Towards an open-source landscape for 3-D CSEM modelling

Werthmüller, Dieter ; Rochlitz, Raphael ; Castillo-Reyes, Octavio ; Heagy, Lindsey

DOI
10.1093/gji/ggab238

Publication date
2021

Document Version
Final published version

Published in
Geophysical Journal International

Geophysical Journal International

Geophys. J. Int. (2021) **227,** 644–659
Advance Access publication 2021 June 21
GJI Marine Geosciences and Applied Geophysics

https://doi.org/10.1093/gji/ggab238

Towards an open-source landscape for 3-D 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, the Netherlands. E-mail: dieter@werthmuller.org*
[2] *Leibniz Institute for Applied Geophysics, Stilleweg* 2, 30655 *Hannover, Germany*
[3] *Barcelona Supercomputing Center (BSC), Nexus II Building c/Jordi Girona,* 29, 08034 *Barcelona, Spain*
[4] *Department of Statistics, University of California, Berkeley, Berkeley, CA* 94720-3860*, USA*

Accepted 2021 June 17. Received 2021 June 8; in original form 2020 October 23

SUMMARY

Large-scale modelling of 3-D 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 available to everyone. Using four different codes in the Python ecosystem, we perform simulations for increasingly complex models in a shallow marine setting. 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. On the one hand, these validations show that the open-source codes are able to compute comparable CSEM responses for challenging, large-scale models. On the other hand, they show many general and method-dependent problems that need to be faced for obtaining accurate results. 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 3-D 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 (EM) fields is therefore of great interest to design survey layouts, to understand the measured data, and for inversion purposes. Publications regarding 3-D modelling in EM methods started to appear as early as the 1970s and 1980s. 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 *Electromagnetic Theory for Geophysical Applications*, which is widely viewed as an authoritative publication on EM geophysics.

In the 1990s, computers became sufficiently powerful that 3-D 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 3-D 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) and 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 2000s 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 (FDs; Wang & Hohmann 1993; Druskin & Knizhnerman 1994; Mackie *et al.* 1994; Streich 2009; Sommer *et al.* 2013), finite elements (FEs; 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 (FVs; Madsen & Ziolkowski 1990; Clemens & Weiland 2001; Haber & Ascher 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 3-D EM modelling (and inversion) are given by Avdeev (2005) and Börner (2010). But there was also a tremendous publications output with regards to 3-D EM modelling in the last 10-15 yr, 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 (SLEs). 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), iterative solvers (Mulder 2006; Jaysaval *et al.* 2015) or a combination of both, the so-called hybrid solvers (Liu *et al.* 2018). The solvers often use pre-conditioners 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 normally 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 landscape in the field of 3-D CSEM modelling changed in the last five years quite dramatically. This paper introduces and compares four projects that make all source code openly available for use and adaptation. 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, and having all openly available facilitates reproducibility of our results. Analytical and semi-analytical solutions only exist for simple half-space 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 yield the same result, it gives confidence in their accuracy. This is the principal motivation of our study, together with the necessity for more reproducible models and modelling results in the realm of 3-D CSEM computations. Miensopust *et al.* (2013) present 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. Such a comparison reveals avenues for further development for each of the codes. 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 paper 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 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. For the numerical examples we show, all codes use the total-field formulation for the electric field. 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. PEC (PMC) implies that the tangential electric (magnetic) field vector components are zero at the boundary. 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-processings (e.g. mesh generation) are 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$\times$Runtime. Memory refers to the maximum memory increase at any point of solving the SLE.

2.1 custEM

The customizable EM modelling Python toolbox `custEM` was developed for simulating arbitrary complex 3-D CSEM geometries with a focus on semi-airborne set-ups, but it supports also land-based, airborne, coastal and marine environments. Multiple electric or magnetic field or potential FE approaches were implemented as total or secondary field formulations. The FE kernel, including higher order basis functions and parallelization, 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 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 polarization parameters in frequency-domain computations 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 3-D CSEM code `emg3d` is a multigrid solver (Fedorenko 1964) for EM diffusion following Mulder (2006), including tri-axial electrical anisotropy. The matrix-free solver can be used as main solver or as pre-conditioner 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 an FV generalization 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 characterized 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. In addition to functionality for modelling arbitrarily rotated electric sources and receivers, it also can simulate magnetic sources and receivers as well as model time-domain responses. It further has routines for obtaining the gradient of the misfit function which can be used by inversion algorithms. 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 3-D 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,

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	emg3d.emsig.xyz	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`

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 20.04	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 420 GB RAM Ubuntu 16.04	Lindsey Heagy

`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 multicore computers and cutting-edge clusters such as 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 polarization, EM (time and frequency domain, controlled and 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 FV 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 EM implementations use a staggered grid FV 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 3-D CSEM codes yield the same EM 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 verified with semi-analytical solutions. The layered model also serves

