Open-source landscape for 3D CSEM modelling

Dieter Werthmüller[1], Raphael Rochlitz[2], Octavio Castillo-Reyes[3] and Lindsey Heagy[4]

[1] TU Delft, Building 23, Stevinweg 1 / PO-box 5048, 2628 CN Delft (NL). E-mail: Dieter@Werthmuller.org

[2] Leibniz Institute for Applied Geophysics, Stilleweg 2, 30655 Hannover (DE)

[3] Barcelona Supercomputing Center (BSC), Nexus II Building c/Jordi Girona, 29, 08034 Barcelona (ES)

[4] University of California Berkeley, Department of Statistics (USA)

Received 202Y MMMM DD; in original form 2020 October 23

SUMMARY

Large-scale modelling of three-dimensional controlled-source electromagnetic (CSEM) surveys used to be feasible only for large companies and research consortia. This has changed over the last few years, and today there exists a selection of different open-source codes. Using four different codes in the Python ecosystem, we perform simulations for increasingly complex models. We first verify the computed fields with semi-analytical solutions for a simple layered model. Then we validate the responses of a more complex block model by comparing results obtained from each code. Finally we compare the responses of a real world model with results from the industry. These validations show that the open-source codes are able to compute comparable CSEM responses for challenging, large-scale models. Our comparison includes finite-element and finite-volume codes using structured rectilinear and octree meshes as well as unstructured tetrahedral meshes. Accurate responses can be obtained independently of the chosen method and the chosen mesh type. The runtime and memory requirements vary greatly based on the choice of iterative or direct solvers. However, we have found that much more time was spent on designing the mesh and setting up the simulations than running the actual computation. The challenging task is, irrespective of the chosen code, to appropriately discretize the model. We provide three models, each with their corresponding discretization and responses of four codes, which can be used for validation of new and existing codes. The collaboration of four code maintainers trying to achieve the same task brought in the end all four codes a significant step further. This includes improved meshing and interpolation capabilities, resulting in shorter runtimes for the same accuracy. We hope that these results may be useful for the CSEM community at large and that we can build over time a suite of benchmarks that will help to increase the confidence in existing and new 3D CSEM codes.

Key words: Controlled source electromagnetics (CSEM); Numerical modelling; Electrical properties.

1 INTRODUCTION

Controlled-source electromagnetic (CSEM) measurements are a frequently applied method in various areas of geophysical exploration, such as geothermal, groundwater, oil and gas, mining, civil engineering, or geohazards. 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 (EM) methods started to appear as early as the 1970's and 1980's. These early publications were integral equation (IE) methods for simulating an anomaly embedded within a layered medium, mostly for loop-loop type transient EM measurements (Raiche 1974; Hohmann 1975; Das & Verma 1982; Newman et al. 1986) and magnetotelluric (MT) measurements (Wannamaker et al. 1984). Ward & Hohmann (1988) assemble many of these solutions in *Electromag-*

netic Theory for Geophysical Applications, which is widely viewed as an authoritative publication on electromagnetic geophysics.

In the 1990's, computers 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* (Oristaglio & Spies 1999), and another book by Wannamaker & 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 & Knizhnerman (1994) for frequency- and time-domain modelling using a Yee grid and a global Krylov subspace approximation; and Alumbaugh et al. (1996); Newman & 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 commonly applied methods to solve Maxwell's equation are the IE method (Raiche 1974; Hursán & Zhdanov 2002; Zhdanov et al. 2006; Tehrani & Slob 2010; Kruglyakov et al. 2016; Kruglyakov & Bloshanskaya 2017) and different variations of the differential equation (DE) method, such as finite differences (FD) (Wang & Hohmann 1993; Druskin & Knizhnerman 1994; Mackie et al. 1994; Streich 2009; Sommer et al. 2013), finite elements (FE) (Commer & Newman 2004; Schwarzbach et al. 2011; da Silva et al. 2012; Grayver et al. 2013; Puzyrev et al. 2013; Zhang & Key 2016), and finite volumes (FV) (Madsen & Ziolkowski 1990; Haber & Ascher 2001; Clemens & Weiland 2001; Jahandari & Farquharson 2014). There are also many different types of discretization, where the most common ones are regular grids (Cartesian, rectilinear, curvilinear), mostly using a Yee grid (Yee 1966) or a Lebedev grid (Lebedev 1964), but also unstructured tetrahedral grids (Zhang & Key 2016; Cai et al. 2017), hexagonal meshes (Cai et al. 2014), or octree meshes (Haber & Heldmann 2007).

Very well written overviews about the different approaches to 3D EM modelling (and inversion) 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 & Kolev 2015; Oh et al. 2015; Wang et al. 2018), indirect solvers (Mulder 2006; 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 & 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.* » Similarly, 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.

Today 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 consideration are `pyGIMLi` (Rücker et al. 2017), `Fatiando` (Uieda 2018), `GemPy` (de la Varga et al. 2019), and `PyVista` (Sullivan & Kaszynski 2019).

The open-source landscape in the field of 3D CSEM modelling changed in the last five years quite dramatically, and here we introduce and compare four recent open-source projects. 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. Having different codes which use different methods and different meshes is ideal to address the topic of validation. Analytical and semi-analytical solutions only exist for simple halfspace or layered-Earth models, which served mostly to verify new codes. The only objective possibility to ensure the accuracy of solutions beyond these simple models is by comparing results from different modellers. If different discretizations and implementations of Maxwell's equations

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 & Gärtner 2004).

	`custEM`	`emg3d`	`PETGEM`	`SimPEG`
Website	custem.rtfd.io	empymod.github.io	petgem.bsc.es	simpeg.xyz
License	GPL-3.0	Apache-2.0	BSD-3-Clause	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, octree
BC	PEC; PMC	PEC	PEC	PEC; PMC
Solver	`MUMPS`	`emg3d`	`PETSc`; `MUMPS`	`PARDISO`; `MUMPS`

yield the same result, it gives confidence in their accuracy. Miensopust et al. (2013) presents a review of two workshops dealing with the validation of MT forward and inversion codes, but we are not aware of any comparable comparison study or benchmark suite for CSEM data.

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.

2 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` (Harris et al. 2020) and `SciPy` (Virtanen et al. 2020). The four of them 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` simplify installation for all of these codes on any major operating system

We briefly present our codes in this section. 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 codes is given in Table 1. All four 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 usually applied in CSEM. The machines on which the different codes were run are listed in Table 2 together with the responsible operator. 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-world 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 time or CPU time is therefore much higher than the reported runtime for the codes that run in parallel, with an upper limit of #Procs×Runtime. Memory refers to the maximum memory increase at any point of solving the SLE.

2.1 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).

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

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)** - 2 sockets Intel Xeon Platinum (Skylake generation) 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

The toolbox considers generally anisotropic petrophysical properties. Even though changes of the conductivity are 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-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.

2.2 emg3d

The 3D CSEM code `emg3d` is a multigrid solver (Fedorenko 1964) for electromagnetic diffusion following Mulder (2006), including tri-axial electrical anisotropy. The matrix-free solver can be used as main solver or 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 primarily a solver, which solves Maxwell's equations for a single frequency and a single, electric source. 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 to obtain time-domain responses. For the underlying discretization the Python package `discretize` is used, which is part of the larger `SimPEG` ecosystem.

2.3 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 to discretize the governing equations in its diffusive form. This technique provides a suitable mechanism to obtain stable numerical solutions and a good trade-off between the 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). 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 [HPC] architectures).

2.4 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, EM (time and frequency, as well as natural sources), and fluid flow. It is a community driven project that aims to support researchers and practitioners by providing a flexible, extensible framework, and common interface to a variety of geophysical methods.

Meshes and finite volume differential operators are implemented in the `discretize` package, which currently includes rectilinear, 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 ($\vec{e}$) and the magnetic flux density ($\vec{b}$) and (II) the HJ-formulation, which discretizes the magnetic field ($\vec{h}$) and the current density ($\vec{j}$). 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`.

3 NUMERICAL VALIDATION

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

3.1 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 uses a 200 m long, 800 A source with frequency $f = 1\,\text{Hz}$ at a single position, and three receiver lines of 101 receivers each. The centre of the x-oriented source is at $x = y = 0\,\text{m}$, 50 m above the seafloor. The x-directed receivers are placed on the seafloor every 200 m from $x = -10\,\text{km}$ to $x = +10\,\text{km}$ in three lines with $y = -3; 0; +3\,\text{km}$.

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 verified with semi-analytical results, and is therefore still a valuable test. The results are shown in Figure 1, 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, for which we used `empymod` (Werthmüller 2017). The second and the third row show the relative percentage error of the inline and the broadside receivers, respectively; note that for the 1D case the two broadside lines for $y = \pm 3\,\text{km}$ are identical.

