Electronic structure E-CAM modules I

E-CAM Deliverable 2.1
Deliverable Type: Report
Delivered in Month 13– October2016

E-CAM
The European Centre of Excellence for
Software, Training and Consultancy
in Simulation and Modelling

Funded by the European Union under grant agreement 676531

Project and Deliverable Information

Project Title	E-CAM: An e-infrastructure for software, training and discussion in simulation and modelling
Project Ref.	Grant Agreement 676531
Project Website	https://www.e-cam2020.eu
EC Project Officer	Juan PELEGRIN
Deliverable ID	D2.1
Deliverable Nature	Report
Dissemination Level	Public
Contractual Date of Delivery	Project Month 13(October2016)
Resubmission Date	04.09.2017
Deliverable Description	9 Software modules delivered to the E-CAM library in electronic structure including solvers for localised orbitals, for computing on a grid and solvers, and for transport, and their documentation.

Document Control Information

Document	Title:	Electronic structure E-CAM modules I
	ID:	D2.1
	Version:	As of September 4, 2017
	Status:	Accepted by WP leader
	Available at:	https://www.e-cam2020.eu/deliverables
	Document history:	Internal Project Management Link
Review	Review Status:	Reviewed
Authorship	Written by:	Liang LIANG(Maison de la Simulation, France)
	Contributors:	Micael Oliveira (Max Planck Institute for the Structure and Dynamics of Matter), Fabiano Corsetti (Imperial College London), Yann Pouillon (University of the Basque Country), Catarina Mendonça (EPFL)
	Reviewed by:	Alan O'Cais (JSC), Sara Bonella (EPFL), Ignacio Pagonabarraga (EPFL)
	Approved by:	Mike Payne (University of Cambridge)

Document Keywords

Keywords:	E-CAM, Electronic Structure, Module, Solvers, Localised Orbitals, CECAM, Documentation, Data Format...

September 4, 2017

***Disclaimer**:This deliverable has been prepared by the responsible Work Package of the Project in accordance with the Consortium Agreement and the Grant Agreement. It solely reflects the opinion of the parties to such agreements on a collective basis in the context of the Project and to the extent foreseen in such agreements.*

[1] liang.liang@idris.fr

Contents

Executive Summary

In this report for Deliverable 2.1 of E-CAM, 9 software modules in electronic structure, which are related to an Extended Software Development Workshop ESDW held by E-CAM in Zaragoza in June 2016, are presented.

These modules are libraries intended to be leveraged during the development process of new applications that span most of the electronic structure community, but the main focus are codes that implement Density Functional Theory, which can be used to determine a wide range of properties of atoms, molecules, and materials.

9 modules are respectively named: LibOMM, `MatrixSwitch`, Libpspio, Libescdf, `Poke`, SQARE radial grid & function, SQARE ODE solvers, SQARE states and FDF. These include codes for solvers of localised orbitals, for computing on a grid and their documentation. In addition to the solvers, data format modules, a pseudopotential data file I/O operation module and intermediary interface layer module are also present in this document.

A short description is written for each module as well as the codes where they have been implemented, followed by a link to the respective Merge-Request on the GitLab service of E-CAM. These merge requests contain detailed information about the code development, testing and documentation of the modules.

1 Introduction

Work Package 2 of E-CAM focuses on selecting software functionalities that are common to many electronic structure implementations, important for the coding and efficiency of codes, and mature enough to allow for a good definition of standards and interfaces. E-CAM collaborates with the Electronic Structure Library Project (ESL) whose goal is to build a community-maintained library of software of use for electronic structure simulations.

Starting from an E-CAM Extended Software Development Workshop (ESDW) in Zaragoza organized by the ESL developers team, the development of new libraries revolved around the broad theme of solvers. A solver is a piece of mathematical software, possibly in the form of a stand-alone computer program or as a software library, that 'solves' a mathematical problem. A solver takes problem descriptions in some sort of generic form and calculates their solution.