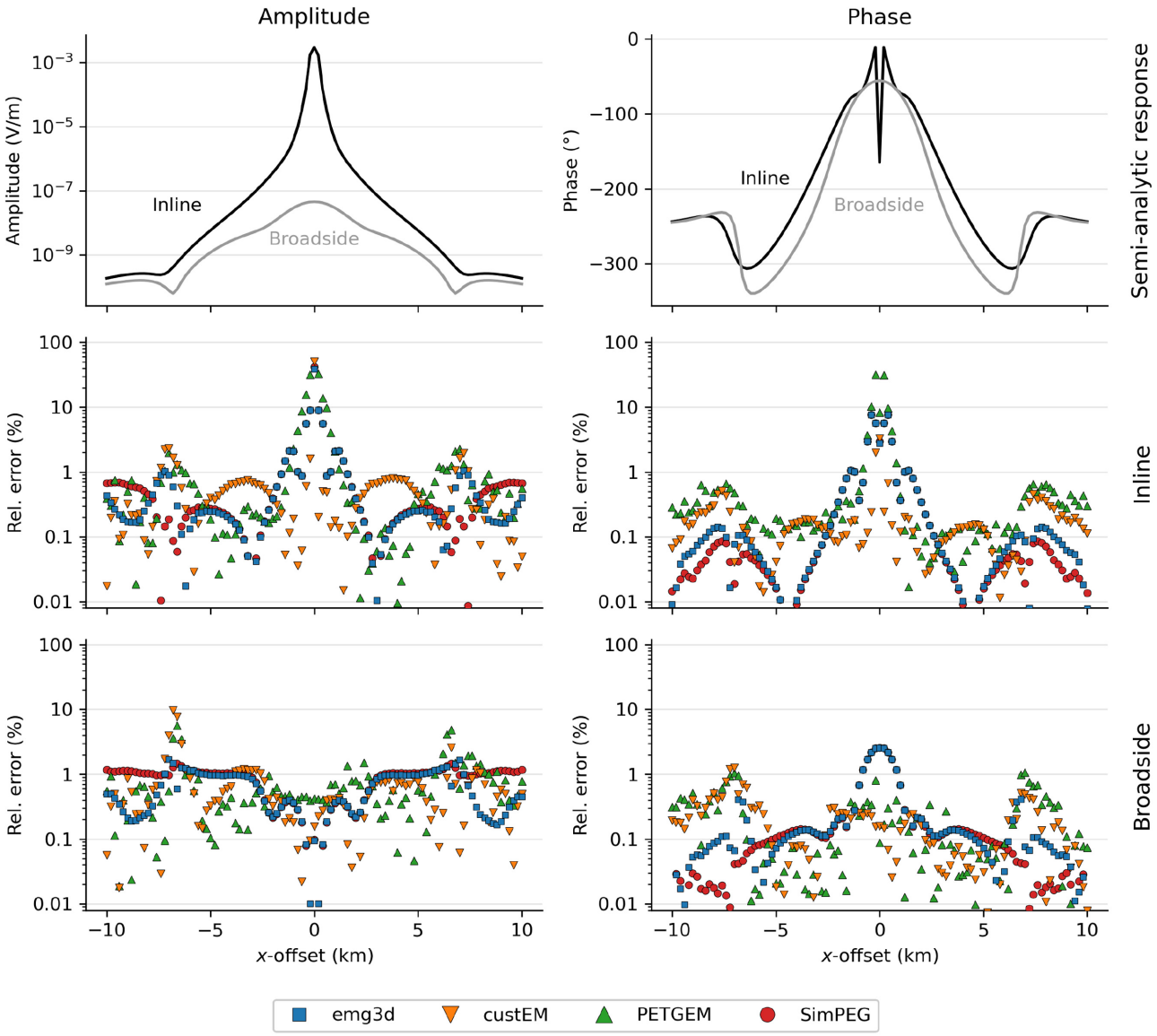

Figure 1. Results of the layered model: the semi-analytical inline and broadside responses are shown in the top row. The relative percentage errors of the four 3-D modellers are shown in the second and third rows for the inline and broadside responses, respectively.

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 a large-scale resistivity model that is openly released, together with computed CSEM responses.

3.1 Layered model

The layered (1-D) model consists of an upper half-space 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 half-space of $1000\,\Omega\,\text{m}$. The survey uses a 200 m long, 800 A source with frequency $f = 1$ Hz at a single position, and three receiver lines of 101 receivers each. The centre of the x-oriented source is at $x = y = 0$ 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 3-D modellers in general and FE codes in particular, and in reality one would not choose a 3-D code to compute responses for such a problem. However, it is one of the few examples that can be verified with semi-analytical results, and it is therefore an important first test. The results are shown in Fig. 1, in the left column the amplitudes and in the right column the phases. 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 1-D case the x-directed electric fields for the two broadside lines for $y = \pm 3$ km are identical.

The misfit is generally in the order of 1% or less, the biggest misfits are on the one hand close to the source and on the other hand where there are rapid phase changes associated to amplitudes having

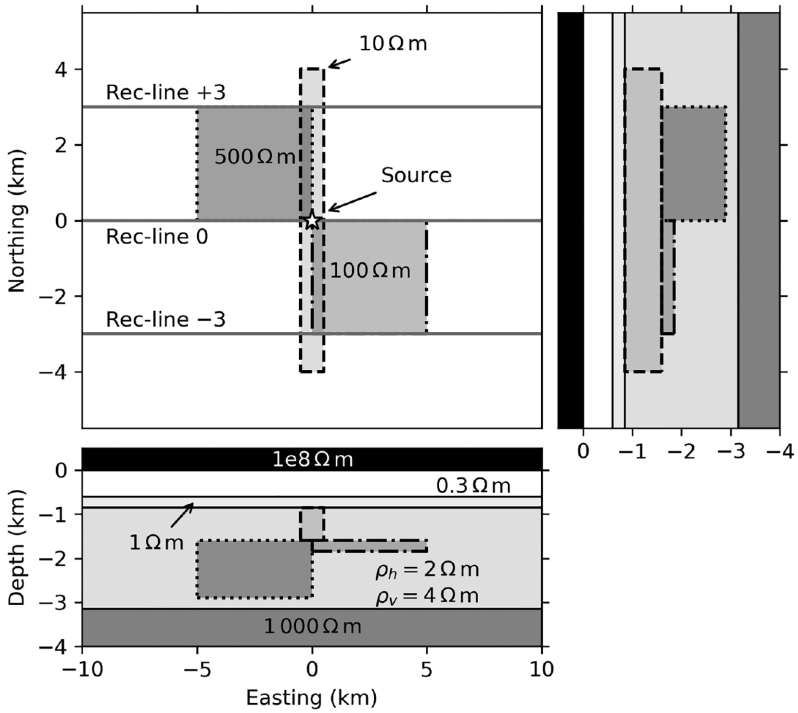

Figure 2. Sketch of the block model, consisting of the layered background model with three resistive blocks, embedded in the thick background layer that has VTI with $\lambda = \sqrt{\rho_v/\rho_h} = \sqrt{2}$.

a dip towards zero. The increased errors towards the location of the 200 m long source are related to the gridding, and finer discretizations (resulting in longer runtime and higher memory consumption) would reduce these errors.

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 in an octree mesh. 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. Although we only show the x-directed field in this comparison, the y-directed field yields a similar result with respect to the misfit. A different story in this scenario is the z-directed field, which is not continues across the boundary on the seafloor. Having receivers close to such a non-continues boundary requires advanced interpolation routines (e.g. Wirianto *et al.* 2011; Shantsev & Maaø 2015). The FE codes handle this problem automatically with the used basis functions, but the used FV codes do currently not have such routines implemented. The grid for the FV codes in this first example with a focus on precision is created in such a way that the receivers are located on the corresponding nodes, so no interpolation is required to obtain the responses.

3.2 Block model

The block model is a CSEM adaption of the MT *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 Fig. 2. Additionally, we add our *Layered Model* as a VTI background. The three resistive blocks have resistivities of 10 Ω m (shallow beam perpendicular to survey lines), 100 Ω m (thin plate, southeast), and 500 Ω m (cube, northwest).

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 normalized root-mean square difference (NRMSD) between two responses R_1 and R_2, given by

$$\text{NRMSD}\ (\%) = 100\ \frac{|R_1 - R_2|}{(|R_1| + |R_2|)/2}\ , \tag{1}$$

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

The results for the three receiver lines $y = -3; 0; +3$ km are shown in the left, middle, and right columns of Fig. 3, respectively. The top row shows the result of `emg3d`, as an example, and the bottom row shows the normalized differences between the absolute responses of the codes. Note that for visual reasons we only show the normalized differences between the two FV codes, the two FE codes, and between `emg3d` and `custEM` as an example of a cross-FV-FE comparison; the three other cross-method comparisons look quantitatively similar.

A few interesting points stand out:

(i) The misfit levels between the different codes are overall comparable; the normalized difference is generally below a few per cent, increasing towards larger offsets.

(ii) The normalized difference between the FV codes `emg3d` and `SimPEG` is comparably low except at the boundary. The generally low normalized difference is because these codes use the same rectilinear mesh for modelling. The increasing normalized difference towards the boundary is because `emg3d` uses the PEC boundary condition, while `SimPEG` uses the PMC boundary condition in this example. This difference from the boundary condition can also be seen in the relative error of the layered model in Fig. 1.