There are a few noteworthy points:

(i) The error of the real and imaginary responses are dominated by their zero-crossings, which blows up the error as is the nature of the relative error formula, it has no real-world implications.

(ii) It can be seen from the absolute-value errors that the misfit is generally in the order of 1 % or less.

(iii) The errors become substantial towards zero offset, particularly for the results of the FV codes. These comparably significant errors could be significantly reduced by choosing a finer discretization around the source, at the cost of increased computation time.

(iv) The error of the responses from the FE codes, 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

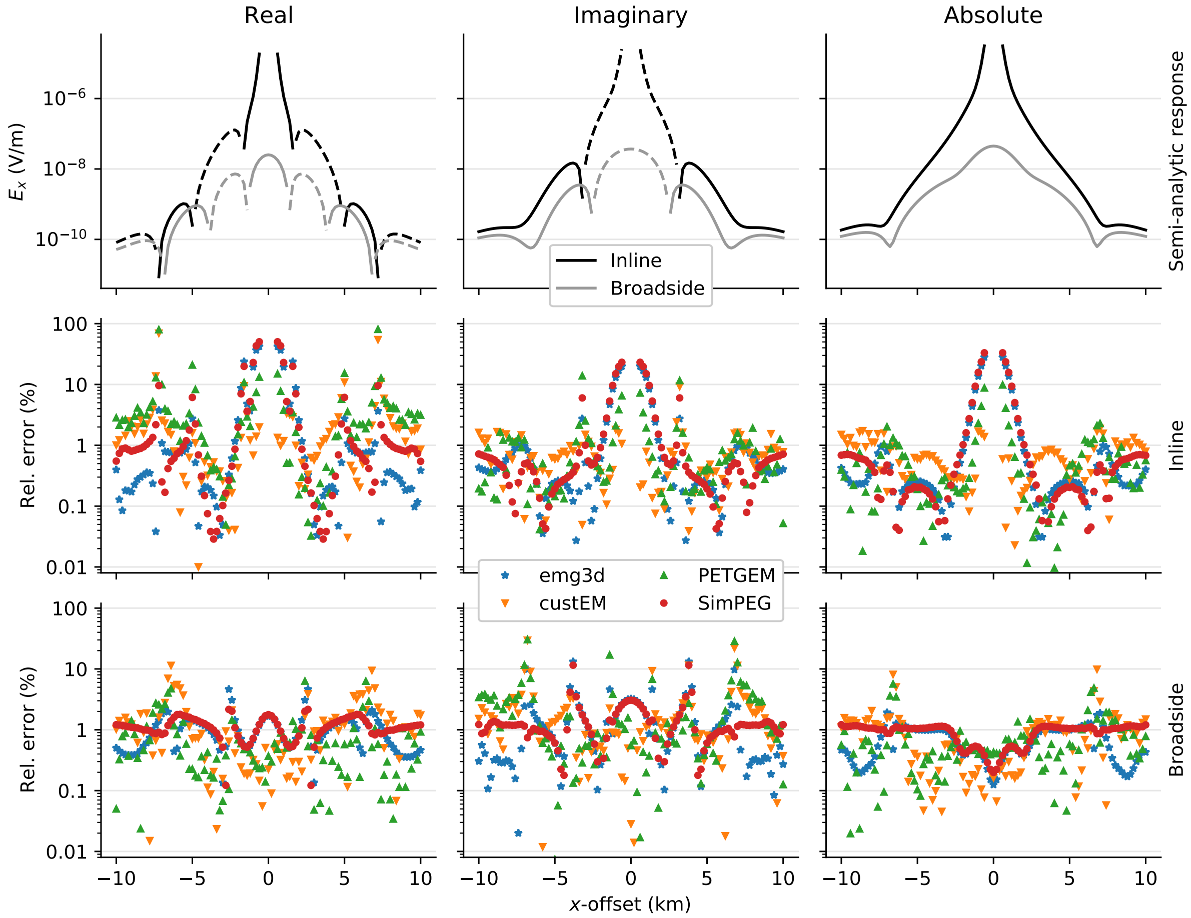

Figure 1. Results of the layered model: The semi-analytical inline and broadside responses are shown in the top row. The relative percentage error of the four 3D modellers are shown in the second and the third row for the inline and broadside responses, respectively.

up to 30 km from the origin, surrounded by a ten times larger halfspace-like boundary mesh with water conductivities assigned to the subsurface. This approximation significantly reduces the problem size compared to considering the layered-earth geometry for the whole computational domain.

Note that `emg3d` and `SimPEG` used the same rectilinear mesh in this example. The use of an octree mesh in `SimPEG` could greatly reduce the size of the mesh and resultant computation time, but at the cost of accuracy, which is reduced when averaging over layer interfaces to coarsen cells and because of the factor-of-two expansion rate of coarsened cells. As the aim of this study is to focus on accuracy of the simulations and compare discretization approaches, we chose to use the same mesh as in the `emg3d` simulation.

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

3.2 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

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.

Code	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	24	118	97.8	1 531 824
`emg3d`	1	116	0.4	4 813 216
`PETGEM`	24	252	151.7	2 455 868
`SimPEG`	4	10 089	280.3	4 813 216

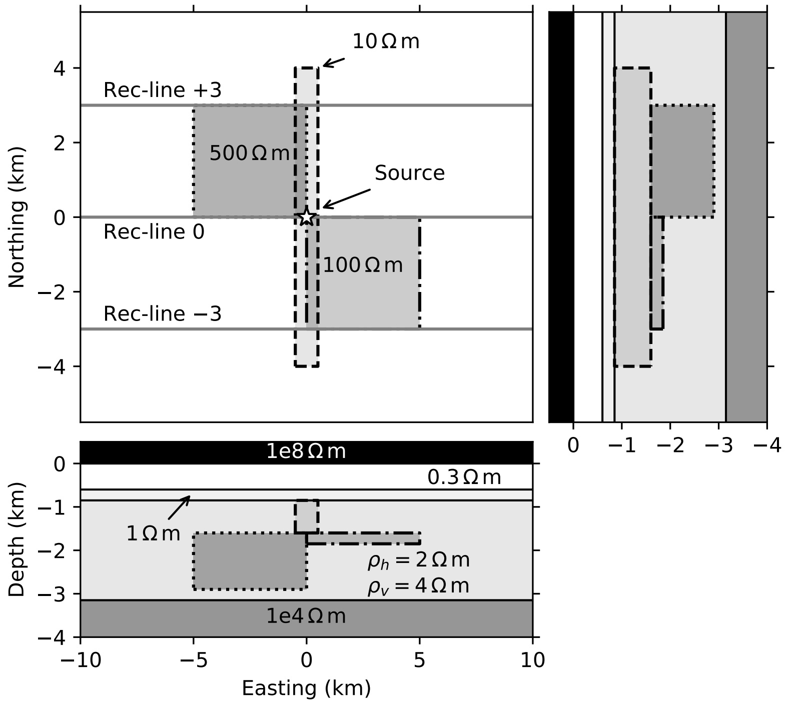

Figure 2. Sketch of the block model, consisting of the layered background model with three resistive blocks, embedded in the thick background layer which has VTI with $\lambda = \sqrt{\rho_\text{v}/\rho_\text{h}} = \sqrt{2}$.

typical marine CSEM problem, as shown in Figure 2. Additionally, we add our *Layered Model* as a VTI background. The three resistive blocks have resistivities of $\rho = 10\,\Omega\,\text{m}$ (shallow beam perpendicular to survey lines), $\rho = 100\,\Omega\,\text{m}$ (thin plate, South-East), and $\rho = 500\,\Omega\,\text{m}$ (cube, North-West).

There are no semi-analytical solutions for such a model to verify our results, and we can only validate them by comparing different results. For this we use the normalised root-mean square difference (NRMSD) between two responses R_1 and R_2, given by

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

to which we refer as simply the *normalised difference* throughout the paper.

The results for the three receiver lines $y = -3; 0; +3\,\text{km}$ 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 normalised differences between the absolute responses of the codes.

A few interesting points stand out:

(i) The error levels between the different codes are overall comparable; the normalised difference is generally below a few percent, increasing towards larger offsets.

(ii) The normalised difference between the FV codes `emg3d` and `SimPEG` is comparably low except at the boundary. The generally low normalised difference is because these codes use the same rectilinear mesh for modelling. The increasing normalised difference towards the boundary is because `emg3d` uses the PEC boundary condition, while `SimPEG` uses the PMC boundary condition in this example.

(iii) The FE codes `custEM` and `PETGEM` show similar boundary effects, but are influenced by the anomaly response (higher errors on right side and now unsymmetrical error distribution).

(iv) The agreement within the same differential equation method (FE or FV) is better than across methods.

