D3.2

First release of MAX software: report on the identified actions, update of the software development plan, and software release

Luigi Genovese, Stefano Baroni, Uliana Alekseeva, Augustin Degomme, Pietro Delugas, Stefano de Gironcoli, Andrea Ferretti, Alberto Garcia, Paolo Giannozzi, Anton Kozhevnikov, Andrea Marini, Ivan Marri, Daniel Wortmann, and Davide Sangalli

Due date of deliverable	30/11//2019 (**month 12**)
Actual submission date	29/11/2019
Lead beneficiary	CEA (participant number 5)
Dissemination level	PU - Public

Document information

Project acronym	MAX
Project full title	Materials Design at the Exascale
Research Action Project type	European Centre of Excellence in materials modelling, simulations and design
EC Grant agreement no.	824143
Project starting/end date	01/12/2018 (month 1) / 30/11/2021 (month 36)
Website	http://www.max-centre.eu
Deliverable no.	D3.2
Authors	Luigi Genovese, Stefano Baroni, Uliana Alekseeva, Augustin Degomme, Pietro Delugas, Stefano de Gironcoli, Andrea Ferretti, Alberto Garcia, Paolo Giannozzi, Anton Kozhevnikov, Ivan Marri, and Daniel Wortmann
To be cited as	Genovese et al. (2019): First release of MAX software: report on the identified actions, update of the software development plan, and software release. Deliverable D3.2 of the H2020 CoE MAX (final version as of 29/11/2019). EC grant agreement no: 824143, CEA, France.

Disclaimer

Contents

1 Executive Summary

The report contains a description of the code development activities of each of the group belonging to MAX consortium. Each code consortium will illustrate in this document the planned activities and their present status. At the end of the document, an updated Gantt chart of the foreseen activities is drawn. The basis for this document is the deliverable of M6 that has been written to coordinate the development efforts. This document essentially represents the first update of the M6 deliverable, and presents the actual status of the planned software activities which have been listed in the M6 Gantt chart.

2 Code Related Activities

We here present, for each code, the update to the activities of the M6 document. At the end of the report the Gantt chart of the planned activities is also updated.

2.1 BigDFT code

B1 Input-file manipulation and handling: wildcard approaches.

The input-file manipulation technique has been implemented and will be out in the version 1.9.0 of the BigDFT code. Such technique has been implemented in the code package suite, but it is available also to other codes via the `futile` library. Essentially, there are various level in which the input file of the code is handled, described in the following list.

- The database of the input variables is handled *via* a text information written in `yaml` format, with a dedicated syntax, that is inserted in the executable as a string buffer at compile time. Such database, written by the code developers, should provide specifications as the expected ranges of values of a variable, or the expected (exclusive) values that a variable is supposed to have. Also the default value of each variable is expected to be provided. In addition, cross-check mechanisms among the variables values are specified such as, for example, to prevent the user to forget to set some ancillary input parameters is a particular run mode has been chosen.
- Such specification is parsed and converted into a `futile` dictionary at the runtime. The same parsing is done on the `yaml` input file of the code such as provided by the user.
- The two dictionaries are then compared, at runtime, by watching if any of the expected input variables is correctly spelled and if it is provided with correct ranges. Exceptions at runtime are raised if the variables specifications are not met.
- Within this scheme, the code developers can define some profiles, which is a set of pre-defined input variables, tailored for specific execution modes. Such profiles have the advantage to make the usage of the code easier for the non-expert users, thereby enabling the employment of more advanced functionalities. Such set of profiles (which can be invoked in groups) are pre-loaded in the user input dictionary before the updated made by the input

file specification. The user is therefore able to override such information if needed.

An example on how this database is defined, together with a tutorial on how to handle the input variables in the BigDFT code, is provided in the URL [1].

On top of this technology, which as already stated, is provided by the `Fortran` library `futile`, we have added in the `PyBigDFT` python module a way of handling the same input files specifications. In this way the `yaml` input file of BigDFT can be generated via a `python` dictionary, filled either explicitly or by some predefined actions on the `BigDFT.Inputfile` class. Examples and tutorials on how to define input parameters from PyBigDFT are provided at the URL[2].

This task has then been concluded in due time.

