, Werner U and Draxl C 2014 *J. Phys.: Condens. Matter* **26** 363202
[15] Ljungberg M.P., Koval P, Ferrari D, Foerster D and Sánchez-Portal D 2015 *Phys. Rev.* B **92** 075422
[16] Leng X, Jin F, Wei M and Ma Y 2016 *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **6** 532
[17] Ruffieux P *et al* 2016 *Nature* **531** 489
[18] Samarakoon D K, Chen Z, Nicolas C and Wang X Q 2011 *Small* **7** 965
[19] Peelaers H, Hernández-Nieves A D, Leenaerts O, Partoens B and Peeters F M 2011 *Appl. Phys. Lett.* **98** 1
[20] Vanin M, Mortensen J J, Kelkkanen A K, Garcia-Lastra J M, Thygesen K S and Jacobsen K W 2010 *Phys. Rev.* B **81** 081408
[21] Filip M R, Eperon G E, Snaith H J and Giustino F 2014 *Nat. Commun.* **5** 5757
[22] Wright A D, Verdi C, Milot R L, Eperon G E, Pérez-Osorio M A, Snaith H J, Giustino F, Johnston M B and Herz L M 2016 *Nat. Commun.* **7** 11755
[23] Bernardi M, Palummo M and Grossman J C 2013 *Nano Lett.* **13** 3664
[24] Molina-Sánchez A, Sangalli D, Hummer K, Marini A and Wirtz L 2013 *Phys. Rev.* B **88** 045412
[25] Gao G *et al* 2012 *Nano Lett.* **12** 3518
[26] Palummo M, Bernardi M and Grossman J C 2015 *Nano Lett.* **15** 2794
[27] Hummelshøj J S, Blomqvist J, Datta S, Vegge T, Rossmeisl J, Thygesen K S, Luntz A C, Jacobsen K W and Nørskov J K 2010 *J. Chem. Phys.* **132** 071101
[28] Luo G *et al* 2011 *Phys. Rev.* B **84** 075439
[29] Chiodo L, García-Lastra J M, Iacomino A, Ossicini S, Zhao J, Petek H and Rubio A 2010 *Phys. Rev.* B **82** 045207
[30] Kang W and Hybertsen M S 2010 *Phys. Rev.* B **82** 085203
[31] Cudazzo P, Attaccalite C, Tokatly I V and Rubio A 2010 *Phys. Rev. Lett.* **104** 1
[32] Varsano D, Sorella S, Sangalli D, Barborini M, Corni S, Molinari E and Rontani M 2017 *Nat. Commun.* **8** 1461
[33] Cannuccia E and Marini A 2011 *Phys. Rev. Lett.* **107** 1
[34] Hogan C, Palummo M, Gierschner J and Rubio A 2013 *J. Chem. Phys.* **138** 024312
[35] http://yambo-code.org
[36] www.etsf.eu
[37] www.max-centre.eu
[38] www.nffa.eu
[39] Giannozzi P *et al* 2009 *J. Phys.: Condens. Matter* **21** 395502
[40] Giannozzi P *et al* 2017 *J. Phys.: Condens. Matter* **29** 465901
[41] Gonze X *et al* 2002 *Comput. Mat. Sci.* **25** 478
[42] Gonze X *et al* 2009 *Comput. Phys. Commun.* **180** 2582
[43] Marques M A, Oliveira M J and Burnus T 2012 *Comput. Phys. Commun.* **183** 2272
[44] Lehtola S, Steigemann C, Oliveira M J and Marques M A 2018 *SoftwareX* **7** 1
[45] Caliste D, Pouillon Y, Verstraete M, Olevano V and Gonze X 2008 *Comput. Phys. Commun.* **179** 748
[46] Rew R and Davis G 1990 *IEEE Comput. Graph.* **10** 76
[47] Brown S A, Folk M, Goucher G, Rew R and Dubois P F 1993 *Comput. Phys.* **7** 304
[48] Qiu D Y, da Jornada F H and Louie S G 2013 *Phys. Rev. Lett.* **111** 216805
[49] Sole R D and Girlanda R 1993 *Phys. Rev.* B **48** 11789