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

3.3 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 & 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 under the open creative common license *CC BY 4.0*. Correa & Menezes (2019) computed CSEM data

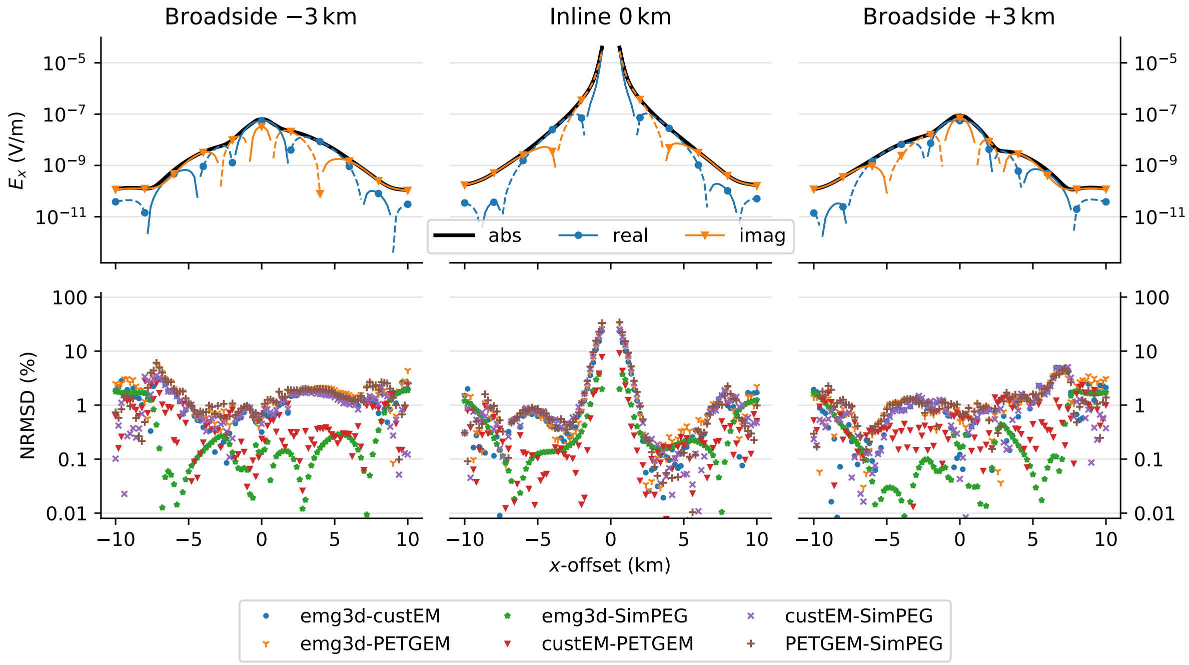

Figure 3. Results of the block model comparison: The responses of `emg3d`, as an example, are shown in the top row, and the normalised differences (%) between the absolute responses $|E_x|$ of the different codes are shown in the bottom row.

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, on one hand, an ideal case to validate our open-source codes against, as it is a complex, realistic model and the data were computed by an industry-proofed code. On the other hand it is impossible to reproduce it exactly, as it is a closed-source code and we cannot know exactly what was done internally. Additionally, that code is a time-domain code obtaining the frequency-domain responses through a transform, and it includes the air layer via a non-local boundary condition at the water-air interface (Mittet 2010). We compute all the results in this study in the frequency domain, where we model the entire domain including the air layer by putting the boundaries of the computational domain far away.

The full MR3D model consists of 1022 × 371 × 1229 cells, totalling to almost 466 million cells, where each cell has dimensions of 25 by 75 by 5 m. The model was upscaled for the computation 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 publication 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.

The responses for all six frequencies and all three electric components are shown in Figure 5, both inline and broadside. The published responses are shown in colour with markers for the frequency, and the responses from our codes are shown beneath in grey colours in this overview plot. Two CSEM data-sets were published with MR3D,

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.

Code	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	24	158	115.9	1 646 736
`emg3d`	1	115	0.4	4 813 216
`PETGEM`	24	238	152.8	2 455 868
`SimPEG`	4	10 295	280.4	4 813 216

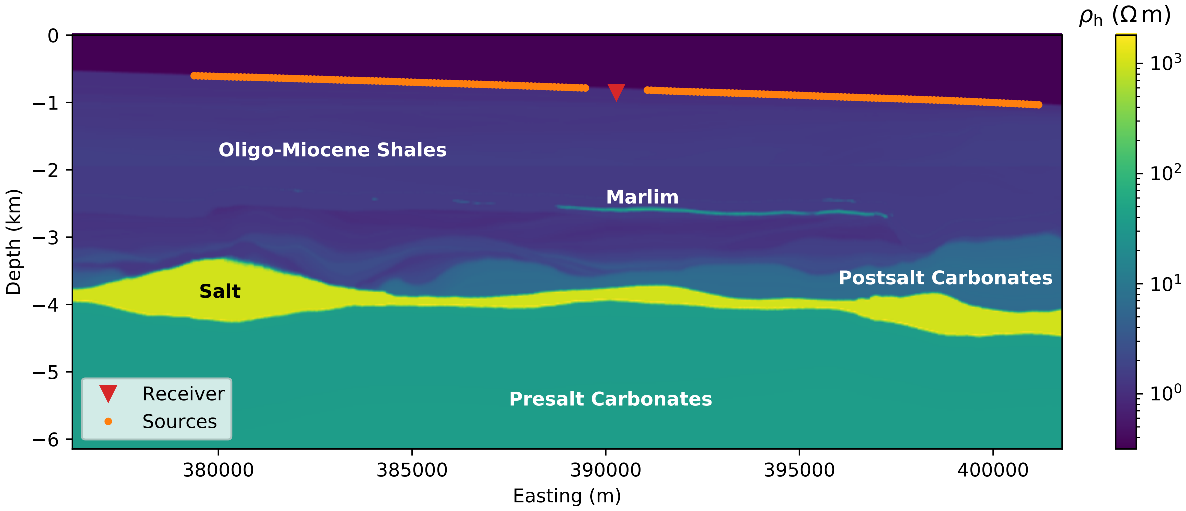

Figure 4. MR3D horizontal resistivity model, x-z-slice through the receiver y-position and with major formations annotated. Air (not shown in the model), seawater, and the salt layer are electrically isotropic, everything else has VTI with $\lambda = \sqrt{2}$. The receiver is located on the seafloor,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

one containing clean responses, and one responses where 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 well. However, there are a few notable points where this is not the case, and they are marked with numbers in the figure.

(i) There are some noticeable differences for positive offsets of the broadside E_y and E_z components, which are likely related to the bathymetry. The reason why we cannot reproduce the published results exactly is that the code is not open-source, and we do not know in detail what it does internally. See the discussion around Figure 6.

(ii) The three highest frequencies become 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. This explains their smooth behaviour, without saying anything about their accuracy.

(iii) 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 comes from 3D effects. These responses are very low, roughly two orders of magnitude lower than the E_x component. Not the published nor our responses are stable, and any of the responses is as bad as the other. Tiniest differences in meshing or different interpolation algorithms will have a huge effect here.

It is important to note that the published results nor any of our results is the *thruth*, but the closer they agree with each other the more certain we can be of the outcomes. Figure 6 shows the normalised differences between the published results and our codes for the three strongest components, the inline E_x and the broadside E_x and E_y components, for three frequencies. A few conclusions can be drawn from this figure: In general the normalised difference is roughly 10 % or less. The structured meshes have in some cases a smaller normalised difference, e.g., for negative offsets in the inline and broadside E_x responses, and in other cases the unstructured tetrahedral meshes, e.g., for positive offsets for all broadside E_y responses, and often they have a comparable NRMSD. Some normalised differences have an interesting step-pattern, which is more pronounced for the structured meshes than for the unstructured meshes; this is related to the annotated point (3) in Figure 4. The origin of this zigzag pattern is most likely the bathymetry. We suspect that the code of the published data did probably use the bathymetry information in addition to the resistivity cube. This would explain why the codes using the tetrahedra mesh, which uses the bathymetry as well, does not show this feature.

The importance of the meshing to the result can be seen in Figure 7, where we compute the normalised differences between our codes. The most obvious result in that figure is that `custEM` and `PETGEM` produce results that are almost identical, their normalised difference is generally below 0.03 %. The reason is simple: `custEM` and `PETGEM` use the exactly same mesh for the computation of this example, the only difference is therefore the solver. This exemplifies that the biggest difference between different codes is the discretization; the solver is important when it comes to runtime and memory, but not for the accuracy of the result. As these two 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 normalised differences between our codes look similar as in the comparison with the published results, and the principal source of the difference must be in the different discretizations.