B2 Mixed-precision techniques for convolutions and Poisson Solver.

The design of the techniques for the mixed precision technique has started. In particular, we are testing various ways for the code developer to handle mixed precision algorithms without the need to perform explicitly out-of-place memory copies. The buffers would have then to be allocated with the size of double floating-point precision, and it should then be up to some computation kernels (e.g. a suitable version of `sdaxpy` routine) to convert the data. This would be of great advantage because the algorithm can be designed and coded without the need to specify the precision, which can be used as a run-time variable in the calling routines.

In the forthcoming months we will deeply investigate the feasibility of such strategy in the design of algorithms for complex codes. We will then implement the chosen approach in the `libconv` and `Psolver` wrappers.

B3 Usage of Pseudo-Fragment approaches for extended systems in the Support Functions formalism.

During 2019, we have implemented this task in the BigDFT code in the $\mathcal{O}(N)$ algorithm. This task has then been concluded in advance with respect to the expected schedule. A paper about the technique [3] has been published and a book chapter on the subject is in press.

The most important point which is enabled by the technique is the *reduction* of the number of degrees of freedom, within a paradigm with *controllable accuracy*. The linear scaling approaches such as those adopted BIGDFT (but also in other codes like ONETEP or CONQUEST) represent a first fundamental step in this direction: the support functions are adapted to the chemical environment surrounding each region. Such adaptivity allows the reduction of the degrees of freedom, with essentially no cost in terms of accuracy, as the occupied Kohn-Sham orbitals are well represented *by design* within such an approach.

This suggested us that for large systems the degrees of freedom may be reduced further, by identifying the various regions which might be expressed with the same set of support functions, in other terms those regions which have a similar local chemical environment. This is important not only in view of computational savings, but also to gain further insights into the systems constituents. The reduction of the complexity of the description is very important in this context: performing a set of production quantum mechanical simulations with an unnecessarily costly approach would provide a study of poor quality, as the simulation scheme would entangle interactions of different length scales and couplings.

A framework for identifying and exploiting regions with similar chemical environments has already proven to be very useful in the context of molecular systems, where the search for system constituents naturally leads to fragmentation. We have provided in this task a similar approach for extended systems. In such cases, the concept of a separable fragment is not meaningful. However, we have provided indicators which help to distinguish regions of an extended system which may be treated with identical degrees of freedom from regions which require a different set of support functions for an accurate simulation.

This pseudo-fragment approach permits the exploitation of support function similarity in a whole new range of materials, and since there are no restrictions on cutting covalent bonds between pseudo-fragments, the user is free to define them according to their preference.

Both the setup and analysis of the calculations presented in the paper (including the generation of input files) has been done *via* the use of a Jupyter Notebook that is publicly available [4], with the aim of easing both reproducibility of the results and the adoption by interested users. We also note that in cases where a pseudo-fragment approach is not directly applicable, e.g. when there is significant distortion in the geometry of the system (as indicated by a cost function), one might still use it to generate a better input guess for the support functions. This can significantly reduce the number of iterations required to further optimize the support functions.

The set of methods and indicators defined in this paper might also be considered as a first step towards the control of the setup of more complex approaches, such as the treatment of defects with various levels of theory. Currently we keep the support functions fixed, however this might be extended in future to allow the support functions to be further optimized in an active region and remain fixed in an environment region, i.e. to define an embedding method at the level of the basis. Long ranged interactions, which are not associated to a modification of the support functions, might therefore be treated with a different level of theory than short-ranged ones. In such cases, the indicators presented in this work could be used to predict the size of the active region, thereby enabling an *a priori* control of approaches that so far have primarily relied on the physical intuition of the user.

B4 Exact exchange for $\mathcal{O}(N)$ implementation.

The work defining the computational approach to extend this functionality to the $\mathcal{O}(N)$ code is foreseen to start at the beginning of 2020.

2.2 SIESTA code

S1 Basis-set contraction.

A blueprint for the basis-set contraction strategy is being developed, based on similar work done for BigDFT.

S2 Exact-exchange.

A reference implementation using fits of gaussians to Siesta's basis of pseudo-atomic-orbitals is basically ready and undergoing last stages of testing. This is the first time that exact-exchange is being implemented in Siesta, and is already uncovering issues that will be relevant to a more advanced implementation, notably the effective sparsity of the Hamiltonian and thus the interaction with non-diagonalization solvers, and the impact on efficiency of tight tolerance levels.