[50] Rozzi C A, Varsano D, Marini A, Gross E K and Rubio A 2006 *Phys. Rev.* B **73** 205119
[51] Ismail-Beigi S 2006 *Phys. Rev.* B **73** 233103
[52] Bussi G, Martin-Samos L, Varsano D, Ferretti A and De Gironcoli S Wigner–Seitz cell cutoff to handle coulomb divergences in anisotropic systems (to be published)
[53] Bruneval F and Gonze X 2008 *Phys. Rev.* B **78** 085125
[54] Berger J A, Reining L and Sottile F 2010 *Phys. Rev.* B **82** 041103
[55] Deslippe J, Samsonidze G, Jain M, Cohen M L and Louie S G 2013 *Phys. Rev.* B **87** 165124
[56] Gao W, Xia W, Gao X and Zhang P 2016 *Sci. Rep.* **6** 36849
[57] Rocca D, Gebauer R, Saad Y and Baroni S 2008 *J. Chem. Phys.* **128** 154105
[58] Aryasetiawan F and Gunnarsson O 1998 *Rep. Prog. Phys.* **61** 237
[59] Larson P, Dvorak M and Wu Z 2013 *Phys. Rev.* B **88** 125205
[60] Coccia E, Varsano D and Guidoni L 2017 *J. Chem. Theory Comput.* **13** 4357
[61] Faber C, Boulanger P, Duchemin I, Attaccalite C and Blase X 2013 *J. Chem. Phys.* **139** 11B612_1
[62] Thygesen K S 2017 *2D Mater.* **4** 022004
[63] Filip M R and Giustino F 2014 *Phys. Rev.* B **90** 245145
[64] Giorgi M P K G 2018 *J. Phys. Chem.* C **9** 5891
[65] Godby R and Needs R 1989 *Phys. Rev. Lett.* **62** 1169
[66] Stankovski M, Antonius G, Waroquiers D, Miglio A, Dixit H, Sankaran K, Giantomassi M, Gonze X, Côté M and Rignanese G M 2011 *Phys. Rev.* B **84** 241201
[67] Cazzaniga M 2012 *Phys. Rev.* B **86** 035120
[68] Marini A, Onida G and Del Sole R 2001 *Phys. Rev. Lett.* **88** 016403
[69] Liu P, Kaltak M, Klimeš J and Kresse G 2016 *Phys. Rev.* B **94** 165109
[70] Giantomassi M, Stankovski M, Shaltaf R, Grüning M, Bruneval F, Rinke P and Rignanese G M 2011 *Phys. Status Solidi* B **248** 275

[71] Rangel T *et al* 2019 (arXiv:1903.06865)
[72] Pickett W E, Krakauer H and Allen P B 1988 *Phys. Rev.* B **38** 2721
[73] Mostofi A A, Yates J R, Lee Y S, Souza I, Vanderbilt D and Marzari N 2008 *Comput. Phys. Commun.* **178** 685
[74] Ferretti A, Mallia G, Martin-Samos L, Bussi G, Ruini A, Montanari B and Harrison N M 2012 *Phys. Rev.* B **85** 235105
[75] Marzari N, Mostofi A A, Yates J R, Souza I and Vanderbilt D 2012 *Rev. Mod. Phys.* **84** 1419
[76] Coccia E, Varsano D and Guidoni L 2014 *J. Chem. Theory Comput.* **10** 501
[77] Varsano D, Giorgi G, Yamashita K and Palummo M 2017 *J. Phys. Chem. Lett.* **8** 3867
[78] Hogan C, Magri R and Del Sole R 2011 *Phys. Rev.* B **83** 155421
[79] Molina-Sánchez A 2018 *ACS Appl. Energy Mater.* **1** 6361
[80] Hogan C, Pulci O, Gori P, Bechstedt F, Martin D S, Barritt E E, Curcella A, Prevot G and Borensztein Y 2018 *Phys. Rev.* B **97** 195407
[81] Varsano D, Marini A and Rubio A 2008 *Phys. Rev. Lett.* **101** 133002
[82] Prezzi D, Varsano D, Ruini A, Marini A and Molinari E 2008 *Phys. Rev.* B **77** 041404
