1 Novel accurate and scalable 3-D MT forward solver 2 based on a contracting integral equation method

3 M. Kruglyakov^{a,b,*}, A. Geraskin^a, A. Kuvshinov^a

4 ^a*ETH Zurich, Sonneggstrasse 5, 8092 Zurich, Switzerland*

5 ^b*Lomonosov Moscow State University, Leninskiye Gory 1-52, 119991 Moscow, Russia*

6 Abstract

We present a novel, open source 3-D MT forward solver based on a method of integral equations (IE) with contracting kernel. Special attention in the solver is paid to accurate calculations of Green's functions and their integrals which are cornerstones of any IE solution. The solver supports massive parallelization and is able to deal with highly detailed and contrasting models. We report results of a 3-D numerical experiment aimed at analyzing the accuracy and scalability of the code.

7 *Keywords:* MT modelling, 3-D conductivity models, integral equation
8 method, HPC, C#

9 1. Introduction

10 Three-dimensional (3-D) electromagnetic (EM) studies of the Earth, from
11 near surface to regional and global scales, have advanced significantly over the
12 past decade. However 3-D EM simulations – which are a core part of any 3-D
13 interpretation – with realistic levels of complexity, accuracy and spatial detail
14 still remain challenging from the computational point of view. The approaches
15 to improve the efficacy of 3-D simulations include: 1) parallelization of the
16 codes; 2) development or implementation of faster and more accurate numerical
17 algorithms; and 3) careful coding of the proposed numerical algorithms to make
18 a code readable, maintainable and extendable. In this paper we present a new
19 EM forward code (solver), named *extrEMe*, in which all three aforementioned
20 approaches are realized. The solver is based on an integral equation (IE) method

Preprint submitted to Computers & Geosciences

August 19, 2016

*Corresponding author

Email address: M.Kruglyakov@gmail.com (M. Kruglyakov)

21 with contracting kernel (cf. [1, 2]) hereafter denoted as the contracting integral
22 equation (CIE) method.

23 The paper is organized as follows. Section 2 sketches the CIE approach.
24 Section 3 discusses the programming concepts, underlying new CIE code, and
25 explains an implementation of parallelization. Section 4 reports results of a
26 numerical experiment aimed at analyzing the accuracy and scalability of the
27 code. The paper also includes a number of appendices which detail numerical
28 aspects of the code.

29 2. CIE approach in a nutshell

30 There are three basic numerical simulation techniques for computing frequency-
31 domain 3-D EM fields or responses, namely finite-difference (FD) methods,
32 finite-element (FE) methods and volume integral equation (IE) methods. For
33 decades FD schemes ([3, 4, 5, 6], among others) dominated in EM, however in
34 the last years FE ([7, 8, 9, 10, 11, 12, 13], among others) and IE ([14, 15, 16,
35 17, 18, 19], among others) methods have gained the growing popularity. The
36 solver discussed in this paper is based on an IE approach. The application of
37 IE to 3-D EM modelling dates to the works of [20, 21, 22]. Since then, the IE
38 concept has progressed dramatically thus making modern IE realizations com-
39 petitive with the FD- and FE-based codes, irrespective of the complexity and
40 extension of the 3-D conductivity models under consideration. This is especially
41 true for a particular class of IE – the IE with contracting kernel, which possesses
42 a remarkable property: its system matrix is well-conditioned irrespective of dis-
43 cretization, frequency and contrasts of conductivity. What follows is a sketch
44 of the CIE approach.

45 Let (real-valued) σ describe a 3-D conductivity distribution in the model. We
46 assume that this distribution is represented by anomaly(ies) with conductivity
47 $\sigma_a(\mathbf{r})$, $\mathbf{r} = (x, y, z)$, embedded in a layered media of background conductivity
48 $\sigma_b(z)$ (see Figure 1). We also assume that all materials are non-magnetic and
49 magnetic permeability is μ_0 everywhere, and frequencies are low enough to use

50 **quasi-static approximation.** We further assume that the model is excited by an
 51 electric source $\mathbf{j}^{\text{imp}}$.

We search for the electric field, $\mathbf{E}$, and magnetic field, $\mathbf{H}$, excited by $\mathbf{j}^{\text{imp}}$ in the model with conductivity σ . These fields obey Maxwell's equations

$$\begin{cases} \nabla \times \mathbf{H} = \sigma \mathbf{E} + \mathbf{j}^{\text{imp}}, \\ \nabla \times \mathbf{E} = i\omega\mu_0 \mathbf{H}. \end{cases} \quad (1)$$

52 Here $i = \sqrt{-1}$, ω is an angular frequency, and μ_0 is the magnetic permeability
 53 of free space. Time dependence of the fields is accounted for by $e^{-i\omega t}$, which
 54 reads, for example, for electric field as $\check{\mathbf{E}}(\mathbf{r}, t) = \int \mathbf{E}(\mathbf{r}, \omega) e^{-i\omega t} d\omega$. All fields, $\mathbf{E}$,
 55 $\mathbf{H}$, and $\mathbf{j}^{\text{imp}}$, are complex-valued functions of ω and $\mathbf{r}$.

To solve eqs (1), let us first consider a solution of Maxwell's equations in background conductivity model

$$\begin{cases} \nabla \times \mathbf{H}_b = \sigma_b \mathbf{E}_b + \mathbf{j}^{\text{imp}}, \\ \nabla \times \mathbf{E}_b = i\omega\mu_0 \mathbf{H}_b. \end{cases} \quad (2)$$

56 We will call $\mathbf{E}_b$ and $\mathbf{H}_b$ the background electric and magnetic fields, respectively.
 57 If we are able to construct and calculate fundamental solutions (“current-to-
 58 electric”, $G_{\sigma_b}^{ej}$, and “current-to-magnetic”, $G_{\sigma_b}^{hj}$, tensor Green's functions) of
 59 eqs (2), then $\mathbf{E}_b$ and $\mathbf{H}_b$ can be determined via convolution integrals

$$\mathbf{E}_b(\mathbf{r}) = \int_{\mathbb{V}^{ext}} G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') \mathbf{j}^{\text{imp}}(\mathbf{r}') dv', \quad (3)$$

$$\mathbf{H}_b(\mathbf{r}) = \int_{\mathbb{V}^{ext}} G_{\sigma_b}^{hj}(\mathbf{r}, \mathbf{r}') \mathbf{j}^{\text{imp}}(\mathbf{r}') dv'. \quad (4)$$

60 Here $\mathbf{r} \in \mathbb{R}^3$, $\mathbf{r}' \in \mathbb{V}^{ext}$, and $\mathbb{V}^{ext}$ is the volume occupied by $\mathbf{j}^{\text{imp}}$. Note that
 61 in all quantities discussed (except conductivities) the dependence on ω is im-
 62 plicit, but omitted for the simplicity of presentation. The subscript “ σ_b ” means
 63 that Green's functions depend not only on $\mathbf{r}, \mathbf{r}'$ (and ω) but also – on the 1-D
 64 conductivity distribution $\sigma_b(z)$.

