Open-source landscape for 3D CSEM modelling

Dieter Werthmüller[*], *Raphael Rochlitz*[†], *Octavio Castillo-Reyes*[‡], *and Lindsey Heagy*[§]

ABSTRACT

Large-scale modelling of three-dimensional controlled-source electromagnetic (CSEM) responses used to be feasible only for large companies and research consortia. However, this has changed over the last few years, and today there exists a selection of different open-source codes which are up to the task. We have a closer look at four different ones, where in all cases at least the user-facing functionality is written in Python. For the numerical comparison we use three models: a simple layered model; a model containing three resistive blocks in a layered background; and a complex, real-world resistivity model. All three models are vertically transverse isotropic. The analysis includes three different mesh-types, simple tensor meshes, octree meshes, and unstructured tetrahedral meshes. Two of the codes use finite volumes and the other two use finite elements. The comparison shows that all four codes are able to compute the different models to sufficient precision. The biggest difference between codes is actually the discretization and the ease of creating a suitable mesh, particularly as that part often takes much more time than the actual computation. The required runtime and memory varies greatly between the codes. However, these depend mainly on the underlying solver, and not on the actual CSEM codes. The collaboration of four code maintainer trying to achieve the same task brought in the end all four codes a significant step further. We hope that these results may be useful for the entire CSEM community at large, and we invite and encourage the community to make more code, modelling scripts, and results publicly available. All our scripts to reproduce the shown results are open-source.

⇓ ⇓ List of things ToDo, Notes, etc ⇓ ⇓

- **This version is quite different from the previous you revised. However, from now on we do not expect too many changes. To make it easier for everyone to see the new changes please make sure that from now on "Track Changes" is always on in Overleaf.**
- Notes on orthography: I settled on discretize with a «z». My dictionary says that discretize is AE and discretise is BE, but that «z» is also used in BE in mathematical contexts. Is that fine or any other opinions?
- We probably should be careful to not mention the actual code names too often. E.g., I removed them from the abstract.
- I tried to avoid double reference. E.g., SciPy, MUMPS, pyGIMLi: I referenced them the first time they are mentioned, and not afterwards. Current exceptions: Miensopust et al., 2013, is cited twice.
- Regarding submission, **please let me know**:
  - If you think the **order of authors** should be different. The last thing we want in this collaboration is that someone feels unhappy about that.
  - If you have **another manuscript under revision with GJI**. When submitting to GJI one has to indicate when you have other manuscripts currently under review with GJI.
  - For the submission I will create a **new repository on swung-research**. The idea is to make a single, new commit with everything (without the git history, for various reasons). Please let me know if you would prefer to commit your files yourself as a PR instead.
  - **Wanted and not wanted reviewers:** Let me know if you have any names for these. I do not have any *not wanted*, but I do have a few I will put on the *wanted* list, mainly people I would be interested to have their feedback and that have knowledge about 3D modelling:
    * Kerry Key (domain knowledge with `MARE3DEM`, also experience with open-sourcing with `MARE2DEM`);
    * Rune Mittet (domain knowledge; he will know what SBLwiz actually does, so it would be very interesting to have his feedback);
    * Dimitry Avdeev and Ralph-Uwe Börner (they both did extensive 3D code overview/review papers in 2005 and 2010, respectively, and it would be interesting to get their feedback).
- **Everyone: Please ensure that the tables have the correct info:**
  1. machine info;
  2. runtime;
  3. memory;
  4. nproc;

5. dof.

I got all the info out of the ".nc" files, so make sure you wrote the correct info into the nc-files. Further notes:

runtime Should only include the wall time (or elapsed real time, wikipedia.org/-wiki/Elapsed_real_time) of solving the SLE, for ALL frequencies (in `emg3d` some frequencies are slower than others, but in any case, we report the sum of all of them); any pre-/postprocessing such as mesh generation or model interpolation can be ignored.

memory It should be the maximum memory increase when solving the SLE (again, ignoring any pre-/postprocessing). Again, of all frequencies. In `emg3d` some frequencies require more RAM than others (more iterations), but we only report the "worst", the heaviest consumer.

nproc Not sure we reached agreement on Slack, but I took no everything in here (threads and (virtual) processes); please note here if you think it should be different.

dof Similarly, not sure we reached agreement on Slack. I put it here now for complex value, hence halving the dof of custEM. Again, please write here if we should change it.

We could discuss if we should report CPU time instead (hence summing all parallel processes); but I think for the user of interest is the real-world time, how long do I have to wait, so I think it is good as we do it now. (Either way, the reader can make a quick estimation in one or the other direction; important is that we are consistent throughout the paper.)

⇑ ⇑ List of things ToDo, Notes, etc ⇑ ⇑

INTRODUCTION

Controlled-source electromagnetic (CSEM) measurements are a frequently applied method in various geophysical exploration fields, such as geothermal, groundwater, oil and gas, mining, civil engineering, or geo-hazards. Modelling these electromagnetic fields is therefore of great interest to design survey layouts, to understand the measured data, and for inversion purposes. Publications regarding three-dimensional (3D) modelling in electromagnetic methods started to appear as early as the 1970's and 1980's. These early publications were integral equation (IE) methods, having an anomaly embedded within a layered medium, mostly for loop-loop type transient EM measurements (Raiche, 1974; Hohmann, 1975; Das and Verma, 1982; Newman et al., 1986) and magnetotelluric (MT) measurements (Wannamaker et al., 1984). Ward and Hohmann (1988) assemble many of these solutions in *Electromagnetic Theory for Geophysical Applications*, which is widely viewed as an authoritative publication on electromagnetic geophysics.

In the 1990's, computer became sufficiently powerful that 3D modelling gained traction, and in 1995 the first International Symposium on Three-Dimensional Electromagnetics took place. This symposium resulted eventually in the book *Three-Dimensional Electromagnetics* by the SEG (Oristaglio and Spies, 1999), and another book by Wannamaker and Zhdanov (2002) with the exactly same title came out only three years later. Often cited publications from that time are Mackie et al. (1994) for 3D MT computation; Druskin and Knizhnerman (1994) for frequency- and time-domain modelling using a Yee grid and a global Krylov subspace approximation; and Alumbaugh et al. (1996); Newman and Alumbaugh (1997) for low-to-high frequency computation on massively parallel computers.

The continuous improvement of computing power and the CSEM boom in the early 2000's in the hydrocarbon industry led to a wealth of developed numerical solutions and according publications. The most common applied methods to solve Maxwell's equation are the IE method (Raiche, 1974; Hursán and Zhdanov, 2002; Zhdanov et al., 2006; Tehrani and Slob, 2010; Kruglyakov et al., 2016; Kruglyakov and Bloshanskaya, 2017) and different variations of the differential equation (DE) method, such as finite differences (FD) (Wang and Hohmann, 1993; Mackie et al., 1994; Druskin and Knizhnerman, 1994; Streich, 2009; Sommer et al., 2013), finite elements (FE) (Commer and Newman, 2004; Schwarzbach et al., 2011; da Silva et al., 2012; Grayver et al., 2013; Puzyrev et al., 2013; Zhang and Key, 2016), and finite volumes (FV) (Madsen and Ziolkowski, 1990; Haber and Ascher, 2001; Clemens and Weiland, 2001; Jahandari and Farquharson, 2014). There are also many different types of discretization, where the most common ones are regular grids (Cartesian, rectilinear), mostly using a Yee grid (Yee, 1966) or a Lebedev grid (Lebedev, 1964), but also unstructured tetrahedral grids (Zhang and Key, 2016; Cai et al., 2017), hexagonal meshes (Cai et al., 2014), or octree meshes (Haber and Heldmann, 2007).

Very well written overviews about the different approaches to 3D EM modelling are given by Avdeev (2005) and Börner (2010). But there was also a tremendous publications output with regards to 3D EM modelling in the last 10-15 years, at least partly driven by the ever increasing computing power. One reason why there are so many publications about this topic results from the variety of techniques to solve the systems of linear equations (SLE). They can be distinguished between direct solvers (Streich, 2009; Chung et al., 2014; Jaysaval et al., 2014; Grayver and Kolev, 2015; Oh et al., 2015; Wang et al., 2018), indirect solvers (Jaysaval et al., 2015), or a combination of both, so-called hybrid solvers (Liu et al.,

2018). The solvers often use preconditioners such as the multigrid method (Aruliah and Ascher, 2002; Jaysaval et al., 2016).

Many of the advancements made in the EM modelling community over the past decades have required that authors develop new implementations from scratch. These codes often provided the research group or company with a competitive advantage for a time, and thus the source codes were often kept internal. In some cases, executables have been made available for academic purposes upon request or to sponsoring consortium members. As the field continues to mature, advancements become more incremental, and particularly in an applied field, many advancements are driven by new use-cases and applications that were not considered by the original authors. In the aforementioned review on EM modelling and inversion Dmitry Avdeev concludes with the following statement: « *The most important challenge that faces the EM community today is to convince software developers to put their 3-D EM forward and inverse solutions into the public domain, at least after some time. This would have a strong impact on the whole subject and the developers would benefit from feedback regarding the real needs of the end-users.*» Further, Oldenburg et al. (2019) argue that an open-source paradigm has the potential to accelerate multidisciplinary research by encouraging the development of modular, interoperable tools that are supported by a community of researchers.

It is becoming more common for researchers in many domains of science to release source-code with an open license that allows use and modification (e.g., see the Open Source Initiative approved licenses: opensource.org/licenses). This is an important step for improving reproducibility of research, «*to provide the means by which the reader can verify the validity of the results and make use of them in further research*» (Broggini et al., 2017). Going a step beyond releasing open-source software, many groups adopt an *open model of development* where code is hosted and versioned in an online repository, all changes are public, and users can engage by submitting and track issues. Community oriented projects further engage by encouraging pull-requests, which are suggested changes to the code. Successful, well-maintained projects often have unit-testing and continuous integration that runs those tests with any changes to the code. Additionally, documentation that includes examples and tutorials is an important component for on-boarding new users and contributors. As a result, in many areas of the geosciences, we are seeing a shift away from a one-way distribution of (open-source) code towards building global communities around open projects. Other notable Python projects with an open model of development within the same realm as the codes under considerations are `pyGIMLi` (Rücker et al., 2017), `Fatiando` (Uieda, 2018), `GemPy` (de la Varga et al., 2019), and `PyVista` (Sullivan and Kaszynski, 2019).

We discuss the topic of open-source developments with respect to 3D CSEM modelling. Even though it comprises still the minority of developments, the open-source landscape in this field of research changed in the last five years quite dramatically. Previously, there existed only closed-source codes owned by companies and consortia, or individual projects that were only available directly from the authors upon request. These codes often came without proper documentation. But more importantly, they also often came without a proper license, which essentially means that they are not open source (no license means legally speaking that, in most countries of the world, all rights are reserved by the author; see, for instance, the *Berne Convention*). We introduce and compare four recent open-source projects with a focus on 3D CSEM modelling. All presented codes are in the Python ecosystem and use either the FV method on structured grids or the FE method on unstructured

tetrahedral meshes.

Furthermore, we deal with the topic of verification. Analytical and semi-analytical solutions only exist for simple halfspace or layered-Earth models, which served mostly to verify new codes. Beyond these simple models, the only objective possibility to ensure the accuracy of solutions is by comparing results from different modellers. If different discretizations and implementations of Maxwell's equations yield the same result, it gives confidence in their accuracy. Miensopust et al. (2013) presents a review of two workshops dealing with the verification of magnetotelluric forward and inversion codes, but we are not aware of any comparable study or benchmark suite for CSEM data. This is, to the best of our knowledge, the first comparison exercise for CSEM codes. We hope our efforts help to ensure the reliability of simulations for any new or existing, open-source or closed-source 3D CSEM project.

