D3.1

Report on identified algorithmic advances, and their software development plan

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

Due date of deliverable	31/05/2019 (**month 6**)
Actual submission date	31/05/2019
Final version date	31/05/2019
Revised version date	19/02/2021
Revised version submission date	19/02/2021
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.	D3.1

Authors	Luigi Genovese and Stefano Baroni, Stefano de Gironcoli, Pietro Delugas, Andrea Ferretti, Alberto Garcia, Paolo Giannozzi, Anton Kozhevikov, Andrea Marini, Ivan Marri, Pablo Ordejon, Davide Sangalli, Daniel Wortmann.
To be cited as	Genovese et al. (2019): Report on identified algorithmic advances, and their software development plan. Deliverable D3.1 of the H2020 CoE MaX (final version as of 19/02/2021). EC grant agreement no: 824143, CEA, France.

Disclaimer

	Change	Author	Note
Change 1	Sections 1 (Executive Summary), 2 (Introduction), 3 (Organization of the activity)	L. Genovese, A. Ferretti	largely changed to clarify the logic and structure of the planned work and SDP. Connections with other WP activity have been highlighted.
Change 2	Section 4 (Identified Algorithms and Software Development Plan)	L. Genovese, A. Ferretti	Updated with more information about input/output from other WPs.
Change 3	Section 5 (Summary of activities: Staging, timeline, link to WPs)	L. Genovese, A. Ferretti	General description of the association of the activities of WP3 with other WPs (including pre-requisites), complexity, and timeline added.
Change 4	Section 6 (Conclusions and outlook)	L. Genovese, A. Ferretti	General conclusions added.

Contents

1 Executive Summary

MAX has an entire work package (WP) devoted to "*Code Evolution: exploiting algorithmic advantages enabled by the exascale transition*". It deals with the implementation of algorithms in the MAX flagship codes, in order to best exploit pre-exa- and exa-scale architectures, by improving resilience and performance and enhancing code capabilities (e.g. to enable the computation of new features which was not possible before). The main actions fall mainly into three categories:

- Actions oriented to robustness, which improve "software resilience" at the pre-exascale level (T3.1);
- Implementation of new functionalities on the existing code bases, aimed at the exploitation of pre-exa- and exa-scale machines (T3.2);
- Algorithmic-restructuring actions (T3.3), i.e. initiatives that may potentially require a redesign of the application.

This document presents the identified "algorithmic advances" which are planned to be implemented in the codes of the MAX consortium. The actual implementation details of such algorithms are presented in the form of a software development plan. This includes a list of the planned algorithms for each code, a classification of each algorithm according to the three categories above, and a tentative timeline in the form of a Gantt chart. Finally, this documents highlights how the above activities are expected to provide/receive input to/from other technical WorkPackages of MAX .

2 Introduction

The increasing computational power of modern architectures represents at the same time an opportunity and a challenge, for both code users and developers. On the one hand, once the performance portability of scientific codes to new HPC machines was achieved,[1] increased computational capabilities would enable new investigation directions within present-day research paradigms (i.e. today's theoretical frameworks and computational approaches). On the other hand, today's workhorse codes and algorithms were conceived, for the great majority of the cases, in an era – remarkably not long ago – when routine production calculations were significantly different from those running today, and even more so in the perspective of the coming new HPC architectures. It appears therefore not surprising that the usage of large scale supercomputing facilities is pushing for deep changes in the adopted computational approaches and algorithms themselves (see e.g. the impact of machine learning or data-oriented methods in computational sciences).

In the context of materials science and electronic structure methods, the situation mirrors the general frame. Given an increase in computational resources, one can think to address larger systems (i.e. to add complexity to the description of the problem at hand), possibly requiring ad hoc algorithms, or to widen the scope of scientific codes by enabling the calculation of new properties (out of reach in the past) or advocating levels of theory with improved accuracy. Importantly, the development of algorithms along these lines is one of the core actions of WP3.

[1] This is not trivial and is the subject of MAX WP2.

Next, one can also consider to change the computational approaches or the usage models of HPC resources. Examples include: the use of machine learning methods to devise force fields for classical molecular dynamics trained on (and displaying the accuracy of) quantum mechanical simulations; the use of stochastic sampling applied to density functional theory, many-body perturbation theory, or quantum chemistry methods; the use of quantum embedding approaches to address large scale systems; or the use of high throughput screening applied to materials science. The question that arises is then which approaches or models will be best fitting on novel HPC platforms. This paradigm (computational hardware as a driving force for the development of theoretical methodologies and algorithms) would be even more evident when quantum computing approaches were to be included in the scenario.

We insist here on this point: the focus is on the science that a novel, potentially disruptive set of computational paradigms might enable to address. But to meet such objective, essential actions about software restructuring and end-user code interfaces need to be identified and implemented (see MAX WP1 for the design of the software architecture), together with the basic algorithms encoding the new approaches. Therefore, a second goal of this WorkPackage is to identify such actions and algorithms, and to implement them for some use-cases of the consortium codes. This is particularly relevant also in view of the scientific cases to be addressed by the Demonstration work package, MAX WP6. Given their relevance, capacity-computing tasks like high-throughput screening are the subject of a dedicated work-package, MAX WP5. Note, however, that algorithmic advances related e.g. to robustness developed within WP3 have been identified as a necessity for these approaches.

This document is organised as follows. In Sec. 3 we first introduce a classification of the algorithms to be addressed by the WP activity. Next we discuss how these developments interact (sketched in terms of input/output streams) with other technical WorkPackages active on the software development. Eventually, the rationale for scheduling and timeline of the planned actions is presented. In Sec. 4, the actual Software Development Plan (SDP) for each code in the MAX consortium is provided. A summary of the WorkPackage activities and the conclusions of the present document are presented in Secs. 5 and 6, respectively.