Introducing further “scattered” fields, $\mathbf{E}_s = \mathbf{E} - \mathbf{E}_b$ and $\mathbf{H}_s = \mathbf{H} - \mathbf{H}_b$, and subtracting eqs (2) from eqs (1), one can write Maxwell's equations for

scattered fields as

$$\begin{cases} \nabla \times \mathbf{H}_s = \sigma_b \mathbf{E}_s + \mathbf{j}^q, \\ \nabla \times \mathbf{E}_s = i\omega\mu_0 \mathbf{H}_s, \\ \mathbf{E}_s(\mathbf{r}), \mathbf{H}_s(\mathbf{r}) \longrightarrow 0 \quad \text{as } |\mathbf{r}| \longrightarrow \infty. \end{cases} \quad (5)$$

where

$$\mathbf{j}^q = (\sigma - \sigma_b) \mathbf{E}_s + \mathbf{j}^s, \quad (6)$$

and

$$\mathbf{j}^s = (\sigma - \sigma_b) \mathbf{E}_b. \quad (7)$$

65 Comparing eqs (5) and eqs (2) one can deduce that the scattered fields, $\mathbf{E}_s$ and
66 $\mathbf{H}_s$, in analogy with eqs (3) and (4), can be calculated as

$$\mathbf{E}_s(\mathbf{r}) = \int_{\mathbb{V}^{mod}} G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') \mathbf{j}^q(\mathbf{r}') dv', \quad (8)$$

$$\mathbf{H}_s(\mathbf{r}) = \int_{\mathbb{V}^{mod}} G_{\sigma_b}^{hj}(\mathbf{r}, \mathbf{r}') \mathbf{j}^q(\mathbf{r}') dv', \quad (9)$$

67 where $\mathbf{r} \in \mathbb{R}^3$, $\mathbf{r}' \in \mathbb{V}^{mod}$, and $\mathbb{V}^{mod}$ is a (modelling) region where $\sigma_a(\mathbf{r}) =$
68 $\sigma(\mathbf{r}) - \sigma_b(z)$ differs from 0, i.e. a volume occupied by anomalies (see, again,
69 Figure 1). If we restrict ourselves to $\mathbf{r} \in \mathbb{V}^{mod}$, we obtain from eq. (8) a
70 (conventional) integral equation with respect to the unknown scattered electric
71 field, $\mathbf{E}_s$

$$\mathbf{E}_s(\mathbf{r}) - \int_{V^{mod}} G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') [\sigma(\mathbf{r}') - \sigma_b(z')] \mathbf{E}_s(\mathbf{r}') dv' = \mathbf{E}_0(\mathbf{r}), \quad (10)$$

where the free term, $\mathbf{E}_0(\mathbf{r})$, is given by

$$\mathbf{E}_0(\mathbf{r}) = \int_{V^{mod}} G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') \mathbf{j}^s(\mathbf{r}') dv'. \quad (11)$$

Note that in Cartesian geometry (which is our case) an action of Green's operator is a convolution with respect to horizontal coordinates, which means that

$$G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') \equiv G_{\sigma_b}^{ej}(x - x', y - y', z, z'). \quad (12)$$

72

73 After discretization, eq. (10) can be solved using Krylov subspace iterations
 74 (cf. [23]). However, for high-contrasting models the resulting system of linear
 75 equations is poorly conditioned which makes the numerical solution of eq. (10)
 76 unstable.

77 The remedy to overcome this problem is to modify eq. (10) to an integral
 78 equation with contraction operator (cf. [1, 2])

$$\chi(\mathbf{r}) - \int_{V^{mod}} K(\mathbf{r}, \mathbf{r}') R(\mathbf{r}') \chi(\mathbf{r}') dv' = \chi_0(\mathbf{r}), \quad (13)$$

79 where

$$R = \frac{\sigma - \sigma_b}{\sigma + \sigma_b}, \quad (14)$$

$$K(\mathbf{r}, \mathbf{r}') = \delta(\mathbf{r} - \mathbf{r}') I + 2\sqrt{\sigma_b(z)} G_{\sigma_b}^{ej}(\mathbf{r}, \mathbf{r}') \sqrt{\sigma_b(z')}, \quad (15)$$

$$\chi_0(\mathbf{r}) = \int_{V^{mod}} K(\mathbf{r}, \mathbf{r}') \frac{\sqrt{\sigma_b(z')}}{\sigma(\mathbf{r}') + \sigma_b(z')} \mathbf{j}^s(\mathbf{r}') dv', \quad (16)$$

$$\chi = \frac{1}{2\sqrt{\sigma_b}} \{(\sigma + \sigma_b) \mathbf{E}_s + \mathbf{j}^s\}. \quad (17)$$

80 Here $\delta(\mathbf{r} - \mathbf{r}')$ is Dirac's delta function and I is the identity operator. The ad-
 81 vantage of this form of integral equation is that after discretization the resulting
 82 system of linear equations is well conditioned even for high-contrasting models
 83 (cf. [24]). Note that the specific form of eq. (13) is motivated by the energy
 84 inequality for the scattered EM field, which expresses a fundamental physical
 85 fact that the energy flow of the scattered field outside the domain with inhom-
 86ogeneities is always non-negative (cf. [1, 2]). We call scattering equation (13) a
 87 contracting integral equation (CIE).

88 Figure 2 sketches the generic flow-chart of MT field computations if the
 89 CIE approach is invoked. After discretization of the 3-D modeling domain
 90 (cf. Figure 1) by $N_x \times N_y \times N_z$ rectangular prisms, finding unknown EM field

91 on a grid includes two key steps: 1) calculation of elements of A which are
 92 expressed via integrals of corresponding elements of Green's operator $G_{\sigma_b}^{ej}$, and
 93 2) numerical solution of CIE, which is nothing but the solution of system of
 94 linear equations in the form $A\mathbf{x} = \mathbf{b}$, where $\mathbf{x} = \chi$, $\mathbf{b} = \chi_0$ and $A = (I - KR)$.
 95 A numerical scheme for calculating matrix elements is discussed in appendices
 96 to this paper.

97 Note that A is a compact (but dense) matrix. The reason for compactness
 98 is that the boundary conditions are exactly accounted for via Greens functions,
 99 and thus the modeling region is confined only to 3-D anomalies. An efficient
 100 (parallelized) scheme to solve CIE numerically is discussed in Section 3.3.

101 3. Software implementation

102 3.1. Overview

103 The *extrEMe* has modular architecture which allows the interchange and
 104 combination of components to follow different specifications. For example, cal-
 105 culation of elements of A can be performed by either the collocation method
 106 (which is described in appendices to this paper) or the Galerkin method (an
 107 implementation from [19]). For details about the difference between these two
 108 methods the reader is referred to Appendix A. Most of the code is written in
 109 the C# language; some performance-critical parts and wrappers to common C
 110 and Fortran libraries are implemented using C and Fortran languages. The
 111 program can be executed on both Windows and Linux systems, and only C and
 112 Fortran parts of the code need to be recompiled. C# code is compiled into
 113 platform-independent Common Intermediate Language (CIL) and does not re-
 114 quire recompilation [25] to run on different platforms. Both desktop machines
 115 and supercomputers are supported.

116 Massive parallelization is done by means of a Message Passing Interface
 117 (MPI). All common implementations of MPI standard (OpenMPI, Microsoft
 118 MPI, MPICH and Intel MPI) are supported. Threaded parallelization is per-
 119 formed with the use of OpenMP in C and Fortran parts of the code, and Task

Figure 1: Sketch of 3-D model under consideration. Generic discretization is shown, which stresses the fact that it is uniform in lateral direction and nonuniform in the vertical direction. Triangles represent the observation sites.

Figure 2: Generic flow-chart of *extrEMe* code.

120 Parallel Library (TPL) [25] in **C#**. For mathematical calculations, such as matrix-
121 vector multiplications and fast convolution, corresponding libraries are used,
122 including **Intel MKL**, **FFTW3**, **OpenBlas**. Some of them can be interchanged,
123 depending on which platform one runs the program.

124 The overall design of the program is built with a view to optimize it for
125 each platform individually and to implement various use cases, still keeping the
126 program as solid as possible.

127 3.2. *C# language*

128 As it was already stated the code is written using **C#** which is a rather uncon-
129 ventional language in the geophysical community. What follows is a summary
130 of the most important facts about **C#**.