We simulate EM fields for a layered background model with vertical transverse isotropy, with and without three resistive blocks, as well as for the complex marine, open-source MR3D model. These three models as well as the corresponding results from four different codes provide a benchmark for new (and existing) codes to be compared to and validated with. First we introduce the codes under consideration, and present afterwards the considered benchmark cases in detail together with the modelling results of our four codes in terms of accuracy and computational performance. Beyond that, we extensively discuss important points that control the performance and suitability of our FV and FE codes, including considerations about the mesh design and the choice of solvers. We conclude with a discussion and conclusions, and a motivation for the EM community at large to not only continue to extend the landscape of open-source codes but to also create a landscape of open-source benchmark models.

Ones were ready check if we should introduce Maxwell's equations or not. Miensopust et al., 2013, for instance, do not touch on any equation.

CODES

The four codes under consideration are, in alphabetical order, `custEM` (Rochlitz et al., 2019), `emg3d` (Werthmüller et al., 2019), `PETGEM` (Castillo-Reyes et al., 2018, 2019), and `SimPEG` (Cockett et al., 2015; Heagy et al., 2017). All four codes have their user-facing routines written in Python, and all of them make heavy use of `NumPy` (van der Walt et al., 2011) and `SciPy` (Virtanen et al., 2020). The four of them are not just open-source but follow the open model of development, meaning that they come with both, an open-source license and an online-hosted version-control system with tracking possibilities (raising issues, filing pull requests). All developments comprise an extensive online documentation with many examples and have continuous integration to some degree. Newer package-management systems such as `conda`, `docker`, or `pip` significantly supported us to make our codes easily available and installable on any of the major operating system without any user-side compilations. Each one can be downloaded and installed with a single command.

[DW]: I guess this statement is currently a bit of a stretch for `PETGEM`, correct Octavio?

In the following, we provide a quick overview of the codes. It is, however, beyond the scope of this article to go into every detail of the different modellers, and we refer to their documentations for more details. An overview comparison of the different codes is given in Table 1. All codes have in common that they solve Maxwell's equation in its differential

form (DE) under the quasi-static or diffusive approximation, hence neglecting displacement currents, which is a common approximation for the low frequency range commonly applied in CSEM. The machines on which the different codes were run are listed in Table 2 together with the responsible operator.

[DW]: I took out «the weak formulation of», as it actually doesn't apply to `emg3d`. Please make sure that everything stated above («have in common that...») actually applies to your code.

A few clarifying words on abbreviations and definitions: For the boundary conditions (BC) in the comparison table we use the abbreviations PEC and PMC, which stand for perfect electric conductor and perfect magnetic conductor, respectively. Another abbreviation used in the tables is dof for degree of freedom, which is equivalent to the size of the SLE we are solving. Finally, we use *runtime* as the wall time or elapsed real time from start to end of solving the SLE; pre- and post-processing (e.g., mesh generation) is not measured. Note that the actual computation or CPU time is therefore much higher than the reported runtime for the codes that run in parallel. Memory refers to the maximum memory increase at any point of solving the SLE.

custEM

The customisable electromagnetic modelling Python toolbox `custEM` was developed for simulating arbitrary complex 3D CSEM geometries with a focus on semi-airborne setups, but it supports also land-based, airborne, coastal, and marine environments. Multiple electric or magnetic field or potential finite-element approaches were implemented as total or secondary field formulations. The finite-element kernel, including higher order basis functions and parallelisation, relies on the `FEniCS` project (Logg et al., 2012; Langtangen et al., 2016). The resulting SLEs are solved with `MUMPS`, which is a very robust but memory consuming choice. Primary field solutions are supplied by the `COMET` package (Skibbe et al., 2020).

The toolbox considers generally anisotropic petrophysical properties. Even though changes of the conductivity is mainly of interest for CSEM modelling, the electric permittivity and magnetic permeability can be taken into account using the preferred electric field approach on Nédélec elements. Recently, induced polarisation parameters in frequency-

	`custEM`	`emg3d`	`PETGEM`	`SimPEG`
Home	custem.rtfd.io	empymod.github.io	petgem.bsc.es	simpeg.xyz
License	GPL-3.0	Apache-2.0	GPL-3.0	MIT
Installation	`conda`	`pip`; `conda`	`pip`	`pip`; `conda`
Comp. Dom.	frequency & time	frequency	frequency	frequency & time
Method	FE	FV	FE	FV
Mesh	tetrahedral	rectilinear	tetrahedral	recti- & curvilinear
BC	PEC; PMC	PEC	PEC	PEC; PMC
Solver	`MUMPS`	`emg3d`	`PETSc`; `MUMPS`	`PARDISO`; `MUMPS`

Table 1: Comparison of the four codes under consideration. Note that `emg3d` is a solver on its own, while the other codes implement third-package solvers such as `PETSc` (Abhyankar et al., 2018), `MUMPS` (Amestoy et al., 2001), or `PARDISO` (Schenk and Gärtner, 2004).

Code	Computer and Operating System	Operator
`custEM`	**PowerEdge R940 (server)** - 144 Xeon Gold 6154 CPU @2.666 GHz - ≈ 3 TB DDR4 RAM - Ubuntu 18.04	Raphael Rochlitz
`emg3d`	**Dell Latitude (laptop)** - i7-6600U CPU@2.6 GHz x4 - 15.5 GiB RAM - Ubuntu 18.04	Dieter Werthmüller
`PETGEM`	**Marenostrum4 (supercomputer)** - Intel Xeon Platinum from Skylake generation; 2 sockets Intel Xeon Platinum 8160 CPU with 24 cores each @2.10GHz for a total of 48 cores per node - 386 GB DDR4 RAM per node - SuSE Linux Enterprise	Octavio Castillo-Reyes
`SimPEG`	**GKE n2-custom (Google cloud)** - Intel Cascade Lake, 8 vCPUs - 355 GB RAM - Ubuntu 16.04	Lindsey Heagy

Table 2: List of hardware, software, and operator with which the different codes were run.

Could everyone check if their RAM is GB or GiB?

domain and three methods for simulating time-domain responses were added to `custEM`. The provided meshing tools are based on `TetGen` (Si, 2015) and functionalities of `pyGIMLi` facilitate the generation of tetrahedral meshes including layered-earth geometries with topography or bathymetry and anomalous bodies which are allowed to be connected or to reach the surface.

The `custEM` toolbox is under steady development and maintenance. Future enhancements comprise adding iterative solution techniques, provided by `FEniCS`, as well as further functionalities for automatically incorporating intersecting bodies such as folds or faults in meshes. Furthermore, current work is on using `custEM` for inverse modelling applications.

emg3d

The 3D CSEM code `emg3d` is a multigrid solver (Fedorenko, 1964) for electromagnetic diffusion following Mulder (2006), with tri-axial electrical anisotropy, isotropic electric permittivity, and isotropic magnetic permeability. The matrix-free solver can be used as main solver, or alternatively as preconditioner for one of the Krylov subspace methods implemented in SciPy. The governing equations are discretized on a staggered grid by the finite-integration technique (Weiland, 1977), which is a finite-volume generalisation of a Yee grid. The code is written completely in Python using the NumPy and SciPy stacks, where the most time- and memory-consuming parts are sped up through jitted (just-in-time compiled) functions using Numba (Lam et al., 2015). The strength of `emg3d` is the matrix-free multigrid implementation, which is characterised by almost linear scaling both in terms of runtime and memory usage, and it is therefore a solver that uses comparably very little memory.

As such the code is, contrary to the other three, primarily a solver, which solves the

Maxwell's equation for a single frequency and a single, electric source. However, recent developments added functionalities such that `emg3d` can be used directly as a more general EM modeller too. It features now routines to obtain the magnetic field due to an electric source, to obtain the electric and magnetic fields due to a magnetic source, and also to obtain time-domain responses. The most recent development added the possibility to compute the responses for entire surveys, and also to compute the misfit and the gradient of the misfit function, and `emg3d` is as such ready to be used in inversion schemes.

For the underlying discretization the Python package `discretize` is used, which is part of the larger `SimPEG` ecosystem. Interoperability between `SimPEG` and `emg3d` is therefore straightforward.

PETGEM

`PETGEM` is a parallel code for frequency-domain 3D CSEM data for marine and land surveys. The high-order edge FE method (HEFEM) is used for the discretization of the governing equations in its diffusive form. This technique provides a suitable mechanism to obtain stable numerical solutions as well as a good trade-off between number of dof and computational effort. The current implementation supports up to sixth-order tetrahedral vector basis functions. Moreover, because the HEFEM belongs to the FE family, the unstructured meshes can be used efficiently on complex geometries, e.g., models with topography and bathymetry.

`PETGEM` permits to locate the source and receivers anywhere in the computational domain (e.g., sediments, seafloor, sea, ground), allowing to analyse the physical environment of the electric responses and how parameters impact them (e.g., frequency, conductivity, dependence on mesh setup, basis order, solver type, among others). Furthermore, `PETGEM` implements a semi-adaptive mesh strategy (*hp* mesh refinement) based on physical parameters and on polynomial order to satisfy quality criteria chosen by the user. Nonetheless, only Horizontal Electric Dipole (HED) has been implemented.

For the parallel forward modelling computations, a highly scalable MPI (message passing interface) domain decomposition allows reducing runtimes and the solution of large-scale modelling cases. This strategy is capable of exploiting the parallelism offered by both modest multi-core computers and cutting-edge clusters (e.g., High-Performance Computing architectures).

Based on the current state of the `PETGEM` project, there are many possibilities to improve or to add additional features. For instance, in the short-term, more modelling routines and support for multi-source set-ups will be implemented. For the long-term, the obvious next step is to move to 3D CSEM data inversion.

SimPEG

`SimPEG` is a modular toolbox for simulations and gradient based inversions in geophysics. Current functionality includes modelling and inversion capabilities for gravity, magnetics, direct current resistivity, induced polarisation, electromagnetics (time and frequency, controlled and natural sources such as magnetotellurics), and fluid flow (Richards equation).

It is a community driven project that aims to support researchers and practitioners by providing a flexible, extensible framework and to facilitate integration of data sets from a variety of geophysical methods, including joint inversions (Astic et al., 2020), by providing a common interface to each method.

[DW]: I added «and practitioners», as I think «support researchers» downplays the fact that `SimPEG` is used by various companies as well...

Meshes and finite volume differential operators are implemented in the `discretize` package, which currently includes tensor, octree, cylindrical, and logically rectangular meshes. Each mesh type inherits from a common structure and uses the same naming conventions for methods and properties. This allows us to decouple the implementation of a discretized partial differential equation from the details of the mesh geometry and therefore we can write a single implementation of discretized partial differential equation (or set of partial differential equations) in `SimPEG` that will support all mesh types implemented in `discretize`.

The electromagnetic implementations use a staggered grid finite volume approach where physical properties are discretized at cell centres, fields on cell edges and fluxes on cell faces. There are two different discretization strategies implemented for Maxwell's equations: (I) the EB-formulation, which discretizes the electric field ($\vec{e}$) on cell edges and the magnetic flux density ($\vec{b}$) on cell faces, and (II) the HJ-formulation, which discretizes the magnetic field ($\vec{h}$) on cell edges and the current density ($\vec{j}$) on cell faces. The physical properties electrical conductivity / resistivity and magnetic permeability are discretized in both cases at cell centres. Having multiple implementations allows for testing that compares results from each approach, as well as the representation of both electrical and magnetic sources on cylindrically symmetric meshes. `SimPEG` supports variable magnetic permeability and full-tensor anisotropy for the physical properties.

`SimPEG` interfaces to various solvers including `PARDISO` and `MUMPS`, and also to some implemented in SciPy. For problems with many sources (e.g. airborne EM), `SimPEG` contains machinery for domain-decomposition approaches; see for example Fournier et al. (2020).