3 Organisation of the activity

3.1 Main classes of algorithms

The identified algorithmic advances at the core of MaX activity can be clustered into three main classes, according to the main very goal of the algorithms to be implemented:

T3.1 Software-related fault resilience algorithms and solutions: the items of this task are meant to *prevent* a waste of computing time due to incorrect behaviour of the code or incorrect setup of the input parameters. This is not associated to traditional fault resilience intended as a feedback to a hardware problem, but rather to a *software* fault, in the sense explained above. Such "fault resilience" might be implemented in two ways:

- By creating new approaches for fault tolerance at the level of the algorithm itself, that can detect a runtime misbehaviour and provide a fallback solution.

- By correcting possible weaknesses of the code components in certain regimes; for example, by preventing the code to run in a regime for which a selected functionality has not been conceived.

T3.2 Enabling new code functionalities with an exascale mindset: the goal is to open research avenues that have up to now been considered unfeasible or impractical within the mindset of current HPC tools, but that could be explored thanks to the increasing computing power. These actions will be by no means simple development initiatives aimed at the traditional expansion of code functionalities. The emphasis here is on the unlocking of computations whose time-to-solution would be too long, or simply not possible, nowadays.

T3.3 Exploitation of new algorithms at the pre-exascale: the code modifications which belong to this task will be associated to algorithmic advances which are enabled by the pre-exascale computing power. New workflows and different I/O will be implemented, in a context with high thread concurrency and increasing computational workload. Actions in this tasks will not only be associated to new functionalities but rather to new computational approaches for exploitation of supercomputers.

3.2 Connection with other Work Packages

The algorithmic developments that are presented here depend on or relate to the activities performed in other Work Packages. Examples of such interactions are discussed in the following.

- WP1: algorithmic development may depend on libraries modularised within WP1 (input from WP1, e.g. YAMBO calculations without empty states will take advantage of LRlib from QUANTUM ESPRESSO) or may become part of a library once ready (output to WP1, eg action B1 from Sec. 4 will be developed within the FUTILE lib from BigDFT). Other examples of "input from WP1" are represented by cases where the algorithmic development requires a restructuring of the code architecture (e.g. the KS_solver lib from QUANTUM ESPRESSO).

- WP2: this WorkPackage deals with performance portability and it is therefore natural to envisage a situation where algorithms are first implemented without GPU support by WP3 and then, when relevant, ported to heterogeneous accelerated systems by WP2 (output to WP2). Conversely, some specialised algorithms relevant for performance portability to be developed in the context of the CP2K codes (e.g. within the COSMA library) are to be considered as input to WP2.

- WP4: dealing with technology and co-design, WP4 can provide indications about trends and emerging standards (e.g. which standards can/should be adopted to address non-volatile memory), which can in turn be relevant for algorithm adoption and implementation. Conversely, WP3 can provide kernels to be used as co-design vehicles (e.g. a new KS_solver for density functional theory).

- WP5: The whole class of algorithms for robustness is quite relevant for WP5, which deals among others with high throughput calculations (e.g. having robust

algorithms for the solution of the DFT Kohn-Sham problem is very desirable when tens of thousands of calculations are run automatically for sampling total energies of materials).

- WP6: deals with scientific and technological demonstration of MAX code.

An indication with the main connections with other WPs will be provided when presenting the details of the algorithms (Sec. 4).

3.3 Complexity assessment, Scheduling, Timeline

WP3 addresses the progressive adaptation in terms of new available algorithms and computable physical quantities of MAX software codes (and in turn of the related research community) to the advent of pre- and exascale machines. As discussed in Sec. 3.2, the algorithmic advances targeted by WP3 may (and in many cases do) depend on the activity of other WorkPackages (see Sec. 4), e.g. via requirements on libraries or code restructuring. Such dependencies represent pre-requisites that need to be taken into account when scheduling the start of the activity for each algorithm/feature (action) to be implemented. Moreover, it is also important to assess the complexity of the software development required to adopt and implement these actions, as well as to evenly distribute personnel effort during the MAX time-span. In turn, this information allows us to define the timeline/duration and scheduling of each WP3 action, which is reported in the form of a Gantt chart, Fig. 2.

Concerning dependencies, the algorithms and features to be implemented within the scope of WP3 can be grouped into 3 classes (or stages):

Stage1: No prerequisites, direct implementation. The algorithmic actions to be implemented do not have relevant dependencies on libraries or code restructuring, and can be directly addressed. This stage includes developments at short and medium term, that can be realised even without altering the present structure of the related codes. Longer term actions are in principle possible, but given the implementation complexity involved, it is unlikely that they will not depend on code restructuring (see Stage 3). The features implemented at this stage will provide interesting functionalities in the codes whilst at the same time helping to raise the the awareness of software *design limitations* that have to be overcome by code restructuring (output to WP1 and WP2).

Stage2: Mild prerequisites. This Stage addresses algorithms and physical features that only have limited dependencies on WP1/WP2 libraries and may require minor code restructuring at the global level. Situations where important changes are involved in limited parts of the code are also considered. This is e.g. the case when the upgrade of the I/O of a code is required in view of an algorithmic redesign or implementation. In this specific situation, thorough discussions will be performed, jointly with WP4, to address the best practices for I/O in the scenario of evolving technologies.

