Code-Verification Techniques for the Method-of-Moments Implementation of the Magnetic-Field Integral Equation

Brian A. Freno[a], Neil R. Matula[a]

[a]*Sandia National Laboratories, Albuquerque, NM 87185*

Abstract

For computational physics simulations, code verification plays a major role in establishing the credibility of the results by assessing the correctness of the implementation of the underlying numerical methods. In computational electromagnetics, surface integral equations, such as the method-of-moments implementation of the magnetic-field integral equation, are frequently used to solve Maxwell's equations on the surfaces of electromagnetic scatterers. These electromagnetic surface integral equations yield many code-verification challenges due to the various sources of numerical error and their possible interactions. In this paper, we provide approaches to separately measure the numerical errors arising from these different error sources. We demonstrate the effectiveness of these approaches for cases with and without coding errors.

Keywords: method of moments, magnetic-field integral equation, code verification, manufactured solutions

1. Introduction

In computational electromagnetics, surface integral equations are frequently used to solve Maxwell's equations on the surfaces of electromagnetic scatterers. The electric-field integral equation (EFIE) relates the surface current to the scattered electric field that arises from an incident electric field, whereas the magnetic-field integral equation (MFIE) relates the surface current to the scattered magnetic field that arises from an incident magnetic field.

Through the method-of-moments (MoM) implementation of these surface integral equations, the surface of the electromagnetic scatterer is discretized using planar or curvilinear mesh elements, and four-dimensional integrals are evaluated over two-dimensional source and test elements. However, the presence of a Green's function in these equations yields singularities when the test and source elements share one or more edges or vertices, and near-singularities when they are otherwise close. Many approaches have been developed to address the singularity and near-singularity for the inner, source-element integral [1–10], as well as for the outer, test-element integral [11–15].

Code verification is an important step towards establishing the credibility of the results of a computational physics simulation [16–18]. Through code verification, the correctness of the implementation of the numerical methods is assessed. The discretization of differential, integral, or integro-differential equations incurs some truncation error, and thus the approximate solutions produced from the discretized equations will incur an associated discretization error. If the discretization error tends to zero as the discretization is refined, the consistency of the code is verified [16]. This may be taken a step further by examining not only consistency, but the rate at which the error decreases as the discretization is refined, thereby verifying the order of accuracy of the discretization scheme. The correctness of the numerical-method implementation may then be verified by comparing the expected and observed orders of accuracy obtained from numerous test cases with known solutions.

Because exact solutions are generally limited and may not sufficiently exercise the capabilities of the code, manufactured solutions [19] are a popular alternative, permitting the construction of problems of arbitrary complexity with known solutions. Through the method of manufactured solutions (MMS), a solution is manufactured and substituted directly into the governing equations to yield a residual term, which is added as a source term to coerce the solution to the manufactured solution.

Email address: `bafreno@sandia.gov` (Brian A. Freno)

However, integral equations yield an additional challenge. While analytical differentiation is straightforward, analytical integration is not always possible. Therefore, the residual source term arising from the manufactured solution may not be representable in closed form, and its implementation may be accompanied by numerical techniques that carry their own numerical errors. Furthermore, in many applications, such as the MoM implementation of the MFIE, singular integrals appear, which can further complicate the numerical evaluation of the source term. Therefore, many of the benefits associated with MMS are lost when applied to integral equations in a straightforward manner.

Code verification has been performed on computational physics codes associated with several physics disciplines, including aerodynamics [20], fluid dynamics [21–27], solid mechanics [28], fluid–structure interaction [29], heat transfer in fluid–solid interaction [30], multiphase flows [31, 32], radiation hydrodynamics [33], electrostatics [34], electrodynamics [35], and ablation [36–40]. For surface integral equations in computational electromagnetics, code-verification activities that employ manufactured solutions have been limited to the EFIE [41–44].

As with the EFIE, the numerical solution to the MoM implementation of the MFIE incurs numerical error from three sources:

1. **Domain discretization.** While planar surfaces can be represented exactly by planar elements, the approximation of sufficiently smooth curved surfaces with planar elements introduces a second-order numerical error [45, Chap. 3]. This error can be reduced by employing curved elements [46].