The work being done is on three specific library bundles, all relating to crucial parts of any density-functional theory code:

- Kohn-Sham eigensolvers: the ESL is coordinating efforts in this area, also through the new "Electronic Structure Infrastructure" (ELSI) project and by connecting U.S. based researchers to the wider ESL effort. ELSI will bring together several different approaches to solving or circumventing the Kohn-Sham eigenvalue problem, including a density matrix solver (`LibOMM`[2], a module developed during the ESDW in Zaragoza), a resolvent-based method (PEXSI), and a scalable diagonalization method (ELPA), and will be open to integrating other open source solvers that are compatible and/or complementary;
- Poisson solvers: similarly to the eigensolvers, the aim is to implement in a single package several different algorithms of use in different situations, providing a unified and clean interface for the user. Special attention goes to allowing different Fast Fourier Transform (FFT) back ends to be connected to the library. The `Poke`[3] module was developed during the ESDW;
- Atomic solvers: this is one of the most notable examples of replicated development in different code bases, with no modern independent library available. The aim here is to create a new library with a modern, coherent list of features, ultimately targeting spherical Hartree-Fock, semi-local and hybrid Density Functional Theory (DFT), and different optional relativistic levels of theory. It will also be connected to LibXC, a library of exchange-correlation functionals for density-functional theory, for immediate access to many different exchange-correlation functionals (and, in doing so, will avoid replicated development). Three Solvers for Quantum Atomic Radial Equations (SQARE) modules related to Atomic solvers were developed during the ESDW in Zaragoza.

In addition to the solvers, work was also done on maintaining existing ESL developments, such as: the development of the Electronic Structure Common Data Format Library (Libescdf), of the Flexible Data Format (FDF), of the Library for PSeudoPotential data files I/O operations (Libpspio) and of the intermediary interface layer module (`MatrixSwitch`[4]) between routines of different levels.

The module developments are done either on E-CAM's `GitLab` server (for 8 of the 9 modules) or on `Github` (for the FDF module). Both the GitLab and GitHub use powerful version control system (Git) service for software development, but E-CAM's own GitLab Server allows more control of software development, for example, the private hosting of the code.

In order to describe each module, a module documentation repository is created in an E-CAM GitLab repository dedicated to Electronic Structure which automatically updates documentation for the E-CAM Electronic Structure Module Documentation on Read the Docs. To submit a module, a developer can fork the GitLab repository and create his own feature branch in which he documents his latest progress about module development, source code link/patch file, new feature commit, etc. The developer can ask to merge their branch into the E-CAM master branch, which is called the `Merge-Request`. The E-CAM software manager and developers can then run the module through it's acceptance criteria (coding style and structure, source code documentation and module testing). The documentation will only appear as part of the Work Package 2 Software Modules on ReadTheDocs.org once the `Merge-Request` is accepted.

In section 2, a short description is provided for each module created, followed by a link to the `Merge-Request` on GitLab of E-CAM. More detailed information about code development, testing and documentation can be found there. Additionally, we also point to the DFT codes already interfaced with some of these modules.

[2] Orbital Minimization Method Library

[3] Solver for the Poisson equation designed for electronic structure codes

[4] Intermediary interface layer module between high-level routines for physics-related algorithms and low-level routines dealing with matrix storage and manipulation

1.1 Overall scope of the module set

The intelligent design and exploitation of materials for technological applications, or of new drugs, relies on our ability to describe and manipulate matter at a microscopic level. To do this, it is essential to know how atoms interact to form molecules and more complex materials. Interactions can be described empirically by creating models that reproduce known macroscopic properties of the material (e.g. melting temperature). They can also, in principle, be computed by solving the microscopic equation that describes the physics of interacting electron and nuclei in atomic systems. This equation, known as the time-independent Schroedinger equation, is one of the cornerstones of quantum mechanics. It is unfortunately too complex to solve, in general, both analytically and numerically. Several approximate methods then exist to tackle the problem. Currently, the best compromise between efficiency and accuracy is provided by a framework known as Density Functional Theory (DFT), which has been successfully applied to determine a wide range of properties of atoms, molecules, and complex materials. The software modules developed in the electronic structure work package in E-CAM, whose specific functionality is discussed below, tackle specific ingredients necessary for DFT calculations of the interactions. They have been conceived to be transferable (i.e. they can be incorporated in many of the codes in use) and scalable (i.e. they can be used on computers at the highest end of current technologies), and are available to the entire community of practitioners in the field.