Stage3: Strong prerequisites and code restructuring. In this stage, long term and extended actions (algorithms/features) are addressed. Typically, the requirements in

terms of libraries and code re-organisation will be considered first (likely by other WPs, such as WP1 and WP2). Special requirements that may be emerging during WP3 activities connected to these algorithms will be promptly communicated to the relevant WPs in order to be addressed in advance.

4 Identified algorithms & Software Development Plan

In this Section we present, on a code basis, the detailed list of algorithms to be adopted and implemented (actions), as identified by each code development team within MaX. To each of the actions we assign a label which will be used to identify its expected timeline in the Gantt chart shown in Fig. 2. Additionally, for each algorithm/feature we also indicate the connection and flow (to/from) of information with respect to other WorkPackages, as outlined above.

4.1 BigDFT code

B1 Input-file manipulation and handling: wildcard approaches.

The variety of the input parameters that is present in a DFT code is often a source of mistakes by the non-expert users, especially in the context of non-conventional simulation on massively parallel architectures. In this situation it is important for the developers to provide internal cross-checks of the code input parameters. We plan to insert in the input file system of BigDFT a system of automatic cross-check between input variables to avoid inconsistencies in the specification of the input variables. This would limit – if not avoid – the potential wasting of computational resources which would result from a bad specification of input files in a massively parallel calculations. In BigDFT, like in other DFT codes, there are many parameters which can be varied, however it is possible to achieve robust and reliable results using a pre-defined set of values for the majority of these variables. We can exploit this by importing a profile, which already defines a set of common values. All these techniques are going to be achieved with the help of the `FUTILE` library, delivered in the context of WP1. For this reason, they might be also imported in other DFT codes which use a representation of the input file as key/value pairs.

Output to: WP1 (FUTILE lib)

B2 Mixed-precision techniques for convolutions and Poisson Solver.

The BigDFT code has been proved to handle calculation of very large systems – up to many thousands electrons [1] – with Hybrid Functionals thanks to an accurate and unconstrained application of the Fock operator, that may also benefit from acceleration by Graphic Processing Units. We plan to further improve this functionality by introducing a mixed precision approach for the self-consistency cycle of hybrid functional calculations, by calculating with single precision the first part of the SCF cycle and having double precision Fock operator for highly precise calculations. In addition, the same techniques might be used for the wavelets convolutions – important for the representation of the Kohn-Sham Hamiltonian operator – thanks to the extensibility of the `libconv` library, which will be shipped with the BigDFT suite.

Output to: WP1 (LibConv, PSolver, FFTXlib, SpFFT);
Input from: WP2 (Boast, GPU)

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

Thanks to the specific properties of the BigDFT code, we plan to present a computational approach which is tailored for reducing the complexity of the description of extended systems in density functional theory. The idea is to define a recipe for generating a set of localised basis functions which are optimised either for the accurate description of pristine, bulk-like Wannier functions, or for the in-situ treatment of deformations induced by defective constituents such as boundaries or impurities. With this approach, we would like to identify the regions of an extended system which require dedicated optimisation of the Kohn–Sham degrees of freedom, and provide the user with a reliable estimation of the errors – if any – induced by the locality of the approach. Such a method facilitates on the one hand an effective reduction of the computational degrees of freedom needed to simulate systems at the nanoscale, while in turn providing a description that can be straightforwardly put in relation to effective models, like tight binding Hamiltonians. The wavelet-based method employed in this works allow for the calculation of systems with different dimensionalities, including slabs and fully periodic systems. Such an approach will be made possible by the usage of `PyBigDFT` library released in the context of WP1.

Input from: WP1 (PyBigDFT)

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

We plan to extend the calculation of hybrid functionals to the formalism of the Support Functions which is used in the Linear-Scaling algorithm of the BigDFT code. Such an extension will be done by further specialising the communication method of a round-robin scheme that is now adopted in the `FUTILE` library. Such calculations will benefit from the sparsity of the density kernel matrix that would preserve the $\mathcal{O}(N)$ scaling for this very challenging calculations, albeit with a considerably larger prefactor than today's calculations with semilocal functionals. The scheduling of the communications which will be needed to do exact exchange operator contracted with a sparse density kernel is of a different kind of what exists now, as it has to consider a different workflow than the round-robin scheme implemented for the cubic-scaling approach presented in [1]. Extensive tests are planned in order to find the crossover points between the implemented approach in the cubic-scaling code and this implementation.

Input from: WP1 (FUTILE, PSolver);
Output to: WP2 (GPU)

4.2 SIESTA code

S1 Basis-set contraction.

We plan to generalise the internal interfaces in SIESTA that deal with the information about basis orbitals to provide two levels of operation: one using the "primitive" basis set employed now, and the other using a "contracted" basis set with lower cardinality. The latter will be constructed "on the fly" [see [2] and refs. therein; also similar work in BigDFT] by an extra optimisation step. These 'minimal basis sets' will have a very large impact on performance in general, and are

essential in particular for the efficiency of the CheSS $O(N)$ solver, whose operation count depends on the size of the spectrum of the Hamiltonian.

Output to: WP6 (demonstration)

S2 Exact-exchange.

A "classic" algorithm based on intermediate conversion of the PAOs to Gaussians is currently being (re)-implemented and is almost ready. But we have in our plans also the potentially better performing ACE (Adaptively Compressed Exact-Exchange) [3] algorithm.

Input from: WP1 (PSolver)

S3 Improved algorithms for scf convergence.