S4 Break-even points for various solvers and automatic dispatch based on learned heuristics.

Work has already started taking advantage of the recently implemented interface to the ELSI library of solvers. A uniform interface is available to the PEXSI, NTPoly, and OMM solvers, as well as to diagonalization (ELPA). The initial dispatch strategy takes into account the size, problem dimensionality, and sparsity of the Hamiltonian, using the ELSI decision-layer API introduced in recent versions.

S5 Re-design of the legacy $\mathcal{O}(N)$ solver.

The work started at the beginning of the project has already advanced to the point at which the machinery of matrix-formats interconversion is functional. The distributed block sparse format used by the DBCSR library is already supported by the MatrixSwitch gluing library, and the latest stage has involved providing a connection between the Siesta internal formats and MatrixSwitch itself.

As an intermediate step, the libOMM library, implementing a cubic-scaling orbital minimization method (without the extra localization constraints that would make it linear scaling) has been used to perform the calculations of the density matrix and energy, handling dense matrices throughout. The next steps will convert *via* MatrixSwitch the sparse Hamiltonian and overlap matrices from the internal Siesta format to the sparse pdcsr format supported by libOMM, and finally implement the handling of a sparse coefficient matrix reflecting the localization constraints. This will make it possible to perform efficient order-N calculations. The computational effort can be further reduced through the additional analysis of sparsity of the Hamiltonian and overlap matrices and reorganizing them in the block-compressed sparse form.

2.3 QUANTUM ESPRESSO code

Q1 Improved diagonalization algorithms.

Work on improving an alternative diagonalization algorithm based on the Parallel Orbital update scheme [5] has greatly progressed during a visit to Profs. Zhou and Dai's group at the Chinese Academy of Science (CAS) in Beijing. Improvements over the previous implementation include: reduction of memory requirement by computing correction vectors of only a fixed fraction of the target eigenvalues and adaptively optimising additional ones as the former converge; more efficient hamiltonian evaluation over batches of trial wavefunctions and elimination of redundant evaluation by reusing the previously accumulated information; minimisation of communication time spent in the dense-matrix diagonalization bottleneck, as well as reduction of the number of diagonalizations by computing additional correction vectors when only few eigenvalues are yet to converge in order to increase variational flexibility and speed-up convergence. The effectiveness of these additions has been verified in a few test cases. The algorithm is being developed in the context of the code-agnostic `KS_library` driver collection. Work is currently focusing on the CPU version; the GPU accelerated version will follow.

Q2 Direct energy minimisation schemes.

High throughput (HTP) applications rely on the robust performance of the underlying quantum engine. The fast convergence of the self-consistent cycle scheme used in typical use cases is plagued by the occasional failure of the procedure in a small fraction of the cases (in the order of a few percent). Alternative, more robust minimisation strategies based on global optimisation of the energy functional are highly desirable as fall-back solutions (even if possibly less efficient). Work on the implementation of global functional optimisation in the QUANTUM ESPRESSO main engine is underway in collaboration with the already mentioned CAS group in Beijing, also though exchange visits. A preliminary implementation of the global optimisation for norm-conserving pseudopotentials has been developed and extension to the more complex ultrasoft and PAW cases is currently underway. Initial tests on non-critical, insulating, systems show good performance of the adopted preconditioning for the wavefunction update; the critical metallic system case, involving the need to optimise additional and very different variables describing orbital occupation, is being addressed. Two routes are explored in parallel and compared for efficiency: the first one is based on the Ensemble DFT idea [6] of extending occupation numbers to a matrix form to make the corresponding minimisation convex at the expenses of increased memory and operation cost. The second one devises smart preconditioning of the two very different set of variables involved. Preliminary tests on a very limited number of norm conserving metallic systems show promising performance. A number of "difficult test cases", met during recent HTP material screening, has been collected and will be used as test-bed for further tuning and development as soon as the ultrasoft extension of the algorithm becomes available.

Q3 RPA-based advanced exchange and correlation functionals.