The actual differences in meshing is shown in the next two figures. Figure 8 shows on the top row the rectilinear

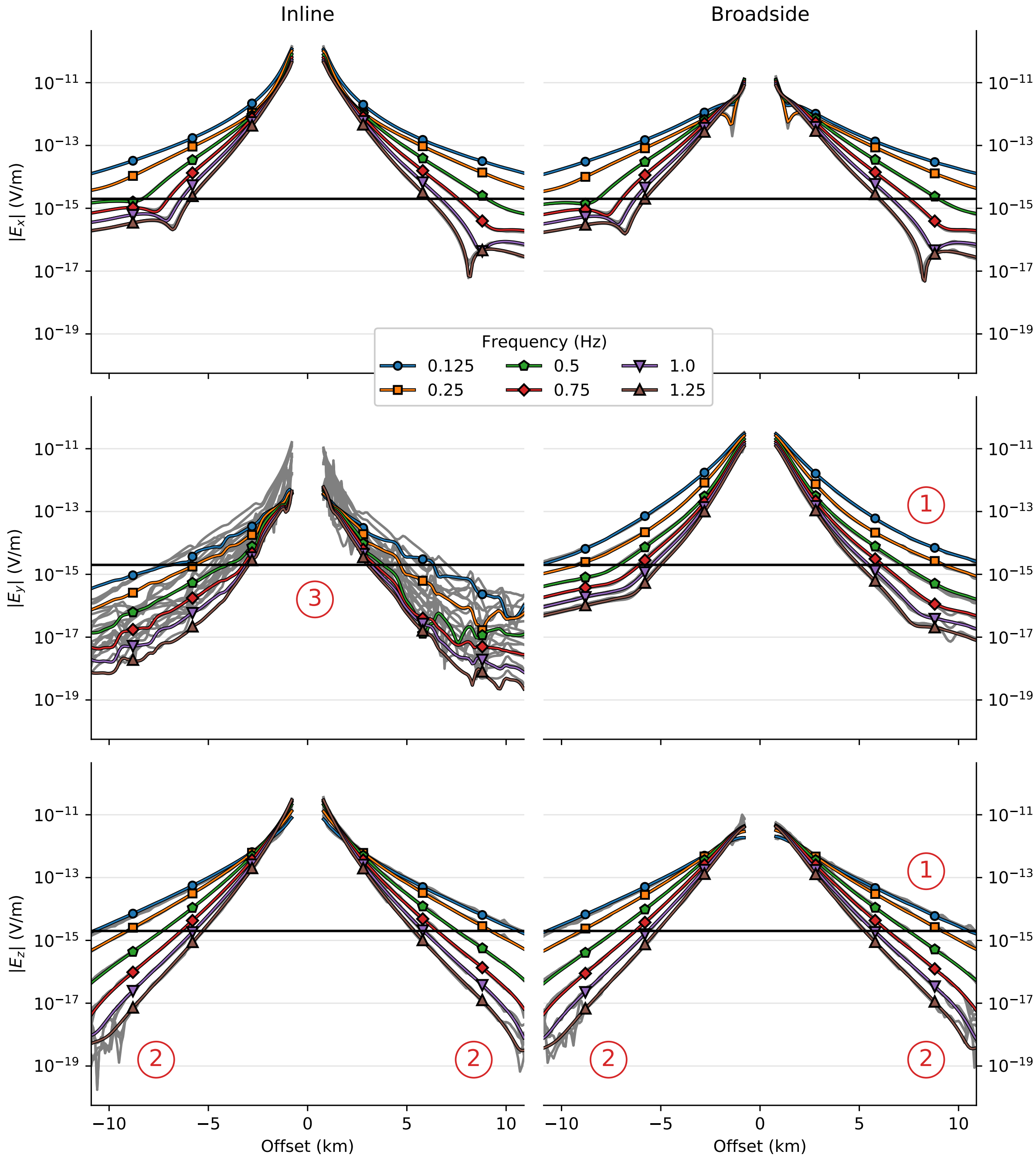

Figure 5. MR3D comparison between our codes (grey lines) and the published data (coloured lines). The grey lines under the coloured lines are not visible in most areas, meaning that they are very similar. However, there are three notable zones, (1) to (3), which are explained further in the text.

mesh of `emg3d`, on the middle row the octree mesh of `SimPEG`, and on the bottom row the tetrahedral mesh of `custEM` and `PETGEM`. The actual mesh and the broadside $|E_x|$ and $|E_y|$ fields are shown in the left, middle, and right columns, respectively. The extent of the plot includes the entire computational domain for the tetrahedral mesh (in x and z direction); the rectilinear and octree mesh extents are larger than what is shown (rectilinear: $x = 327$ to 454 km, $z = -57$ to 66 km; octree: $x = 288$ to 492 km, $z = -123$ to 41 km). This figure is interesting for various points, as the differences between the three meshes are clearly visible. Particularly the advantage of octree and tetrahedra to create large cells outside of the region of interest to push away the boundary can be seen, whereas a rectilinear grid, even though stretched, has to compute many cells far away of the zone of interest. The different boundary condition also play a part, where the top row used PEC, and the other two rows PMC.

Figure 9 shows a zoom-in to the survey domain, the domain of interest, which shows how the different codes discretized the zones of interest. The most striking point of these mesh comparison plots is that, while the overall

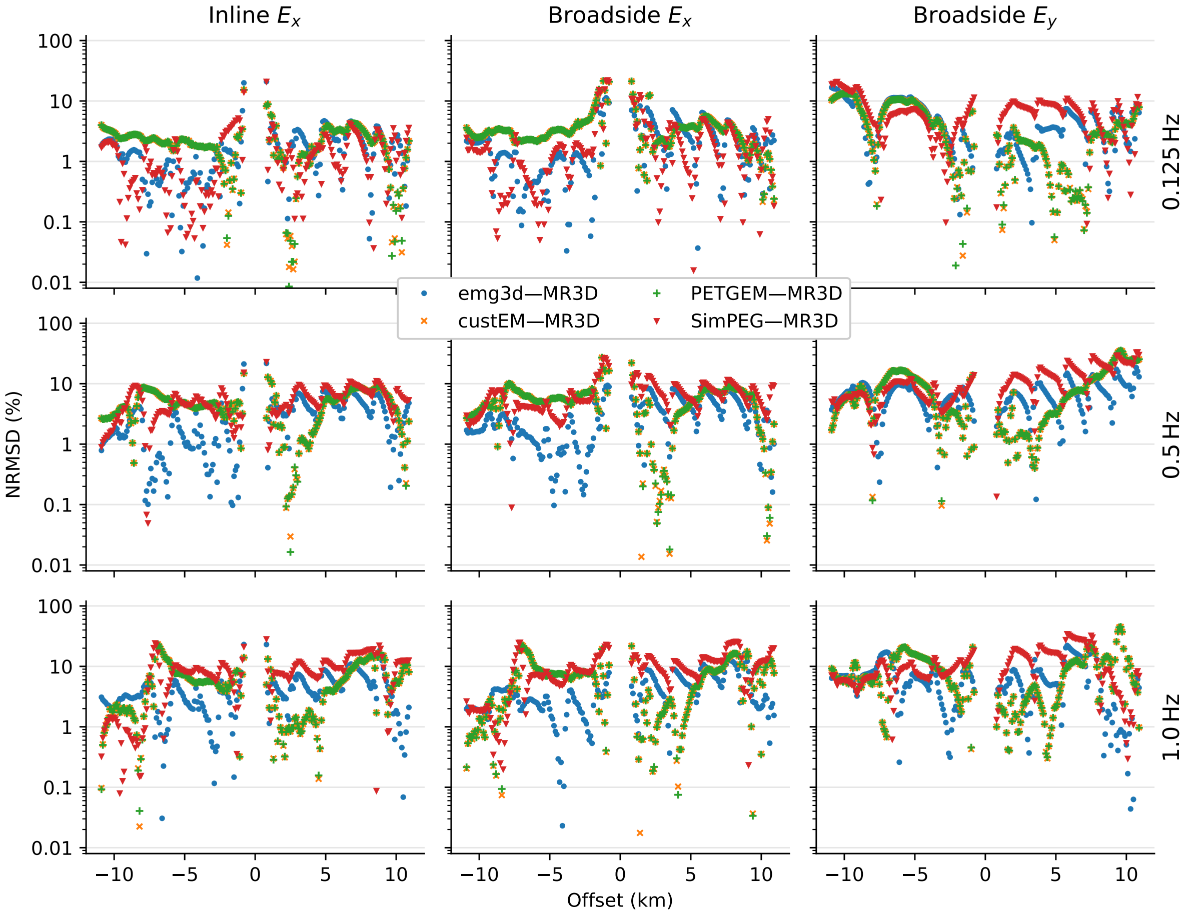

Figure 6. Normalised differences for all four codes in comparison with the published data, for the inline E_x field and the broadside E_x and E_y fields in the left, middle, and right column, respectively. Shown are the three frequencies 0.125 Hz, 0.5 Hz, and 1.0 Hz in the top, middle, and bottom row,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

fields are similar and the responses at receiver locations agree well, the fields can vary quite a lot in different parts of the model. Two kilometres below the seafloor, as an example, there are already significant changes visible, purely due to meshing. This is an important insight that has to be taken into account when interpreting modelling results and the feasibility for inversions.