NUMERICAL VERIFICATION

We computed the responses for three different models to verify that the four 3D CSEM codes yield the same electromagnetic responses and to compare different solver types (FE, FD) and different mesh types (unstructured tetrahedra, tensor, octree) in different scenarios. The first model is an anisotropic layered model, which can be compared to semi-analytical solutions. The layered model serves also as background model for the second verification, where we add three resistive blocks into the subsurface. The final comparison is based on the realistic, anisotropic marine model MR3D, which is an open resistivity model which comes with computed CSEM responses.

Layered Model

The layered (1D) model consists of an upper halfspace of air ($\rho_{\text{air}} = 10^8\,\Omega\,\text{m}$), a 600 m deep water layer ($\rho_{\text{sea}} = 0.3\,\Omega\,\text{m}$), followed by a 250 m thick, isotropic layer of $1\,\Omega\,\text{m}$, a 2.3 km thick, anisotropic (vertical transverse isotropy, VTI) layer of $\rho_{\text{h}} = 2\,\Omega\,\text{m}$ and $\rho_{\text{v}} = 4\,\Omega\,\text{m}$, and finally a resistive, isotropic basement consisting of a lower halfspace of $1000\,\Omega\,\text{m}$. The survey consists of a 200 m long and 800 A strong source at a single position with frequency $f = 1\,\text{Hz}$, and three receiver lines of 101 receivers each. The centre of the x-directed source

is at $x = y = 0$ m, 50 m above the seafloor. The x-directed receivers are placed on the seafloor every 200 m from $x = -10$ km to $x = +10$ km in three lines with $y = -3; 0; +3$ km.

A layered model fails to show the strength of 3D modellers, and in reality one would not choose a 3D code to compute responses for a layered model. However, it is one of the few examples that can be compared to semi-analytical results, and is therefore still a valuable test. All layered-model responses are compared to results computed with the semi-analytical code `empymod` (Werthmüller, 2017).

Figure 1 shows the result of the layered model verification, where the left, middle, and right columns are for the real part, imaginary part, and absolute value, respectively (dashed lines indicate negative values). The top row shows the actual, semi-analytically computed responses, and the second and the third rows show the relative error (in percent) of the inline and the broadside receivers, respectively (note that for the 1D case the two broadside lines for $y = \pm 3$ km are identical).

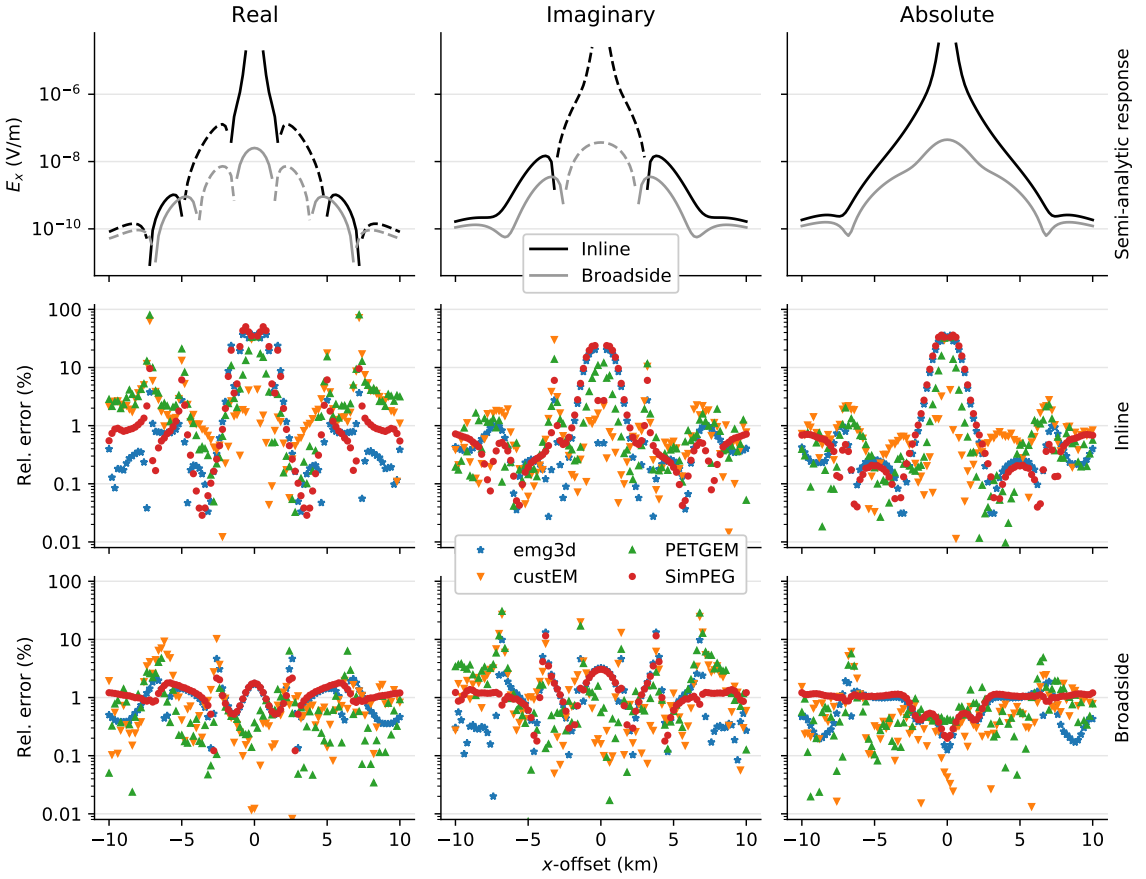

Figure 1: Inline and broadside responses (top row) for the layered model. The relative error of the four 3D modellers are shown in the second and the third row for the inline and broadside responses, respectively.

There are a few noteworthy points:

1. The error plots of the real and imaginary responses are dominated by the zero-crossings, which blows the error up as is the nature of the relative error formula,

it has no real-world implications.

2. However, it can be seen from the absolute-values errors that the error is generally in the order of 1 % or less.

3. The relative errors close to the source are getting huge, particularly for the inline receivers and particularly for the results of `emg3d` and `SimPEG`. These comparably huge errors are caused by a too coarse discretization around the source. They could be significantly reduced by choosing a finer discretization, at the cost of increased computation time.

4. The relative error of the responses from `custEM` and `PETGEM`, on the hand, are relatively big at large offsets, particularly for the inline real responses. This has, again, its origin in the chosen discretization. The thin isotropic layer beneath the water requires many tetrahedra for a good-quality discretization. Therefore, the FE codes only extend this thin layer up to 30 km for the origin, surrounded by a ten times larger halfspace-like boundary mesh with water conductivities assigned to the subsurface. This approximation significantly reduced the problem size compared to considering the layered-earth geometry for the whole computational domain.

Please not that `emg3d` and `SimPEG` used the same tensor mesh in this example.

[DW]: Lindsey: A sentence why Tensor over OcTree.

[DW]: Raphael/Octavio: A sentence about the differences in the meshes of `custEM` and `PETGEM`.

The runtime and memory requirements for the layered model are listed in Table 3.

[DW]: Lindsey: A note about high RAM usage of `SimPEG`? About PARDISO or Mumps or similar. Could be used as a general note that it boils down to the underlying solver.

Block Model

The block model is a derivation of the *Dublin Test Model 1* from the first EM modelling workshop described by Miensopust et al. (2013). We use the same layout of the blocks but adjust the dimensions and resistivities to a typical marine CSEM problem, as shown in Figure 2. Additionally, we add our *Layered Model* as a VTI background. The three

	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	24	118	97.8	1 530 808
`emg3d`	1	116	0.4	4 813 216
`PETGEM`	24	90	34.2	2 455 868
`SimPEG`	4	10 089	280.3	4 813 216

Table 3: Comparison of number of processes, runtime, and memory, as well as the degree of freedom of the discretization used by the different codes for the layered model.

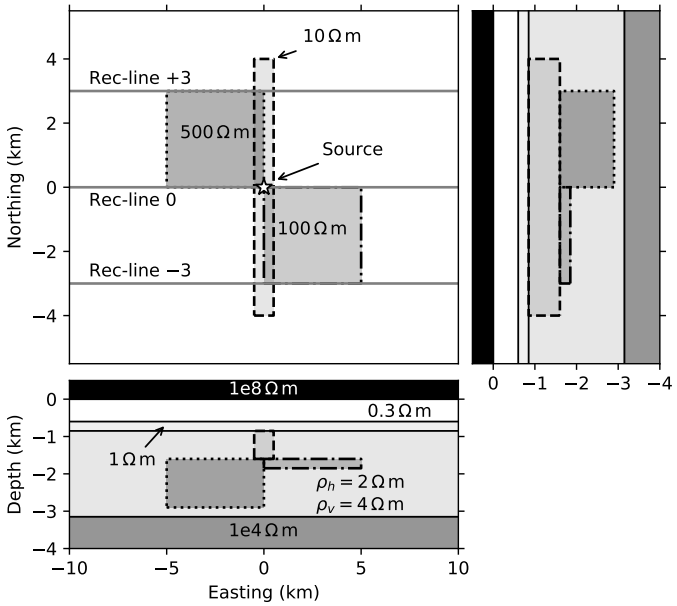

Figure 2: Sketch of the block model, consisting of the layered background model with three resistive blocks, embedded in the thick background layer which has VTI with $\lambda = \sqrt{\rho_\mathrm{v}/\rho_\mathrm{h}} = \sqrt{2}$.

resistive blocks have resistivities of $\rho = 10\,\Omega\,\mathrm{m}$ (shallow beam perpendicular to survey lines), $\rho = 100\,\Omega\,\mathrm{m}$ (thin plate, South-East), and $\rho = 500\,\Omega\,\mathrm{m}$ (cube, North-West).

There are no (semi-)analytical solutions for such a model. We therefore use the normalised root-mean square difference (NRMSD) between the results of different codes, where the NRMSD of two responses R_1 and R_2 is given by

$$\mathrm{NRMSD}\ (\%) = 200\frac{|R_1 - R_2|}{|R_1| + |R_2|}\ . \tag{1}$$

The results for the three receiver lines $y = -3, 0, +3\,\mathrm{km}$ are shown in the left, middle, and right columns of Figure 3, respectively. The top row shows the result of `emg3d`, as an example, and the bottom row shows the NRMSDs between the absolute responses of the codes.

We might wait for the final `SimPEG` results before jumping to conclusions; but generally, the NRMSD differences is below 1-2 %, except for large offsets.

The corresponding required runtime and memory are listed in Table 4.

Marlim R3D

The Marlim oil field is a giant reservoir in a turbidite sandstone horizon in the north-eastern part of the Campos Basin, offshore Brazil, which was discovered in 1985. Carvalho and Menezes (2017) created from seismic data and well log data a realistic, three-dimensional resistivity model with vertical transverse isotropy (VTI), called MR3D, which they released

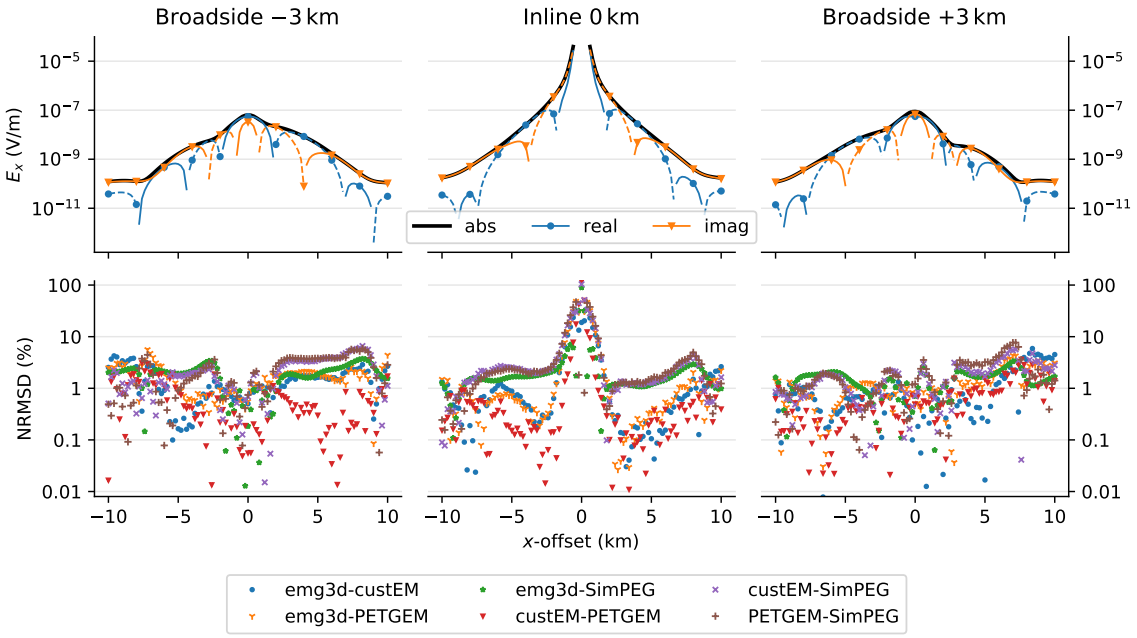

Figure 3: Results of the block model comparison: The responses of `emg3d`, as an example, are shown in the top row, and the NRMSDs (%) between the absolute responses ($|E_x|$) of the different codes are shown in the bottom row.