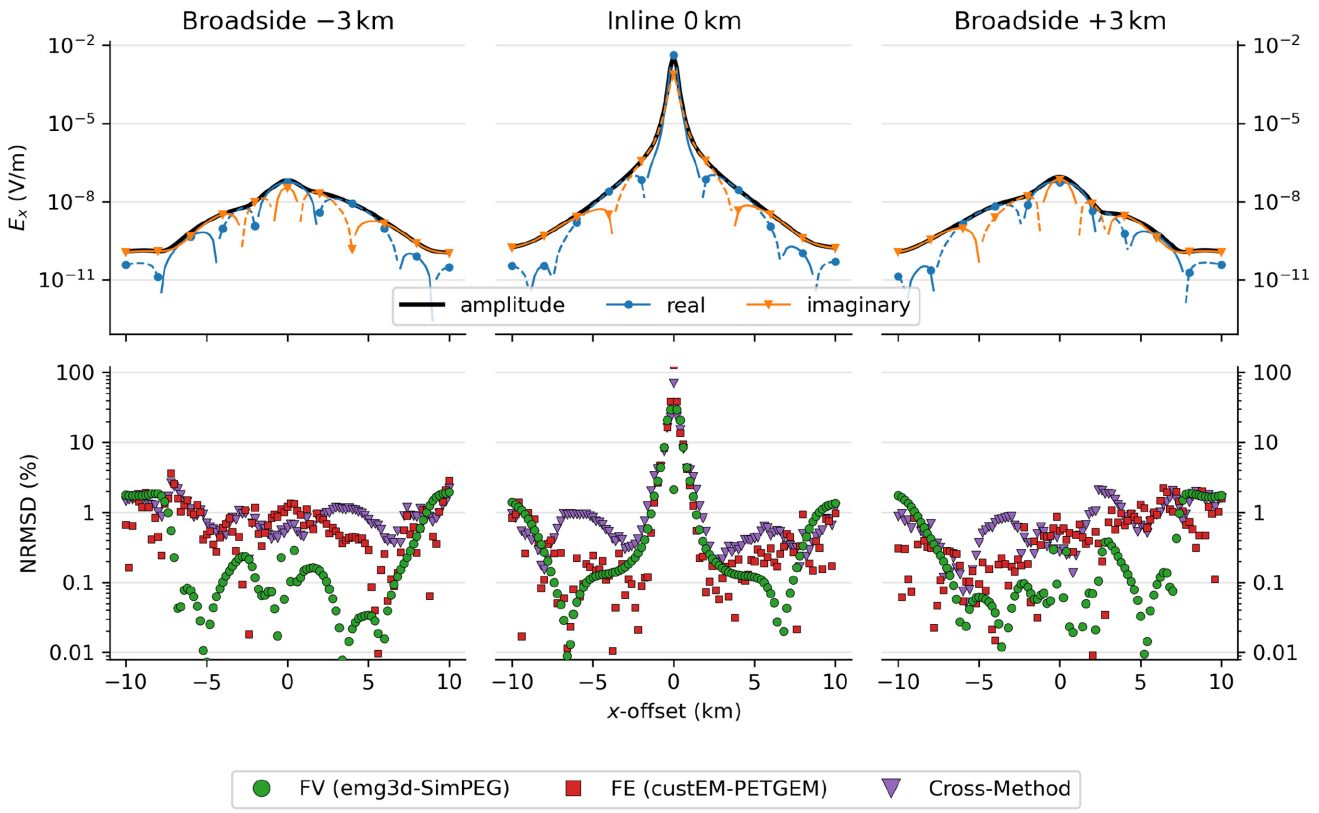

Figure 3. Results of the block model comparison: the responses of `emg3d`, as an example, are shown in the top row and the normalized differences (%) between the amplitudes of the different codes are shown in the bottom row.

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 block model.

Code	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	48	312	281.8	6014440
`emg3d`	1	213	0.5	6004144
`PETGEM`	24	238	152.8	2455868
`SimPEG`	4	20000	387.9	6004144

(iii) The FE codes `custEM` and `PETGEM` show similar boundary effects, but are influenced by the anomaly response (higher misfits over the anomalous blocks).

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

The corresponding required runtime and memory are listed in Table 3.

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, 3-D 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 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 code from the industry (Maaø 2007, *SBLwiz* software from *EMGS*). It is therefore, on the 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 × 75 × 5 m. The model was upscaled for the computation to 515 × 563 × 310 cells, totalling to almost 90 million cells, where each cell has dimensions of 100 × 100 × 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 Fig. 4, together with the receiver and sources positions. All layer have a VTI with $\lambda = \sqrt{2}$,

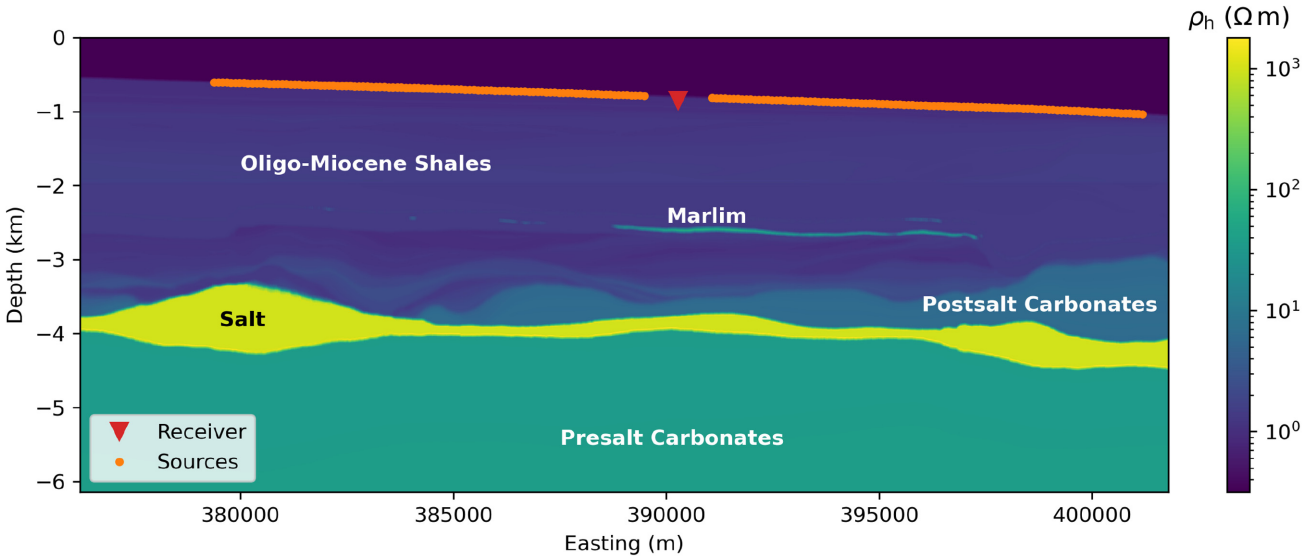

Figure 4. MR3D horizontal resistivity model, x–z-slice through the receiver y-position and with major formations annotated. Air (not shown in the model), seawater and the salt layer are electrically isotropic, everything else has VTI with $\lambda = \sqrt{2}$. The receiver is located on the seafloor,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

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 Fig. 5, both inline and broadside (we only show the amplitude, as the conclusions for the phase are very similar). 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, 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.

(1) 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 Fig. 6.

(2) The three highest frequencies of the E_z fields become noisy at large offsets for all of our codes (well below any real-world noise level).

(3) The responses for the E_y component of the inline acquisition line do not agree at all. In the 1-D case, the inline E_y component would be zero, and the only response we can measure here comes from 3-D effects. These responses are very low, roughly two orders of magnitude lower than the E_x component. Neither 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 there is no *true solution*; but the closer that the previously published results and the results obtained from the four codes run here agree, the more confident we can be of the outcomes. This emphasizes the importance of comparing 3-D results and one of our main objectives, as there is no other way to check the results of 3-D codes for complex models. Fig. 6 shows the normalized 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 normalized difference is roughly 10% or less. The structured meshes have in some cases a smaller normalized difference, for example for negative offsets in the inline and broadside E_x responses, and in other cases the unstructured tetrahedral meshes, for example for positive offsets for all broadside E_y responses, and often they have a comparable NRMSD. Some normalized 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 (1) in Fig. 4. The origin of this zigzag pattern is most likely the bathymetry, as the code of the published data uses the bathymetry information in addition to the resistivity cube. This explains why the codes using the tetrahedra mesh, which use the bathymetry as well, do not show this feature. None of our codes has currently special interpolation routines implemented to take particular care of a source injection close to a dipping interface, such as the bathymetry, with discontinuities in the normal electrical field and the normal derivatives of both electric and magnetic tangential fields. This effect alone can potentially contribute several per cents of NRMSD, as shown by Shantsev & Maaø (2015).