Preliminary work necessary to align existing specialised code performing total energy RPA [7] and RPAx [8] calculations to the current QUANTUM ESPRESSO develop version has been undertaken in conjunction with an analysis of the code refactoring needs to fully exploit high-level parallelisation in the imaginary frequency integration in order to enable scalability to pre-exascale machines. Actual code refactoring will start with incorporation of a Postdoc (early in 2020).

Q4 Extension of the localised inner-projection method to EXX.

Work in ongoing to study methods for improving the convergence properties of the method. In the current implementation the capability to push convergence to very low threshold may be limited by the presence of orbitals having a marginal overlap oscillating over or below the threshold. We are testing different methods to freeze or to make such contribution negligible when close to convergence.

Q5 Adaptive parallelisation schemes.

Preliminary work has been done, both in code-specific routines and in the routines managing MPI communicator setup, to extract and encapsulate the initialisation of linear-algebra MPI communicator. Run-time dynamical initialisation of linear-algebra communicators, although not yet implemented, is now possible with little effort. A similar effort for the more complex case of the k-point parallelisation level is planned for the next months.

2.4 YAMBO code

Y1 Restart structure, parallel IO and database re-organization.

With the introduction of parallel I/O based on NetCDF and HDF5, YAMBO databases can now be created independently of the parallel structure used during the simulation. Thanks to that, the user can restart an interrupted run using a different parallel structure. This new feature is useful in general to address software resilience, and particularly important in specific cases, as listed below.

Application to BSE and QP calculations
An example is the calculation and solution of the excitonic Hamiltonian. Its construction is very time consuming, and may not be finalised within a single run of the code (consider e.g. all issues threatening resilience). As scheduled, we coded a technique which allows for a restart of the calculation regardless of the parallel structure used in the different runs. The restart also works if the simulation ends abruptly due to crashes or wall-time limit. This is obtained *via* the use of character matrix, shadow to the complex BSE matrix. The extra disk space required is almost negligible. Both the shadow character and the complex matrix are stored exploiting parallel I/O. The restart was tested on different machines and parallel configuration and has been proven to be robust and reliable. The restart procedure turns out to be a very important feature to be exploited in automatised workflows that need automatic restart. It was thus included in YAMBO version 4.4 which was released in September 2019.

Moreover we exploited the parallel I/O to also recover the restart of the iterative Haydock solver for the BSE solution. The restart procedure, which was already present for serial runs now works also in parallel runs. This feature makes it possible, for example, to get a first solution with lower precision of the excitonic problem and later to restart the solver to reach a more refined solution. Also this feature was tested and then released with version 4.4.

Application to QP calculations
Starting from the experience with the excitonic Hamiltonian, we have also discussed the strategy to restart the calculation of quasi-particles (QP) exploiting the use of a character variable to be store in the QP database. Though not initially foreseen, we are now planning to exploit a similar restart strategy also for full frequency GW calculations (see Y4).

Y2 Exploitation of mixed precision algorithms.

YAMBO supports both single- and double-precision (DP) floating-point arithmetic. The single-precision (SP), in particular, is adopted to reduce the computational and

DP reference	SP (old)	Error (old)	SP (new)	Error (new)
-0.666	-0.849	0.183	-0.664	-0.002
-0.476	-0.620	0.144	-0.474	-0.002
-0.513	-0.737	0.225	-0.510	-0.002
3.791	3.867	-0.076	3.792	-0.001
3.753	3.773	-0.019	3.753	0.000
3.832	3.912	-0.080	3.832	0.000

Table 1: QP corrections at the Γ point for selected across the band gap in a TiO_2 nano-crystal. All energies are in eV. Columns 2 to 5 reports the QP correction and the deviation from the reference value using the SP and the mixed precision algorithms.

storage cost of simulations. In some cases, however, the SP can produce a lack of accuracy that impacts negatively on the final result. For this reason, some critical parts of YAMBO calculations (for instance, accumulations in the calculation of the Hartree-Fock matrix elements or correlation self-energy) are always executed in DP, even if the code has been compiled in SP. In order to identify additional critical sections of YAMBO, we have performed an extensive comparison of runs executed in both SP and DP.