	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	24	158	115.9	1 654 242
`emg3d`	1	115	0.4	4 813 216
`PETGEM`	24	72	35.8	2 455 868
`SimPEG`	4	9993	280.6	4 813 216

Table 4: Comparison of number of processes, runtime, and memory, as well as the degree of freedom of the discretization used by the different codes for the block model.

under the open creative common license *CC BY 4.0*. Correa and Menezes (2019) computed CSEM data for MR3D for six frequencies from 0.125 Hz to 1.25 Hz, and released them under the same CC license. To compute the data they used a state-of-the-art code from the industry (Maaø, 2007, *SBLwiz* software from *EMGS*). It is therefore an ideal case to verify our open-source codes against, as it is a complex, realistic model and the data were computed by an industry-proofed code. Additionally, that code is a time-domain code, whereas the four codes under consideration here compute the results all in the frequency-domain.

The full MR3D model consists of $1022 \times 371 \times 1229$ cells, totalling to almost 466 million cells, where each cell has dimensions of 25 by 75 by 5 m. For the computation the model is upscaled to 515 by 563 by 310 cells, totalling to almost 90 million cells, where each cell has dimensions of 100 by 100 by 20 m. Both the full model and the upscaled computational model are released openly. The published data set consists of a regular grid of receivers of 20 in eastern direction by 25 in northern direction, 500 receivers in total, with 1 km spacing located on the irregular seafloor. 45 source-towlines were located on the same grid above each receiver line, 50 m above the seafloor, with shots every 100 m. The computed responses for one of the receivers, `04Rx251a`, are shown in the original paper for the East-West inline-source `04Tx013a` and the East-West broadside-source `04Tx014a` (broadside offset of 1 km). We reproduce the responses for this receiver and corresponding source-lines in our comparison. The x-z cross-section of the horizontal resistivity model at the receiver position is shown in Figure 4, together with the receiver and sources positions. All layer have a VTI with $\lambda = \sqrt{2}$, except for the air layer, the seawater, and the salt layer, which are all isotropic.

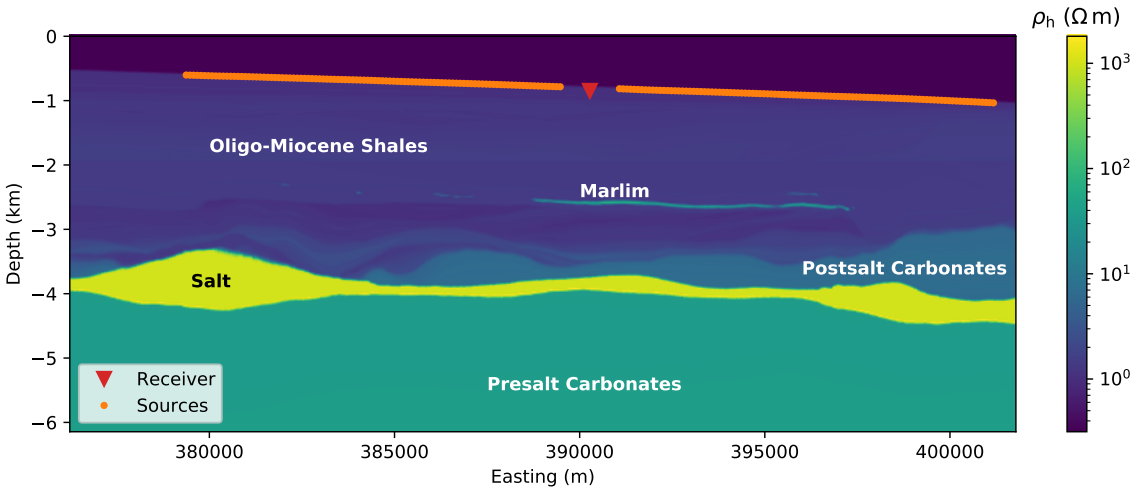

Figure 4: MR3D horizontal resistivity model, x-z-slice through the receiver y-position and with major formations annotated. Air (not shown in the model), seawater, and the salt layer are electrically isotropic, everything else has VTI with $\lambda = \sqrt{2}$. The receiver is located on the seafloor, and the sources fly 50 m above the seafloor.

The responses for all six frequencies, both inline and broadside, for all three electric components are shown in Figure 5. The published responses are shown in color (and with markers), and the responses from our codes are shown beneath in grey colours in this overview plot. Two datasets were published, one containing clean responses, and one where

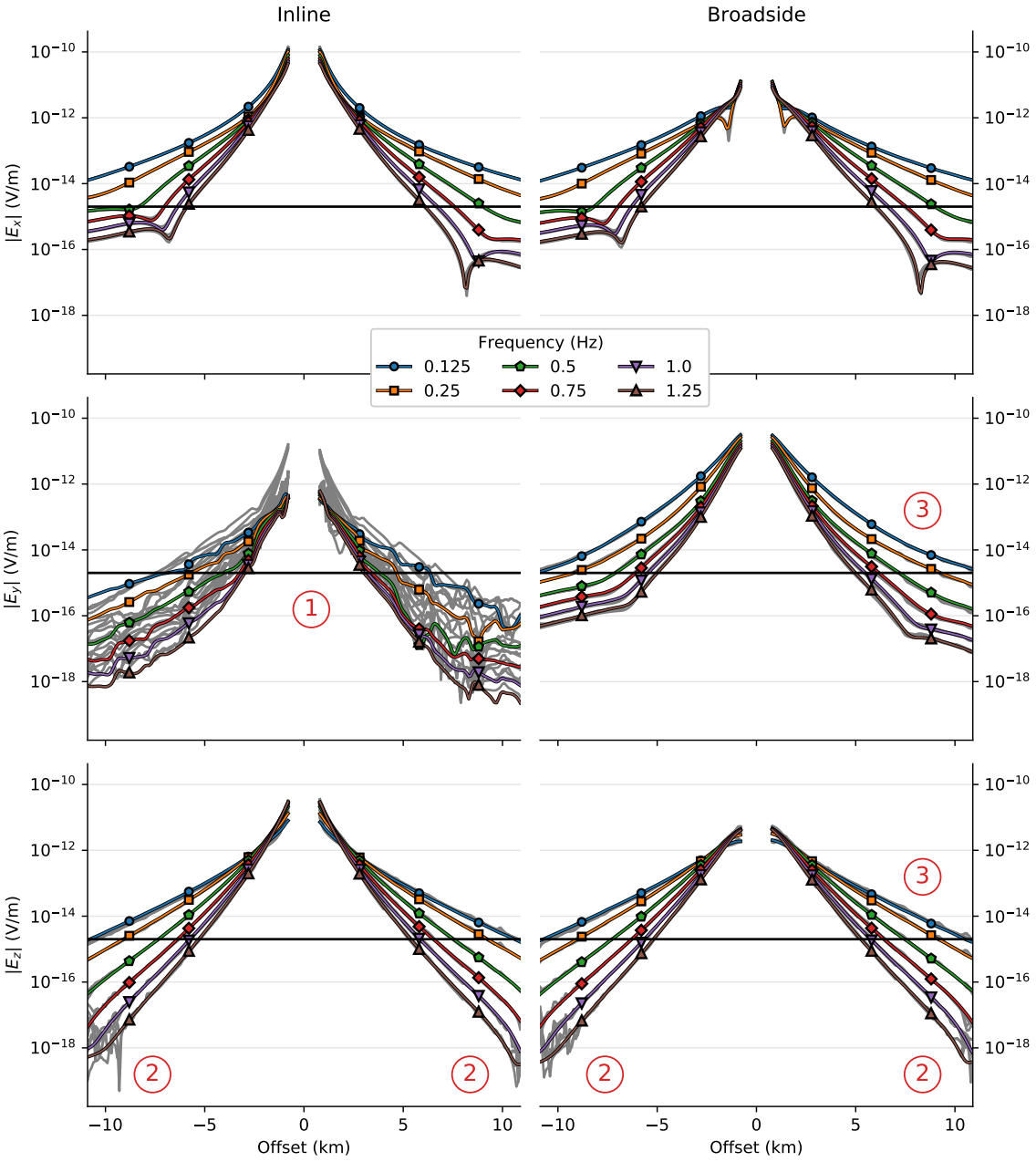

Figure 5: MR3D comparison between our codes (grey lines) and the published data (coloured lines). The grey lines under the coloured lines are not visible in most areas, meaning they are very similar. However, there are three notable zones, (1) to (3), which are explained further in the text.

realistic noise was added. For the comparison we use the clean data, but we indicate the chosen noise level by the horizontal line at 2×10^{-15} V/m. It can be seen that the grey lines are not visible for most parts, which means that the data agree very well. However, there are a few notable points where this is not the case, and they are marked with numbers in the figure.

1. The responses for the E_y component of the inline acquisition line do not agree at all. In the 1D case, the inline E_y component would be zero, and the only response we can measure here come from 3D effects, but they are very low, roughly two orders of magnitude lower than the E_x component. The published responses are not stable either, and therefore any of the responses is as bad as the other. Also, tiniest differences in meshing will have a huge effect here.

2. The highest three frequencies became noisy at large offsets for all of our codes (well below any real-world noise level). The reason why the published responses are not noisy is probably because they were computed in the time domain, and transformed to the frequency domain.

3. There are some noticeable differences in the positive offsets of the broadside E_y and E_z components. The differences are related to the bathymetry. And the reason why we cannot reproduce the published results exactly is that the code is not open-source, and we do not exactly know what it does internally. See the discussion around Figure 9.

Figure 6 shows the NRMSDs between the published results and our codes for the inline E_x and the broadside E_x and E_y components, for three frequencies. A few general conclusions can be drawn from this figure: In general the NRMSD is roughly 10 % or less, particularly for the inline E_x component it is only a few % or less. In some cases the regular meshes with `emg3d` and `SimPEG` have a smaller NRMSD, such as, e.g., negative offsets in the inline and broadside E_x responses, in some cases the unstructured tetrahedral meshes with `custEM` and `PETGEM` have a smaller NRMSD, such as, e.g., positive offsets for all broadside E_y responses, and often they have a comparable NRMSD. Some NRMSD have an interesting step-pattern, which is more pronounced for `emg3d` and `SimPEG` than for `custEM` and `PETGEM`; this is related to the annotated point (3) in Figure 4, and will be discussed in more detail with Figure 9.