Here we have started by extending the basic toolbox, and adding mechanisms for changes in the mixing recipe as the calculation progresses, under various conditions (see the new blocks description in the manual). However, more work needs to be done to advance the robustness and efficiency, both by refinement of existing techniques and by the implementation of new algorithms. The exploration of the proper heuristics can be made more efficient in terms of programmer's time by the use of Lua scripts, which can access the data structures of the program and modify the convergence parameters on-the-fly. New ideas can be tried without recompilation. This Lua-embedding is a major feature in SIESTA that now sets it apart from other codes, but that is going to be offered as a module in WP1. This action might be considered to span two task realms: that of the fault resilience (non-convergence should be avoided), and that of the implementation of novel algorithms (the scf convergence loop might be substituted by other techniques, such as, e.g., direct minimisation).

Input from: WP1 (LUA Flook library);
Output to: WP1 (KS_solver), WP5 (robustness)

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

The solvers currently available in SIESTA have different complexity dependencies on the size of the system, and different use-case features: Diagonalisation is a cubic scaling direct method, and it comes in several flavours (ScaLAPACK: standard, MRRR, etc; ELPA as a more efficient re-implementation of the basic algorithms); The Orbital Minimisation Method (OMM) is a cubic scaling solver algorithm that allows for the reuse of previous information. The Pole Expansion and Selected Inversion (PEXSI) solver affords a multi-level parallelisation with reduced scaling (at most quadratic with system size), and CheSS, the density-matrix purification method within ELSI-NTPoly, and the legacy $O(N)$ method in SIESTA based on OMM-plus-localisation, are linear-scaling algorithms.

Each solver has an effective prefactor which depends on the details of the system and the use case (e.g. single-point vs. MD runs). We will investigate the break-even points as a function of system size and characteristics, and collect the information into a form that can be dynamically re-used in actual runs for the most efficient

selection of the solver. Steps in this direction are also being taken in the ELSI project. This action can also be seen as dealing with resilience (the use of the wrong solver should be avoided) and with novel algorithmic ideas (including the dynamical combination of algorithms).

Input from: WP1 (ELSI-interface);
Output to: WP5 (robustness)

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

The original linear-scaling functionality in SIESTA was based on the combination of the OMM method with orbital localisation. We plan to re-design it by exploiting the libDBCSR library for sparse-matrix multiplication produced in T1.1 (WP1). In fact, whole new families of $O(N)$ methods can be implemented on that foundation (connection to the use of the Algorithm Development Platform of T1.2 (WP1)).

Input from: WP1 (libDBCSR)

4.3 QUANTUM ESPRESSO code

Q1 Improved diagonalisation algorithms.

A possible point of failure for density-mixing QUANTUM ESPRESSO `pw.x` runs is the appearance of non positive-definite overlap matrices during the iterative diagonalisation. For this we will try alternative diagonalisation algorithms with reduced subspace diagonalisation, that are both more robust and more scalable, these new developments will be incrementally added to the `KS_library` released by WP1.

Output to: WP1 (KS_Solvers)

Q2 Direct energy minimization schemes.

For non periodic systems a global minimisation approach will fix both diagonalisation failures as well provide a more robust self consistent method.

Output to: WP1 (KS_Solvers), WP5 (robustness)

Q3 RPA based advanced exchange and correlation functionals.

Advanced exchange and correlation functionals based on the Random Phase Approximation (RPA) [4] and its extensions including exchange-kernel corrections (RPAx) [5] have recently emerged as promising, yet extremely costly, accurate functionals naturally incorporating static and dynamical correlation effects. In this action we will modify existing specialised codes in QUANTUM ESPRESSO performing total energy RPA and RPAx calculations to fully exploit high-level parallelisation of the required integration in the imaginary frequency domain, enabling scalability to pre-exascale machines. We will apply this approach to several benchmark systems probing different correlation regimes.

Output to: WP6 (demonstration), WP2 (GPu)

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

We will extend the localised inner-projection approach [3, 6] to exact exchange, successfully demonstrated for the ground state to excited-state properties within time-dependent DFT and MBPT, as well as to real-time simulations of electronic excitations. We expect that this action will finally close the gap between accurate but expensive excited-state simulations using plane waves, with less expensive, but less accurate, ones using localised basis sets.

Output to: WP6 (demonstration)

Q5 Adaptive parallelisation schemes.

On the hardware architectural side, we plan to implement and adopt advanced adaptive parallelisation schemes. Currently the parallelisation strategy is defined via command-line options at startup. We will implement a dynamical definition of communicators and grids in real and reciprocal space, to be activated at run time, switching between different data distributions according to the task being performed, in order to optimise communication balance. Actions of this side are also planned in WP2. The developments of this side will be eventually added to the `UtilXlib` library distributed by WP1.

Output to: WP1 (UtilXlib)

4.4 YAMBO code

Y1 Restart structure, parallel IO and database re-organisation: Application to BSE and QP calculations

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

An example is the calculation and solution of the excitonic Hamiltonian. The construction of the excitonic Hamiltonian is very time consuming, and may not be finalised within a single run of the code (consider e.g. all issues threatening resilience). The restart technique that we plan to code allows for a proper restart which is independent from the parallel structure used and from the way the simulation ended. This will be obtained via the use of character matrix, shadow to the complex BSE matrix. The extra disk space required is almost negligible. We also plan to apply the same strategy to restart the calculation of quasi-particles (QP). As above, this computational step is typically very time-consuming and it would be definitely important to store the data (QP corrections) already computed for later use or restart. A flexibly and parallel I/O database handling is foreseen to be a very effective solution also for this case.