The importance of the meshing to the result can be seen in Fig. 7, where we compute the normalized differences between our codes. The most obvious result in that figure is that `custEM` and `PETGEM` produce results that are almost identical, their normalized difference is generally below 0.03%. This is why they are shown as one case in Fig. 6. 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 Fig. 7, as the comparisons to `PETGEM` would look the same. But in general the normalized differences between our codes look similar as in the comparison with the published

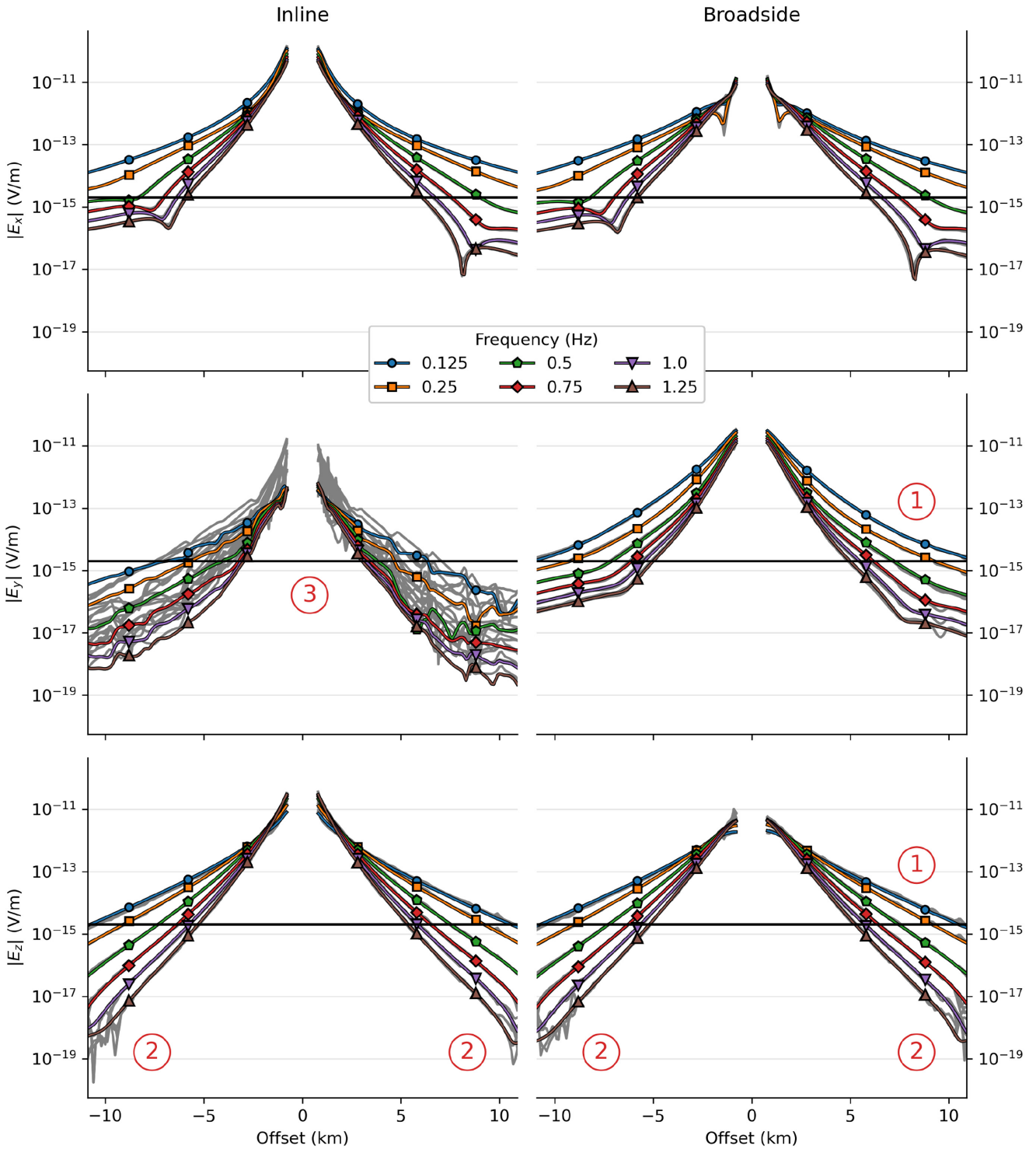

Figure 5. MR3D comparison between our codes (grey lines) and the published data (coloured lines). The grey lines under the coloured lines are not visible in most areas, meaning that they are very similar. However, there are three notable zones, (1) to (3), which are explained further in the text.

results, and the principal source of the difference must be in the different discretizations.

The actual differences in meshing are shown in the next two figures. Fig. 8 shows on the top row the rectilinear 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 directions); 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