One might argue that 5-10 % NRMSD is a lot. We argue that this is probably as good as it gets, given that the code with which we compare is not open-source. This has been shown within the process of these results. Our first attempt were based on the originally published, fine MR3D model. This model is very detailed, but too small in x- and y-direction for CSEM computation, and each of our codes did its own extrapolation. However, none of the codes were able to reproduce the published results. The reason is that there are many ways how you can extrapolate a model. We reached out to the authors of MR3D, and they kindly agreed to also published the upscaled and extended computational model. It is only with this model that we were able to get comparable results, visually mostly the same as shown in Figure 4.

However, even though the computational input model is know, we still do not know what the code SBLwiz exactly does inside. From the information we have we know that it computes the data in the time domain transforming them afterwards to the frequency

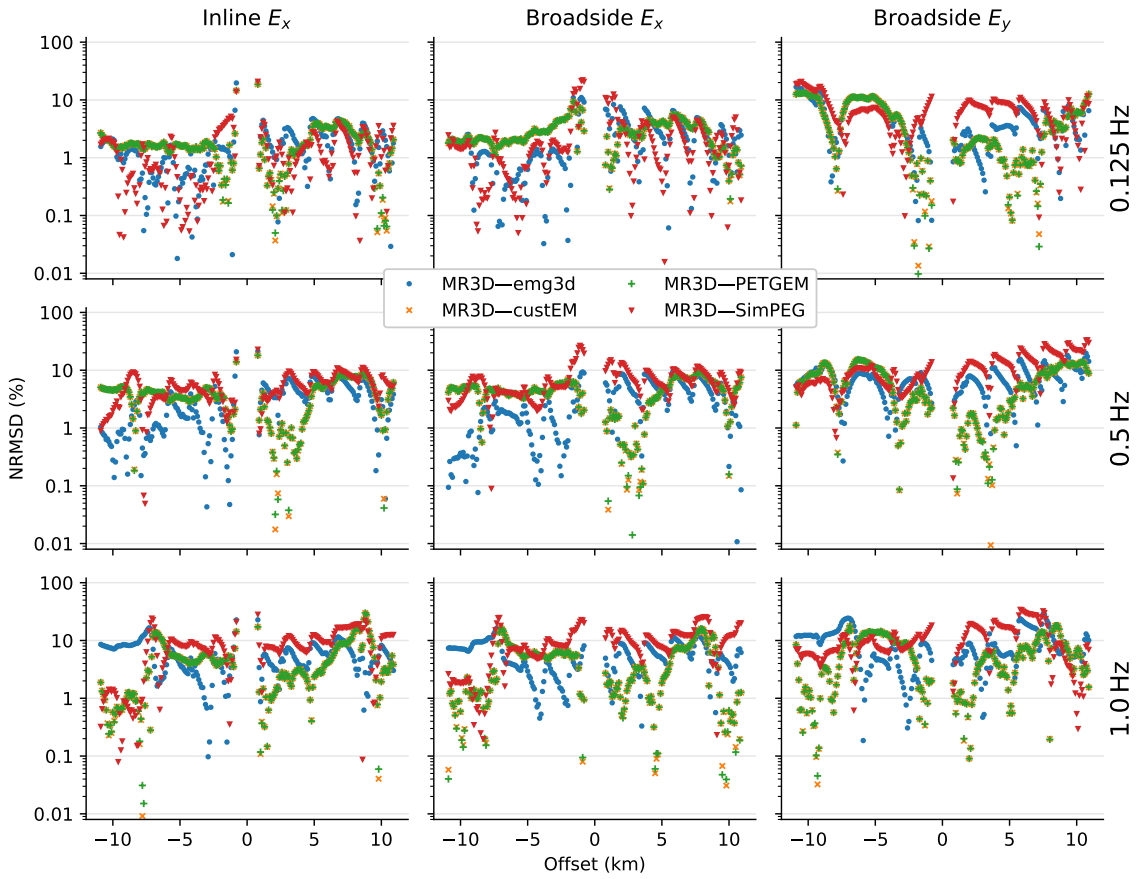

Figure 6: NRMSDs for all four codes in comparison with the published data, for the inline E_x field and the broadside E_x and E_y fields in the left, middle, and right column, respectively. Shown are the three frequencies 0.125 Hz, 0.5 Hz, and 1.0 Hz in the top, middle, and bottom row, respectively.

domain, and it includes the air layer via a non-local boundary condition at the water-air interface (Mittet, 2010). All our codes, on the other hand, compute in the frequency domain and include the air layer in the model (putting the boundaries sufficiently far away).

The importance of the meshing to the result can be seen in Figure 7, where we compute the NRMSDs between our codes. The most obvious result in that figure is that `custEM` and

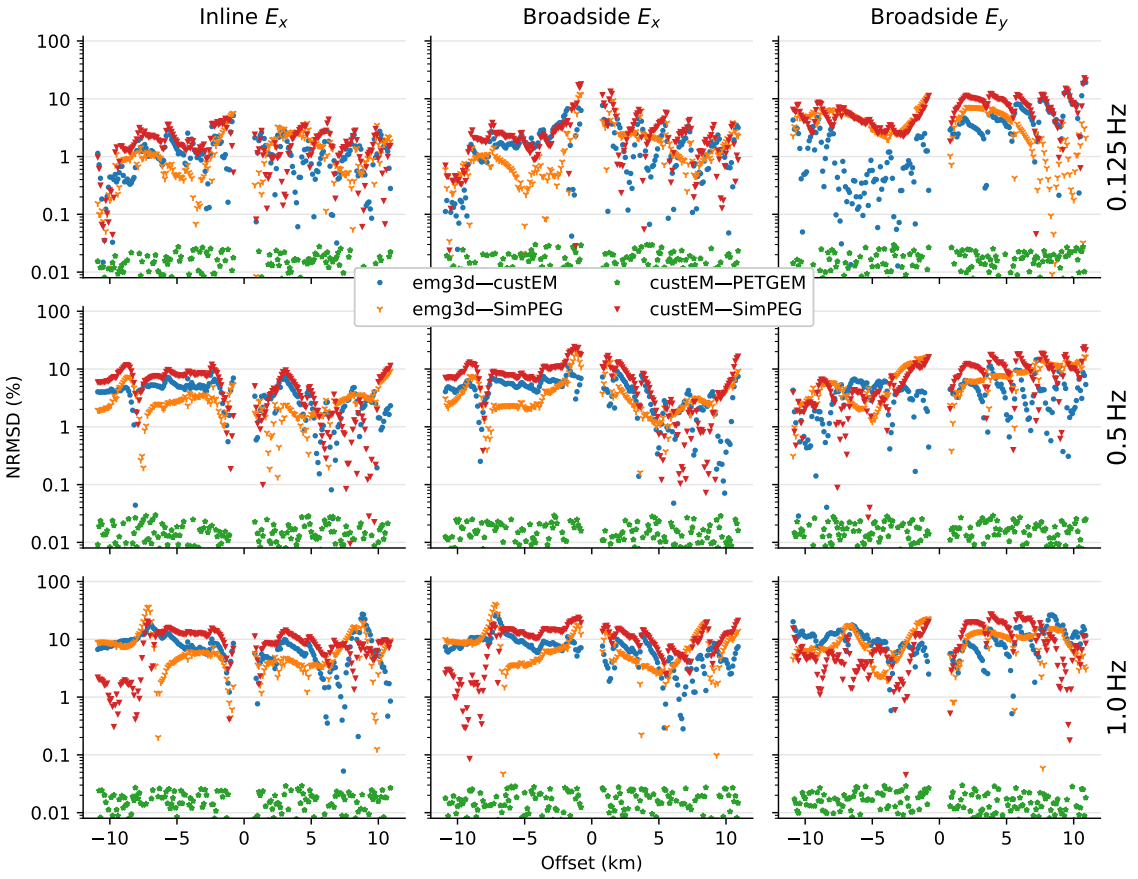

Figure 7: NRMSDs, just as in Figure 6, but comparing some of the four codes with each other. Good visible is that `custEM` and `PETGEM` produce very similar results, which is due to the fact that they use the same mesh.

`PETGEM` produce results that are almost identical, their NRMSD is generally below 0.03 %. The reason is simple: `custEM` and `PETGEM` use the exactly same mesh for the computation, the only difference is therefore the solver. As they are so similar we compare `emg3d` and `SimPEG` only to `custEM` in Figure 7, as the comparisons to `PETGEM` would look the same. But in general the NRMSDs between our codes look similar as in the comparison with the published results, which is due to the different meshes in use.

The differences in meshing is shown in the next two figures, in Figure 8 for the entire computational domain, and in Figure 9 a zoom-in to the survey domain, the domain of interest. The top rows show the tensor mesh of `emg3d`, the middle rows the octree mesh of `SimPEG`, and the bottom rows the tetrahedra mesh of `custEM` and `PETGEM`. The actual mesh, the $|E_x|$ field, and the $|E_y|$ field are shown in the left, middle, and right columns, respectively.

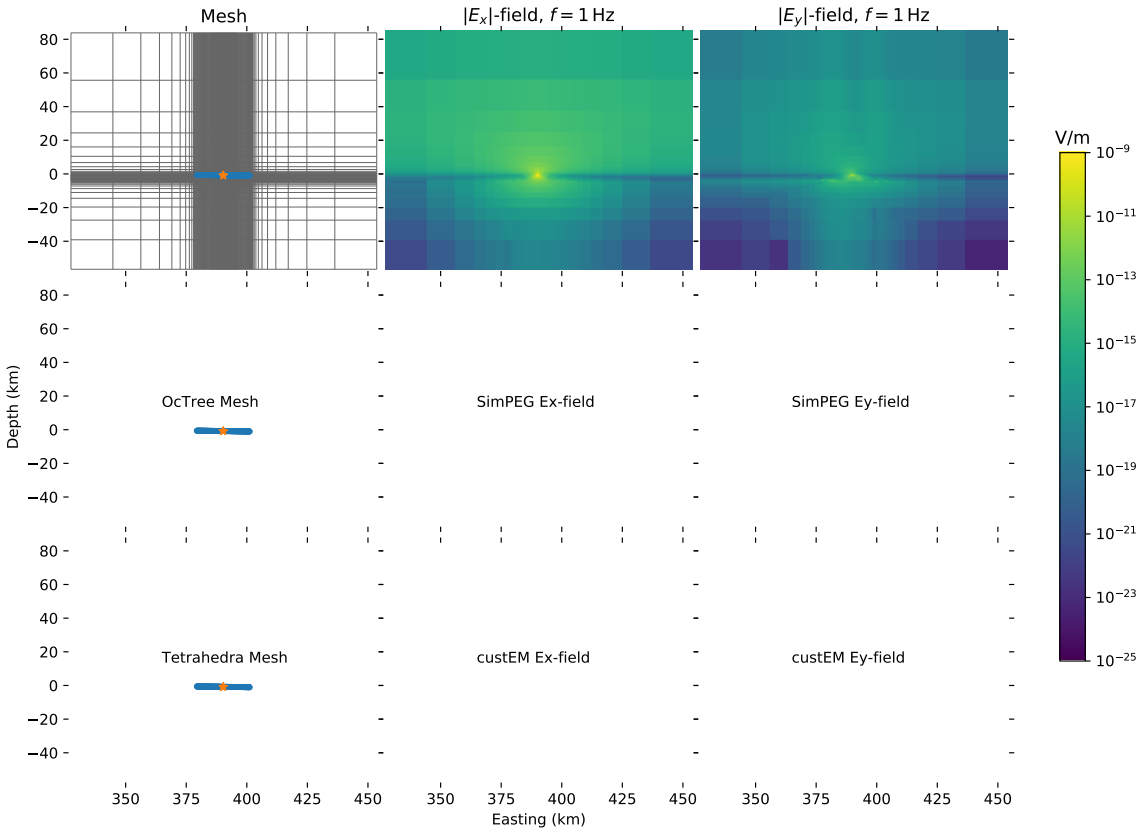

Figure 8: Meshes (left column), E_x field (middle column), and E_y field (right column) for a tensor mesh (top row), octree mesh (middle row), and tetrahedra mesh (bottom row) for the entire computational domain.