[83] Varsano D, Caprasecca S and Coccia E 2016 *J. Phys.: Condens. Matter* **29** 013002
[84] Varsano D, Coccia E, Pulci O, Conte A M and Guidoni L 2014 *Comput. Theor. Chem.* **1040** 338
[85] Kammerlander D, Botti S, Marques M A L, Marini A and Attaccalite C 2012 *Phys. Rev.* B **86** 125203
[86] Hernandez V, Roman J E and Vidal V 2005 *ACM Trans. Math. Softw.* **31** 351
[87] Marini A 2001 Optical and electronic properties of copper and silver: from density functional theory to many body effects *PhD Thesis* University of Roma Tor Vergata
[88] Blackford L *et al* 1997 *ScaLAPACK Users' Guide* (Philadelphia, PA: SIAM)
[89] Haydock R 1980 *Solid State Physics* vol 35, ed F S H Ehrenfest and D Turnbull (New York: Academic) pp 215–98
[90] Grüning M, Marini A and Gonze X 2009 *Nano Lett.* **9** 2820
[91] Grüning M, Marini A and Gonze X 2011 *Comput. Mat. Sci.* **50** 2148
[92] Ma Y, Rohlfing M and Molteni C 2009 *Phys. Rev.* B **80** 241405
[93] Palummo M, Hogan C, Sottile F, Bagalá P and Rubio A 2009 *J. Chem. Phys.* **131** 84102
[94] Balay S, Gropp W D, McInnes L C and Smith B F 1997 *Modern Software Tools in Scientific Computing* ed E Arge *et al* (Basel: Birkhäuser Press) pp 163–202
[95] Molina-Sánchez A, Hummer K and Wirtz L 2015 *Surf. Sci. Rep.* **70** 554
[96] Marsili M, Sangalli D, Molina-Sánchez A, Palummo M and Marini A 2018 (preprint)
[97] Sangalli D, Marini A and Debernardi A 2012 *Phys. Rev.* B **86** 125139
[98] Sangalli D, Dal Conte S, Manzoni C, Cerullo G and Marini A 2016 *Phys. Rev.* B **93** 195205
[99] Sangalli D, Berger J A, Attaccalite C, Grüning M and Romaniello P 2017 *Phys. Rev.* B **95** 155203
[100] Paleari F, Galvani T, Amara H, Ducastelle F, Molina-Sánchez A and Wirtz L 2018 *2D Mater.* **5** 045017
[101] Giustino F 2017 *Rev. Mod. Phys.* **89** 015003
[102] Poncé S, Margine E R and Giustino F 2018 *Phys. Rev.* B **97** 121201
[103] Forster F, Molina-Sánchez A, Engels S, Epping A, Watanabe K, Taniguchi T, Wirtz L and Stampfer C 2013 *Phys. Rev.* B **88** 085419
[104] Molina-Sánchez A, Sangalli D, Wirtz L and Marini A 2017 *Nano Lett.* **17** 4549
[105] Wang Z *et al* 2018 *Nano Lett.* **18** 6882
[106] Allen P B and Heine V 1976 *J. Phys. C: Solid State Phys.* **9** 2305
[107] Allen P B and Cardona M 1983 *Phys. Rev.* B **27** 4760
[108] Fan H Y 1951 *Phys. Rev.* **82** 900
[109] Antončík E 1955 *Cechoslovackij Fiziceskij Zurnal* **5** 449
[110] Cannuccia E and Marini A 2012 *Eur. Phys. J.* B **85** 320
[111] Marini A, Poncé S and Gonze X 2015 *Phys. Rev.* B **91** 224310
[112] Baroni S, de Gironcoli S, Dal Corso A and Giannozzi P 2001 *Rev. Mod. Phys.* **73** 515
[113] Poncé S, Antonius G, Boulanger P, Cannuccia E, Marini A, Côté M and Gonze X 2014 *Comput. Mat. Sci.* **83** 341
[114] Cannuccia E and Marini A 2011 *Phys. Rev. Lett.* **107** 255501