1.2 General applications and possible exploitation of the codes

All the modules described in this deliverable are not stand-alone programs, but are instead intended to be used from within other codes. Therefore, the users for these modules can be broadly divided into two groups: end users and code developers. The end users of codes that are interfaced with these module will benefit from the functionalities they provide, while the developers of those codes will save time and resources, as the implementation and maintenance of these functionalities will be greatly simplified.

The codes that can benefit from these modules' functionalities span most of the electronic structure community, but the main focus are codes that implement Density Functional Theory, which can be used to determine a wide range of properties of atoms, molecules, and materials. Several popular electronic structure codes are already interfaced with some of the modules described in this deliverable. The ones we are aware of, in no particular order, are:

- Discontinuous Galerkin Method for Density Functional Theory DGDFT, which goal is to develop and implement a new discontinuous approach to quantum mechanical materials calculations on an unprecedented scale, and to reach for the first time the length and time scales necessary to accurately model solid-electrolyte interfaces in lithium-ion batteries, thus paving the way for breakthroughs in understanding, device performance, and safety [3];
- Fritz Haber Institute ab initio molecular simulations package FHI-aims, an all-electron electronic structure code based on numeric atom-centered orbitals to capture a wide range of molecular and materials properties from quantum-mechanical first principles[2];
- High-Performance Computational Chemistry software package NWChem, that aims to provide its users with computational chemistry tools that are scalable both in their ability to treat large scientific computational chemistry problems efficiently, and in their use of available parallel computing resources from high-performance parallel supercomputers to conventional workstation clusters[5];
- SIESTA, the Spanish Initiative for Electronic Simulations with Thousands of Atoms, which is both a method and its computer program implementation, to perform electronic structure calculations and ab initio molecular dynamics simulations of molecules and solids, using standard norm-conserving pseudo-potentials and a flexible, numerical linear combination of atomic orbitals basis set, which includes multiple-zeta and polarization orbitals [4];
- Octopus, a pseudo-potential real-space package aimed at the simulation of the electron-ion dynamics of one-, two-, and three-dimensional finite systems subject to time-dependent electromagnetic fields, with applications on the determination of linear and non-linear absorption spectra, harmonic spectra, laser induced fragmentation, etc. of a variety of systems[1];
- Atomic Pseudopotentials Engine APE, a density functional theory atomic program and pseudo-potentials generator, that produces pseudo-potential files suitable for use with DFT codes such as SIESTA and OCTOPUS.

By performing tasks common to many DFT codes, these modules allow scientists to focus on developing new ideas, and new science can be coded without needing to rewrite functionalities that are already well-established, and without needing to know more software engineering than science. In other words, these modules allow to separate the coding effort for cutting-edge research from the software infrastructure it rests on top of, which needs maintaining and rewriting at every step of the hardware race.

2 Modules

In this section, a short description is written for each module, followed by a link to the `Merge-Request` on GitLab service of E-CAM, which shows detailed information about code development, testing and documentation.

2.1 LibOMM

LibOMM solves the Kohn-Sham equation as a generalized eigenvalue problem for a fixed Hamiltonian. It implements the orbital minimization method (OMM), which works within a density matrix formalism. The basic strategy of the OMM is to find the set of Wannier functions (WFs) describing the occupied subspace by direct unconstrained minimization of an appropriately-constructed functional. The density matrix can then be calculated from the WFs. The solver is usually employed within an outer self-consistency (SCF) cycle. Therefore, the WFs resulting from one SCF iteration can be saved and then re-used as the initial guess for the next iteration.