Write more once the other results are in the figures too. Points to compare: See if the different boundary conditions are visible. See and comment on the effects of the different, fast coarsening outside the domain of interest.

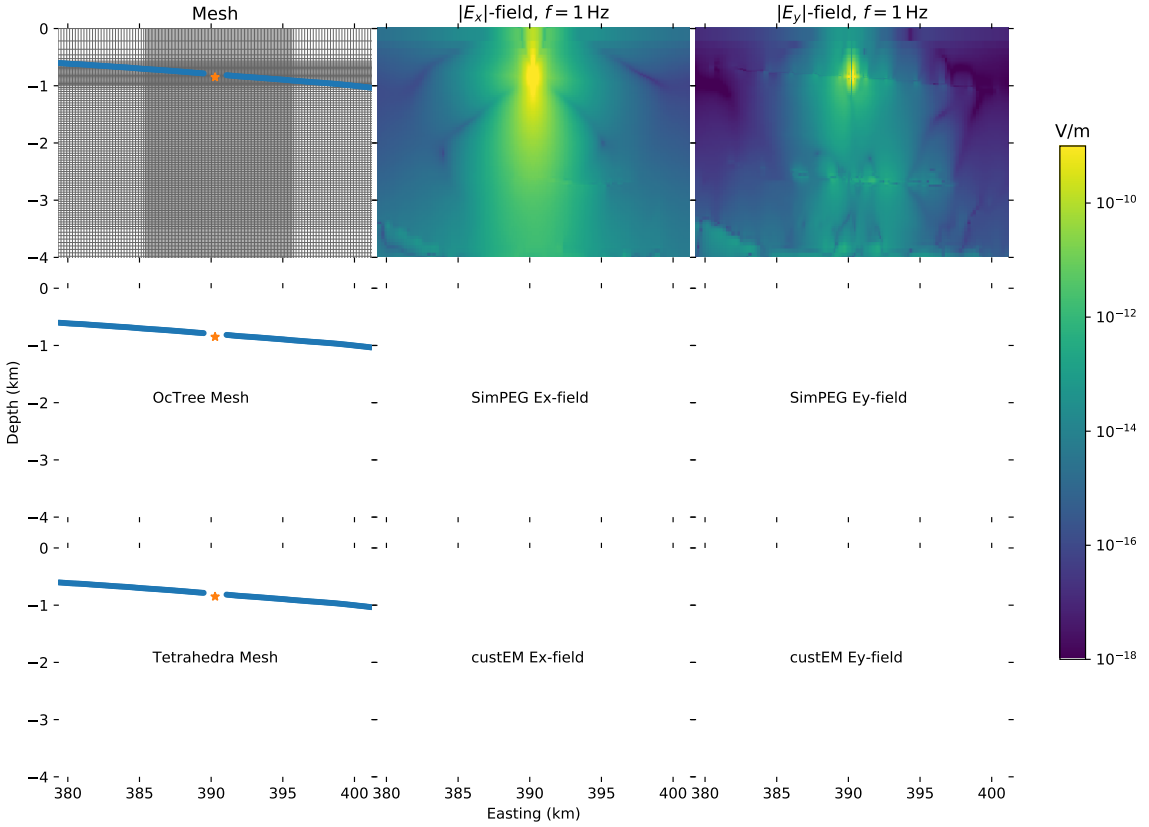

Figure 9: Meshes (left column), E_x field (middle column), and E_y field (right column) for a tensor mesh (top row), octree mesh (middle row), and tetrahedra mesh (bottom row) for the survey domain.

Most importantly, once all results are in the figures: Discuss about the effect of bathymetry, particularly on the E_y field. Mark it with a number in the figure. Compare between the meshes. State that the results will not have a smaller NRMSD unless we know exactly how the bathymetry is treated.

The required runtime and memory to compute the shown MR3D responses for the four codes are listed in Table 5. The memory requirement varies from $0.5 - 558.5$ GiB, and the runtime from roughly 7 min to 1h 23 min, where the codes were run on very different machines from servers to supercomuters (see Table 2) and use between 1 and 96 processes in parallel).

DISCUSSION

Even if modelling results may look completely valid, they could not be correct at all. Verifying the performance of 3D CSEM codes for real 3D problems is only possible by cross-validating multiple solutions. Overall, we observed an excellent match between all solutions. The relative misfits, almost always smaller than 10 %, were related to the either weak or

	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	64	4984	558.5	4 539 474
`emg3d`	1	1200	0.5	5 998 992
`PETGEM`	96	3527	437.8	4 539 474
`SimPEG`	4	422	12.8	720 146

Table 5: Comparison of number of processes, runtime, and memory, as well as the degree of freedom of the discretization used by the different codes for the MR3D model.

strong amplitudes of real or imaginary parts and can be attributed to particularities regarding the discretization, boundary effects, and interpolation. Considering this satisfactory result in terms of accuracy and robustness, it is more interesting to discuss the characteristics of the four considered codes with focus on the demand of computational resources and the capability of handling the investigated CSEM geometries, including the very important underlying mesh design. The following arguments are based on a comparison of only marine CSEM models, which are the focus of industry applications. Nevertheless, we are confident that many observations would hold in a similar manner for land-based, airborne, or mixed setups.

We point out that making a synthetic and a real benchmark model available to cross-validate any 3D CSEM code was the determining reason for their choice, regardless of the suitability of the considered FD and FE codes for these problems. The first model, including the block-anomalies variation, is exactly reproducible by using any numerical method with either regular or irregular computational domains. Aside from the idea of analysing an industrially relevant benchmark example, the MR3D model was also considered since the primary design from Correa and Menezes (2019) uses a regular discretization. At least theoretically, the exact geometry might be reconstructed with unstructured meshes, but it is never possible to convert an irregular geometry to a regular grid without approximations. Hence, the chosen examples favoured the FD codes, since they used either regular grids or octree-meshes, whereas the two FE codes were required to tetrahedralise the model geometries.

In the first model, the necessity of discretizing regular structures with tetrahedra, especially thin layers, led to comparatively large FE problem sizes and computational resource demands. FE codes with unstructured meshes were probably the poorest choice of all commonly used methods for this setup in terms of computational performance. However, thinking about introducing just one slight irregularity such as bathymetry, a thin dipping plate or an anticline, the performance behaviour could likely switch. All of these changes would barely change the problem size for the FE codes, but significantly increase it for the FD codes if the true geometry should be approximated sufficiently accurate.

The real MR3D model was, ironically, the ideal case for FE codes because it's structure was characterised by irregular lithological horizons obtained from seismic data. Since the corresponding published resistivity model was defined on a regular grid, we were forced to approximate the comparatively fine regular discretization by an unstructured one and interpolate the resistivity data for being able to apply the FE codes to this problem. We are completely aware about the ineptness of this re-approximation procedure, unless it served for

the cross-validation purposes. Nevertheless, the chosen tetrahedral discretization, especially in combination with second order basis functions, showed clearly the intrinsic strength of the FE codes, requiring significantly less dof for meshing irregular geometries in general. The p2 FE solution required only ≈9 M dof instead of ≈90 M dof for the FD system by Correa and Menezes (2019), disregarding the structure of the system matrices.

The codes under consideration use direct and indirect solvers. In general, the biggest advantage of iterative solvers with appropriate preconditioners are the comparatively small memory requirements. Direct solvers can be considered as most robust to solve any ill-conditioned SLE, which is most important for systems of the FE method on unstructured meshes. As the consumed computational resources indicate, a more powerful machine than a laptop is required for most 3D CSEM problems, but no high-performance-computing architecture.

The computation times can differ significantly for specific simulations, but there is no clear general advantage for one of the solver types. In our models, the runtime comparison favoured the iterative solver and therefore, `emg3d`. Even though not shown here, direct solvers exhibit their strongest advantage in terms of computation times if responses of multiple CSEM transmitters need to be computed in the same computational domain. As the system matrix factorisation requires 98-99 % of the solution time, computations for additional sources come at almost no cost for direct solvers, i.e., MUMPS, whereas it would come at the same cost as the first source for `emg3d`. For the sake of completeness, note that there are also advanced techniques for iterative solvers to make use of preconditioners, factorisations, or intermediate solutions of the previous SLE to speed up the computation for multiple sources, but this argumentation is beyond the scope of this work.

We see this work as the start for future elaboration on the validation of modelling codes, not necessarily restricted to CSEM problems. Much more comparisons and examples are required which not only result in the finding that a sufficient accuracy was observed. 3D CSEM modelling is a difficult task, which requires many considerations. It starts with the selection of the right code for the problem. We found that the meshing task, which is most performance determining, is particularly difficult and relevant, choosing cells small enough to appropriately represent the model yet to be as coarse as possible still achieving the desired precision. The required model extent has to be considered as well, thinking also about effects of the so-called airwave in marine setups or boundary meshes in general. This is not a new finding but rather well known fact. We are surprised that only the minority of publications in the field of 3D EM modelling considers a sufficiently detailed elaboration of the meshing process.

A completely objective evaluation of codes is only possible by cross-validation. Nevertheless, we point out that using different discretizations or formulations of the numerical approach, e.g. by choosing field or potential approaches, second or total field formulations, different polynomial order basis functions, and others, is a suitable alternative to the cross-comparison of multiple codes for self-validating complex 3D results. This is a very powerful method to confirm the general functionality of codes for any modelling problem, which might not be covered by existing cross-validation benchmark examples. Considering for instance this study, it would be very optimistic to estimate the accuracy of the utilised codes for a land-based mineral exploration setup with a highly-conductive, steeply-dipping conductor. However, each modeller could run multiple simulations with different configurations to increase the reliability of the obtained results.

It is worth noting that the reported runtime is just one of the aspects. But to generate an appropriate mesh, with all the testing involved, can easily take days of work. As such the computation times becomes relative, and the ease of usability of a code has to be considered too. For `custEM`, `PETGEM`, and `SimPEG` it is therefore also a question of how easy everything is to set-up, because after the set-up phase it depends largely on the used solver, which might change in the future; `SimPEG`, e.g., is looking at moving to similar solvers as currently are used by `custEM` and `PETGEM`. It is, however, slightly different for `emg3d`, as that is primarily a solver on its own. It is also important to state that while `emg3d`, `custEM`, and `PETGEM` are purely CSEM codes, `SimPEG` is a much bigger framework which includes other geophysical methods as well.

The shown examples consider the geophysical problem of marine CSEM. However, the codes could, and many frequently are, applied to very different applications, including any other geophysical exploration in the sea or on land (oil, mining, geothermal, groundwater, environmental) and also non-geophysical tasks. The current main applications of the different codes are mining (`custEM`), mining and environmental (`SimPEG`), oil and more recent medical and geothermal (`PETGEM`), and oil, methane, and deep sea mining (`emg3d`).

TODOS

Add what improvements were initiated by article ("But also, a plus-point of such collaboration is if it brings the realm as such further. So we should note down if this collaboration resulted in new features implemented in some codes, new releases etc. What did it push forward? It would be a motivation for further cross-project/community collaborations. Something for the discussion or conclusion part."):

- Lindsey: For SimPEG - this will result in new utilities for volume averaging, and likely more documentation on tips for designing tree meshes. This could be a pretty compelling piece to bring in to the discussion
- Raphael:
  - cell-wise resistivity interpolation added
  - multi-layer subsurface topography first time applied
  - reciprocity modeling (even though that has not too much to do with implementation)
  - mesh design (resolution, refinement, etc.)
- Lindsey: Within SimPEG: This work has motivated feature development including: (a) interpolation and averaging strategies from mapping physical properties on a fine mesh to a coarser mesh for computation and (b) new examples to include in the documentation for designing octree meshes. Furthermore, as a part of a broader development objective of inter-operating with other forward-simulation engines, connecting `emg3d` and `SimPEG` provided a motivating use-case for the latest refactor and release of `SimPEG` 0.14.0.