As a result, we have identified particular conditions (generally when large systems are addressed adopting a small number of MPI tasks) where the use of SP leads to a lack of accuracy. For instance, quasi-particle corrections calculated for a TiO_2 nanocrystal in SP (240 electrons, 7 Ry in the response block size, 240 bands in both the polarisability and self-energy), differ up to 0.2 eV with respect to the ones obtained in DP when the simulation is performed on 8 MPI tasks. The difference decreases by increasing the number of threads used in the calculation of the self-energy (specified by the parameter `SE_Threads` in the input file) and rapidly disappears when increasing the number of MPI tasks. To overcome this problem, and thus to improve the numerical accuracy and stability of the code, we have modified the `QP_ppa_cohsex` routine, always defining in double precision only the variable which is used for the extensive accumulation. This modification does not affect the execution time and fixes the problem (see Table 1). We plan to extend the implementation of the mixed precision paradigm to other parts of YAMBO in order to improve memory, disk, and bandwidth usage without affecting the overall accuracy.

Y4 Advanced approaches for full-frequency GW.

This activity is aimed at making YAMBO able to exploit the increasing computational power provided by today's and tomorrow's HPC machines to push further the bar of accuracy in GW and many-body perturbation theory calculations by removing some common approximations. Among those, here we focus on the improving on the plasmon pole approximation, still very popular used in GW calculations. During the first year of MAX , we have developed a new scheme based on a multipole expansion of the screened Coulomb interaction. Preliminary tests done on metallic systems indicate that few poles (e.g. 5 or 6, corresponding to 10-12 frequencies) are enough for an accurate description of the full frequency dependence of the response function. This allows us to obtain accurate GW correlation self-energies by computing the dielectric matrix for a few frequencies thereby reducing the computational costs needed for other full frequency calculations (e.g. real-axis integration or contour deformation techniques) that typically require a number of frequencies ranging from several tens to hundreds. The implementation of this newly developed methodology in the YAMBO code is currently ongoing.

Y5 Advanced self-energies from MBPT.

With the boost in computational capabilities expected with pre-exascale machines, the calculation of advanced self-energies in many-body perturbation theory methods may become possible and usable for realistic systems. In the WP3 MaX work-plan, we considered two types of such developments and we report here the work done during the first year of MaX , which focused mostly on the calculation of GW corrections in the presence of an environment (such a solvent).

- **GW self-energies plus environment**: with the aim of describing the quasi-particle levels of systems (molecules, nanostructures, layers) in the proximity of a dielectric/metallic environment (such as a surface or a solvent), we are extending the calculation of the GW self-energy by including environment effects (e.g. following the polarisable continuum model, PCM, approach) [9, 10]. We have set up the formalism needed to include the effect of the environment into the GW framework and we have interfaced YAMBO with the ENVIRON module,[1] already available in QUANTUM ESPRESSO. We are now starting the testing phase of the implemented scheme.

Y7 Real time parallelisation.

The time propagation of the one–body Green's function is extremely demanding. This is due to the fact that each element of the function must be evolved in time. This has a dramatic effect in terms of total computational time, parallel communications and memory consumption. However the translational invariance imposes the Green's function to be $\mathbf{k}$-point diagonal, a property that can be used in the parallel code development. Indeed now YAMBO distributes the one–body Green's function over k-points. In this way each element of the matrix is evolved independently with a dramatic gain of CPU time and communication. All steps of the time evolution are now distributed: the time–step solvers, the real–time databases I/O, the observable dumping.

The code activities Y3 and Y6 will start as planned in the next few months, according to the Gantt chart shown in Fig. 1.

2.5 FLEUR code

Out of the implementation tasks related to the FLEUR code, the focus of the current release lies in the redesign of the Hamiltonian and overlap matrix setup. In addition, we started to refactor existing code in order to extend the strategies successfully introduced

[1] http://www.quantum-environment.org/

in the matrix setup into the computation of the Coulomb matrix in LAPW. To improve charge density convergence we implemented the Kerker preconditioning for the charge density mixing.

F1 Finishing the restructuring of the Hamiltonian setup into high-level operations
The matrix setup, i.e. the calculation of the matrix elements of the Hamiltonian and the overlap matrix, is one of the most time consuming operations in a `FLEUR` self-consistency cycle and thus one of the determining factors for the total time of the calculation. The contribution of the interstitial region (and also the vacuum for film setups) can be efficiently obtained from the Fourier transformation of the potential and of the LAPW step-function. These calculations scale quadratically with the basis set size and are easily parallelized. Much more relevant and difficult is the setup within the spheres that shows a cubical dependence on the basis set size and hence is dominating.