Input from: WP1 (IO_Ylib)

Y2 Exploitation of mixed precision algorithms

YAMBO supports both single and double precision (DP) arithmetic. The single precision (SP), in particular, allows the user to reduce the overall computational

and storage requirements and often the time-to-solution. In some cases, however, the use of SP floating point during the whole computational workload could produce a lack of precision in the final results. For this reason, some specific parts of YAMBO (eg extensive accumulations as in the calculation of the GW self-energy) are executed in DP. We plan to extend the implementation of the mixed precision paradigm to other parts of YAMBO (these parts will be identified by performing specific simulations in both SP and DP) in order to significantly improve memory, disk and bandwidth usage without affecting the results precision. Moreover, we aim at improving numerical accuracy and numerical robustness in highly parallelised applications, especially when the Open-MP paradigm is extensively exploited. Finally, we will also consider and evaluate the use of the mixed-precision FFT library to be developed within the QUANTUM ESPRESSO consortium.

Input from: WP1 (FFTXlib)

Y3 Real–time propagation with atomic motion and interface with QE

The real time module of the YAMBO code allows the user to describe electronic propagation using non local self-energies in an efficient way. This is achieved projecting the equation of motion (EOM) in the basis-set of the Kohn–Sham (KS) wave–functions. The propagation of the EOM has, then, the same cost with local and with non-local potentials. Moreover the choice of the number of states in the KS basis–set naturally imposes a cutoff on the maximum frequency oscillation permitted. This makes possible the use of larger time-steps compared to plane–wave or real–space implementations. The drawback is that the dynamics of electrons away from the equilibrium position of the nuclei (where the KS wave–functions are centred) can be hardly described. This is a limitation if atomic motion, for example within Ehrenfest dynamics, is considered. Indeed the atomic displacement carries the electronic density away from the initial position. The solution is to update the basis–set during the dynamics. We will implement a scheme which has been already used for a pure TD-DFT code [7]. This will require an on the fly interface between YAMBO and the `pw.x` code from QUANTUM ESPRESSO, where YAMBO will feed `pw.x` with the atomic positions (and eventually the electronic density) and receive the KS wave–functions.

Input from: WP1 (UPF_pseudolib, FFTXlib);
Output to: WP6 (demonstration)

Y4 Advanced approaches for full-frequency GW

Simple approximations (such as the plasmon-pole models) to deal with dynamical nature of the screened Coulomb interaction W, and to ease the frequency convolution needed to compute the GW self-energies are still very common in today's GW calculations, but may break down and severely limit the overall accuracy of the implementation. In the exascale perspective, removing this limitation not only becomes feasible for a large class of systems, but is also an interesting way to exploit the available computing power. A number of methods are already proposed in the literature, such as real axis integration [8], analytic continuation [9, 10, 11], contour deformation [12] to name a few. Here we plan to improve the full frequency implementation of YAMBO (real axis method) by implementing more advanced

techniques (either among the existing ones or newly developed) to address this point.

Input from: WP1 (DevXlib);
Output to: WP2 (GPU), WP5 (validation)

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 MAX work-plan we consider two types of such developments:

- GW self-energies plus environment: with the aim of describing the quasiparticle levels of systems (molecules, nanostructures, layers) in the proximity of a dielectric/metallic environment (such as a surface or a solvent), we plan to extend the calculation of the GW self-energy by including environment effects (e.g. following the polarisable continuum model, PCM, approach) [13, 14]. We plan to interface YAMBO with the ENVIRON module,[2] already available also from QUANTUM ESPRESSO.
- SOSEX-like self-energies: beyond GW methods are emerging as higher accuracy MBPT methods for excited state properties. Upon a critical evaluation of the accuracy of this class of methods (ongoing work by some of the YAMBO developers on model systems), we consider to implement the SOSEX approximation [15, 16] also for 3D realistic system in YAMBO. This approach is numerically very challenging and would fit naturally the capabilities of pre- and exascale machines.

Input from: WP1 (Driver_Ylib, LA_Ylib, CoulCut_Ylib, DevXlib);
Output to: WP2 (GPU), WP6 (demonstration)

Y6 YAMBO without empty states

Excited state calculations are severely limited by the large number of (empty) states involved in sum-over-states operations used to converge the physical quantities (e.g., the self–energy or the screened Coulomb interaction). Lanczos-based algorithms have been applied to GW, BSE, and TD-DFT methods [17, 18, 19, 20, 21, 22, 23] and demonstrated as a feasible option to address the above problems. In fact, when applied to solvers for the Dyson equation, for the electronic Green's function and for the polarisation function, these algorithms do not require the use of virtual orbitals nor the explicit inversion of dielectric matrices. In this context we plan to interface YAMBO with the linear response module of QUANTUM ESPRESSO to take advantage of Lanczos-based solvers in the calculation of the screened Coulomb interaction W and possibly of the GW self-energy.

Input from: WP1 (LRlib, FFTXlib, LAXlib, UPF_pseudolib)

Y7 Real-time parallelisation.

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

The time propagation of the equation of motion for a time dependent function is intrinsically serial. This is a serious drawback for simulations using a large number of CPUs, since frequent all-to-all communications are required. However YAMBO does not propagate a simple function but a time dependent matrix. This offers the opportunity to distribute the matrix over different groups of CPUs, thus parallelising the propagation and reducing the communication process inside the CPU subgroups. In this action such approach will be developed to perform NEQ–MBPT simulations on (pre-)exascale machines.

Input from: WP1 (DevXlib);
Output to: WP2 (GPU)

4.5 FLEUR code

Most development effort on the FLEUR code will focus on the implementation of ideas that enable more efficient utilisation of current and future exascale technology. In particular, we will spend some efforts in the redesign of existing functionality in order to achieve performance, scalability and robustness by modifications and improvements of the fundamental algorithms used.