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 $0.5 - 230$ GiB, and the runtime from roughly 7 min to 20 min, where the codes were run on very different machines from servers to supercomputers (see Table 2) and use between 1 and 96 processes in parallel.

The actual discretized resistivity model of the three mesh types are shown in Figure 10. The models where obtained using volume averaging (`emg3d`, `SimPEG`) and linear interpolation (`custEM`, `PETGEM`), in both cases on $\log_{10}(\rho)$ values. The actual representation of the original resistivities in the tetrahedra mesh is not as smooth as in the other meshes since this would require significantly smaller tetrahedra and more dof on top of the inclusion of subsurface layer constraints. However, note that the combination of slicing through tetrahedra and the vertical exaggeration leads to a very distorted view of the tetrahedral mesh which is barely representative for the overall

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.

Code	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	64	872	230.1	1 918 106
`emg3d`	1	1254	0.6	5 998 992
`PETGEM`	96	524	175.4	1 918 106
`SimPEG`	4	422	12.8	720 146

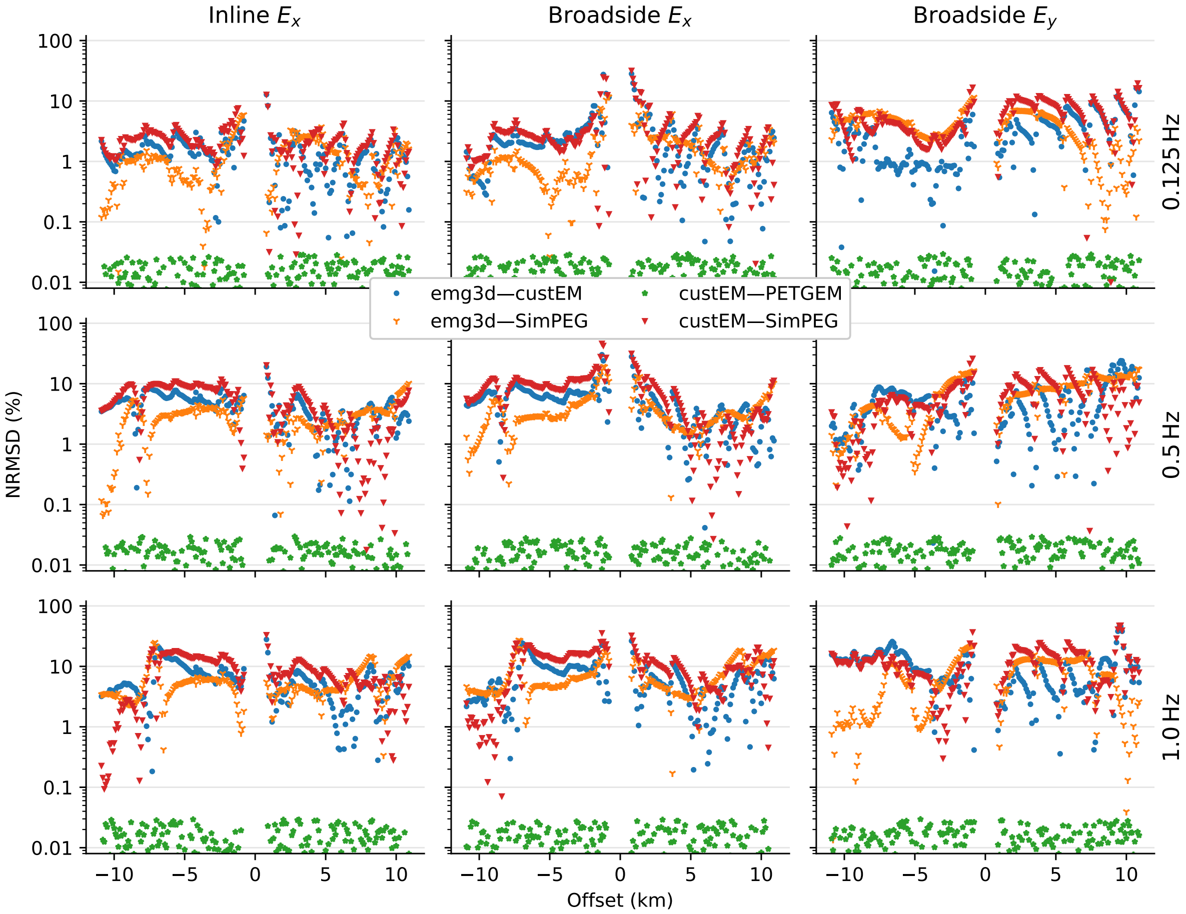

Figure 7. Normalised differences, just as in Figure 6, but comparing some of the four codes with each other. `custEM` and `PETGEM` produce very similar results, which is due to the fact that they use the same mesh. This also shows that the biggest impact in 3D forward modelling comes from the discretization.

averaged approximation of resistivities. The actual responses at the receivers (field interpolation) are obtained by spline interpolation (`emg3d`), linear interpolation (`SimPEG`), and through the second-order basis functions (`custEM`, `PETGEM`). It is worth mentioning as a final note that we used frequency independent meshes in this study. Creating frequency dependent meshes could potentially improve both the result and the runtime.

4 DISCUSSION

Today we are in the fortunate situation where we have several open-source codes available to model CSEM data for arbitrarily sized complex 3D models. It is therefore comparably easy for anyone to run a simulation and obtain reasonable-looking, robust and precise results, something that was impossible a few years ago. However, just because one obtains a good-looking result does not say anything about its accuracy. Though we may be able to confirm the accuracy of codes in some simple scenarios, validating the performance of 3D CSEM codes for complex problems is only possible by cross-validating multiple solutions. This is the core motivation for this study.

Overall, we observed an excellent match between the solutions. The amplitudes of the layered model have mostly a relative error of less than 1–2 %. Looking individually at the real or imaginary part results in much higher error values, but those are related to the zero crossings of the responses. The cross-validation of the results for the block model yields a similar picture, with normalised differences within a few percents. For the large, complex MR3D model it is a bit different. First it can be noted that the misfits are higher for higher frequencies. For the lowest frequency, 0.125 Hz, the normalised difference is a few percent for inline and broadside E_x components, and up to 10 % for the broadside E_y component. For higher frequencies these numbers rise to the order of ±10 %. The reason for the higher misfit for MR3D than for the layered or block model lies mostly in the discretization, boundary effects, and interpolation strategy. While 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 the responses we compare to were computed is not open-source. This has been shown within the process of this study. Our first attempts were based on the originally published, fine