131 **C#** is a multi-paradigm general-purpose programming language. It embraces
132 strong typing, imperative, declarative, functional, generic and object-oriented
133 programming disciplines [25]. Like **C++**, **Fortran** (newer versions) and Python
134 it allows one perform rapid development and quickly write graceful code without
135 losing it's readability and maintainability.

136 There are a few myths about **C#** which we would like to dispel. One of the
137 myths claims that **C#** is very slow and therefore it is not suitable for high perfor-
138 mance computing. However, although **C#** runs on top of a virtual machine which
139 abstracts the native architecture, before a first run it compiles to native code
140 with maximum optimizations for the target processor. Note that math-intensive
141 operations like matrix-vector multiplications and Fast Fourier Transform can
142 not be efficiently implemented directly in **C#**. However even with **Fortran** these
143 operations are not implemented, but reused from standard libraries. And there
144 are no obstacles to do the same with **C#**. Following the Pareto principle [26],
145 only a small amount of code performs time-consuming calculations. All other
146 parts of the code build infrastructure: a data model (description of the objects
147 represented by a computer system), data flows, input and output format data
148 readers/writers etc. These parts usually do not take significant time during the
149 execution, but can take much time and efforts during coding. And the quality

150 of these parts influences how easy it will be to maintain the code in the future.

151 **C#** allows one to build infrastructure more efficiently than **C** or **Fortran**.

152 Another myth is about the proprietary status of **C#**. **C#** is the main pro-
153 gramming language designed for the Common Language Infrastructure (CLI).
154 The CLI is an *open* specification developed by Microsoft and standardized by
155 ISO(ISO/IEC 23271:2012 - Information technology – CLI) and ECMA (Stan-
156 dard ECMA-335 - 6th edition, June 2012) that describes the executable code
157 and run time environment. The specification defines an environment that al-
158 lows multiple high-level languages to be used on different computer platforms
159 without being rewritten for specific architectures.

160 One more preconception is the rarity of **C#**. However, according to TIOBE
161 programming community index – which is a measure of the popularity of pro-
162 gramming languages, calculated from a number of search engine results for
163 queries containing the name of the language – by January 2016 **C#** takes the
164 fourth place (after **Java**, **C**, and **C++**). By contrast **Fortran** takes the 24-th
165 place.

166 Lastly, there is a delusion that **C#** is available only for Windows. But in
167 reality, programs, written in **C#**, can be readily executed under Linux-based and
168 Mac OS systems using the Mono project, which is an open source implemen-
169 tation of Microsoft’s .NET Framework based on the ECMA standards for **C#**
170 and the Common Language Runtime. Moreover, the next version of .NET stack
171 called .NET Core was announced at the end of 2014 to be fully open source.
172 This means that the future implementations of .NET for Windows, Linux and
173 Mac OS will have the same code base, and there will be no place for different
174 behavior of the run time, which is still presently possible (but in rather rare
175 cases).

176 The *extrEMe* code was successfully tested on machine with Windows 8.1 64-
177 bit with Intel Core 7 processor, on the Brutus supercomputer (ETH Zurich) with
178 CentOS 6.6 and AMD Opteron processors, and on the Piz Daint supercomputer
179 (Swiss National Supercomputing Centre) with Cray Linux OS and Intel Xeon
180 processors. Note also, that if **C** and **Fortran** parts of the program should be

181 recompiled with different OS-dependent compiler keys, the main part of the
182 program written in **C#**, after being compiled under Windows, successfully runs
183 without modification and recompilation on any platform.

184 3.3. Parallelization implementation

185 Generally speaking, two levels of parallelization can be distinguished: these
186 are shared memory (or multithreaded) parallelization, and distributed memory
187 parallelization. Distributed memory refers to a multiple-processor computer
188 system in which each processor has its own private memory. For shared mem-
189 ory systems parallel tasks share a global address space which they read and
190 write to asynchronously. Nowadays supercomputers very often have mixed ar-
191 chitecture: multi-core processors are combined into nodes, within which they
192 have shared memory architecture; communication between nodes is organized
193 in a distributed way. Multi-core desktop machines can work in both paradigms.
194 The MPI has been developed for distributed memory parallel programming.
195 The standard contains a set of routines which provides synchronization and
196 communication functionality between a set of processes. The great part of par-
197 allelization in *extrEMe* is done through MPI. However MPI routines are quite
198 low-level. This means that using them directly leads to rather “dirty” code
199 with a mixture of different abstractions. That is why a few levels of abstraction
200 were introduced in *extrEMe*. On the low level there exists a wrapper, which
201 envelops native MPI methods and provides access to them for **C#**. This allows
202 for changing MPI implementation without touching (and without recompiling)
203 other parts of the code. Currently, work with **OpenMPI**, **MPICH**, and **MS-MPI**
204 implementations has been realized and tested. Our parallelization scheme is
205 designed to be especially efficient for laterally large models. The scheme follows
206 a concept introduced in [19] and is sketched below.

Recall that one of the key steps of any numerical implementation of IE is the
solution of the system of linear equations $A\mathbf{x} = \mathbf{b}$ with a dense matrix A . To
solve it as efficiently as practicable we use specific properties of the CIE kernel,
including its convolution nature (cf. eqs 12 and 15). It allows one to write the

system matrix A in the form [19]

$$A = S + R_1 B R_2, \quad (18)$$

where S , R_1 , R_2 are the diagonal matrices, and B is a block-Toeplitz matrix. We solve the system $A\mathbf{x} = \mathbf{b}$ by Krylov subspace iterations, using our own distributed implementation of the FGMRES method [27]. It is known that the most time-consuming part of any iterative method for solving the system $A\mathbf{x} = \mathbf{b}$ is the matrix-vector multiplication. Since matrix B is a block Toeplitz matrix, we can use the 2D Fast Fourier Transform (FFT) to speed up this convolution-type operation [28]. This, in particular, means that instead of matrix B we need to store its 2D discrete Fourier transformation, $\tilde{B}$. Therefore, we need to store $8 \times 2N_x \times 2N_y \times N_z^2$ complex values (cf. [28]). Here, 8 stands for the number of entries of 3×3 Green function dyad minus 1, due to symmetry of two dyad elements (cf. eq. A.2). This symmetry is trivially checked by substituting A.5 into A.2. Note also that duplicating the horizontal dimension is required to calculate the convolution.

The algorithm for the multiplication of the block Toeplitz matrix B with some vector $\mathbf{v}$ (to obtain vector $\mathbf{u} \equiv B\mathbf{v}$) can be represented as a sequence of the following steps:

1. Compute $3 \times N_z$ times forward 2D FFT of vector $\mathbf{v}$ to obtain $\tilde{\mathbf{v}}$ of dimension $3 \times 2N_x \times 2N_y \times N_z$. Here, 3 stands for the number of components (x , y and z) of vector $\mathbf{v}$;
2. Compute $9 \times 2N_x \times 2N_y = 36 \times N_x \times N_y$ algebraic matrix-vector multiplications to obtain vector $\tilde{\mathbf{u}}$;
3. Compute $3N_z$ backward 2D FFTs of vector $\tilde{\mathbf{u}}$ to obtain $\mathbf{u}$ of dimension $3 \times N_x \times N_y \times N_z$.

Note, that $36 \times N_x \times N_y$ matrix-vector multiplications at the second step can be trivially separated into $4 \times N_x \times N_y$ groups which are data independent. This allows one to use the following scheme of distributed data storage and parallel algorithm of IE solver. Let us assume, for the sake of simplicity, that

we have $2N_x$ nodes. The distributed storage of the matrix is organized as follows. The block-vector of dimension $8 \times 2N_y \times N_z^2$ is stored at the k -th node, $k = 1 \dots 2N_x$. This storage organization allows for using only local algebraic matrix-vector multiplications and hiding all data transfer inside the distributed 2D FFT algorithm. The algebraic matrix-vector multiplication is processed by MKL or OpenBLAS depending on their availability, whereas the distributed 2D FFT is processed by our own implementation based on the ideas from [29] and the specifics of our problem. In brief, we perform multiple 1D FFTs concurrently, then transpose (between different nodes) the output, perform 1D FFTs along the other dimension and transpose the data back. The 1D local FFT is processed by the FFTW3 package [29].