[115] Gali A, Demján T, Vörös M, Thiering G, Cannuccia E and Marini A 2016 *Nat. Commun.* **7** 11327
[116] Marini A 2013 *J. Phys.: Conf. Ser.* **427** 012003
[117] Bernardi M, Vigil-Fowler D, Lischner J, Neaton J B and Louie S G 2014 *Phys. Rev. Lett.* **112** 257402
[118] Villegas C E P, Rocha A R and Marini A 2016 *Nano Lett.* **16** 5095
[119] Kawai H, Yamashita K, Cannuccia E and Marini A 2014 *Phys. Rev.* B **89** 085202
[120] Molina-Sánchez A, Palummo M, Marini A and Wirtz L 2016 *Phys. Rev.* B **93** 155435
[121] Marini A 2008 *Phys. Rev. Lett.* **101** 106405
[122] Li Y, Chernikov A, Zhang X, Rigosi A, Hill H M, van der Zande A M, Chenet D A, Shih E M, Hone J and Heinz T F 2014 *Phys. Rev.* B **90** 205422
[123] Kaasbjerg K, Thygesen K S and Jacobsen K W 2012 *Phys. Rev.* B **85** 115317
[124] Andrade X *et al* 2015 *Phys. Chem. Chem. Phys.* **17** 31371
[125] Noda M *et al* 2019 *Comput. Phys. Commun.* **235** 356
[126] Attaccalite C, Grüning M and Marini A 2011 *Phys. Rev.* B **84** 245110
[127] Shoji I, Kondo T, Kitamoto A, Shirane M and Ito R 1997 *J. Opt. Soc. Am.* B **14** 2268
[128] Jang J I, Park S, Clark D J, Saouma F O, Lombardo D, Harrison C M and Shim B 2013 *J. Opt. Soc. Am.* B **30** 2292
[129] Attaccalite C and Grüning M 2013 *Phys. Rev.* B **88** 235113
[130] Souza I, Íñiguez J and Vanderbilt D 2004 *Phys. Rev.* B **69** 085106
[131] Resta R 1994 *Rev. Mod. Phys.* **66** 899
[132] King-Smith R and Vanderbilt D 1993 *Phys. Rev.* B **47** 1651
[133] Attaccalite C, Cannuccia E and Grüning M 2017 *Phys. Rev.* B **95** 125403
[134] Attaccalite C, Grüning M, Amara H, Latil S and Ducastelle F M C 2018 *Phys. Rev.* B **98** 165126
[135] Strinati G 1988 *Riv. Nuovo Cimento (1978–1999)* **11** 1
[136] Grüning M, Sangalli D and Attaccalite C 2016 *Phys. Rev.* B **94** 035149
[137] Grüning M and Attaccalite C 2016 *Phys. Chem. Chem. Phys.* **18** 21179
[138] Riefer A and Schmidt W G 2017 *Phys. Rev.* B **96** 235206
[139] Denk R *et al* 2017 *Nanoscale* **9** 18326
[140] http://yambo-code.org/performance
[141] Pizzi G, Cepellotti A, Sabatini R, Marzari N and Kozinsky B 2016 *Comput. Mater. Sci.* **111** 218
[142] Larsen A H *et al* 2017 *J. Phys.: Condens. Matter* **29** 273002
[143] https://github.com/abinit/abipy
[144] http://yambopy.readthedocs.io

[145] Hunter J D 2007 *Comput. Sci. Eng.* **9** 90
[146] https://travis-ci.com/
[147] https://github.com/yambo-code/yambo-aiida
[148] https://aiida-yambo.readthedocs.io
[149] https://github.com/aiidateam/aiida-quantumespresso
[150] http://www.yambo-code.org/robots/index.php
[151] Attaccalite C, Bockstedte M, Marini A, Rubio A and Wirtz L 2011 *Phys. Rev.* B **83** 144115
[152] Amato M, Kaewmaraya T, Zobelli A, Palummo M and Rurali R 2016 *Nano Lett.* **16** 5694
[153] Giorgi G, Yamashita K and Palummo M 2018 *J. Phys. Chem. Lett.* **9** 5891