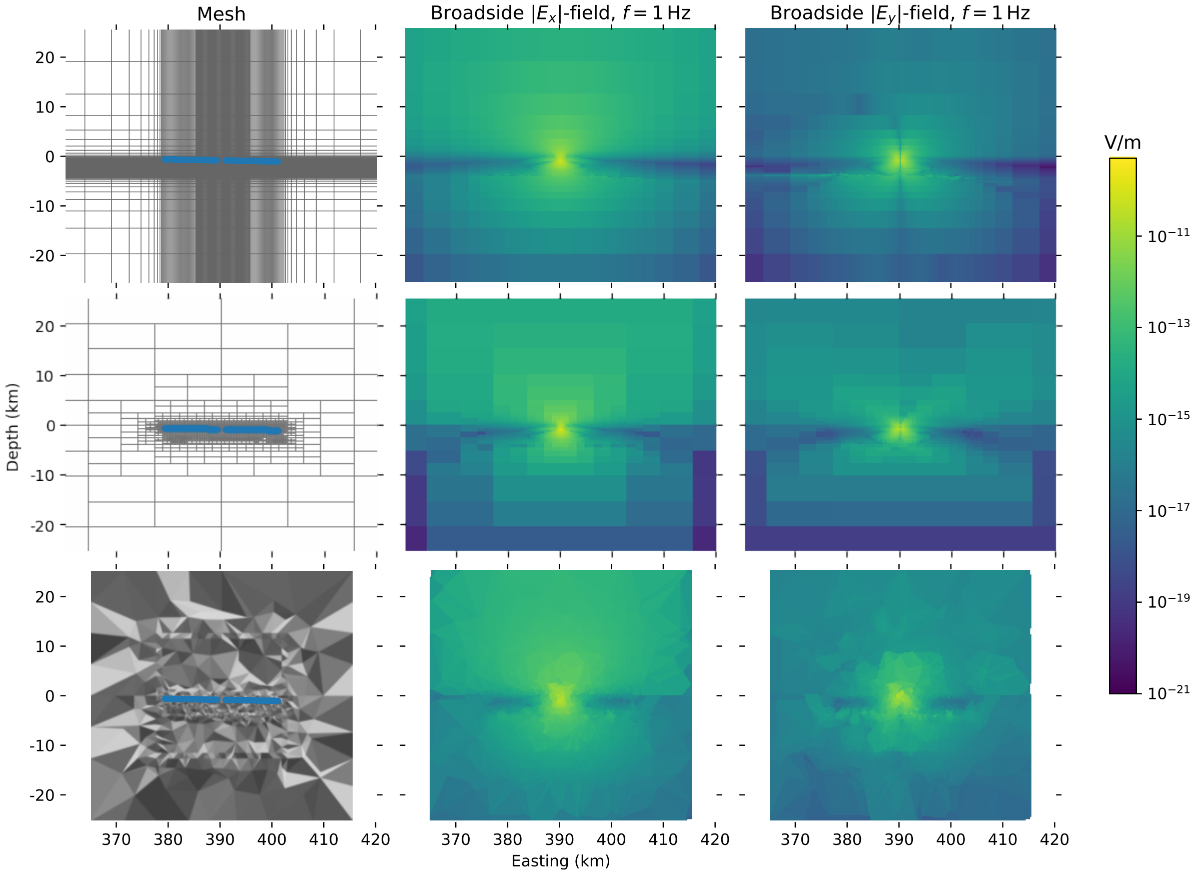

Figure 8. Meshes and broadside E_x and E_y fields in the left, middle, and right column, respectively, for the rectilinear, octree, and tetrahedral meshes in the top, middle, and bottom row, respectively. The shown dimensions shows the entire computational domain for the tetrahedra mesh, whereas the entire rectilinear and octree meshes are bigger.

MR3D model. This model is very detailed, but its extent is too small in x- and y-directions for CSEM computations. In order to represent the fine-scale model on a mesh suitable for computation, each of our codes did its own upscaling and extrapolation. However, we were unsuccessful in our attempt to reproduce the published results. The reason is that there are many ways how you can extrapolate a model, and without knowing the details of how MR3D represents the model on the computational mesh, we could not re-produce this step. 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 obtain comparable results.

One of the key insights we gained from this study is that creating appropriate meshes is a difficult and time-consuming task. While there now exist several open-source 3D CSEM codes the capability for automatic meshing lacks behind and currently remains a largely manual task requiring experience in the field. It is important to state that the chosen models favour structured meshes. The first two examples can easily and accurately be represented by regular meshes. Using, for instance, a dipping layer instead of three resistive cubes would turn it around and make the model more favourable for unstructured meshes. MR3D would be an ideal scenario for unstructured meshes, as it is based on also published irregular lithological horizons. Since the corresponding resistivity model, however, was defined on a regular grid, the FE codes 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. Both cases show that automatic and adaptive meshing capabilities are needed, which will increase the flexibility of 3D CSEM modelling. Such meshing codes should also take into account the physics of CSEM computation with its diffusive behaviour. It is also important to state that the unstructured mesh used for the MR3D model incorporates the entire domain and could be used for all source and receiver positions, whereas the rectilinear and octree meshes were designed for the shown receiver and corresponding source lines.

Whereas proper discretization is the main driver for accurate results, the used solver is the main discriminator in terms of runtime and memory. Both iterative and direct solvers are used in this study, however, the chosen models clearly favour iterative solvers. The biggest advantage of iterative solvers are the comparatively small memory requirements, even more so if the iterative solver is matrix free. Direct solvers exhibit their strongest advantage,

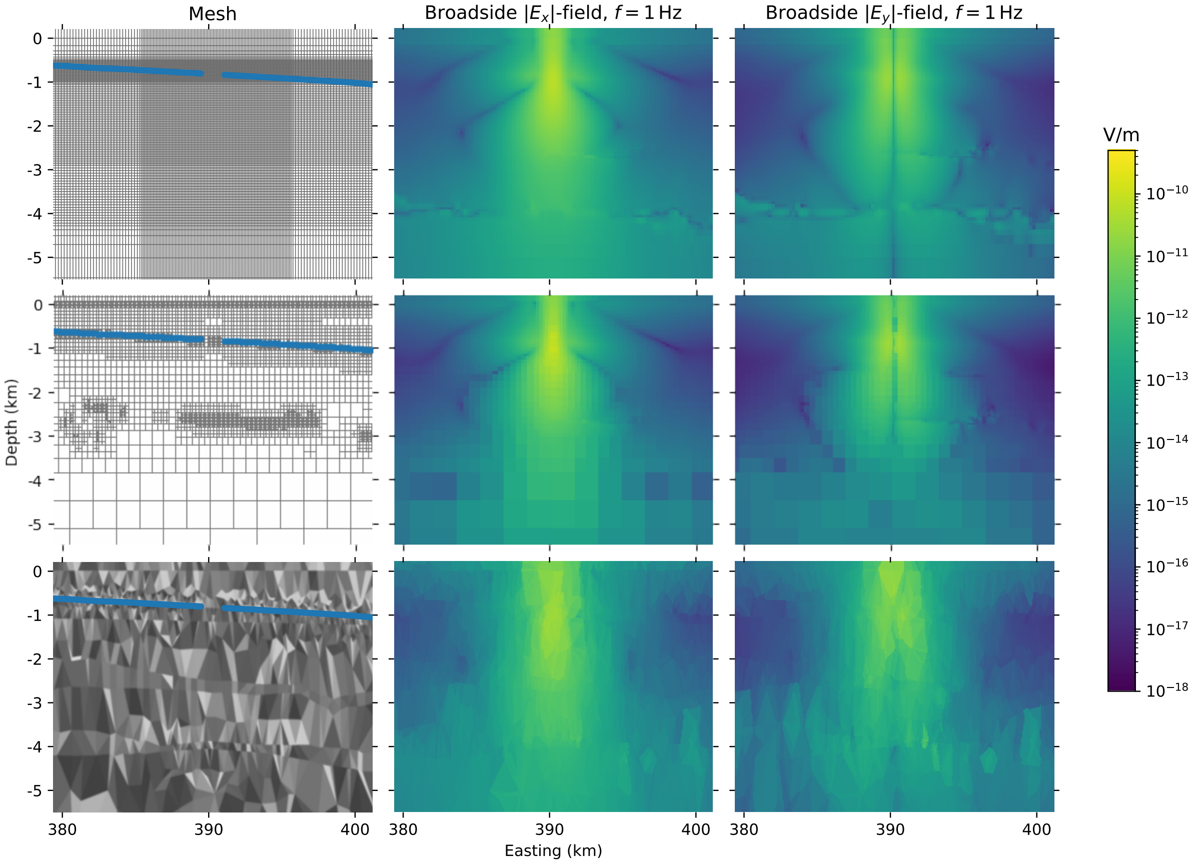

Figure 9. Meshes and broadside E_x and E_y fields in the left, middle, and right column, respectively, for the rectilinear, octree, and tetrahedral meshes in the top, middle, and bottom row, respectively (3× vertical exaggeration).

besides stability and robustness, 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, whereas it would come at the same cost as the first source for iterative solvers. Independently of the solver there is a trade-off between high accuracy and runtime. As an example, the octree mesh made for a very fast computation in the MR3D example thanks to the low number of dof, even though a direct solver was used. But the octree mesh has, due to its 2:1 aspect ratio, some limitations on the accuracy, why it was not used in the first two examples. Having discussed the discretization and the solver it is important to state that the reported runtime and memory is just one of the aspects, and we would like to emphasise that neither was at the core of this comparison, but the validation of the results. As such, no special efforts were undertaken to minimise either, as this is an entire different task.