2. **Solution discretization.** Common in the solution to differential, integral, and integro-differential equations, the approximation of the solution in terms of a finite number of basis functions, or alternatively the approximation of the underlying equation operators in terms of a finite amount of solution queries, is the most common contributor to the numerical error. For sufficiently smooth solutions, this error can be reduced by employing higher-order basis functions [46] or stencils.

3. **Numerical integration.** The analytical evaluation of the integrals in integral equations is usually not possible. For well-behaved integrals, quadrature rules or other integration methods can be used, with the expectation that the associated numerical error is at least of the same order as that arising from the solution-discretization error. A less rigorous expectation is that the error from numerical integration decreases as the fidelity of the numerical integration algorithm is increased (e.g., increasing the number of quadrature points). However, for singular or nearly singular integrals, such convergence is not assured [47].

For the EFIE, Marchand et al. [41, 42] compute the MMS source term using additional quadrature points. Freno et al. manufacture the Green's function, permitting the numerical-integration error to be eliminated and the solution-discretization error to be isolated [43]. They also provide approaches to isolate the numerical-integration error [44]. Unlike the EFIE, however, the MFIE is used to model only closed surfaces, introducing additional constraints on the ability to manufacture solutions.

In this paper, we present code-verification techniques for the MoM implementation of the MFIE that measure these three error sources separately, to the extent possible. As in [43], to eliminate the numerical-integration error, we manufacture the Green's function. When the manufactured Green's function makes the equations practically singular, we select a unique solution by minimizing the error in two different norms, each with their own trade-offs. To isolate the numerical-integration error, we present two approaches and show how the numerical-integration error is related to the discretization error. Finally, to address domains with curvature, we present code-verification approaches that account for and neglect the curvature.

It is important to note that, by manufacturing the Green's function, we avoid the challenges associated with evaluating the aforementioned (nearly) singular integrals. Given the computational expense of computing accurate reference solutions, assessing these integral evaluations is best accomplished through extensive unit testing as complementary code verification. Examples for the EFIE are included in [15, 47].

This paper is organized as follows. In Section 2, we describe the MoM implementation of the MFIE. In Section 3, we describe the challenges of using MMS with the MoM implementation of the MFIE, and we describe our approach to mitigating them. In Section 4, we demonstrate the effectiveness of our approaches for cases with and without coding errors and with and without curvature. In Section 5, we summarize this work.

2. The Method-of-Moments Implementation of the MFIE

In time-harmonic form, the scattered magnetic field $\mathbf{H}^{\mathcal{S}}$ due to induced surface currents on a scatterer can be computed by [48]

$$\mathbf{H}^{\mathcal{S}}(\mathbf{x}) = \frac{1}{\mu} \nabla \times \mathbf{A}(\mathbf{x}), \tag{1}$$

where the magnetic vector potential $\mathbf{A}$ is defined by

$$\mathbf{A}(\mathbf{x}) = \mu \int_{S'} \mathbf{J}(\mathbf{x}')G(\mathbf{x}, \mathbf{x}')dS'. \tag{2}$$

In (1) and (2), the integration domain is the closed surface S of a perfectly conducting scatterer. Additionally, $\mathbf{J}$ is the electric surface current density, and G is the Green's function

$$G(\mathbf{x}, \mathbf{x}') = \frac{e^{-jkR}}{4\pi R}, \tag{3}$$

where $R = |\mathbf{x} - \mathbf{x}'|$, $k = \omega\sqrt{\mu\epsilon}$ is the wavenumber, and μ and ϵ are the permeability and permittivity of the surrounding medium.