4. Numerical Experiment

For a numerical experiment we used the COMMEMI 3D-II model (shown in Figure 3), but with much higher maximum lateral contrast. Resistivity A in the left block is $0.1 \Omega\text{m}$ and resistivity B in the right block is $1000 \Omega\text{m}$, giving a maximum lateral contrast of 10000 (recall that in the standard COMMEMI 3D-II model the maximum lateral contrast is 100). The anomalous volume comprising two adjacent blocks was discretized by $N_x \times N_y \times N_z = 512 \times 512 \times 128$ cells. Note that anomalous volume coincides with the modeling volume, and the grid is equidistant in the x - and y -directions. Figures 4 - 6 show the magnetotelluric (MT) responses at a period of 1 sec computed by the *extrEMe* code and by the modern finite element (FE) code of [13] (for the definition of MT responses the reader is referred to the book [30]). Since in all figures one can hardly see the discrepancy between the results, in Figure 7 the relative differences between *extrEMe* and FE apparent resistivities are demonstrated, and Figures 8 and 9 show the differences between *extrEMe* and FE phases and tippers, respectively. One can see that in overall the agreement is very good, however – and as expected – there is a slight disagreement between the results at the contacts of the blocks.

# of nodes	Green's tensors	CIE solution (x-pol)	CIE solution (y-pol)
64	92	767	800
128	54	426	400
256	28	195	180
512	20	98	94
1024	18	63	59

Table 1: CPU times (see columns 2-4; in seconds) of *extrEMe* code with respect to an increasing number of nodes (see column 1). Note that for MT scenario the integrals of Green's tensor for a given period is computed only once. The CPU times for $A\mathbf{x} = \mathbf{b}$ solution are summarized in the third and fourth columns for x - and y -polarizations, respectively. It was assumed that the approximation to the solution, $\mathbf{x}^{(n)}$, was obtained once the following inequality hold: $\|A\mathbf{x}^{(n)} - \mathbf{b}\|/\|\mathbf{b}\| < \epsilon$, with $\epsilon = 10^{-8}$. Number of matrix-vector multiplications $A\mathbf{x}$ was 574 for x -polarization and 552 for y -polarization.

Table 1 demonstrates the scalability of *extrEMe* code in terms of CPU time (in seconds) with respect to an increasing number of nodes. The same numerical grid, introduced above, was exploited for all runs. Note that the number of degrees of freedom (DoF) in the CIE solution is $3 \times 2 \times N_x \times N_y \times N_z$ where 3 indicates three components of the χ field, and 2 stands for their real and imaginary parts. The computations were performed on Cray XC30 supercomputer "Piz Daint" of the Swiss National Supercomputer Centre. Figure 10 shows the scalability of the code in an alternative way. It is seen that *extrEMe* is indeed scalable, i.e. the run time is inversely proportional to the number of the involved nodes.

5. Conclusions

The *extrEMe* code introduced here shows impressive performance in terms of accuracy and computational loads. Accuracy of the code is achieved by careful (and new) analytical and semi-analytical estimation of Green's functions and their integrals. A new – compared with alternative IE solvers [15, 17] – scheme of parallelization based on distributed FFTs provides very good scalability of

Figure 3: COMMEMI-3D model: side view (left) and plane view (right).

Figure 4: Apparent resistivities, ρ_{xy} (left) and ρ_{yx} (right), along the profile shown in Figure 3 by the dashed line.

Figure 5: Phases of impedance, ϕ_{xy} (left) and ϕ_{yx} (right), along the profile shown in Figure 3 by the dashed line.

Figure 6: Real (left) and imaginary (right) parts of tipper's component W_{zx} , along the profile shown in Figure 3 by the dashed line.

Figure 7: Relative difference between *extrEMe* and FE apparent resistivities, ρ_{xy} (left) and ρ_{yx} (right), along the profile shown in Figure 3 by the dashed line.

Figure 8: Difference between *extrEMe* and FE phases of impedance, φ_{xy} (left) and φ_{yx} (right), along the profile shown in Figure 3 by the dashed line.

Figure 9: Difference between *extrEMe* and FE real (left) and imaginary (right) parts of tipper's component W_{zx} , along the profile shown in Figure 3 by the dashed line.

Figure 10: Strong scalability of *extrEMe* code at COMMEMI 3D-II model. Note, that in contrast to Table 1 the CPU time for CIE is shown for two polarizations.

the code up to thousands of cores. We believe that *extrEMe* could be a competitive alternative to the most advanced FE- and FD-based codes, irrespective of complexity, contrasts, expansion and spatial detail of 3-D conductivity models under consideration.

In this paper we discuss implementation of the code to an MT problem. The extension of the code on the controlled-source case is rather straightforward and is the topic of ongoing realization.

In contrast to existing IE solvers which are proprietary, the *extrEMe* is the open source and free available for academia software. The sources can be cloned from git repository by `git clone git@bitbucket.org/em_geophysics/extrEMe.git` or downloaded by direct link: https://bitbucket.org/em_geophysics/extreme/get/master.zip.

6. Acknowledgements

The authors would like to thank Alexander Grayver for many seminal discussions over the course of this work and for his input to Section 4. This work was supported by a grant from the Swiss National Supercomputing Centre (CSCS)

under project ID s577, and in part by the Russian Foundation for Basic Research under grant no. 13-05-12111. Mikhail Kruglyakov also acknowledges the support of the Swiss National Science Foundation (grant no. IZK0Z2 163494) and the ETH Zurich during his stay as a visiting fellow.

References

- [1] O. Pankratov, D. Avdeyev, A. Kuvshinov, Electromagnetic field scattering in a heterogeneous Earth: A solution to the forward problem, *Izvestiya, Physics of the Solid Earth* (3) (1995) 201–209.
- [2] B. Singer, Method for solution of Maxwell’s equations in non-uniform media, *Geophys. J. Int.* 120 (1995) 590–598.
- [3] R. Mackie, J. Smith, T. Madden, 3-Dimensional electromagnetic modeling using finite-difference equation – The magnetotelluric example, *Radio Science* 29 (4) (1994) 923–935.
- [4] E. Haber, U. M. Ascher, Fast finite volume simulation of 3D electromagnetic problems with highly discontinuous coefficients, *SIAM Journal of Scientific Computing* 22(6) (2001) 1943–1961.
- [5] G. Newman, D. Alumbaugh, Three-dimensional induction logging problems, Part 2: A finite-difference solution, *Geophysics* 61 (2002) 484–491.
- [6] G. D. Egbert, A. Kelbert, Computational recipes for electromagnetic inverse problems, *Geophys. J. Int.* 189 (1) (2012) 251–267. doi:{10.1111/j.1365-246X.2011.05347.x}.
- [7] C. Schwarzbach, R.-U. Börner, K. Spitzer, Three-dimensional adaptive higher order finite element simulation for geo-electromagnetics - a marine CSEM example, *Geophys. J. Int.* 187 (2011) 63–74.
- [8] C. G. Farquharson, M. P. Miensopust, Three-dimensional finite-element modelling of magnetotelluric data with a divergence correction, *Journal of Applied Geophysics* 75 (2011) 699–710.