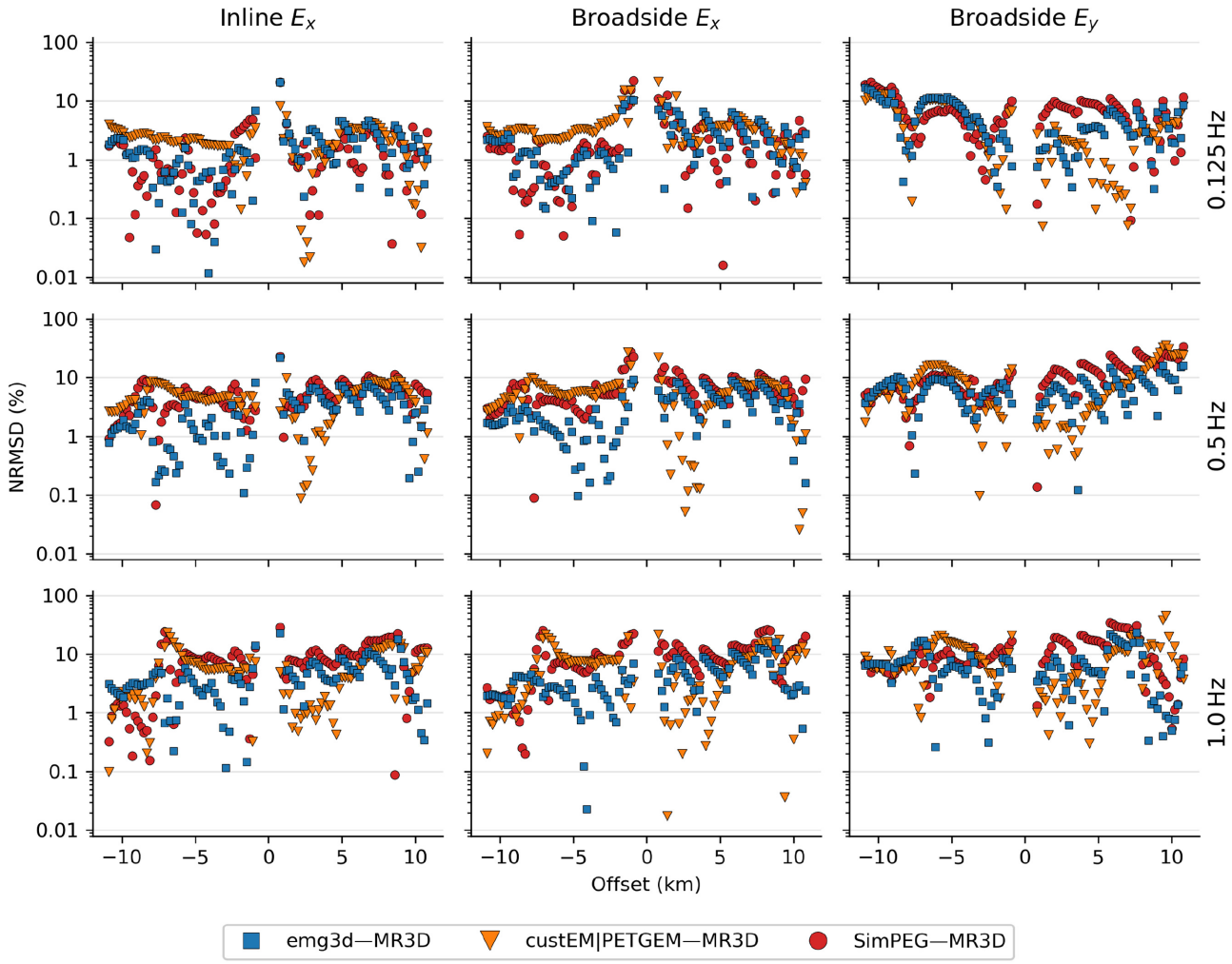

Figure 6. Normalized differences for all four codes in comparison with the published data, for the inline E_x field and the broadside E_x and E_y fields in the left, middle and right columns, respectively. Shown are the three frequencies 0.125, 0.5 and 1.0 Hz in the top, middle and bottom rows, respectively. The results of the two FE codes are almost identical, see Fig. 7, and are therefore combined here.

interesting for various reasons, 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 plays a part, where the top row used PEC and the other two rows used PMC. Also visible is that there is more energy close to the boundary in the tetrahedral mesh, due to the smaller computational domain with corresponding reflections. However, this seems not to have influenced the responses at the receiver locations, as the misfit between the FE and FV codes does not increase particularly for receivers with larger offsets.

Fig. 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 fields are similar and the responses at receiver locations agree well, the fields can vary quite a lot in different parts of the model. 2 km 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 4. The memory requirement varies from 0.5–230 GiB, and the runtime from roughly 7 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 Fig. 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 averaged approximation of resistivities. The actual responses

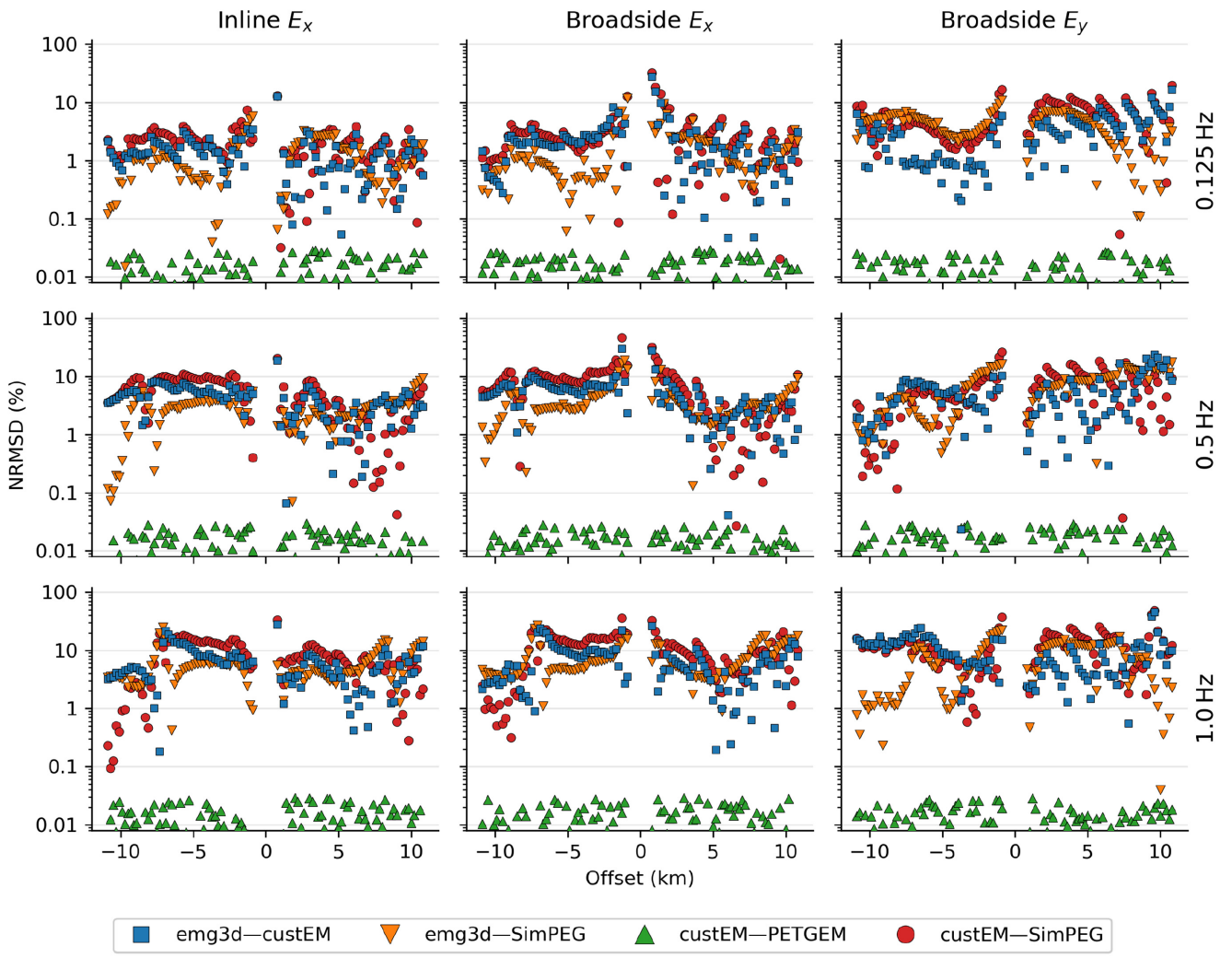

Figure 7. Normalized differences, just as in Fig. 6, but comparing some of the four codes with each other. `custEM` and `PETGEM` produce very similar results, which is due to the fact that they use the same mesh. This also shows that the biggest impact in 3-D forward modelling comes from the discretization.

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 3-D 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 3-D CSEM codes for complex problems is only possible by cross-validating multiple solutions. This plus the necessity for more reproducible modelling results are the core motivations for this study.

Overall, we observed an excellent match between the solutions. The amplitudes and phases of the layered model have mostly a relative error of less than 1–2%. The cross-validation of the results for the block model yields a similar picture, with normalized differences within a few per cents. 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 normalized difference is a few per cent for inline and broadside E_x components, and up to 10% for the broadside E_y component. For higher frequencies these numbers increase to the order of $\pm 10\,\%$. The reason for the higher misfit for MR3D than for the layered or block model lies mostly in the discretization, boundary effects, and interpolation strategies for the model, the source distribution on the mesh, and to obtain the response at receiver locations. 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 MR3D model. This model is very detailed, but its extent is too small in the 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