The contribution to the matrices from a single sphere can be written as

$$M_{ij} = \sum_{LL'} A^*_{iL} M^{\mathrm{loc}}_{LL'} A_{jL'} \; . \tag{1}$$

While this equation looks like a straight forward matrix multiplication of three matrices, the actual implementation is substantially more complex to:

- use the additional fact that the resulting matrix M_{ij} is Hermitian;
- exploit the data-locality of the operation even in a distributed matrix setup;
- exploit the fact that the local matrix $M^{\mathrm{loc}}_{LL'}$ is dominated by its diagonal entries and that different cut-off parameters for the L and L' summations can be used.

Hence, while we try to keep our interface of the corresponding routines simple and the program flow to closely follow the simple mathematical formulation, we also try to ensure performance of the underlying computational kernels by exploiting these physical properties. The resulting code has already been proven to enable the efficient implementation of new functionalities, as for example the more complete treatment of non-collinear magnetism in the spheres due to its simpler overall structure and the clear separation of high-level operations from the underlying complex and optimized algorithms.

F2 Evaluation of the Coulomb kernel in LAPW Similar to the redesigned matrix setup discussed above, we also work on an implementation of a high-level, easy to use set of operations related to the coulomb kernel in the LAPW basis set. Since these operations are key to the implementation of hybrid functionals as well as for similar schemes like the reduced density matrix functional theory, providing such a set of high-level operations will simplify such code and enable the implementation of new functionality while simultaneously providing performance for such operations. In the current release we already prepared for these developments by implementing a strict set of regression tests and by starting to refactor existing code parts.

F3 Improved charge-density mixing schemes We implemented the Kerker preconditioner to increase convergence speed in metallic systems in `FLEUR`. As we do not use a pure plane-wave basis set, the actual implementation of this well-established preconditioner is somewhat more complex due to its non-local correction. The basic idea we followed was to map the algorithm onto the solver for the Hartree potential by generalizing it to handle also the case of a Yukawa potential. The Kerker correction can then be evaluated based on the solution of the Yukawa potential for the change of the charge density. Our implementation covers the representation of the charge in terms of plane-waves as used in the interstitial volume between the atoms as well as more atomic-like expansions used in the MT-spheres. Hence it can be considered the mathematically rigorous Kerker preconditioner for the LAPW method.

Extensive tests have shown that the convergence for large metallic systems is substantially improved up to the point in which the number of iterations needed to achieve convergence becomes independent of the system size. Further test for more inhomogeneous setups are currently performed to explore the limits of the method's applicability.

2.6 CP2K code

Implementation of forces and stress tensor for RPA and double-hybrid functionals in CP2K requires a distributed multiplication of tall and skinny matrices (matrices with very different dimension sizes). At the moment this functionality is provided by the ScaLAPACK library which is CPU-only and which has a poor performance for this kind of matrices. The work is in progress to create a communication-optimal library COSMA for the efficient multiplication of matrices of arbitrary shapes. The technical report on the COSMA algorithm is available [2].
The following developments have been completed in the past 6 months:

- *pdgemm* wrapper for COSMA library which substitutes the ScaLAPACK implementation; the performance of COSMA-*pdgemm* is shown in Fig. **??**
- added AMD ROCm backed for COSMA

The following actions are planned to be finished by M12:

C1 Integrate COSMA library into CP2K using an intermediate ScaLAPACK matrix layout.

The COSMA-*pdgemm* wrapper is ready. It has to be included in the build system of CP2K, making sure that there is no conflict of symbols between ScaLAPACK and COSMA libraries. This work is almost finished.

The following actions are planned next:

C3 Switch to COSMA in RPA calculations.

At this point COSMA library will be fully integrated into CP2K which will accelerate the RPA and related calculations (such as forces and stress tensor) where multiplication of tall and skinny matrices is a bottleneck.