F1 Finishing the restructuring of the Hamiltonian setup into high-level operations

In a standard DFT self-consistency cycle the bulk of the computational effort goes into the matrix-diagonalisation and the Hamiltonian-setup. While this Hamiltonian setup can be written as straight-forward matrix operations, such an implementation does not take advantage of the specific structure of the problem and hence leads to significant inefficiencies. The full-exploitation of the symmetries and the specific structure on the other hand does not allow the use of optimised libraries available for the standard matrix operations and thus also does not provide satisfactory performance. Hence these two approaches have to be combined to keep the possibility to harvest the performance and performance-portability proved by the use of standard operations and libraries with the exploitation of a certain set of symmetries present in the problem. We already made significant advances in this process and will finish the implementation within this project. Besides providing a general performance boost, our effort will also lead to a better utilisation of computing resources in different parallelisation strategies and hence lower the burden on the user to identify an optimal parallel setup. The utilisation of standard libraries also enable us to push the code to the limits of machine performance. The functionality implemented here will constitute a key component of the LAPWlib developed in WP1.

Output to: WP1 (LAPWlib)

F2 Evaluation of the Coulomb kernel in LAPW

The evaluation of the coulomb kernel in LAPW is a key requirement in the efficient implementation of hybrid functionals. This operation is significantly more computational challenging than the tasks of a standard DFT calculation and hence must be optimised carefully. We plan to utilise our knowledge from the construction of the Hamiltonian to adopt a similar strategy for this code and thus proceed

by analysing the existing implementation, identify the possibilities to exploit standard HPC libraries and to keep optimisations exploiting the specific structure of the problem. Similar to the previous action item the work done here will be strongly linked to the library on LAPW operations (LAPWlib) as developed in WP1.

Output to: WP1 (LAPWlib)

F3 Improved charge-density mixing schemes

A key task in any DFT calculation is to achieve a self-consistent solution of the electronic density. This is usually done by relatively simple iteration schemes in which convergence is not guaranteed. Convergence of this iterative approach usually requires the use of quasi-Newton methods for mixing the density. The use of preconditioners can also increase the convergence rate and the stability of the procedure. Within the linearised augmented plane-wave approach implemented in `FLEUR` many of these established preconditioners are difficult to implement due to the complexity of the representation of the charge. Hence, we plan to explore some ideas that are more adopted to our basis set. These developments are crucial for HTC and needed to enable the efficient use of the code on (pre-)exascale machines.

Output to: WP5 (robustness), WP6 (demonstration)

4.6 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 here:
http://spcl.inf.ethz.ch/Publications/.pdf/mmm-tr.pdf.

The following developments have been completed:

- a single-node tiled multiplication of large matrices which outperforms the cublasXt; source code is available here:
  https://github.com/kabicm/Tiled-MM
- a grid-to-grid converter to change matrix distribution from one layout to another, for example, from COSMA to ScaLAPCK or vice versa; source code is available here:
  https://github.com/kabicm/grid2grid

The following actions are planned next:

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

In the current implementation of CP2K ScaLAPACK is used to perform distributed matrix-matrix multiplication. As a first step of incorporating COSMA library into CP2K the ScaLAPACK data layout will be preserved and the following chain of

operations will be executed: transformation of wave-functions from internal CP2K format to ScaLAPACK format → transformation from ScaLAPCK to COSMA layout using grid2grid library → matrix multiplication with COSMA → transformation of the resulting matrix from COSMA layout back to ScaLAPACK layout using grid2grid library.

Input from: WP2 (COSMA)

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

In this stage the further optimisation 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.

Input from: WP2 (COSMA)

C3 Switch to COSMA in RPA calculations

In the final step the 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.

Input from: WP2 (COSMA)

4.7 SIRIUS library

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

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

The optimisers have been successfully tested on few structures where the classical density mixing scheme doesn't converge. We plan the following next steps:

U1 Prototype and implement conjugate gradient method

This is an implementation of direct minimisation for ensemble electronic structure calculations by Baarman *et al.* [24] in which the update operator for the electronic orbitals takes the structure of the Stiefel manifold into account. In this method the optimisation scheme for the occupation numbers ensures that the constraints remain satisfied.

Output to: WP1 (KS_solvers)

U2 Prototype and implement proximal gradient method.

This is an implementation of proximal gradient method for ensemble density functional theory by Ulbrich *et al.* [25] which is suitable for metallic systems.

Output to: WP1 (KS_solvers)

U3 Interface advanced density optimisers with QUANTUM ESPRESSO and CP2K

Once the wave-function optimisers are prototyped and proved to be working they will be implemented in a highly-efficient way directly in the SIRIUS library. This will allow QUANTUM ESPRESSO and CP2K codes to call the optimisers from SIRIUS (we consider direct or reverse communication implementations) and find the ground states of systems which can't be converged using standard mixing techniques.

Output to: WP1 (KS_solvers)

5 Summary of the activities: Staging, timeline, link to WPs

We present here a summary of the identified actions (algorithmic and feature developments) of this WorkPackage, with the aim of helping the reader to browse through the detailed description of the activities. In Fig. 2, we present the expected timeline of the actions of the WorkPackage in the form of a Gantt chart.

T3.1 Software-related fault resilience algorithms and solutions.