- [9] V. Puzyrev, J. Koldan, J. de la Puente, G. Houzeaux, M. Vazquez, J. M. Cela, Efficient pre-conditioned iterative solution strategies for the electromagnetic diffusion in the earth: finite-element frequency-domain approach, *Geophys. J. Int.* 193 (2013) 678–693.
- [10] Z. Ren, T. Kalscheuer, H. Greenhalgh, S. Maurer, A goal-oriented adaptive finite-element approach for plane wave 3-D electromagnetic modelling, *Geophys. J. Int.* 194 (2013) 700–718.
- [11] E. S. Um, M. Commer, G. A. Newman, Efficient pre-conditioned iterative solution strategies for the electromagnetic diffusion in the earth: finite-element frequency-domain approach, *Geophys. J. Int.* 193 (2013) 1460–1473.
- [12] A. Grayver, M. Burg, Robust and scalable 3-D geo-electromagnetic modelling approach using the finite element method, *Geophys. J. Int.* 198 (2014) 110–125.
- [13] A. Grayver, T. Kolev, Large-scale 3D geo-electromagnetic modeling using parallel adaptive high-order finite element method, *Geophysics* 80 (6) (2015) 277–291.
- [14] D. Avdeev, A. Kuvshinov, O. Pankratov, G. Newman, Three-dimensional induction logging problems, Part I: An integral equation solution and model comparisons, *Geophysics* 67 (2) (2002) 413–426.
- [15] G. Hursan, M. Zhdanov, Contraction integral equation method in three-dimensional electromagnetic modeling, *Radio Science* 37. doi:10.1029/2001RS002513.
- [16] B. Singer, Electromagnetic integral equation approach based on contraction operator and solution optimization in Krylov subspace, *Geophys. J. Int.* 175 (2008) 857–884.
- [17] T. Koyama, H. Utada, D. Avdeev, Fast and memory-saved 3-D forward modeling code for MT by using integral equation method, in: Abstract

- 350 book, 19th workshop on electromagnetic induction in the Earth, China,
351 2008.
- 352 [18] J. Kamm, L. Pedersen, Inversion of airborne tensor VLF data using integral
353 equations, *Geophys. J. Int.* doi:10.1093/gji/ggu161.
- 354 [19] M. Kruglyakov, L. Bloshanskaya, High-performance parallel solver of elec-
355 tromagnetics integral equations based on Galerkin method, *Mathematical*
356 *Geoscience*, also at <http://arxiv.org/abs/1512.06126> (2016) submitted.
- 357 [20] A. Raiche, An integral equation approach to three-dimensional modelling,
358 *Geophys. J. R. astr. Soc.* 36 (1974) 363–376.
- 359 [21] G. Hohmann, Three-dimensional induced polarization and electromagnetic
360 modeling, *J. Geophys.* 40 (1975) 309–324.
- 361 [22] P. Weidelt, Electromagnetic induction in three-dimensional structures, *J.*
362 *Geophys.* 41 (1975) 85–109.
- 363 [23] A. Greenbaum, Iterative methods for solving linear systems, Philadelphia,
364 USA, 1997.
- 365 [24] O. Pankratov, A. Kuvshinov, Applied mathematics in EM studies with
366 special emphasis on an uncertainty quantification and 3-D IE modelling,
367 *Survey in Geophysics* (2015) doi 10.1007/s10712-015-9340-4.
- 368 [25] J. Mayo, *C# 3.0 Unleashed: With the .NET Framework 3.5*, 1st Edition,
369 Sams, 2008.
- 370 [26] M. J. Joseph, M. G. Frank, *Jurans quality control handbook*, McGraw-Hill
371 (1988).
- 372 [27] Y. Saad, A flexible inner-outer preconditioned GMRES algorithm, *SIAM*
373 *J Sci Comput* 14 (2) (1993) 461–469.
- 374 [28] D. B. Avdeev, A. V. Kuvshinov, O. V. Pankratov, G. A. Newman, High-
375 performance three-dimensional electromagnetic modelling using modified

- 376 Neumann series. wide-band numerical solution and examples, J. Geomagn.
377 Geoelectr. 49 (1997) 1519–1539.
- 378 [29] M. Frigo, S. G. Johnson, The design and implementation of FFTW3, Pro-
379 ceedings of the IEEE 93 (2) (2005) 216–231, special issue on “Program
380 Generation, Optimization, and Platform Adaptation”.
- 381 [30] A. Chave, A. Jones, The magnetotelluric method - Theory and Practice,
382 Cambridge University Press, Cambridge, UK, 2012.
- 383 [31] V. Dmitriev, A. Silkin, R. Farzan, Tensor Green function for the system of
384 Maxwell’s equations in a layered medium, Comput. Math. and Modeling
385 13 (2) (2002) 107–118.
- 386 [32] W. Anderson, Numerical integration of related Hankel transforms of order
387 0 and 1 by adaptive digital filtering, Geophysics 44 (1979) 1287–1305.
- 388 [33] H. K. Johansen, K. Sorensen, Fast Hankel Transforms, Geophysical
389 Prospecting 27 (1979) 876–901.
- 390 [34] N. Christensen, Optimized fast Hankel transform filters, Geophysical
391 Prospecting 38 (1990) 545–568.

392 **Appendix A. Numerical scheme to estimate Green operator inte-** 393 **grals**

394 Before discussing numerical algorithms to estimate Green’s functions and its
395 integrals we would like to notice that there are two approaches to solve CIE nu-
396 merically, namely, collocation method (CM as in [15, 17, 18, 28]), and Galerkin
397 method (GM; as in [16, 19]). The actual difference between the methods lies in
398 a way how to approximate continuous CIE by the system of linear equations.
399 Both methods are based on a piece-wise approximation of the fields, however in
400 CM one computes the field in the center of the cell, whereas in GM one finds the
401 average of the field over the cell. Therefore in Galerkin method matrix elements
402 are the double volumetric integrals of Green’s tensor, whereas in collocation

method they are single volumetric integrals. Thus GM requires much more computational resources to calculate matrix elements. However this disadvantage is somehow counterbalanced by the fact that GM guarantees convergence to the solution, whereas CM does not.

What follows is an explanation of how matrix elements are calculated within our implementation of CM which is a method of choice in *extrEMe* code. Note, that in spite of potential divergence we never experienced the situations when *extrEMe* code failed to converge to the solution.

There exist several techniques within collocation method how to calculate Green’s functions and their integrals (i.e. matrix elements) [15, 17, 18, 28]. Here we present our own way to compute these quantities. First, we use alternative expression for integrands with respect to horizontal variables (cf. this Appendix), and provide analytical formulas for the integrals based on adopted approximation (cf. Appendix C). Second, we introduce the concept of multiple Hankel transformation in order to improve the accuracy of estimating “horizontal” integrals (cf. Appendix B). Lastly, we demonstrate that integration in vertical direction could be performed analytically (cf. Appendix D) and implement such integration in our code. As far as we know, in alternative IE solvers based on collocation method (except [17]), integration in vertical direction is performed numerically.

As can be seen from the explanation in Section 2 the key operation which is multiply invoked in the CIE method is the volumetric integration of type given by the following expression

$$\mathbf{F}(x, y, z) = \int_{\mathbb{V}_{\mathbf{j}}} G_{\sigma_b}^{ej}(x - x', y - y', z, z') \mathbf{j}(x', y', z') dv', \quad (\text{A.1})$$

where $\mathbb{V}_{\mathbf{j}}$ is the volume occupied by $\mathbf{j}$. For example, $\mathbf{F} \equiv \mathbf{E}_b$, $\mathbb{V}_{\mathbf{j}} \equiv \mathbb{V}^{ext}$, and $\mathbf{j} \equiv \mathbf{j}^{imp}$, if background electric field $\mathbf{E}_b$ is calculated (cf. with eq. 3). We now explain a numerical scheme for evaluating integrals given by eq. (A.1).