The total magnetic field $\mathbf{H}$ is the sum of the incident magnetic field $\mathbf{H}^{\mathcal{I}}$ and $\mathbf{H}^{\mathcal{S}}$. On S,

$$\mathbf{n} \times \mathbf{H} = \mathbf{J}, \tag{4}$$

where $\mathbf{n}$ is the unit vector normal to S. From (1) and (2), and noting that $\nabla \times [\mathbf{J}(\mathbf{x}')G(\mathbf{x}, \mathbf{x}')] = \nabla G(\mathbf{x}, \mathbf{x}') \times \mathbf{J}(\mathbf{x}')$ and $\nabla G(\mathbf{x}, \mathbf{x}') = -\nabla' G(\mathbf{x}, \mathbf{x}')$,

$$\mathbf{H}^{\mathcal{S}}(\mathbf{x}) = \int_{S'} [\mathbf{J}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS'$$

when $\mathbf{x}$ is just outside of S. Therefore, at S,

$$\mathbf{n} \times \mathbf{H}^{\mathcal{S}} = \lim_{\mathbf{x} \to S} \mathbf{n} \times \int_{S'} [\mathbf{J}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS' = \frac{1}{2}\mathbf{J} + \mathbf{n} \times \int_{S'} [\mathbf{J}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS', \tag{5}$$

where the final term is evaluated through principal value integration. From (4) and (5) the MFIE at a point on the surface of the scatterer is [49, 50]

$$\frac{1}{2}\mathbf{J} - \mathbf{n} \times \int_{S'} [\mathbf{J}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS' = \mathbf{n} \times \mathbf{H}^{\mathcal{I}}. \tag{6}$$

Projecting (6) onto an appropriate space $\mathbb{V}$ containing vector fields that are tangent to S yields the variational form: find $\mathbf{J} \in \mathbb{V}$, such that

$$\frac{1}{2} \int_{S} \bar{\mathbf{v}} \cdot \mathbf{J}dS - \int_{S} \bar{\mathbf{v}}(\mathbf{x}) \cdot \left(\mathbf{n}(\mathbf{x}) \times \int_{S'} [\mathbf{J}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS' \right) dS = \int_{S} \bar{\mathbf{v}} \cdot (\mathbf{n} \times \mathbf{H}^{\mathcal{I}})dS \tag{7}$$

for all $\mathbf{v} \in \mathbb{V}$, where the bar notation denotes complex conjugation. We can write (7) more succinctly as

$$a(\mathbf{J}, \mathbf{v}) = b(\mathbf{H}^{\mathcal{I}}, \mathbf{v}), \tag{8}$$

where the sesquilinear forms are defined by

$$a(\mathbf{u}, \mathbf{v}) = \frac{1}{2} \int_{S} \bar{\mathbf{v}}(\mathbf{x}) \cdot \mathbf{u}(\mathbf{x})dS - \int_{S} \bar{\mathbf{v}}(\mathbf{x}) \cdot \left(\mathbf{n}(\mathbf{x}) \times \int_{S'} [\mathbf{u}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')]dS' \right) dS, \tag{9}$$

$$b(\mathbf{u}, \mathbf{v}) = \int_{S} \bar{\mathbf{v}}(\mathbf{x}) \cdot [\mathbf{n}(\mathbf{x}) \times \mathbf{u}(\mathbf{x})]dS. \tag{10}$$

To solve the variational problem (8), we discretize S with a mesh composed of triangular elements and approximate $\mathbf{J}$ with $\mathbf{J}_h$ in terms of the Rao–Wilton–Glisson (RWG) basis functions $\mathbf{\Lambda}_j(\mathbf{x})$ [3]:

$$\mathbf{J}_h(\mathbf{x}) = \sum_{j=1}^{n_b} J_j \mathbf{\Lambda}_j(\mathbf{x}), \tag{11}$$

where n_b is the total number of basis functions. There are other suitable basis function choices for the MFIE such as curl-conforming basis functions [51, 52]; however, given that the MFIE is often combined with the EFIE to obtain the combined-field integral equation, we restrict the scope of this work to the RWG basis functions. The RWG basis functions are second-order accurate [45, pp. 155–156], and are defined for a triangle pair by

$$\mathbf{\Lambda}_j(\mathbf{x}) = \begin{cases} \dfrac{\ell_j}{2A_j^+}\boldsymbol{\rho}_j^+, & \text{for } \mathbf{x} \in T_j^+ \\[2mm] \dfrac{\ell_j}{2A_j^-}\boldsymbol{\rho}_j^-, & \text{for } \mathbf{x} \in T_j^- \\[2mm] \mathbf{0}, & \text{otherwise} \end{cases},$$

where ℓ_j is the length of the edge shared by the triangle pair, and A_j^+ and A_j^- are the areas of the triangles T_j^+ and T_j^- associated with basis function j. $\boldsymbol{\rho}_j^+$ denotes the vector from the vertex of T_j^+ opposite the shared edge to $\mathbf{x}$, and $\boldsymbol{\rho}_j^-$ denotes the vector to the vertex of T_j^- opposite the shared edge from $\mathbf{x}$.