Many things to discuss and take into account

- Examples are in favour of FD codes, (rectangle blocks, complex model provided as FD model)
- Examples are in favour of iterative solver, as only one source was computed. Iterative solvers (`emg3d`) will require the same amount of time for each new source, whereas direct solvers have additional source locations almost for free
- Not possible to exactly reproduce published results unless the codes is also open-source, as one cannot know what internally is actually done (show example!)
- In the end the codes (`custEM`, `emg3d`, `PETGEM`, `SimPEG`) define mainly, how easy the interaction is. How easy is it to set-up the mesh, to create sources and receivers, etc etc. The *actual* computation time, however, is mostly NOT defined by these codes, but by the underlying solver. (The exception to that is `emg3d`, which is its own solver.)
- In the end it comes down to the most important, crucial, difficult, AND TIME CONSUMING point: mesh generation. It took all of us MUCH more time to generate appropriate meshes, then to actually compute the responses.
- very different application area. E.g. `emg3d` is for f-domain, CSEM; not too low frequencies, as it becomes slow (null-space). `SimPEG`, on the other hand, has from 1D, 2D, 3D, DC, IP, Magnetic, TEM, inversion, you-name-it things; so from very specialist to very broad.

CONCLUSIONS & OUTLOOK

We compared four different open-source 3D CSEM modellers by computing responses for a layered, anisotropic model, including a modification with three resistive blocks, and the realistic marine Marlim R3D model. All comparisons exhibited an excellent reliability of the solutions. Our data should make it very easy for new codes to have a readily available data set to test against and verify. We are confident that the discussions about runtimes, memory consumption, solver choices, and, most important, discretizations of the 3D CSEM problems help potential users not only to decide which modeller is best suited for specific tasks, but also to support their individual projects.

As none of the considered methods is best suited for all problems, it is important to have more test models for various scenarios in future. Our study is limited to marine CSEM cases in the frequency-domain. Reasonable extensions include providing benchmark models for land, airborne, or mixed CSEM environments as well as time-domain data. Another addition to this work would be focusing on complex irregular models, tailored for an FE or FV code based on unstructured meshes, and to compare how the FD codes can cope with it.

We encourage the community to not only work on new open-source code developments but also to create a landscape of easily accessible benchmark models for increasing the number of reliable and reproducible solutions. Our study could provide one of the primary inputs for this task. We are confident that the mentioned development will and should be taking place, taking into account the current general trend in science to open-source papers, codes or data.

ACKNOWLEDGEMENT

We would like to thank Paulo Menezes for the help and explanations with regards to the Marlim R3D model and corresponding CSEM data, and for making their actual computation model available under an open-source license.

The work of D.W. was conducted within the Gitaro.JIM project funded through MarTERA, a *European Union's Horizon 2020* research and innovation programme, grant agreement N° 728053; martera.eu.

The development of `custEM` as part of the DESMEX project was funded by the Germany Ministry for Education and Research (BMBF) in the framework of the research and development program Fona-r4 under grant 033R130D.

The work of O.C-R. has received funding from the *European Union's Horizon 2020 programme* under the *Marie Sklodowska-Curie* grant agreement N° 777778. Further, the development of `PETGEM` has received funding from the *European Union's Horizon 2020 programme*, grant agreement N° 828947, and from the Mexican Department of Energy, CONACYT-SENER Hidrocarburos grant agreement N° B-S-69926. Furthermore, O.C-R. has been 65% cofinanced by the European Regional Development Fund (ERDF) through the Interreg V-A Spain-France-Andorra program (POCTEFA2014-2020). POCTEFA aims to reinforce the economic and social integration of the French-Spanish-Andorran border. Its support is focused on developing economic, social and environmental cross-border activities through joint strategies favouring sustainable territorial development.

DATA

All files to rerun the different models with the four codes and reproduce the shown results are available at ....

Put up on Zenodo, link here.

REFERENCES

Abhyankar, S., J. Brown, E. M. Constantinescu, D. Ghosh, B. F. Smith, and H. Zhang, 2018, PETSc/TS: A modern scalable ODE/DAE solver library: arXiv preprint arXiv:1806.01437.

Alumbaugh, D. L., G. A. Newman, L. Prevost, and J. N. Shadid, 1996, Three-dimensional wideband electromagnetic modeling on massively parallel computers: Radio Science, **31**, no. 1, 1–23; doi: 10.1029/95RS02815.

Amestoy, P. R., I. S. Duff, J. Koster, and J.-Y. L'Excellent, 2001, A fully asynchronous multifrontal solver using distributed dynamic scheduling: SIAM Journal on Matrix Analysis and Applications, **23**, no. 1, 15–41; doi: 10.1137/S0895479899358194.

Aruliah, D., and U. Ascher, 2002, Multigrid preconditioning for Krylov methods for time-harmonic Maxwell's equations in three dimensions: SIAM Journal on Scientific Computing, **24**, no. 2, 702–718; doi: 10.1137/S1064827501387358.

Astic, T., L. J. Heagy, and D. W. Oldenburg, 2020, Petrophysically and geologically guided multi-physics inversion using a dynamic Gaussian mixture model: (in review) Geophysical Journal International.

Avdeev, D. B., 2005, Three-dimensional electromagnetic modelling and inversion from theory to application: Surveys in Geophysics, **26**, no. 6, 767–799; doi: 10.1007/s10712-005-1836-x.

Broggini, F., J. Dellinger, S. Fomel, and Y. Liu, 2017, Reproducible research: Geophysics papers of the future–Introduction: Geophysics, **82**, no. 6, WBi–WBii; doi: 10.1190/geo2017-0918-spseintro.1.

Börner, R.-U., 2010, Numerical modelling in geo-electromagnetics: Advances and challenges: Surveys in Geophysics, **31**, no. 2, 225–245; doi: 10.1007/s10712-009-9087-x.

Cai, H., X. Hu, J. Li, M. Endo, and B. Xiong, 2017, Parallelized 3D CSEM modeling using edge-based finite element with total field formulation and unstructured mesh: Computers & Geosciences, **99**, 125–134; doi: 10.1016/j.cageo.2016.11.009.

Cai, H., B. Xiong, M. Han, and M. Zhdanov, 2014, 3D controlled-source electromagnetic modeling in anisotropic medium using edge-based finite element method: Computers & Geosciences, **73**, 164–176; doi: 10.1016/j.cageo.2014.09.008.

Carvalho, B. R., and P. T. L. Menezes, 2017, Marlim R3D: a realistic model for CSEM simulations–phase I: model building: Brazilian Journal of Geology, **47**, no. 4, 633–644; doi: 10.1590/2317-4889201720170088.

Castillo-Reyes, O., J. de la Puente, and J. M. Cela, 2018, PETGEM: A parallel code for 3D CSEM forward modeling using edge finite elements: Computers & Geosciences, **119**, 126–136; doi: 10.1016/j.cageo.2018.07.005.

Castillo-Reyes, O., J. de la Puente, L. E. García-Castillo, and J. M. Cela, 2019, Parallel 3D marine controlled-source electromagnetic modeling using high-order tetrahedral nédélec elements: Geophysical Journal International, **219**, 39–65; doi: 10.1093/gji/ggz285.

Chung, Y., J.-S. Son, T. J. Lee, H. J. Kim, and C. Shin, 2014, Three-dimensional modelling of controlled-source electromagnetic surveys using an edge finite-element method with a direct solver: Geophysical Prospecting, **62**, no. 6, 1468–1483; doi: 10.1111/1365-2478.12132.

Clemens, M., and T. Weiland, 2001, Discrete electromagnetism with the finite integration technique: PIER, **32**, 65–87; doi: 10.2528/PIER00080103.

Cockett, R., S. Kang, L. J. Heagy, A. Pidlisecky, and D. W. Oldenburg, 2015, SimPEG: An open source framework for simulation and gradient based parameter estimation in geophysical applications: Computers & Geosciences, **85**, 142–154; doi: 10.1016/j.cageo.2015.09.015.

Commer, M., and G. Newman, 2004, A parallel finite-difference approach for 3D transient electromagnetic modeling with galvanic sources: Geophysics, **69**, no. 5, 1192–1202; doi: 10.1190/1.1801936.

Correa, J. L., and P. T. L. Menezes, 2019, Marlim R3D: A realistic model for controlled-source electromagnetic simulations–Phase 2: The controlled-source electromagnetic data set: Geophysics, **84**, no. 5, E293–E299; doi: 10.1190/geo2018-0452.1.

da Silva, N. V., J. V. Morgan, L. MacGregor, and M. Warner, 2012, A finite element multifrontal method for 3D CSEM modeling in the frequency domain: Geophysics, **77**, no. 2, E101–E115; doi: 10.1190/geo2010-0398.1.

Das, U. C., and S. K. Verma, 1982, Electromagnetic response of an arbitrarily shaped three-dimensional conductor in a layered earth – numerical results: Geophysical Journal International, **69**, no. 1, 55–66; doi: 10.1111/j.1365-246X.1982.tb04935.x.

de la Varga, M., A. Schaaf, and F. Wellmann, 2019, Gempy 1.0: open-source stochastic geological modeling and inversion: Geoscientific Model Development, **12**, no. 1, 1–32; doi: 10.5194/gmd-12-1-2019.

Druskin, V., and L. Knizhnerman, 1994, Spectral approach to solving three-dimensional maxwell's diffusion equations in the time and frequency domains: Radio Science, **29**, no. 4, 937–953; doi: 10.1029/94RS00747.

Fedorenko, R. P., 1964, The speed of convergence of one iterative process: USSR Computational Mathematics and Mathematical Physics, **4**, no. 3, 227–235; doi: 10.1016/0041-5553(64)90253-8.

Fournier, D., L. J. Heagy, and D. W. Oldenburg, 2020, Sparse magnetic vector inversion in spherical coordinates: Geophysics, **85**, no. 3, J33–J49; doi: 10.1190/geo2019-0244.1.

Grayver, A. V., and T. V. Kolev, 2015, Large-scale 3D geoelectromagnetic modeling using parallel adaptive high-order finite element method: Geophysics, **80**, no. 6, E277–E291; doi: 10.1190/geo2015-0013.1.

Grayver, A. V., R. Streich, and O. Ritter, 2013, Three-dimensional parallel distributed inversion of CSEM data using a direct forward solver: Geophysical Journal International, **193**, no. 3, 1432–1446; doi: 10.1093/gji/ggt055.

Haber, E., and U. M. Ascher, 2001, Fast finite volume simulation of 3D electromagnetic problems with highly discontinuous coefficients: SIAM Journal on Scientific Computing, **22**, no. 6, 1943–1961; doi: 10.1137/S1064827599360741.

Haber, E., and S. Heldmann, 2007, An octree multigrid method for quasi-static maxwell's equations with highly discontinuous coefficients: Journal of Computational Physics, **223**, no. 2, 783–796; doi: https://doi.org/10.1016/j.jcp.2006.10.012.

Heagy, L. J., R. Cockett, S. Kang, G. K. Rosenkjaer, and D. W. Oldenburg, 2017, A framework for simulation and inversion in electromagnetics: Computers and Geosciences, **107**, 1–19; doi: 10.1016/j.cageo.2017.06.018.

Hohmann, G. W., 1975, Three-dimensional induced polarization and electromagnetic modeling: Geophysics, **40**, no. 2, 309–324; doi: 10.1190/1.1440527.

Hursán, G., and M. S. Zhdanov, 2002, Contraction integral equation method in three-dimensional electromagnetic modeling: Radio Science, **37**, no. 6, 1–1–1–13; doi: 10.1029/2001RS002513.