We start with explicit formulas for elements of a 3×3 dyadic Green’s function

$G_{\sigma_b}^{ej}$. Using the ideas from [28] and [31] we obtain

$$G_{\sigma_b}^{ej} = \begin{pmatrix} \frac{\partial}{\partial y} \left[\frac{\xi}{\rho} g_{tt} \right] + \frac{\partial}{\partial x} \left[\frac{\eta}{\rho} g_{tp} \right] & \frac{\partial}{\partial y} \left[\frac{\eta}{\rho} (g_{tt} - g_{tp}) \right] & \frac{\partial}{\partial x} g_{tz} \\ \frac{\partial}{\partial x} \left[\frac{\xi}{\rho} (g_{tt} - g_{tp}) \right] & \frac{\partial}{\partial x} \left[\frac{\eta}{\rho} g_{tt} \right] + \frac{\partial}{\partial y} \left[\frac{\xi}{\rho} g_{tp} \right] & \frac{\partial}{\partial y} g_{tz} \\ -\frac{\partial}{\partial x} g_{zt} & -\frac{\partial}{\partial y} g_{zt} & \left(\frac{\partial}{\partial x} \eta + \frac{\partial}{\partial y} \xi \right) \left[\frac{g_{zz}}{\rho} \right] \end{pmatrix}, \quad (\text{A.2})$$

where

$$\eta = x - x' \quad \xi = y - y', \quad (\text{A.3})$$

and

$$\rho = \sqrt{\eta^2 + \xi^2}. \quad (\text{A.4})$$

Functions $g_*(\rho, z, z')$ are called Green's scalar functions and are expressed as the following Hankel transforms

$$\begin{aligned} g_{tt}(\rho, z, z') &= -\frac{i\omega\mu_0}{2} \int_0^\infty J_1(\lambda\rho) U_1(z, z', \lambda) d\lambda, \\ g_{tp}(\rho, z, z') &= \frac{1}{2\sigma_b(z)} \int_0^\infty J_1(\lambda\rho) \frac{\partial^2}{\partial z \partial z'} U_\sigma(z, z', \lambda) d\lambda, \\ g_{tz}(\rho, z, z') &= \frac{1}{2\sigma_b(z)} \int_0^\infty J_0(\lambda\rho) \frac{\partial}{\partial z'} U_\sigma(z, z', \lambda) \lambda d\lambda, \\ g_{zt}(\rho, z, z') &= \frac{1}{2\sigma_b(z)} \int_0^\infty J_0(\lambda\rho) \frac{\partial}{\partial z} U_\sigma(z, z', \lambda) \lambda d\lambda, \\ g_{zz}(\rho, z, z') &= \frac{1}{2\sigma_b(z)} \int_0^\infty J_1(\lambda\rho) U_\sigma(z, z', \lambda) d\lambda. \end{aligned} \quad (\text{A.5})$$

426 Here $U_\gamma(z, z', \lambda)$, $\gamma = 1, \sigma$ are fundamental functions of the layered media [31].

427 The definition and properties of these functions are presented in Appendix D.

Let the volume $V_{\mathbf{j}}$ be represented as

$$\mathbb{V}_{\mathbf{j}} = \bigcup_{l=1}^{N_{\mathcal{M}}} \mathbb{V}_l, \quad (\text{A.6})$$

and within each volume $\mathbb{V}_l$ the field $\mathbf{j}$ be a constant $\mathbf{j}_l$. Then eq. (A.1) can be rewritten as

$$\mathbf{F}(x, y, z) = \sum_{l=1}^{N_{\mathcal{M}}} \mathbf{j}_l \int_{\mathbb{V}_l} G_{\sigma_b}^{ej}(x - x', y - y', z, z') dv'. \quad (\text{A.7})$$

428 We will now show how the integrals over specific $\mathbb{V}_l$ are computed within
 429 our CIE solution. Let rectangular volume $\mathbb{V}_l$ be represented as $\mathbb{V}_l = [q_l, q_l +$
 430 $h_x] \times [p_l, p_l + h_y] \times [d_l, d_l + h_z]$.

The integrals over the vertical domain $[d_l, d_l + h_z]$ can be written as follows

$$\tilde{g}_\nu^l(\rho, z) = \int_{d_l}^{d_l+h_l} g_\nu(\rho, z, z') dz' = \int_0^\infty J_{\mu_1}(\lambda, \rho) \tilde{V}_\nu^l(\lambda, z) \lambda^{\mu_2} d\lambda, \quad (\text{A.8})$$

431 where μ_1, μ_2 depend on ν as in (A.5). Here $\tilde{V}_\nu^l(\lambda, z)$ are integrals over $[d_l, d_l +$
 432 $h_z]$ of functions $U_\gamma(z, z', \lambda)$ and their partial derivatives. These integrals are
 433 calculated analytically using formulas for U_γ (see Appendix D). Hence, the
 434 main computational challenge of the integrating $G_{\sigma_b}^{ej}$ over some volume $\mathbb{V}_l$ is
 435 the integration over the horizontal domain $[q_l, q_l + h_x] \times [p_l, p_l + h_y]$, which is
 436 explained below.

For simplicity of presentation we omit index l in further exposition. Substituting (A.2) into (A.7) one can see that the integrals of Green's tensor components over the horizontal domain $[q, q + h_x] \times [p, p + h_y]$ are expressed as integrals of three types

$$\begin{aligned} I_1(x, y) &= \int_p^{p+h_y} \int_q^{q+h_x} \frac{\partial}{\partial \alpha} f_1(\rho) dx' dy', \\ I_2(x, y) &= \int_p^{p+h_y} \int_q^{q+h_x} \frac{\partial}{\partial \alpha} \left(\frac{\alpha}{\rho} f_2(\rho) \right) dx' dy', \\ I_3(x, y) &= \int_p^{p+h_y} \int_q^{q+h_x} \frac{\partial}{\partial \alpha} \left(\frac{\beta}{\rho} f_3(\rho) \right) dx' dy'. \end{aligned} \quad (\text{A.9})$$

Here f_* is one of $\tilde{g}_*$ functions, $\alpha, \beta = x, y$, $\alpha \neq \beta$. In the following we will consider only the case when $\alpha = x$, $\beta = y$, since integrals for $\alpha = y$, $\beta = x$ are computed in the same manner. Taking the inner integrals in (A.9) analytically,

we obtain

$$\begin{aligned}
I_1 &= \int_p^{p+h_y} \{f_1(\rho_1) - f_1(\rho_2)\} dy', \\
I_2 &= \int_p^{p+h_y} \left\{ f_2(\rho_2) \frac{q+h_x}{\rho_2} - f_2(\rho_1) \frac{q}{\rho_1} \right\} dy', \\
I_3 &= \int_p^{p+h_y} \left\{ f_3(\rho_2) \frac{y'}{\rho_2} - f_3(\rho_1) \frac{y'}{\rho_1} \right\} dy',
\end{aligned} \tag{A.10}$$

where $\rho_1 = \sqrt{(q-x)^2 + (y-y')^2}$ and $\rho_2 = \sqrt{(q+h_x-x)^2 + (y-y')^2}$.

The integrals in (A.10) are found numerically by the following procedure

1. Using multiple Hankel transform (see Appendix B) we calculate functions $\tilde{g}_*$ in (A.8) for N_ρ points ρ_i $i = 1 \dots N_\rho$;
2. For all functions $\tilde{g}_*$ we construct piece-wise linear interpolating functions (see Appendix C);
3. The integrals from (A.10) are approximated by integrals over interpolating functions taken analytically (see again Appendix C).