A positive outcome of a collaboration between different projects such as this is that it brings the realm as a

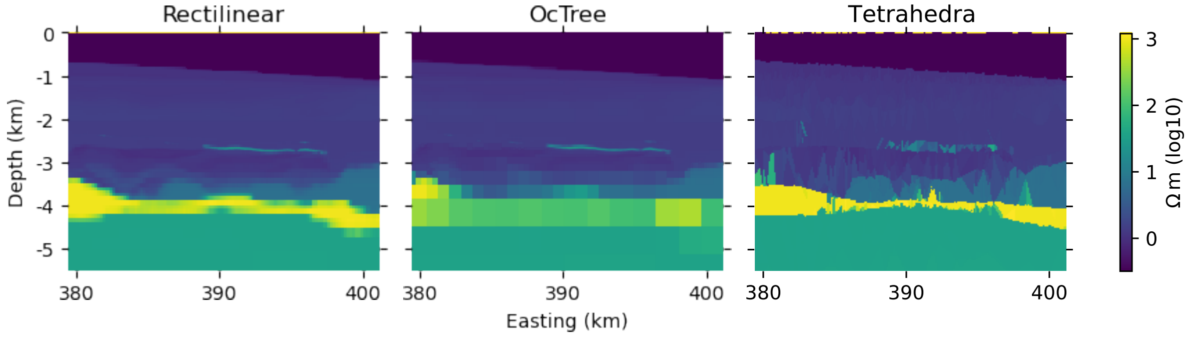

Figure 10. The resistivity model as obtained from the three different discretizations using rectilinear, octree, and tetrahedral meshes (3× vertical exaggeration).

whole further, which should be motivation enough for further collaborations. Within SimPEG, this work has motivated feature development including averaging strategies to map physical properties on a fine mesh to a coarser mesh for computation and new examples to be included 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. Within emg3d, this work pushed a lot of the meshing functionality, and implementation of I/O utilities for different file formats. Within custEM, cell-wise resistivity interpolation was added for this work, multi-layer subsurface topography was the first time applied as well as first time reciprocity modelling, and also improved mesh design (resolution, refinement, etc.). Within PETGEM, this work has promoted the inclusion of routines for magnetic field computation and the implementation of continuous integration functions. It has also improved its test suite and work-flow for the generation of adapted meshes based on a semi-automatic approach that reduces user intervention.

The shown examples consider the geophysical problem of marine CSEM with resistive bodies in the frequency domain. We hope to see in the future similar comparisons for other cases, such as land CSEM with strong topography; airborne and mixed air-ground surveys; looking for conductive bodies; time domain; and models based on horizons instead of blocks which pose a challenge for structured grids and favour unstructured meshes.

5 CONCLUSIONS

We compare the computed fields and underlying meshes of four different open-source 3D CSEM codes by means of three increasingly complex resistivity models. A layered marine, anisotropic model is used as first example to verify our results with semi-analytical responses. All codes show excellent accuracy with a relative error in the order of a few percents. Three resistive blocks are added to the layered model to increase complexity in the second example. The cross-validation of the outcomes of the different codes shows a normalised difference of a few percent, hence in a similar range as the relative error in the layered model. The third example is Marlim R3D, a realistic, complex, marine resistivity model. The model and corresponding CSEM responses are open-source, allowing an independent validation of our results with results from another (closed-source) code. In this scenario the normalised difference is in the order of 5–10 %, where the main source of error can be attributed to differences in discretization. The required runtime and memory consumption is primarily controlled by the used solver. The accuracy, on the other hand, depends to a large degree on the mesh. A proper discretization is therefore key for an accurate result, which makes this step the most time-consuming task, not the actual computation of the responses itself. Validating the correctness of a 3D code is a difficult task, and it is essential to have easily accessible benchmark models with reliable and reproducible solutions. Our study is one example of a collaboration facilitated by open practices including sharing of code and data. 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.

6 DATA

The resulting CSEM responses of the four codes for the three models, as well as all files to rerun the different models with the four codes and reproduce the shown results are available at Zenodo (DOI) *with the final submission.* For the review process we refer to the github page github.com/swung-research/3d-csem-open-source-landscape.

ACKNOWLEDGMENTS

We would like to thank Bane Sullivan for plotting of tetrahedra meshes in matplotlib through PyVista, Seogi Kang for the input and octree mesh design for the SimPEG layer and block models, and Joseph Capriotti for the octree mesh implementation and volume averaging in SimPEG. We would also 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 by R.R. as part of the DESMEX/DESMEX II projects was funded by the Germany Ministry for Education and Research (BMBF) in the framework of the research and development program Fona-r4 under grants 033R130D/033R130DN.

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.

The work of L.H. received funding from the National Science Foundation EarthCube program under award 1928406.

REFERENCES

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

Alumbaugh, D. L., Newman, G. A., Prevost, L., & Shadid, J. N., 1996. Three-dimensional wideband electromagnetic modeling on massively parallel computers, *Radio Science*, **31**, 1–23, doi: 10.1029/95RS02815.

Amestoy, P. R., Duff, I. S., Koster, J., & L'Excellent, J.-Y., 2001. A fully asynchronous multifrontal solver using distributed dynamic scheduling, *SIAM Journal on Matrix Analysis and Applications*, **23**, 15–41, doi: 10.1137/S0895479899358194.

Aruliah, D. & Ascher, U., 2002. Multigrid preconditioning for Krylov methods for time-harmonic Maxwell's equations in three dimensions, *SIAM Journal on Scientific Computing*, **24**, 702–718, doi: 10.1137/S1064827501387358.

Avdeev, D. B., 2005. Three-dimensional electromagnetic modelling and inversion from theory to application, *Surveys in Geophysics*, **26**, 767–799, doi: 10.1007/s10712-005-1836-x.

Broggini, F., Dellinger, J., Fomel, S., & Liu, Y., 2017. Reproducible research: Geophysics papers of the future–Introduction, *Geophysics*, **82**, 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**, 225–245, doi: 10.1007/s10712-009-9087-x.

Cai, H., Xiong, B., Han, M., & Zhdanov, M., 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.

Cai, H., Hu, X., Li, J., Endo, M., & Xiong, B., 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.

Carvalho, B. R. & Menezes, P. T. L., 2017. Marlim R3D: a realistic model for CSEM simulations–phase I: model building, *Brazilian Journal of Geology*, **47**, 633–644, doi: 10.1590/2317-4889201720170088.

Castillo-Reyes, O., de la Puente, J., & Cela, J. M., 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., de la Puente, J., García-Castillo, L. E., & Cela, J. M., 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., Son, J.-S., Lee, T. J., Kim, H. J., & Shin, C., 2014. Three-dimensional modelling of controlled-source electromagnetic surveys using an edge finite-element method with a direct solver, *Geophysical Prospecting*, **62**, 1468–1483, doi: 10.1111/1365-2478.12132.

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

Cockett, R., Kang, S., Heagy, L. J., Pidlisecky, A., & Oldenburg, D. W., 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. & Newman, G., 2004. A parallel finite-difference approach for 3D transient electromagnetic modeling with galvanic sources, *Geophysics*, **69**, 1192–1202, doi: 10.1190/1.1801936.

Correa, J. L. & Menezes, P. T. L., 2019. Marlim R3D: A realistic model for controlled-source electromagnetic simulations–Phase 2: The controlled-source electromagnetic data set, *Geophysics*, **84**, E293–E299, doi: 10.1190/geo2018-0452.1.

da Silva, N. V., Morgan, J. V., MacGregor, L., & Warner, M., 2012. A finite element multifrontal method for 3D CSEM modeling in the frequency domain, *Geophysics*, **77**, E101–E115, doi: 10.1190/geo2010-0398.1.

Das, U. C. & Verma, S. K., 1982. Electromagnetic response of an arbitrarily shaped three-dimensional conductor in a layered earth – numerical results, *Geophysical Journal International*, **69**, 55–66, doi: 10.1111/j.1365-246X.1982.tb04935.x.

de la Varga, M., Schaaf, A., & Wellmann, F., 2019. Gempy 1.0: open-source stochastic geological modeling and inversion, *Geoscientific Model Development*, **12**, 1–32, doi: 10.5194/gmd-12-1-2019.

Druskin, V. & Knizhnerman, L., 1994. Spectral approach to solving three-dimensional maxwell's diffusion equations in the time and frequency domains, *Radio Science*, **29**, 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**, 227–235, doi: 10.1016/0041-5553(64)90253-8.

Grayver, A. V. & Kolev, T. V., 2015. Large-scale 3D geoelectromagnetic modeling using parallel adaptive high-order finite element method, *Geophysics*, **80**, E277–E291, doi: 10.1190/geo2015-0013.1.

Grayver, A. V., Streich, R., & Ritter, O., 2013. Three-dimensional parallel distributed inversion of CSEM data using a direct forward solver, *Geophysical Journal International*, **193**, 1432–1446, doi: 10.1093/gji/ggt055.

Haber, E. & Ascher, U. M., 2001. Fast finite volume simulation of 3D electromagnetic problems with highly discontinuous coefficients, *SIAM Journal on Scientific Computing*, **22**, 1943–1961, doi: 10.1137/S1064827599360741.

Haber, E. & Heldmann, S., 2007. An octree multigrid method for quasi-static maxwell's equations with highly discontinuous coefficients, *Journal of Computational Physics*, **223**, 783–796, doi: https://doi.org/10.1016/j.jcp.2006.10.012.