Q2 Direct minimisation
Y2 Mixed precision algorithms
B1 Approaches for Input-files
S4 Automatic solver dispatch
Q1 Improved diagonalisation algorithms
B2 Mixed-precision techniques
Y1 Restart structure, parallel IO for BSE and QP
F3 Improved charge-density mixing schemes

T3.2 Enabling new code functionalities with an exascale mindset

S2 Exact-exchange
Y4 Full-frequency GW
Y5 Advanced self-energies from MBPT
F2 Coulomb kernel in LAPW
Q3 RPA-based XC functionals
Y3 Real–time propagation with atomic motion and interface with QE
C2 Transform CP2K matrices to COSMA layout
C3 Switch to COSMA in RPA calculations

T3.3 Exploitation of new algorithms at the pre-exascale

B4 Exact exchange for $\mathcal{O}(N)$
Q4 Localised inner-projection method for EXX
S1 Basis-set contraction
B3 Fragment approaches for extended systems
S3 Improved algorithms for SCF
S5 Re-design $\mathcal{O}(N)$ solver
Q5 Adaptive parallelisation schemes
Y6 YAMBO without empty states
Y7 Real time parallelisation
F1 Restructuring of the Hamiltonian
C1 Integrate COSMA library into CP2K

	T3.1	**T3.2**	**T3.3**
Stage 1	Q2, Y2	S2, Y4	B4, Q4
Stage 2	B1, S4, Q1	Y5, F2	S1, B3
Stage 3	B2, Y1, F3	Q3, Y3, C2, C3	S3, S5, Q5, Y6, Y7, F1, C1, U1, U2, U3

Figure 1: Table of the proposed WP3 actions classified according to the Stage and Tasks. Actions of increasing difficulty are associated to higher values of Stages, whereas increasing complexity correspond to tasks of higher number.

U1 Conjugate gradient method

U2 Proximal gradient method

U3 Advanced density optimisers with QUANTUM ESPRESSO and CP2K.

Exploitation of new algorithms at the pre-exascale

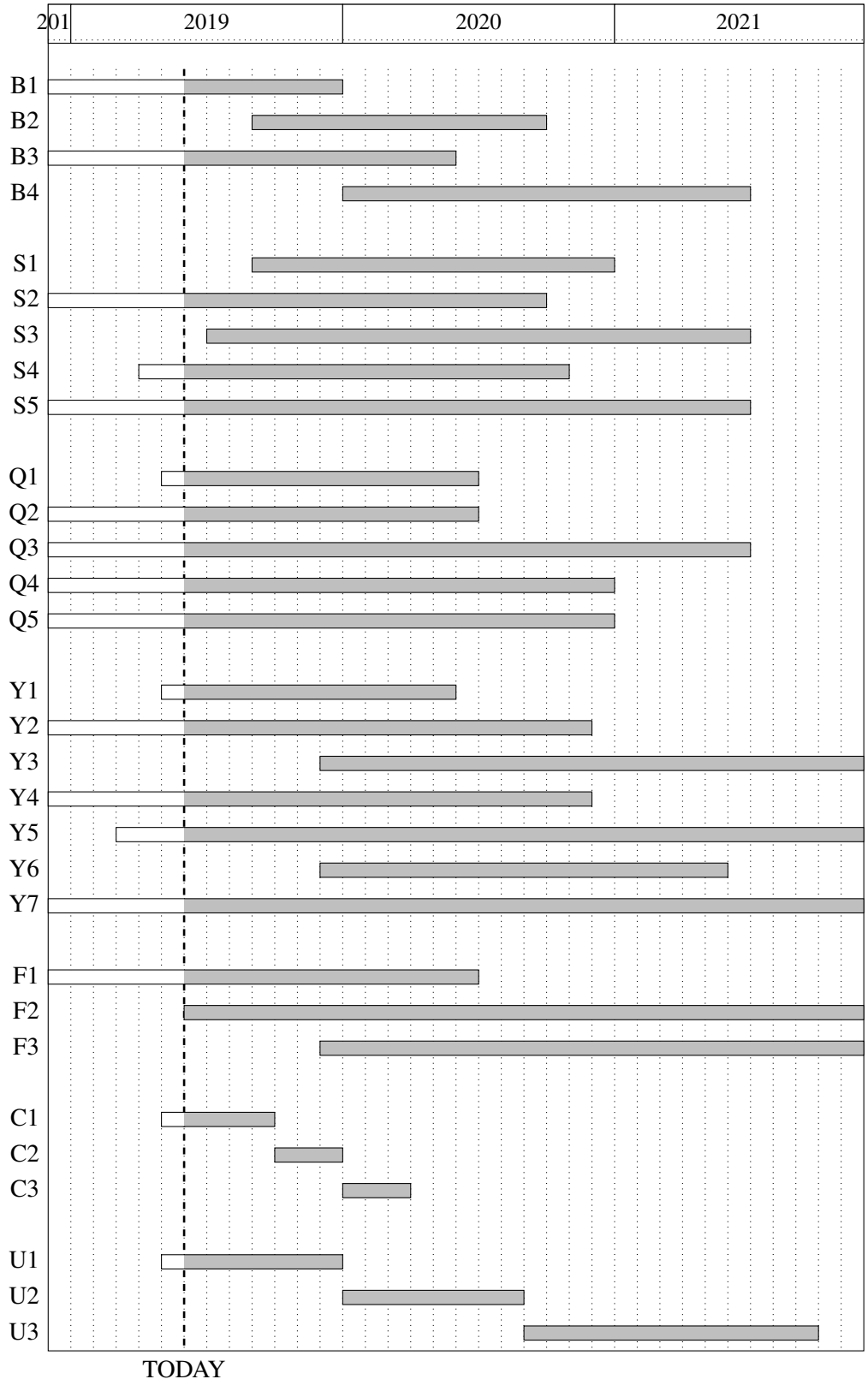

Figure 2: Gantt diagram.