Appendix B. Multiple Hankel Transformation

First we notice that integrals on the RHS of (A.8) can be written in the following general form

$$f(\rho, z) = \int_0^\infty J_{0,1}(\lambda \rho) F(\lambda, z) d\lambda, \tag{B.1}$$

where $f = \tilde{g}_\nu$, $F(\lambda, z) = V_\nu(\lambda, z) \lambda^{0,1}$ for the $\nu \in \{tt, tp, tz, zt, zz\}$. The most popular numerical scheme to compute integrals like (B.1) is a digital filtering [32, 33, 34]. In our study we use Christensen [34] implementation of this approach because it has flexible trade-off between accuracy and computational time. The key feature of this implementation is the possibility to compute $f_i = f(\rho_i)$, $\rho_i = \rho_{min} 10^{\frac{i}{N_{dec}}}$ $i = 0 \dots M-1$ by using only $L+M$ values of $F(\lambda)$. The N_{dec} and L are the method parameters which are specified by desired precision.

453 However, the distance between knots $d_i = \rho_{i+1} - \rho_i = \rho_{min} 10^{\frac{i}{N_{dec}}} \left(1 + 10^{\frac{1}{N_{dec}}}\right)$
 454 increases exponentially with i . If we want to use piece-wise linear interpolation
 455 of f between knots, then large d_i decreases accuracy of this interpolation. This
 456 problem is overcome by using $N \geq 1$ geometric sequences of $\rho_{i,p} = \rho_p 10^{\frac{i}{N_{dec}}}$,
 457 $\rho_p = \rho_{min} \left(1 + \frac{p}{N} 10^{\frac{1}{N_{dec}}}\right)$, $p = 0 \dots N - 1$, $i = 0 \dots M - 1$.

458 Using this scheme we compute $F(\lambda)$ for $N(L + M)$ values of λ and obtain
 459 $f(\rho)$ in $N_\rho = NM$ points. Our computational experiments show that $N = 10$,
 460 $N_{dec} = 40$, and $L = 431$ allow the computation of integrals in (A.10) with the
 461 relative error from 10^{-8} to 10^{-3} depending on the distance between cells. As
 462 expected from the algorithm, the errors increase with the distance, but they are
 463 always sufficiently small for the subsequent calculations.

464 Appendix C. Estimating integrals along horizontal cell

Using $f_i = f(\rho_i)$, $i = 1 \dots N_\rho$ we construct piece-wise linear continuous
 function $\tilde{f}(\rho)$ as

$$\begin{aligned} \tilde{f}(\rho) &= a_i \rho + b_i \text{ if } \rho_i \leq \rho \leq \rho_{i+1}, \\ a_i &= \frac{f_{i+1} - f_i}{\rho_{i+1} - \rho_i}, \quad b_i = f_i - \rho_i a_i \\ i &= 1 \dots N_\rho - 1. \end{aligned} \tag{C.1}$$

465 It is obvious that $\tilde{f}$ interpolates f .

Suppose t, s, u are some numbers $u \geq s$ and there exists $1 \leq i \leq N_\rho - 1$ such
 that $\rho_{i+1}^2 \geq u^2 + t^2$, $s^2 + t^2 \geq \rho_i^2$; then we define $Q_\gamma^i(t, s, u)$, $\gamma = \{1, 2, 3\}$ as
 follows

$$\begin{aligned} Q_1^i(t, s, u) &= \int_s^u \tilde{f}(r) dy, \\ Q_2^i(t, s, u) &= t \int_s^u \tilde{f}(r) \frac{dy}{r}, \\ Q_3^i(t, s, u) &= \int_s^u \tilde{f}(r) \frac{y}{r} dy, \end{aligned} \tag{C.2}$$

where $r = \sqrt{t^2 + y^2}$. Since $\tilde{f}(r)$ is a piece-wise linear function, the integrals in (C.2) are taken analytically as

$$\begin{aligned}
Q_1^i(t, s, u) &= \int_s^u \tilde{f}(r) dy = \int_s^u (a_i r + b_i) dy = \\
&= b_i(u - s) + \frac{a_i}{2} \left(ur_u - sr_s + t^2 \ln \left| \frac{u + r_u}{s + r_s} \right| \right), \\
Q_2^i(t, s, u) &= t \int_s^u \tilde{f}(r) \frac{dy}{r} = \\
&= a_i t(u - s) + t b_i \ln \left| \frac{u + r_u}{s + r_s} \right|, \\
Q_3^i(t, s, u) &= \int_s^u \tilde{f}(r) \frac{y}{r} dy = \int_s^u \left(a_i y + b_i \frac{y}{r} \right) dy = \\
&= \frac{a_i}{2} (u^2 - s^2) + b_i (r_u - r_s),
\end{aligned} \tag{C.3}$$

466 where $r_s = \sqrt{s^2 + t^2}$ and $r_u = \sqrt{u^2 + t^2}$.

If f is one of $\tilde{g}_*$ functions from (A.8) and I_γ is the corresponding integral from (A.10), then using (C.3) we obtain following approximations for I_γ :

$$\begin{aligned}
I_\gamma &\approx Q_\gamma^{n_1}(q - x, p, \tilde{y}_{n_1+1}(q - x)) - \\
&Q_\gamma^{n_2}(q + h_x - x, p, \tilde{y}_{n_2+1}(q + h_x - x)) + \\
&\sum_{j=n_1+1}^{m_1-1} Q_\gamma^j(q - x, \tilde{y}_j(q - x), \tilde{y}_{j+1}(q - x)) - \\
&\sum_{j=n_2+1}^{m_2-1} Q_\gamma^j(q + h_x - x, \tilde{y}_j(q + h_x - x), \tilde{y}_{j+1}(q + h_x - x)) + \\
&Q_\gamma^{m_1}(q - x, \tilde{y}_{m_1}(q - x), p + h_y) - \\
&Q_\gamma^{m_2}(q + h_x - x, \tilde{y}_{m_2}(q + h_x - x), p + h_y),
\end{aligned} \tag{C.4}$$

where $\tilde{y}_k(t)^2 + t^2 = \rho_k^2$, the sign of $\tilde{y}_k(t)$ goes with signs of p , $p + h_y$, and

n_1, n_2, m_1, m_2 are determined from the following inequalities

$$\begin{aligned}
\tilde{y}_{n_1}(q-x) &\leq p \leq \tilde{y}_{n_1+1}(q-x), \\
\tilde{y}_{n_2}(q-x+h_x) &\leq p \leq \tilde{y}_{n_2+1}(q-x+h_x), \\
\tilde{y}_{m_1}(q-x) &\leq p+h_y \leq \tilde{y}_{m_1+1}(q-x), \\
\tilde{y}_{m_2}(q-x+h_x) &\leq p+h_y \leq \tilde{y}_{m_2+1}(q-x+h_x).
\end{aligned} \tag{C.5}$$

467 Appendix D. Fundamental functions of the layered media

468 For fast and accurate calculation of the integrals of Green's tensors over
469 the vertical domain we use the so-called fundamental functions of the layered
470 media [31]. Consider the media with $N_{lay} + 1$ homogeneous layers with conduc-
471 tivities σ_n , $n = 0 \dots N_{lay}$, where 0-th and N_{lay} -th layers represent the upper
472 (air) and the lower halfspaces, respectively.