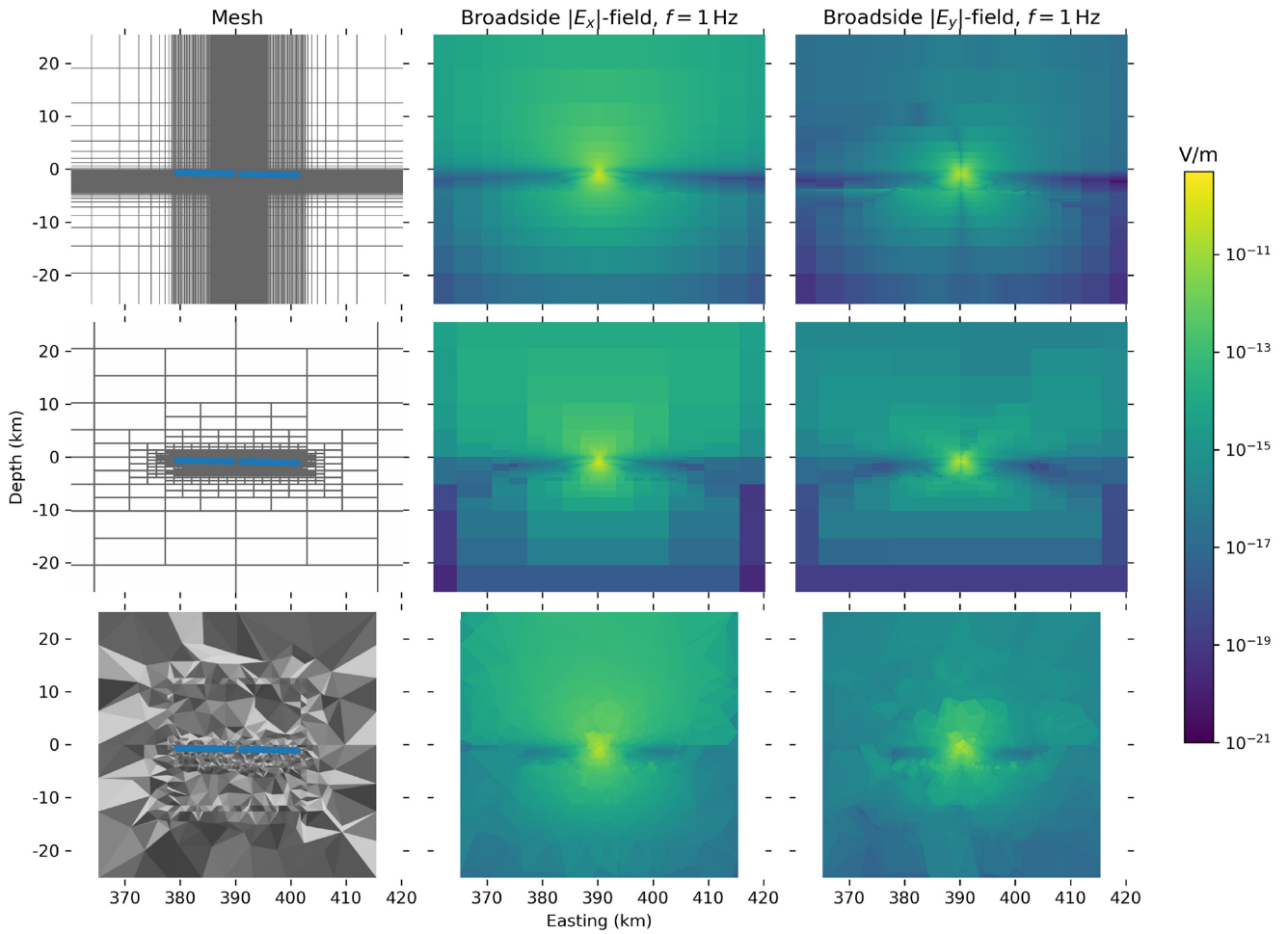

Figure 8. Meshes and broadside E_x and E_y fields in the left, middle and right columns, respectively, for the rectilinear, octree and tetrahedral meshes in the top, middle and bottom rows, respectively. The shown dimensions display the entire computational domain for the tetrahedra mesh, whereas the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

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 publish 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 3-D 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 they are best suited to represent the irregular geometry provided by the 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 3-D CSEM modelling (Schwarzbach *et al.* 2011; Key 2016; Castillo-Reyes *et al.* 2019). 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. Also, creating FV meshes is much simpler than creating FE meshes. However, having a FE mesh yields a lot of flexibility with regards to interpolation and local refinement, making it more powerful once created.

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, besides stability and robustness, in terms of computation times if responses of multiple CSEM transmitters need to be computed in the same

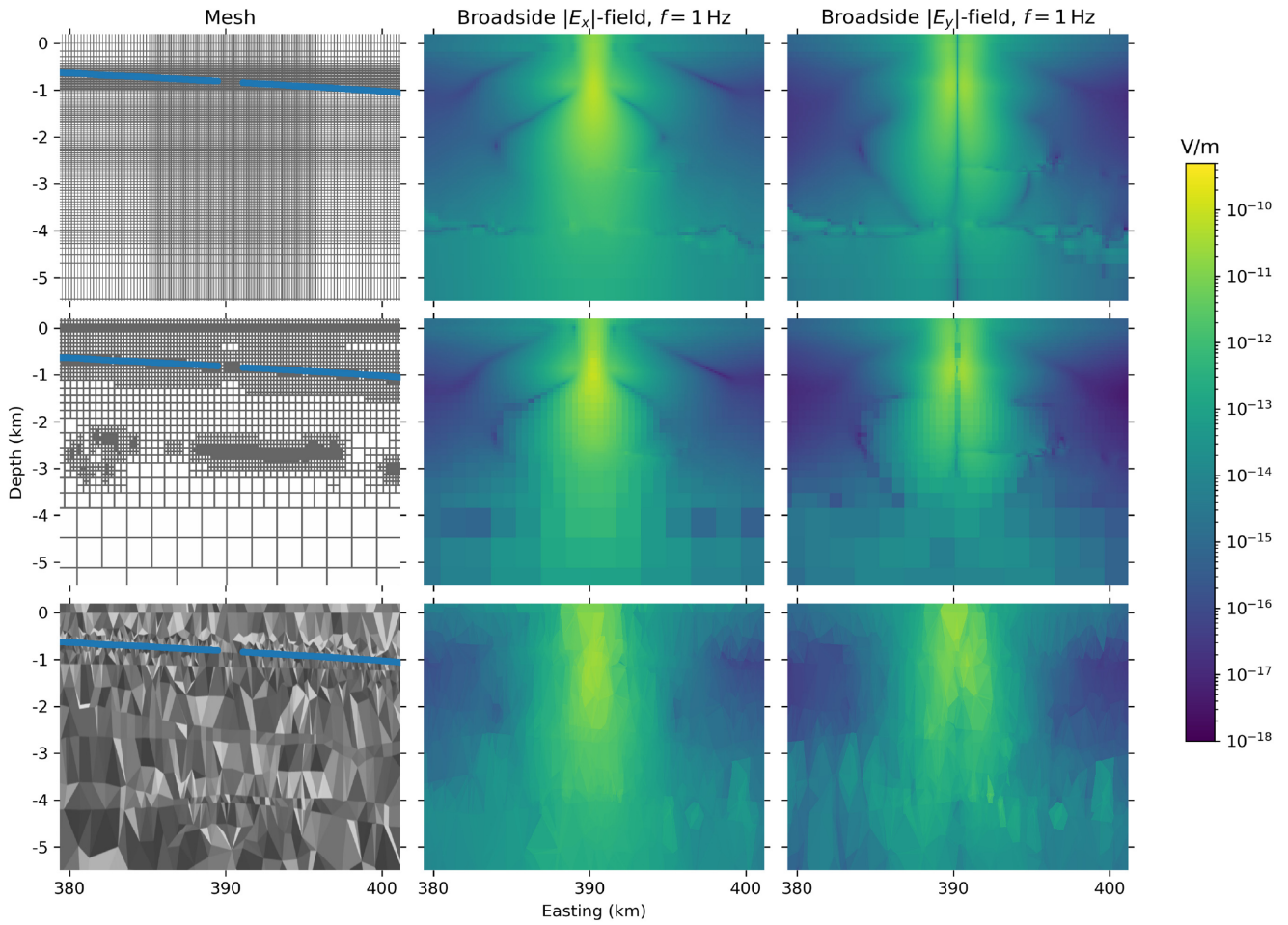

Figure 9. Meshes and broadside E_x and E_y fields of the survey domain in the left, middle and right columns, respectively, for the rectilinear, octree and tetrahedral meshes in the top, middle and bottom rows, respectively (3× vertical exaggeration).