2.1.1 Practical application and exploitation of the code

This module is accessible through the Electronic Structure Infrastructure ELSI, which in turn is interfaced with the DGDFT, FHI-aims, NWChem, and SIESTA codes.

ELSI provides and enhances scalable, open-source software library solutions for electronic structure calculations in materials science, condensed matter physics, chemistry, molecular biochemistry, and many other fields [6]. LibOMM is one of the libraries supported and enhanced in ELSI.

Merge Request	Merge-Request of LibOMM module.
Direct Documentation Link	readme.rst of LibOMM module.

2.2 MatrixSwitch

`MatrixSwitch` is a module which acts as an intermediary interface layer between high-level routines for physics-related algorithms and low-level routines dealing with matrix storage and manipulation. This allows the high-level routines to be written in a way which is physically transparent, and enables them to switch seamlessly between different software implementations of the matrix operations. It is a software library and module to be used within a calling code. It is developed within the same repository project as other libraries (see source code for more details), but all are self-contained within separate directories.

2.2.1 Practical application and exploitation of the code

This module is accessible through ELSI, which in turn is interfaced with the DGDFT, FHI-aims, NWChem, and SIESTA codes.

Merge Request	Merge-Request of MatrixSwitch module.
Direct Documentation Link	readme.rst of MatrixSwitch module.

2.3 Poke

`Poke` is a solver for the Poisson equation designed for electronic structure codes. Similarly to the eigensolvers, the aim is be to implement in a single package several different algorithms of use in different situations, providing a unified and clean interface for the user. Special attention goes to allowing different FFT back ends to be connected to the library, the `Poke` module was developed during the ESDW.

2.3.1 Practical application and exploitation of the code

This module has been interfaced with the Octopus code.

Merge Request	Poke module merge request
Direct Documentation Link	readme.rst of Poke module.

2.4 SQARE modules

SQARE is a library of utilities intended for dealing with functions discretized on radial meshes, wave-equations with spherical symmetry and their corresponding quantum states. The utilities are segregated into three levels: radial grids & functions, ODE solvers, and states.

2.4.1 SQARE radial grid & function

This module provides functions and structures to create radial meshes and define discretized radial functions on those meshes.

Merge Request	Merge-Request of SQARE radial & grid module.
Direct Documentation Link	readme.rst of SQARE radial grid & function module.

2.4.2 SQARE ODE solvers

This module provides functions and structures to solve ordinary differential equations on a radial mesh.

Merge Request	Merge-Request of SQARE ODE solvers module.
Direct Documentation Link	readme.rst of SQARE ODE solvers module.

2.4.3 SQARE states

This module provides functions and structures to solve radial wave-equations in various flavors and obtain the corresponding eigenstates.

Merge Request	Merge-Request of SQARE states module.
Direct Documentation Link	readme.rst of SQARE states module.

2.5 Libescdf

Libescdf is a library containing tools for reading and writing massive data structures related to electronic structure calculations, following the standards defined in the Electronic Structure Common Data Format. It is a library created to exchange electronic-structure-related data in a platform-independent and efficient manner. It is based on the Electronic Structure Common Data Format Specifications, as well as HDF5.

Merge Request	Merge-Request of Libescdf module.
Direct Documentation Link	readme.rst of Libescdf module.

2.6 FDF

FDF(Flexible Data Format) is an input file parser that offers an easy, transferable and practical way for a Fortran program to read its input. It is text (ASCII) based, and conceived for small data (input parameters). Every input piece of data is introduced in a line of an input file (which can be standard input) by writing a name-value pair, that is, a name characterising the data, and its value. If the latter corresponds to a physical magnitude, the units can also be specified after the value. Names can be long and should be descriptive of the value it corresponds to. FDF blocks are used to input structured data, in which case, the program using FDF reads the inside of the block. From the programming point of view, FDF allows for any data to be retrieved whenever, from any part of the code, and in any order. If a piece of data sought by FDF is not found in the input file, FDF will return a default value, as set up in the call to the FDF routine.