Harris, C. R., Millman, J. K., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Matthew, Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., & Oliphant, T. E., 2020. Array programming with NumPy, *Nature*, **585**, 357–362, doi: 10.1038/s41586-020-2649-2.

Heagy, L. J., Cockett, R., Kang, S., Rosenkjaer, G. K., & Oldenburg, D. W., 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**, 309–324, doi: 10.1190/1.1440527.

Hursán, G. & Zhdanov, M. S., 2002. Contraction integral equation method in three-dimensional electromagnetic modeling, *Radio Science*, **37**, 1–1–1–13, doi: 10.1029/2001RS002513.

Jahandari, H. & Farquharson, C. G., 2014. A finite-volume solution to the geophysical electromagnetic forward problem using unstructured grids, *Geophysics*, **79**, E287–E302, doi: 10.1190/geo2013-0312.1.

Jaysaval, P., Shantsev, D., & de la Kethulle de Ryhove, S., 2014. Fast multimodel finite-difference controlled-source electromagnetic simulations based on a Schur complement approach, *Geophysics*, **79**, E315–E327, doi: 10.1190/geo2014-0043.1.

Jaysaval, P., Shantsev, D. V., & de la Kethulle de Ryhove, S., 2015. Efficient 3-D controlled-source electromagnetic modelling using an exponential finite-difference method, *Geophysical Journal International*, **203**, 1541–1574, doi: 10.1093/gji/ggv377.

Jaysaval, P., Shantsev, D. V., de la Kethulle de Ryhove, S., & Bratteland, T., 2016. Fully anisotropic 3-D EM modelling on a Lebedev grid with a multigrid pre-conditioner, *Geophysical Journal International*, **207**, 1554–1572, doi: 10.1093/gji/ggw352.

Kruglyakov, M. & Bloshanskaya, L., 2017. High-performance parallel solver for integral equations of electromagnetics based on Galerkin method, *Mathematical Geosciences*, **49**, 751–776, doi: 10.1007/s11004-017-9677-y.

Kruglyakov, M., Geraskin, A., & Kuvshinov, A., 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., Pitrou, A., & Seibert, S., 2015. Numba: A LLVM-based python JIT compiler, in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, LLVM '15, pp. 1–6, doi: 10.1145/2833157.2833162.

Langtangen, H. P., Logg, A., & Tveito, A., 2016. *Solving PDEs in Python: The FEniCS Tutorial I*, vol. 3 of **Simula SpringerBriefs on Computing**, Springer International Publishing, 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**, 69–92, doi: 10.1016/0041-5553(64)90240-X.

Liu, R., Guo, R., Liu, J., Ma, C., & Guo, Z., 2018. A hybrid solver based on the integral equation method and vector finite-element method for 3D controlled-source electromagnetic method modeling, *Geophysics*, **83**, E319–E333, doi: 10.1190/geo2017-0502.1.

Logg, A., Mardal, K.-A., & Wells, G., 2012. *Automated solution of differential equations by the finite element method: The FEniCS book*, vol. 84 of **Lecture Notes in Computational Science and Engineering**, Springer-Verlag Berlin Heidelberg, 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**, A19–A23, doi: 10.1190/1.2434781.

Mackie, R. L., Smith, J. T., & Madden, T. R., 1994. Three-dimensional electromagnetic modeling using finite difference equations: the magnetotelluric example., *Radio Science*, **29**, 923–935, doi: 10.1029/94RS00326.

Madsen, N. K. & Ziolkowski, R. W., 1990. A three-dimensional modified finite volume technique for Maxwell's equations, *Electromagnetics*, **10**, 147–161, doi: 10.1080/02726349008908233.

Miensopust, M. P., Queralt, P., Jones, A. G., & 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**, 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**(1), F33–F50, doi: 10.1190/1.3278525.

Mulder, W. A., 2006. A multigrid solver for 3D electromagnetic diffusion, *Geophysical Prospecting*, **54**, 633–649, doi: 10.1111/j.1365-2478.2006.00558.x.

Newman, G. A. & Alumbaugh, D. L., 1997. Three-dimensional massively parallel electromagnetic inversion–I. Theory, *Geophysical Journal International*, **128**, 345–354, doi: 10.1111/j.1365-246X.1997.tb01559.x.

Newman, G. A., Hohmann, G. W., & Anderson, W. L., 1986. Transient electromagnetic response of a three-dimensional body in a layered earth, *Geophysics*, **51**, 1608–1627, doi: 10.1190/1.1442212.

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

Oldenburg, D. W., Heagy, L. J., Kang, S., & Cockett, R., 2019. 3D electromagnetic modelling and inversion: a case for open source, *Exploration Geophysics*, **0**, 1–13, doi: 10.1080/08123985.2019.1580118.

Oristaglio, M. & Spies, B., 1999. *Three-Dimensional Electromagnetics*, vol. 7 of **Geophysical Developments**, Society of Exploration Geophysicists, doi: 10.1190/1.9781560802154.

Puzyrev, V., Koldan, J., de la Puente, J., Houzeaux, G., Vázquez, M., & Cela, J. M., 2013. A parallel finite-element method for three-dimensional controlled-source electromagnetic forward modelling, *Geophysical Journal International*, **193**, 678–693, doi: 10.1093/gji/ggt027.

Raiche, A. P., 1974. An integral equation approach to three-dimensional modelling, *Geophysical Journal International*, **36**, 363–376, doi: 10.1111/j.1365-246X.1974.tb03645.x.

Rochlitz, R., Skibbe, N., & Günther, T., 2019. custEM: customizable finite element simulation of complex controlled-source electromagnetic data, *Geophysics*, **84**, F17–F33, doi: 10.1190/geo2018-0208.1.

Rücker, C., Günther, T., & Wagner, F. M., 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. & Gärtner, K., 2004. Solving unsymmetric sparse systems of linear equations with PARDISO, *Future Generation Computer Systems*, **20**, 475–487, doi: 10.1016/j.future.2003.07.011.

Schwarzbach, C., Börner, R.-U., & Spitzer, K., 2011. Three-dimensional adaptive higher order finite element simulation for geo-electromagnetics—a marine csem example, *Geophysical Journal International*, **187**, 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**, 1–36, doi: 10.1145/2629697.

Skibbe, N., Rochlitz, R., Günther, T., & Müller-Petke, M., 2020. Coupled magnetic resonance and electrical resistivity tomography:

An open-source toolbox for surface nuclear-magnetic resonance, *Geophysics*, **85**, F53–F64, doi: 10.1190/geo2019-0484.1.

Sommer, M., Hölz, S., Moorkamp, M., Swidinsky, A., Heincke, B., Scholl, C., & Jegen, M., 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**, F95–F105, doi: 10.1190/1.3196241.

Sullivan, C. B. & Kaszynski, A. A., 2019. PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK), *Journal of Open Source Software*, **4**, 1450, doi: 10.21105/joss.01450.

Tehrani, A. M. & Slob, E., 2010. Fast and accurate three-dimensional controlled source electromagnetic modelling†, *Geophysical Prospecting*, **58**, 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**, 957, doi: 10.21105/joss.00957.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., & 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., Morten, J. P., & Spitzer, K., 2018. Anisotropic three-dimensional inversion of CSEM data using finite-element techniques on unstructured grids, *Geophysical Journal International*, **213**, 1056–1072, doi: 10.1093/gji/ggy029.

Wang, T. & Hohmann, G. W., 1993. A finite-difference, time-domain solution for three-dimensional electromagnetic modeling, *Geophysics*, **58**, 797–809, doi: 10.1190/1.1443465.

Wannamaker, P. E. & Zhdanov, M., 2002. *Three-Dimensional Electromagnetics*, no. 35 in Methods in Geochemistry and Geophysics, Elsevier Publishing Company, ISBN 978-0-080-54299-7.

Wannamaker, P. E., Hohmann, G. W., & Ward, S. H., 1984. Magnetotelluric responses of three-dimensional bodies in layered earths, *Geophysics*, **49**, 1517–1533, doi: 10.1190/1.1441777.

Ward, S. H. & Hohmann, G. W., 1988. *Electromagnetic Theory for Geophysical Applications*, chap. 4, pp. 130–311, Society of Exploration Geophysicists, 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**, 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**, WB9–WB19, doi: 10.1190/geo2016-0626.1.

Werthmüller, D., Mulder, W. A., & Slob, E. C., 2019. emg3d: A multigrid solver for 3D electromagnetic diffusion, *Journal of Open Source Software*, **4**, 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**, 302–307, doi: 10.1109/TAP.1966.1138693.

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

Zhdanov, M. S., Lee, S. K., & Yoshioka, K., 2006. Integral equation method for 3D modeling of electromagnetic fields in complex structures with inhomogeneous background conductivity, *Geophysics*, **71**, G333–G345, doi: 10.1190/1.2358403.