[2] http://spcl.inf.ethz.ch/Publications/.pdf/mmm-tr.pdf

The following actions have been postponed:

C2 Transform CP2K matrices directly to COSMA layout without a need of ScaLAPACK.

In this stage the further optimization will be implemented and the wave-functions stored in the internal CP2K format will be converted to COSMA layout directly without an intermediate ScaLAPCK matrix representation. Due to the complexity of the internal matrix representation in COSMA this action is postponed in favour of more achievable deliverables.

2.7 SIRIUS software development platform

The domain-specific SIRIUS software development platform aims at providing efficient algorithms for the DFT total energy minimization. The work has been started to implement the direct solvers for the wave-function optimization. The following optimizers for the plane-wave pseudopotential method have been implemented:

- orbital transformation method for insulators
- direct minimization for ensemble density-functional theory

The optimizers have been successfully tested on few structures where the classical density mixing scheme doesn't converge. The following actions have been completed:

U1 Prototype and implement conjugate gradient method.

This is an implementation of direct minimization for ensemble electronic structure calculations by Baarman *et al.* [11] in which the update operator for the electronic orbitals takes the structure of the Stiefel manifold into account. In this method the optimization scheme for the occupation numbers ensures that the constraints remain satisfied. The prototype code was written in Python and tested on several structure where conventional mixing fails.

The following actions are in development stage:

U2 Prototype and implement proximal gradient method.

This is an implementation of proximal gradient method for ensemble density functional theory by Ulbrich *et al.* [12] which is suitable for metallic systems. The prototype will be done in Python using SIRIUS bindings.

The following actions remain to be implemented:

U3 Interface advanced density optimizers with QUANTUM ESPRESSO and CP2K.

Once the wave-function optimizers are prototyped and proved to be working they will be implemented in a highly-efficient way in a standalone library using reverse communication and interfaced with SIRIUS. This will allow QUANTUM ESPRESSO and CP2K codes to call the optimizers from SIRIUS and find the ground states of systems which cannot be converged using standard mixing techniques.

3 Conclusions and perspectives

The development efforts continue as expected in M6. Some activities have been completed before the expected date. Overall, the updated Gantt chart of Fig. 1 is in line with the expected work at M6.