Function $U_\gamma(z, z', \lambda)$ is defined as a unique solution of the problem

$$\left\{ \begin{aligned}
&\frac{\partial^2}{\partial z^2} U_\gamma(z, z', \lambda) - \eta_0^2 U_\gamma(z, z', \lambda) = 0, z < d_1, z \neq z', \\
&\frac{\partial^2}{\partial z^2} U_\gamma(z, z', \lambda) - \eta_n^2 U_\gamma(z, z', \lambda) = 0, d_n < z < d_{n+1}, z \neq z', \\
&\frac{\partial^2}{\partial z^2} U_\gamma(z, z', \lambda) - \eta_{N_{lay}}^2 U_\gamma(z, z', \lambda) = 0, z > d_{N_{lay}}, z \neq z', \\
&[U_\gamma(z, z', \lambda)]_{z=d_n} = 0, \\
&\left[\frac{1}{\gamma} \frac{\partial}{\partial z} U_\gamma(z, z', \lambda) \right]_{z=d_n} = 0, \\
&[U_\gamma(z, z', \lambda)]_{z=z'} = 0, \\
&\left[\frac{\partial}{\partial z} U_\gamma(z, z', \lambda) \right]_{z=z'} = -2, \\
&|U_\gamma(z, z', \lambda)| \rightarrow 0 \text{ as } z \rightarrow \pm\infty, \\
&\eta_m^2 = \lambda^2 - k_m^2, \quad k_m^2 = i\omega\mu_0\sigma_m, \\
&m = 0 \dots N_{lay}, \quad n = 1 \dots N_{lay} - 1, \quad 0 < \lambda < \infty.
\end{aligned} \right. \tag{D.1}$$

The following procedure is performed to obtain an explicit expression for U_γ thus allowing analytical (vertical) integration. Let $l_0 = 0$, $l_{N_{lay}} = 0$, $l_n =$

$d_{n+1} - d_n$, $n = 1 \dots N_{lay} - 1$. Define p_m^γ , q_m^γ , $m = 0 \dots N_{lay}$ by the recurrent expressions

$$\begin{aligned}
p_0^\gamma &= 0; & q_{N_{lay}}^\gamma &= 0; \\
p_1^\gamma &= \frac{1 - \alpha_0^\gamma \frac{\eta_0}{\eta_1}}{1 + \alpha_0^\gamma \frac{\eta_0}{\eta_1}}; & q_{N_{lay}-1}^\gamma &= \frac{1 - \beta_{N_{lay}}^\gamma \frac{\eta_{N_{lay}}}{\eta_{N_{lay}-1}}}{1 + \beta_{N_{lay}}^\gamma \frac{\eta_{N_{lay}}}{\eta_{N_{lay}-1}}}; \\
p_{m+1}^\gamma &= \frac{1 + \alpha_m^\gamma \frac{\eta_m}{\eta_{m+1}} \frac{p_m^\gamma e^{-2\eta_m l_m} - 1}{p_m^\gamma e^{-2\eta_m l_m} + 1}}{1 - \alpha_m^\gamma \frac{\eta_m}{\eta_{m+1}} \frac{p_m^\gamma e^{-2\eta_m l_m} - 1}{p_m^\gamma e^{-2\eta_m l_m} + 1}}, & q_{m-1}^\gamma &= \frac{1 + \beta_m^\gamma \frac{\eta_m}{\eta_{m-1}} \frac{q_m^\gamma e^{-2\eta_m l_m} - 1}{q_m^\gamma e^{-2\eta_m l_m} + 1}}{1 - \beta_m^\gamma \frac{\eta_m}{\eta_{m-1}} \frac{q_m^\gamma e^{-2\eta_m l_m} - 1}{q_m^\gamma e^{-2\eta_m l_m} + 1}}, \\
m &\neq N_{lay}; & m &\neq 0; \\
\alpha_m^\gamma &= \begin{cases} 1, \gamma = 1; \\ \frac{\sigma_{m+1}}{\sigma_m}, \gamma = \sigma; \end{cases} & \beta_m^\gamma &= \begin{cases} 1, \gamma = 1; \\ \frac{\sigma_{m-1}}{\sigma_m}, \gamma = \sigma. \end{cases}
\end{aligned} \tag{D.2}$$

Let $d_0 = d_1$, $d_{N_{lay}+1} = d_{N_{lay}}$ and let points z_r , z_s belong to r and s layers respectively, $0 \leq r, s \leq N_{lay}$. Then using (D.2) one obtains

$$U_\gamma(z_r, z_s, \lambda) = \begin{cases} A_{r,s}^\gamma (p_r^\gamma e^{2\eta_r d_r} e^{-\eta_r z_r} + e^{\eta_r z_r}) (e^{-\eta_s z_s} + q_s^\gamma e^{-2\eta_s d_{s+1}} e^{\eta_s z_s}) & \text{for } z_r \leq z_s; \\ A_{s,r}^\gamma (p_s^\gamma e^{2\eta_s d_s} e^{-\eta_s z_s} + e^{\eta_s z_s}) (e^{-\eta_r z_r} + q_r^\gamma e^{-2\eta_r d_{r+1}} e^{\eta_r z_r}) & \text{for } z_r > z_s, \end{cases} \tag{D.3}$$

where

$$\begin{aligned}
A_{r,s}^\gamma &= Q_r^\gamma \times Q_{r+1}^\gamma \times \dots \times Q_{s-1}^\gamma A_{s,s}^\gamma, \text{ for } r < s, \\
Q_m^\gamma &= \frac{1 + p_{m+1}^\gamma}{1 + p_m^\gamma e^{-2\eta_m l_m}} e^{(\eta_{m+1} - \eta_m) d_{m+1}}, m = 1 \dots N_{lay} - 1 \\
A_{n,n}^\gamma &= \frac{1}{\eta_n (1 - p_n^\gamma q_n^\gamma e^{-2\eta_n l_n})}, \text{ for } r = s = n, n = 0 \dots N_{lay} \\
A_{r,s}^1 &= A_{s,r}^1, \quad A_{r,s}^\sigma = \frac{\sigma_r}{\sigma_s} A_{s,r}^\sigma \text{ for } r > s.
\end{aligned} \tag{D.4}$$

473 It can be proved by direct calculations that RHS of (D.3) under (D.2) and (D.4)
474 is the unique solution of (D.1).

In view of (D.3) one can see that integrals in (A.8) (i.e. the integrals over $U_\gamma(z, z', \lambda)$ and their partial derivatives) can be integrated analytically with

respect to z, z' over any domains that do not intersect the layer boundaries. However, the rounding errors arising in addition and multiplication of very small or large quantities make the formula (D.3) impractical for $\lambda \gg 1$. Instead the following formula is used

$$U_\gamma(z_r, z_s, \lambda) = \begin{cases} A_{r,s}^\gamma \left(p_r^\gamma e^{-(\eta_r z_r + \eta_s z_s - 2\eta_r d_r)} + q_s^\gamma e^{-(2\eta_s d_{s+1} - (\eta_r z_r + \eta_s z_s))} + \right. \\ \quad \left. e^{-(\eta_s z_s - \eta_r z_r)} + p_r^\gamma q_s^\gamma e^{-(2(\eta_s d_{s+1} - \eta_r d_r) - (\eta_s z_s - \eta_r z_r))} \right) & \text{for } z_r \leq z_s; \\ A_{s,r}^\gamma \left(p_s^\gamma e^{-(\eta_s z_s + \eta_r z_r - 2\eta_s d_s)} + q_r^\gamma e^{-(2\eta_r d_{r+1} - (\eta_s z_s + \eta_r z_r))} + \right. \\ \quad \left. e^{-(\eta_r z_r - \eta_s z_s)} + p_s^\gamma q_r^\gamma e^{-(2(\eta_r d_{r+1} - \eta_s d_s) - (\eta_r z_r - \eta_s z_s))} \right) & \text{for } z_r > z_s. \end{cases} \quad (\text{D.5})$$

475 Formula (D.5) overcomes the aforementioned problem, since the real parts of all
 476 the exponents are negative. The consequent calculations provide accurate and
 477 robust results for any $0 < \lambda < \infty$.