Jahandari, H., and C. G. Farquharson, 2014, A finite-volume solution to the geophysical electromagnetic forward problem using unstructured grids: Geophysics, **79**, no. 6, E287–E302; doi: 10.1190/geo2013-0312.1.

Jaysaval, P., D. Shantsev, and S. de la Kethulle de Ryhove, 2014, Fast multimodel finite-difference controlled-source electromagnetic simulations based on a Schur complement approach: Geophysics, **79**, no. 6, E315–E327; doi: 10.1190/geo2014-0043.1.

Jaysaval, P., D. V. Shantsev, and S. de la Kethulle de Ryhove, 2015, Efficient 3-D controlled-source electromagnetic modelling using an exponential finite-difference method: Geophysical Journal International, **203**, no. 3, 1541–1574; doi: 10.1093/gji/ggv377.

Jaysaval, P., D. V. Shantsev, S. de la Kethulle de Ryhove, and T. Bratteland, 2016, Fully anisotropic 3-D EM modelling on a Lebedev grid with a multigrid pre-conditioner: Geophysical Journal International, **207**, no. 3, 1554–1572; doi: 10.1093/gji/ggw352.

Kruglyakov, M., and L. Bloshanskaya, 2017, High-performance parallel solver for integral equations of electromagnetics based on Galerkin method: Mathematical Geosciences, **49**, no. 6, 751–776; doi: 10.1007/s11004-017-9677-y.

Kruglyakov, M., A. Geraskin, and A. Kuvshinov, 2016, Novel accurate and scalable 3-D MT forward solver based on a contracting integral equation method: Computers & Geosciences, **96**, 208–217; doi: 10.1016/j.cageo.2016.08.017.

Lam, S. K., A. Pitrou, and S. Seibert, 2015, Numba: A LLVM-based python JIT compiler: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, 1–6;

doi: 10.1145/2833157.2833162.

Langtangen, H. P., A. Logg, and A. Tveito, 2016, Solving PDEs in Python: The FEniCS Tutorial I: Springer International Publishing, volume **3** *of* Simula SpringerBriefs on Computing; doi: 10.1007/978-3-319-52462-7.

Lebedev, V. I., 1964, Difference analogues of orthogonal decompositions, basic differential operators and some boundary problems of mathematical physics. I: USSR Computational Mathematics and Mathematical Physics, **4**, no. 3, 69–92; doi: 10.1016/0041-5553(64)90240-X.

Liu, R., R. Guo, J. Liu, C. Ma, and Z. Guo, 2018, A hybrid solver based on the integral equation method and vector finite-element method for 3D controlled-source electromagnetic method modeling: Geophysics, **83**, no. 5, E319–E333; doi: 10.1190/geo2017-0502.1.

Logg, A., K.-A. Mardal, and G. Wells, 2012, Automated solution of differential equations by the finite element method: The FEniCS book: Springer-Verlag Berlin Heidelberg, volume **84** *of* Lecture Notes in Computational Science and Engineering; doi: 10.1007/978-3-642-23099-8.

Maaø, F. A., 2007, Fast finite-difference time-domain modeling for marine-subsurface electromagnetic problems: Geophysics, **72**, no. 2, A19–A23; doi: 10.1190/1.2434781.

Mackie, R. L., J. T. Smith, and T. R. Madden, 1994, Three-dimensional electromagnetic modeling using finite difference equations: the magnetotelluric example.: Radio Science, **29**, no. 4, 923–935; doi: 10.1029/94RS00326.

Madsen, N. K., and R. W. Ziolkowski, 1990, A three-dimensional modified finite volume technique for Maxwell's equations: Electromagnetics, **10**, no. 1-2, 147–161; doi: 10.1080/02726349008908233.

Miensopust, M. P., P. Queralt, A. G. Jones, and the 3D MT modellers, 2013, Magnetotelluric 3-D inversion—a review of two successful workshops on forward and inversion code testing and comparison: Geophysical Journal International, **193**, no. 3, 1216–1238; doi: 10.1093/gji/ggt066.

Mittet, R., 2010, High-order finite-difference simulations of marine CSEM surveys using a correspondence principle for wave and diffusion fields: Geophysics, **75**, F33–F50; doi: 10.1190/1.3278525.

Mulder, W. A., 2006, A multigrid solver for 3D electromagnetic diffusion: Geophysical Prospecting, **54**, no. 5, 633–649; doi: 10.1111/j.1365-2478.2006.00558.x.

Newman, G. A., and D. L. Alumbaugh, 1997, Three-dimensional massively parallel electromagnetic inversion–I. Theory: Geophysical Journal International, **128**, no. 2, 345–354; doi: 10.1111/j.1365-246X.1997.tb01559.x.

Newman, G. A., G. W. Hohmann, and W. L. Anderson, 1986, Transient electromagnetic response of a three-dimensional body in a layered earth: Geophysics, **51**, no. 8, 1608–1627; doi: 10.1190/1.1442212.

Oh, S., K. Noh, S. J. Seol, and J. Byun, 2015, 3D CSEM frequency-domain modeling and inversion algorithms including topography: SEG Technical Program Expanded Abstracts, 828–832; doi: 10.1190/segam2015-5898964.1.

Oldenburg, D. W., L. J. Heagy, S. Kang, and R. Cockett, 2019, 3D electromagnetic modelling and inversion: a case for open source: Exploration Geophysics, **0**, no. 0, 1–13; doi: 10.1080/08123985.2019.1580118.

Oristaglio, M., and B. Spies, 1999, Three-dimensional electromagnetics: Society of Exploration Geophysicists, volume **7** *of* Geophysical Developments; doi: 10.1190/1.9781560802154.

Puzyrev, V., J. Koldan, J. de la Puente, G. Houzeaux, M. Vázquez, and J. M. Cela,

2013, A parallel finite-element method for three-dimensional controlled-source electromagnetic forward modelling: Geophysical Journal International, **193**, no. 2, 678–693; doi: 10.1093/gji/ggt027.

Raiche, A. P., 1974, An integral equation approach to three-dimensional modelling: Geophysical Journal International, **36**, no. 2, 363–376; doi: 10.1111/j.1365-246X.1974.tb03645.x.

Rochlitz, R., N. Skibbe, and T. Günther, 2019, custEM: customizable finite element simulation of complex controlled-source electromagnetic data: Geophysics, **84**, no. 2, F17–F33; doi: 10.1190/geo2018-0208.1.

Rücker, C., T. Günther, and F. M. Wagner, 2017, pyGIMLi: An open-source library for modelling and inversion in geophysics: Computers & Geosciences, **109**, 106–123; doi: 10.1016/j.cageo.2017.07.011.

Schenk, O., and K. Gärtner, 2004, Solving unsymmetric sparse systems of linear equations with PARDISO: Future Generation Computer Systems, **20**, no. 3, 475–487; doi: 10.1016/j.future.2003.07.011.

Schwarzbach, C., R.-U. Börner, and K. Spitzer, 2011, Three-dimensional adaptive higher order finite element simulation for geo-electromagnetics—a marine csem example: Geophysical Journal International, **187**, no. 1, 63–74; doi: 10.1111/j.1365-246X.2011.05127.x.

Si, H., 2015, Tetgen, a Delaunay-based quality tetrahedral mesh generator: ACM Transactions on Mathematical Software (TOMS), **41**, no. 2, 1–36; doi: 10.1145/2629697.

Skibbe, N., R. Rochlitz, T. Günther, and M. Müller-Petke, 2020, Coupled magnetic resonance and electrical resistivity tomography: An open-source toolbox for surface nuclear-magnetic resonance: Geophysics, **85**, no. 3, F53–F64; doi: 10.1190/geo2019-0484.1.

Sommer, M., S. Hölz, M. Moorkamp, A. Swidinsky, B. Heincke, C. Scholl, and M. Jegen, 2013, GPU parallelization of a three dimensional marine CSEM code: Computers & Geosciences, **58**, 91–99; doi: 10.1016/j.cageo.2013.04.004.

Streich, R., 2009, 3D finite-difference frequency-domain modeling of controlled-source electromagnetic data: Direct solution and optimization for high accuracy: Geophysics, **74**, no. 5, F95–F105; doi: 10.1190/1.3196241.

Sullivan, C. B., and A. A. Kaszynski, 2019, PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK): Journal of Open Source Software, **4**, no. 37, 1450; doi: 10.21105/joss.01450.

Tehrani, A. M., and E. Slob, 2010, Fast and accurate three-dimensional controlled source electromagnetic modelling†: Geophysical Prospecting, **58**, no. 6, 1133–1146; doi: 10.1111/j.1365-2478.2010.00876.x.

Uieda, L., 2018, Verde: Processing and gridding spatial data using Green's functions: Journal of Open Source Software, **3**, no. 29, 957; doi: 10.21105/joss.00957.

van der Walt, S., S. C. Colbert, and G. Varoquaux, 2011, The NumPy array: A structure for efficient numerical computation: Computing in Science Engineering, **13**, no. 2, 22–30; doi: 10.1109/MCSE.2011.37.

Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, 2020, SciPy 1.0: Fundamental algorithms for scientific computing in python: Nature Methods, **17**, 261–272; doi: 10.1038/s41592-019-0686-2.

Wang, F., J. P. Morten, and K. Spitzer, 2018, Anisotropic three-dimensional inversion of CSEM data using finite-element techniques on unstructured grids: Geophysical Journal International, **213**, no. 2, 1056–1072; doi: 10.1093/gji/ggy029.

Wang, T., and G. W. Hohmann, 1993, A finite-difference, time-domain solution for three-dimensional electromagnetic modeling: Geophysics, **58**, no. 6, 797–809; doi: 10.1190/1.1443465.

Wannamaker, P. E., G. W. Hohmann, and S. H. Ward, 1984, Magnetotelluric responses of three-dimensional bodies in layered earths: Geophysics, **49**, no. 9, 1517–1533; doi: 10.1190/1.1441777.

Wannamaker, P. E., and M. Zhdanov, 2002, Three-dimensional electromagnetics: Elsevier Publishing Company. Methods in Geochemistry and Geophysics, No. 35; ISBN 978-0-080-54299-7.

Ward, S. H., and G. W. Hohmann, 1988, 4, *in* Electromagnetic Theory for Geophysical Applications: Society of Exploration Geophysicists, 130–311; doi: 10.1190/1.9781560802631.ch4.

Weiland, T., 1977, Eine Methode zur Lösung der Maxwellschen Gleichungen für sechskomponentige Felder auf diskreter Basis: Archiv für Elektronik und Übertragungstechnik, **31**, no. 3, 116–120; pdf: https://www.leibniz-publik.de/de/fs1/object/display/bsb00064886_00001.html.

Werthmüller, D., 2017, An open-source full 3D electromagnetic modeler for 1D VTI media in Python: empymod: Geophysics, **82**, no. 6, WB9–WB19; doi: 10.1190/geo2016-0626.1.

Werthmüller, D., W. A. Mulder, and E. C. Slob, 2019, emg3d: A multigrid solver for 3D electromagnetic diffusion: Journal of Open Source Software, **4**, no. 39, 1463; doi: 10.21105/joss.01463.

Yee, K., 1966, Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media: IEEE Transactions on Antennas and Propagation, **14**, no. 3, 302–307; doi: 10.1109/TAP.1966.1138693.

Zhang, Y., and K. Key, 2016, MARE3DEM: A three-dimensional CSEM inversion based on a parallel adaptive finite element method using unstructured meshes: SEG Technical Program Expanded Abstracts, 1009–1013; doi: 10.1190/segam2016-13681445.1.

Zhdanov, M. S., S. K. Lee, and K. Yoshioka, 2006, Integral equation method for 3D modeling of electromagnetic fields in complex structures with inhomogeneous background conductivity: Geophysics, **71**, no. 6, G333–G345; doi: 10.1190/1.2358403.