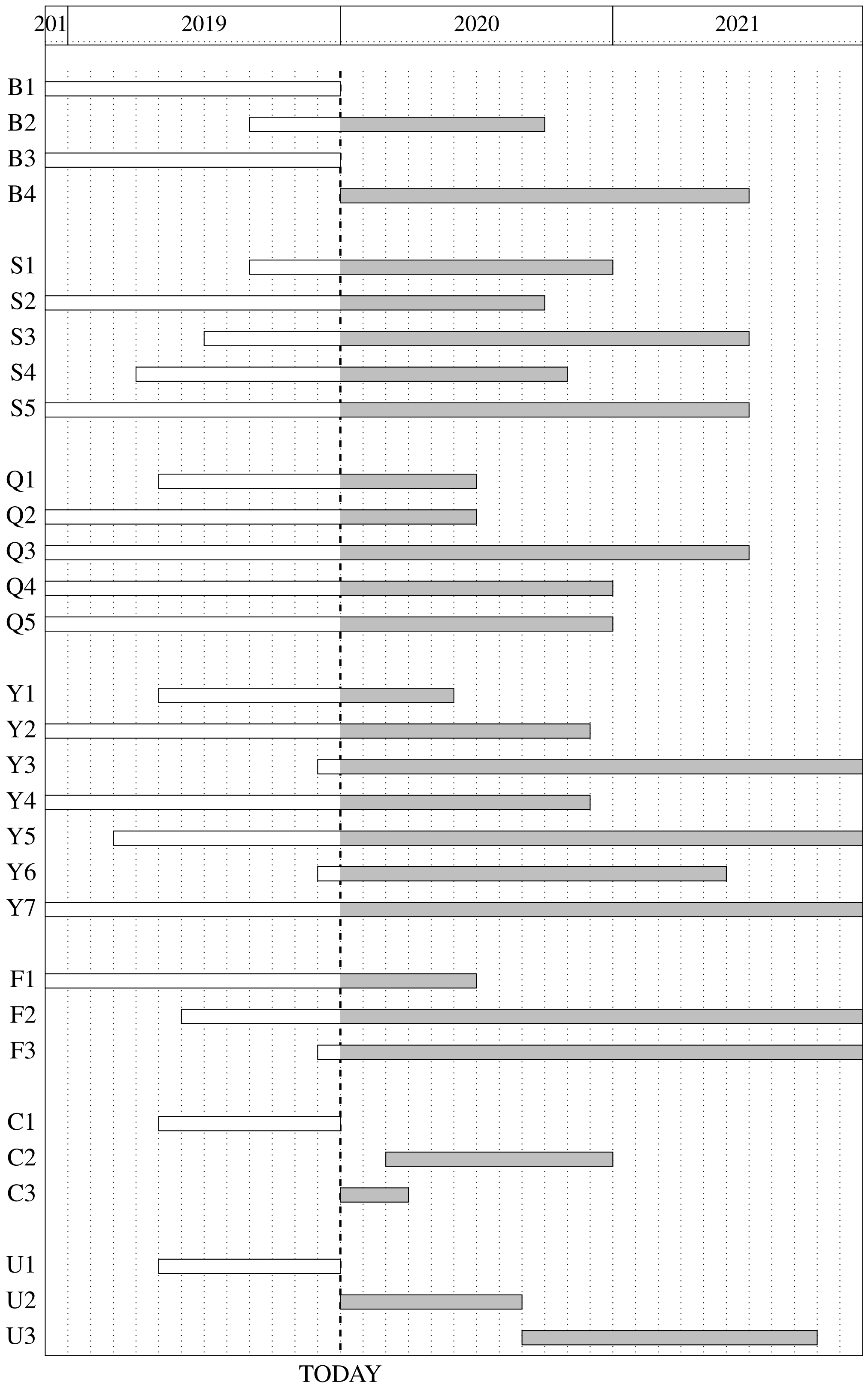

Figure 1: Gantt diagram of the tasks.

Acronyms

RPA Random Phase Approximation. 8

References

[1] Tutorial for Input variables in the BigDFT code. URL http://bigdft.org/Wiki/index.php?title=Inserting_a_new_input_variable_in_the_code.

[2] Tutorial for Input variables in the BigDFT code, from PyBigDFT. URL https://bigdft-suite.readthedocs.io/projects/PyBigDFT/en/latest/inputfiles.html.

[3] Ratcliff, L. E. & Genovese, L. *J. Phys. Cond. Mat.* **31**, 285901 (2019).

[4] Notebooks for handling the Pseudo-Fragment approach in BigDFT code. URL https://gitlab.com/luigigenovese/pfrag-sicnt.

[5] Pan, Y. *et al.* A parallel orbital-updating based plane-wave basis method for electronic structure calculations. *J. Comp. Phys.* **384**, 482–492 (2017).

[6] Marzari, N., Vanderbilt, D. & Payne, M. C. Ensemble density-functional theory for ab initio molecular dynamics of metals and finite-temperature insulators. *Phys. Rev. Letters* **79**, 1337–1340 (1997).

[7] Nguyen, H.-V. & de Gironcoli, S. Efficient calculation of exact exchange and rpa correlation energies in the adiabatic-connection fluctuation-dissipation theory. *Phys. Rev. B* **79**, 205114 (2009).

[8] Colonna, N., Hellgren, M. & de Gironcoli, S. Correlation energy within exact-exchange adiabatic connection fluctuation-dissipation theory: Systematic development and simple approximations. *Phys. Rev. B* **90**, 125150 (2014).

[9] Tomasi, J., Mennucci, B. & Cammi, R. Quantum mechanical continuum solvation models. *Chem. Rev.* **105**, 2999–3094 (2005).

[10] Andreussi, O., Dabo, I. & Marzari, N. Revised self-consistent continuum solvation in electronic-structure calculations. *J. Chem. Phys.* **136**, 064102 (2012).

[11] Baarman, K., Havu, V. & Eirola, T. Direct minimization for ensemble electronic structure calculations. *J. Sci. Comput.* **66**, 1218–1233 (2016).

[12] Ulbrich, M., Wen, Z., Yang, C., Klöckner, D. & Lu, Z. A proximal gradient method for ensemble density functional theory. *SIAM J. Sci. Comput.* **37** (2015).