2.6.1 Practical application and exploitation of the code

This module has been interfaced with the SIESTA code.

Merge Request	Merge-Request of FDF module.
Direct Documentation Link	readme.rst of FDF module.

2.7 Libpspio

Libpspio is a pseudopotentials I/O library for Density-Functional Theory (DFT) calculations. It can both read and write pseudopotential data, which makes it suitable for use with pseudopotential generators and electronic structure codes. The main objective of Libpspio is to let any DFT code access or produce pseudopotential information without having to care about file formats. Libpspio is a valuable alternative to most error-prone homemade implementations and is helpful in improving file format specifications.

2.7.1 Practical application and exploitation of the code

This module has been interfaced with the Octopus and APE codes.

Merge Request	Merge-Request of Libpspio module.
Direct Documentation Link	readme.rst of Libpspio module.

3 Outlook

The report of Deliverable 2.1 of E-CAM describes 9 Software modules in electronic structure related to the ESDW Zaragoza. They include codes for solvers for localised orbitals, for computing on a grid, and their documentation, as well as data format modules, a pseudopotential data file I/O operation module and an intermediary interface layer module.

Solvers for transport were not completed during the ESDW, due to the absence of expert in this subject at the meeting. Future efforts will be made in this direction.

At the time of writing this report, not all modules had been merged into the main E-CAM repository as they hadn't completed the acceptance workflow established within E-CAM. Over the coming months these modules will be available in E-CAM Electronic Structure Library GitLab page.

References

Acronyms Used

CECAM Centre Européen de Calcul Atomique et Moléculaire

DFT Density Functional Theory

ESL Electronic Structure Library Project

ESDW Extended Software Development Workshop

FFT Fast Fourier Transform

LibOMM Orbital Minimization Method Library

Libpspio Library for PSeudoPotential data files I/O operations

Libescdf Electronic Structure Common Data Format Library

SQARE Solvers for Quantum Atomic Radial Equations

FDF Flexible Data Format

URLs referenced

Page ii

https://www.e-cam2020.eu ... https://www.e-cam2020.eu
https://www.e-cam2020.eu/deliverables ... https://www.e-cam2020.eu/deliverables
Internal Project Management Link ... https://redmine.e-cam2020.eu/issues/173
liang.liang@idris.fr ... mailto:liang.liang@idris.fr
http://creativecommons.org/licenses/by/4.0 ... http://creativecommons.org/licenses/by/4.0

Page 2

ESL ... http://esl.cecam.org
ELSI ... http://elsi-interchange.org
GitLab ... https://gitlab.e-cam2020.eu/
module documentation repository ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules
E-CAM GitLab repository dedicated to Electronic Structure ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Struc
E-CAM Electronic Structure Module Documentation ... https://readthedocs.org/projects/e-cam-electronic-stru
Read the Docs ... https://readthedocs.org/
Work Package 2 Software Modules ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/
ReadTheDocs.org ... http://readthedocs.org

Page 3

DGDFT ... http://www.dgdft-scidac.org/
FHI-aims ... https://aimsclub.fhi-berlin.mpg.de/
NWChem ... http://www.nwchem-sw.org/
SIESTA ... https://departments.icmab.es/leem/siesta/
Octopus ... http://octopus-code.org
APE ... https://gitlab.com/ape/ape

Page 4

GitLab service of E-CAM ... https://gitlab.e-cam2020.eu/
ELSI ... http://elsi-interchange.org
DGDFT ... http://aip.scitation.org/doi/full/10.1063/1.4931732
FHI-aims ... https://aimsclub.fhi-berlin.mpg.de/
NWChem ... http://www.nwchem-sw.org/
SIESTA ... https://departments.icmab.es/leem/siesta/
ELSI ... http://elsi-interchange.org
Merge-Request of LibOMM module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/1
readme.rst of LibOMM module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/libOMM/readme.html