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 MR3D model.

Code	#Procs	Runtime (s)	Memory (GiB)	#dof
`custEM`	64	872	230.1	1918106
`emg3d`	1	1246	0.5	5998992
`PETGEM`	96	524	175.4	1918106
`SimPEG`	4	422	12.8	720146

computational domain. As the system matrix factorization 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. For 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 emphasize that neither was at the core of this comparison, but the validation of the results. As such, no special efforts were undertaken to minimize 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 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. Marlim R3D is a rare example of an open-source resistivity model that also comes with simulated CSEM data. The spectrum of geophysical

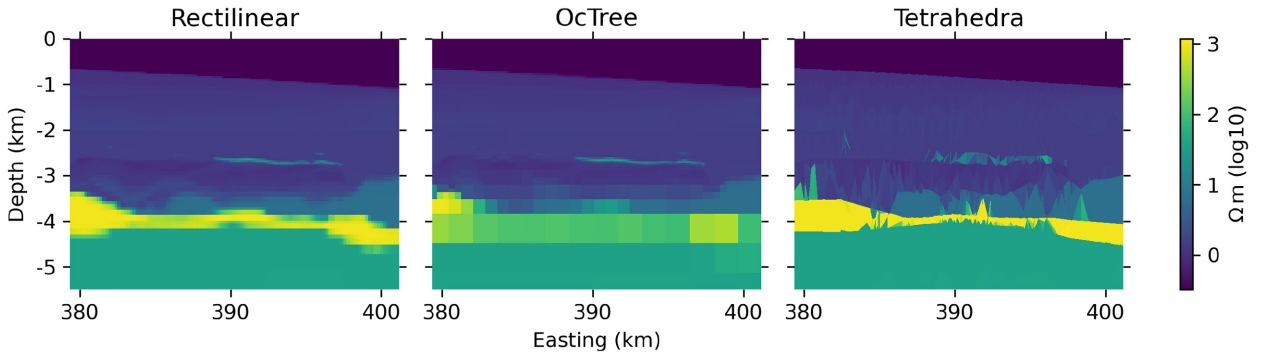

Figure 10. The resistivity model as obtained from the three different discretizations using rectilinear, octree and tetrahedral meshes (3× vertical exaggeration).

EM methods is much wider than marine CSEM. We hope to see in the future more open-source models with accompanying data and 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. An ideal realistic model should be defined as a function `resistivity(x, y, z)` which returns the resistivity for any location in the domain of interest, preferably with the horizons of the major formations. This would allow any meshing strategy to extract its optimal resistivity model.

It was a fruitful path to get four different CSEM codes to work together, and it included a steep learning curve for all involved parties. Our comparison and our results are far from perfect. The chosen models favour regular grids, and the chosen surveys favour iterative solvers. We only consider one particular CSEM case, the shallow marine setting looking for a resistor. Our comparison raised probably more questions regarding the accuracy of 3-D CSEM modelling than it answered. We consider this as an important, initial step which we believe yields already many insights to the reader. We have shown similarities and differences in FV and FE codes and our results show that ideally the CSEM responses of a complex model should be computed with various codes, as there is not a single code that will provide very accurate results. Focusing on reproducibility and open-source software and data forced us to show the misfit between the codes brutally honest, there is no hiding. We think this should be done more often. We do not claim that our codes are superior to other codes in any way. It is our codes' current status, and we hope that we can build on this and that many similar comparisons will follow.

5 CONCLUSIONS

We compare the computed fields and underlying meshes of four different open-source 3-D 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 per cents. 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 normalized difference of a few per cent, 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 MR3D model and corresponding CSEM responses are released openly, allowing an independent validation of our results with results from another (closed-source) code. In this scenario the normalized difference is in the order of 5–10%, where the main source of error can be attributed to differences in discretization and handling of the bathymetry with corresponding advanced interpolations of the EM fields. 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 3-D 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.

ACKNOWLEDGEMENTS

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 regard to the Marlim R3D model and corresponding CSEM data, and for making their actual computation model available under an open-source license. We would further like to thank the editor Ute Weckmann and assistant editor Fern Story as well as the reviewers Colin Farquharson and Rune Mittet for many helpful comments, which improved this manuscript considerably.

The work of DW was conducted within the Gitaro.JIM project funded through MarTERA, a European Union's Horizon 2020 Framework Programme, grant agreement No. 728053.

The development of `custEM` by RR 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 OC-R has received funding from the European Union's Horizon 2020 Framework Programme under the Marie Sklodowska-Curie grant agreement No. 777778. Further, the development of `PETGEM` has received funding from the European Union's Horizon 2020 Framework Programme, grant agreement No. 828947, and from the Mexican Department of Energy, CONACYT-SENER Hidrocarburos grant agreement No. B-S-69926. Furthermore, OC-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 LH received funding from the National Science Foundation EarthCube program under award 1928406.

DATA AVAILABILITY

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 (10.5281/zenodo.4535602).

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,* 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 Sci.,* **31,** 1–23.

Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y. & Koster, J., 2001. A fully asynchronous multifrontal solver using distributed dynamic scheduling, *SIAM J. Matrix Anal. Appl.,* **23,** 15–41.

Aruliah, D. & Ascher, U., 2002. Multigrid preconditioning for Krylov methods for time-harmonic Maxwell's equations in three dimensions, *SIAM J. Sci. Comput.,* **24,** 702–718.

Avdeev, D.B., 2005. Three-dimensional electromagnetic modelling and inversion from theory to application, *Surv. Geophys.,* **26,** 767–799.

Börner, R.-U., 2010. Numerical modelling in geo-electromagnetics: advances and challenges, *Surv. Geophys.,* **31,** 225–245.

Broggini, F., Dellinger, J., Fomel, S. & Liu, Y., 2017. Reproducible research: geophysics papers of the future—introduction, *Geophysics,* **82,** WBi–WBii.

Cai, H., Xiong, B., Han, M. & Zhdanov, M., 2014. 3D controlled-source electromagnetic modeling in anisotropic medium using edge-based finite element method, *Comput. Geosci.,* **73,** 164–176.

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, *Comput. Geosci.,* **99,** 125–134.

Carvalho, B.R. & Menezes, P.T.L., 2017. Marlim R3D: a realistic model for CSEM simulations—phase I: model building, *Braz. J. Geol.,* **47,** 633–644.

Castillo-Reyes, O., de la Puente, J. & Cela, J.M., 2018. PETGEM: a parallel code for 3D CSEM forward modeling using edge finite elements, *Comput. Geosci.,* **119,** 126–136.

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, *Geophys. J. Int.,* **219,** 39–65.

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, *Geophys. Prospect.,* **62,** 1468–1483.

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

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, *Comput. Geosci.,* **85,** 142–154.

Commer, M. & Newman, G., 2004. A parallel finite-difference approach for 3D transient electromagnetic modeling with galvanic sources, *Geophysics,* **69,** 1192–1202.

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.

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.