6 Conclusions and Outlook

In this document we have presented the organisation of the activities of WP3, that are related to the development of algorithmic advances within the codes of the MAX consortium. Importantly, for each code, a list of algorithmic developments relevant to the MAX work plan has been first identified and then presented in detail. Concerning the presentation of WP3 work plan, we have first introduced the description of the Work-Package Tasks, devoted to different classes of algorithms, such as *Software-related fault resilience algorithms and solutions* (T3.1), *Enabling new code functionalities with an exascale mindset* (T3.2), and *Exploitation of new algorithms at the pre-exascale* (T3.3). We have also categorised the different development actions by readiness stages, that identify the level of complexity (including dependencies and pre-requisites from other WP actions) of each algorithmic/feature development for the reference host code. This analysis, together with the constraints of distributing the personnel effort as evenly as possible, has led to the definition of a schedule and a timeline for each development action, summarised in a Gantt chart.

Notably, most of the development actions of WP3 have connections and dependencies on other WorkPackage work. We have carefully analysed them and sketched, for each algorithmic advance of WP3, the main input and output streams to/from other WPs, as detailed in Sec. 3. This analysis was also quite important to define the actual schedule of the activities of WP3. Last, the Key Performance Indicators that are associated to WP3 take into account the performance opportunities that are leveraged by this WorkPackage. We plan to include the most important achievements in the forthcoming deliverables.

Acronyms

DFT Density Functional Theory. 14

MBPT Many Body Perturbation Theory. 14

RPA Random Phase Approximation. 13

References

[1] Ratcliff, L. E., Degomme, A., Flores-Livas, J. A., Goedecker, S. & Genovese, L. Affordable and accurate large-scale hybrid-functional calculations on GPU-accelerated supercomputers. *J. Phys.: Condens. Matter* **30**, 095901 (2018).

[2] Lin, L., Lu, J., Ying, L. & E, W. Adaptive local basis set for kohn–sham density functional theory in a discontinuous galerkin framework i: Total energy calculation. *J. Comput. Phys.* **231**, 2140 – 2154 (2012).

[3] Lin, L. Adaptively compressed exchange operator. *J. Chem. Theory Comput.* **12**, 2242–2249 (2016).

[4] 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).

[5] 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).

[6] Carnimeo, I., Baroni, S. & Giannozzi, P. Fast hybrid density-functional computations using plane-wave basis sets. *Electronic Structure* **1**, 015009 (2019).

[7] Wang, Z., Li, S.-S. & Wang, L.-W. Efficient real-time time-dependent density functional theory method and its application to a collision of an ion with a 2d material. *Phys. Rev. Lett.* **114**, 063004 (2015).

[8] Marini, A., Onida, G. & Del Sole, R. Quasiparticle electronic structure of copper in the GW approximation. *Phys. Rev. Lett.* **88**, 016403 (2001).

[9] Rojas, H. N., Godby, R. W. & Needs, R. J. Ab initio calculations of self-energies and dielectric response functions of solids. *Phys. Rev. Lett.* **74**, 1827–1830 (1995).

[10] Rieger, M. M., Steinbeck, L., White, I. D., Rojas, H. N. & Godby, R. W. The gw space-time method for the self-energy of large systems. *Comput. Phys. Commun.* **117**, 211–228 (1999).

[11] Ren, X. *et al.* Resolution-of-identity approach to Hartree–Fock, hybrid density functionals, RPA, MP2 and GW with numeric atom-centered orbital basis functions. *New J. Phys.* **14**, 053020 (2012).

[12] Fleszar, A. & Hanke, W. Spectral properties of quasiparticles in a semiconductor. *Phys. Rev. B* **56**, 10228–10232 (1997).

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

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

[15] Grüneis, A., Marsman, M., Harl, J., Schimka, L. & Kresse, G. Making the random phase approximation to electronic correlation accurate. *J. Chem. Phys.* **131**, 154115 (2009).

[16] Ren, X., Marom, N., Caruso, F., Scheffler, M. & Rinke, P. Beyond the gw approximation: A second-order screened exchange correction. *Phys. Rev. B* **92**, 081104 (2015).

[17] Rocca, D., Gebauer, R., Saad, Y. & Baroni, S. Turbo charging time-dependent density-functional theory with Lanczos chains. *J. Chem. Phys.* **128**, 154105 (2008).

[18] Umari, P., Stenuit, G. & Baroni, S. Gw quasiparticle spectra from occupied states only. *Phys. Rev. B* **81**, 115104 (2010).

[19] Rocca, D., Lu, D. & Galli, G. Ab initio calculations of optical absorption spectra: Solution of the bethe–salpeter equation within density matrix perturbation theory. *J. Chem. Phys.* **133**, 164109 (2010).

[20] Giustino, F., Cohen, M. L. & Louie, S. G. Gw method with the self-consistent sternheimer equation. *Phys. Rev. B* **81**, 115105 (2010).

[21] Pham, T. A., Nguyen, H.-V., Rocca, D. & Galli, G. gw calculations using the spectral decomposition of the dielectric matrix: Verification, validation, and comparison of methods. *Phys. Rev. B* **87**, 155148 (2013).

[22] Ge, X., Binnie, S. J., Rocca, D., Gebauer, R. & Baroni, S. turbotddft 2.0—hybrid functionals and new algorithms within time-dependent density-functional perturbation theory. *Comput. Phys. Commun.* **185**, 2080 – 2089 (2014).

[23] Govoni, M. & Galli, G. Large scale gw calculations. *J. Chem. Theory Comput.* **11**, 2680–2696 (2015).

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

[25] 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).