ELSI ... http://elsi-interchange.org
DGDFT ... http://aip.scitation.org/doi/full/10.1063/1.4931732
FHI-aims ... https://aimsclub.fhi-berlin.mpg.de/
NWChem ... http://www.nwchem-sw.org/
SIESTA ... https://departments.icmab.es/leem/siesta/
Merge-Request of MatrixSwitch module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/2
readme.rst of MatrixSwitch module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/MatrixSwitch/readme.html
Octopus ... http://octopus-code.org

Page 5

Poke module merge request ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/5
readme.rst of Poke module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/poke/readme.html
Merge-Request of SQARE radial & grid module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modu
merge_requests/6
readme.rst of SQARE radial grid & function module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/sqare-grids-doc/readme.html
Merge-Request of SQARE ODE solvers module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modu
merge_requests/7
readme.rst of SQARE ODE solvers module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/sqare-ode-doc/readme.html
Merge-Request of SQARE states module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/8
readme.rst of SQARE states module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/sqare-states-doc/readme.html
Merge-Request of Libescdf module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/4
readme.rst of Libescdf module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/escdf/readme.html

Page 6

SIESTA ... https://departments.icmab.es/leem/siesta/
Merge-Request of FDF module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/9
readme.rst of FDF module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/FDF/readme.html
Octopus ... http://octopus-code.org
APE ... https://gitlab.com/ape/ape
Merge-Request of Libpspio module. ... https://gitlab.e-cam2020.eu/E-CAM/Electronic-Structure-Modules/merge_requests/3
readme.rst of Libpspio module. ... http://e-cam-electronic-structure-modules.readthedocs.io/en/latest/modules/pspio/readme.html

Page 7

Electronic Structure Library GitLab page ... https://gitlab.e-cam2020.eu/esl

—

[1] Xavier Andrade, David Strubbe, Umberto De Giovannini, Ask Hjorth Larsen, Micael J. T. Oliveira, Joseba Alberdi-Rodriguez, Alejandro Varas, Iris Theophilou, Nicole Helbig, Matthieu J. Verstraete, Lorenzo Stella, Fernando Nogueira, Alan Aspuru-Guzik, Alberto Castro, Miguel A. L. Marques, and Angel Rubio. Real-space grids and the octopus code as tools for the development of new simulation approaches for electronic systems. *Phys. Chem. Chem. Phys.*, 17:31371–31396, 2015.

[2] Volker Blum, Ralf Gehrke, Felix Hanke, Paula Havu, Ville Havu, Xinguo Ren, Karsten Reuter, and Matthias Scheffler. Ab initio molecular simulations with numeric atom-centered orbitals. *Computer Physics Communications*, 180(11):2175 – 2196, 2009.

[3] Wei Hu, Lin Lin, and Chao Yang. Dgdft: A massively parallel method for large scale density functional theory calculations. *The Journal of Chemical Physics*, 143(12):124110, 2015.

[4] José M Soler, Emilio Artacho, Julian D Gale, Alberto García, Javier Junquera, Pablo Ordejón, and Daniel Sánchez-Portal. The siesta method for ab initio order- n materials simulation. *Journal of Physics: Condensed Matter*, 14(11):2745, 2002.

[5] M. Valiev, E.J. Bylaska, N. Govind, K. Kowalski, T.P. Straatsma, H.J.J. Van Dam, D. Wang, J. Nieplocha, E. Apra, T.L. Windus, and W.A. de Jong. Nwchem: A comprehensive and scalable open-source solution for large scale molecular simulations. *Computer Physics Communications*, 181(9):1477 – 1489, 2010.

[6] Alberto García William P. Huhn Mathias Jacquelin Weile Jia Björn Lange Lin Lin Jianfeng Lu Wenhui Mi Ali Seifitokaldani Álvaro Vázquez-Mayagoitia Chao Yang Haizhao Yang Victor Wen-zhe Yu, Fabiano Corsetti and Volker Blum. Elsi: A unified software interface for kohn-sham electronic structure solvers. *submitted*, 2017. https://arxiv.org/abs/1705.11191v1.