Das, U.C. & Verma, S.K., 1982. Electromagnetic response of an arbitrarily shaped three-dimensional conductor in a layered earth—numerical results, *Geophys. J. Int.,* **69,** 55–66.

de la Varga, M., Schaaf, A. & Wellmann, F., 2019. Gempy 1.0: open-source stochastic geological modeling and inversion, *Geosci. Model Dev.,* **12,** 1–32.

Druskin, V. & Knizhnerman, L., 1994. Spectral approach to solving three-dimensional Maxwell's diffusion equations in the time and frequency domains, *Radio Sci.,* **29,** 937–953.

Fedorenko, R.P., 1964. The speed of convergence of one iterative process, *USSR Comput. Math. Math. Phys.,* **4,** 227–235.

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

Grayver, A.V., Streich, R. & Ritter, O., 2013. Three-dimensional parallel distributed inversion of CSEM data using a direct forward solver, *Geophys. J. Int.,* **193,** 1432–1446.

Haber, E. & Ascher, U.M., 2001. Fast finite volume simulation of 3D electromagnetic problems with highly discontinuous coefficients, *SIAM J. Sci. Comput.,* **22,** 1943–1961.

Haber, E. & Heldmann, S., 2007. An octree multigrid method for quasi-static Maxwell's equations with highly discontinuous coefficients, *J. Comput. Phys.,* **223,** 783–796.

Harris, C.R. *et al.*, 2020. Array programming with NumPy, *Nature,* **585,** 357–362.

Heagy, L.J., Cockett, R., Kang, S., Rosenkjaer, G.K. & Oldenburg, D.W., 2017. A framework for simulation and inversion in electromagnetics, *Comput. Geosci.,* **107,** 1–19.

Hohmann, G.W., 1975. Three-dimensional induced polarization and electromagnetic modeling, *Geophysics,* **40,** 309–324.

Hursán, G. & Zhdanov, M.S., 2002. Contraction integral equation method in three-dimensional electromagnetic modeling, *Radio Sci.,* **37,** 1–1-1-13.

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

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.

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, *Geophys. J. Int.,* **203,** 1541–1574.

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, *Geophys. J. Int.,* **207,** 1554–1572.

Key, K., 2016. MARE2DEM: a 2-D inversion code for controlled-source electromagnetic and magnetotelluric data, *Geophys. J. Int.,* **207**(1), 571–588.

Kruglyakov, M. & Bloshanskaya, L., 2017. High-performance parallel solver for integral equations of electromagnetics based on Galerkin method, *Math. Geosci.,* **49,** 751–776.

Kruglyakov, M., Geraskin, A. & Kuvshinov, A., 2016. Novel accurate and scalable 3-D MT forward solver based on a contracting integral equation method, *Comput. Geosci.,* **96,** 208–217.

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, *Simula SpringerBriefs on Computing,* Vol. 3. Springer International Publishing.

Lebedev, V.I., 1964. Difference analogues of orthogonal decompositions, basic differential operators and some boundary problems of mathematical physics. I, *USSR Comput. Math. Math. Phys.*, **4,** 69–92.

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.

Logg, A., Mardal, K.-A. & Wells, G., 2012. Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book, *Lecture Notes in Computational Science and Engineering,* Vol. 84, Springer-Verlag.

Maaø, F.A., 2007. Fast finite-difference time-domain modeling for marine-subsurface electromagnetic problems, *Geophysics,* **72,** A19–A23.

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

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

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, *Geophys. J. Int.,* **193,** 1216–1238.

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.

Mulder, W.A., 2006. A multigrid solver for 3D electromagnetic diffusion, *Geophys. Prospect.,* **54,** 633–649.

Newman, G.A. & Alumbaugh, D.L., 1997. Three-dimensional massively parallel electromagnetic inversion–I. Theory, *Geophys. J. Int.,* **128,** 345–354.

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.

Oh, S., Noh, K., Seol, S.J. & Byun, J., 2015. 3D CSEM frequency-domain modeling and inversion algorithms including topography, in *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, *Explor. Geophys.,* 1–13, doi:10.1080/08123985.2019.1580118.

Oristaglio, M. & Spies, B., 1999. Three-Dimensional Electromagnetics, Geophysical Developments, *Society of Exploration Geophysicists,* **Vol. 7,** 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, *Geophys. J. Int.,* **193,** 678–693.

Raiche, A.P., 1974. An integral equation approach to three-dimensional modelling, *Geophys. J. Int.,* **36,** 363–376.

Rücker, C., Günther, T. & Wagner, F.M., 2017. pyGIMLi: an open-source library for modelling and inversion in geophysics, *Comput. Geosci.,* **109,** 106–123.

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

Schenk, O. & Gärtner, K., 2004. Solving unsymmetric sparse systems of linear equations with PARDISO, *Future Gener. Comput. Syst.,* **20,** 475–487.

Schwarzbach, C., Börner, R.-U. & Spitzer, K., 2011. Three-dimensional adaptive higher order finite element simulation for geo-electromagnetics—a marine CSEM example, *Geophys. J. Int.,* **187,** 63–74.

Shantsev, D.V. & Maaø, F.A., 2015. Rigorous interpolation near tilted interfaces in 3-D finite-difference EM modelling, *Geophys. J. Int.,* **200,** 743–755.

Si, H., 2015. Tetgen, a Delaunay-based quality tetrahedral mesh generator, *ACM Trans. Math. Softw.,* **41,** 1–36.

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.

Sommer, M., Hölz, S., Moorkamp, M., Swidinsky, A., Heincke, B., Scholl, C. & Jegen, M., 2013. GPU parallelization of a three dimensional marine CSEM code, *Comput. Geosci.,* **58,** 91–99.

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.

Sullivan, C.B. & Kaszynski, A.A., 2019. PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK), *J. Open Source Softw.,* **4,** 1450.

Tehrani, A.M. & Slob, E., 2010. Fast and accurate three-dimensional controlled source electromagnetic modelling, *Geophys. Prospect.,* **58,** 1133–1146.

Uieda, L., 2018. Verde: Processing and gridding spatial data using Green's functions, *J. Open Source Softw.,* **3,** 957 , doi:10.21105/joss.00957.

Virtanen, P. *et al.*, 2020. SciPy 1.0: fundamental algorithms for scientific computing in python, *Nat. Methods,* **17,** 261–272.

Wang, F., Morten, J.P. & Spitzer, K., 2018. Anisotropic three-dimensional inversion of CSEM data using finite-element techniques on unstructured grids, *Geophys. J. Int.,* **213,** 1056–1072.

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

Wannamaker, P.E. & Zhdanov, M., 2002. *Three-Dimensional Electromagnetics,* No. 35 in Methods in Geochemistry and Geophysics, Elsevier Publishing Company.

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

Ward, S.H. & Hohmann, G.W., 1988. Electromagnetic theory for geophysical applications, chap. 4, in Electromagnetic Methods in Applied Geophysics: Volume 1, Theory, 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, *Arch. Elektron. Übertrag.tech.,* **31,** 116–120.

Werthmüller, D., 2017. An open-source full 3D electromagnetic modeler for 1D VTI media in Python: empymod, *Geophysics,* **82,** WB9–WB19.

Werthmüller, D., Mulder, W.A. & Slob, E.C., 2019. emg3d: A multigrid solver for 3D electromagnetic diffusion, *J. Open Source Softw.,* **4,** 1463, doi:10.21105/joss.01463.

Wirianto, M., Mulder, W.A. & Slob, E.C., 2011. Applying essentially non-oscillatory interpolation to controlled- source electromagnetic modelling, *Geophys. Prospect.,* **59**(1), 161–175.

Yee, K., 1966. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media, *IEEE Trans. Antennas Propag.,* **14,** 302–307.

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, 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.