These basis functions ensure that $\mathbf{J}_h$ is tangent to the mesh when using planar triangular elements. Additionally, along the shared edge of the triangle pair, the component of $\mathbf{\Lambda}_j(\mathbf{x})$ normal to that edge is unity. Therefore, for a triangle edge shared by only two triangles, the component of $\mathbf{J}_h$ normal to that edge is J_j. The solution is considered most accurate at the midpoint of the edge [45, pp. 155–156]; therefore, we measure the solution at the midpoints.

Defining $\mathbb{V}_h$ to be the span of RWG basis functions associated with the mesh on S, the Galerkin approximation of (8) is now: find $\mathbf{J}_h \in \mathbb{V}_h$, such that

$$a(\mathbf{J}_h, \mathbf{\Lambda}_i) = b(\mathbf{H}^{\mathcal{I}}, \mathbf{\Lambda}_i) \tag{12}$$

for $i = 1, \ldots, n_b$. Letting $\mathbf{J}^h$ denote the vector of coefficients used to construct $\mathbf{J}_h$ (11), (12) can be written in matrix form as $\mathbf{Z}\mathbf{J}^h = \mathbf{V}$, where $J_j^h = J_j$, $V_i = b(\mathbf{H}^{\mathcal{I}}, \mathbf{\Lambda}_i)$, and $Z_{i,j} = a(\mathbf{\Lambda}_j, \mathbf{\Lambda}_i)$.

3. Manufactured Solutions

We define the residual functional for each test basis function as

$$r_i(\mathbf{u}) = a(\mathbf{u}, \mathbf{\Lambda}_i) - b(\mathbf{H}^{\mathcal{I}}, \mathbf{\Lambda}_i). \tag{13}$$

We can write the variational form (8) in terms of (13) as

$$r_i(\mathbf{J}) = a(\mathbf{J}, \mathbf{\Lambda}_i) - b(\mathbf{H}^{\mathcal{I}}, \mathbf{\Lambda}_i) = 0. \tag{14}$$

Similarly, we can write the discretized problem (12) in terms of (13) as

$$r_i(\mathbf{J}_h) = a(\mathbf{J}_h, \mathbf{\Lambda}_i) - b(\mathbf{H}^{\mathcal{I}}, \mathbf{\Lambda}_i) = 0. \tag{15}$$

The method of manufactured solutions modifies (15) to be

$$r_i(\mathbf{J}_h) = r_i(\mathbf{J}_{\mathrm{MS}}), \tag{16}$$

where $\mathbf{J}_{\mathrm{MS}}$ is the manufactured solution, and $r_i(\mathbf{J}_{\mathrm{MS}})$ is computed exactly.

Inserting (14) and (15) into (16) yields

$$a(\mathbf{J}_h, \mathbf{\Lambda}_i) = a(\mathbf{J}_{\mathrm{MS}}, \mathbf{\Lambda}_i). \tag{17}$$

However, instead of solving (17), we can equivalently solve (12) by setting

$$b\big(\mathbf{H}^{\mathcal{I}}, \boldsymbol{\Lambda}_i\big) = a(\mathbf{J}_{\mathrm{MS}}, \boldsymbol{\Lambda}_i). \tag{18}$$

Equation (18) is satisfied by

$$\mathbf{n} \times \mathbf{H}^{\mathcal{I}} = \frac{1}{2}\mathbf{J}_{\mathrm{MS}} - \mathbf{n} \times \int_{S'} \big[\mathbf{J}_{\mathrm{MS}}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')\big] dS'. \tag{19}$$

Noting that $\mathbf{J}_{\mathrm{MS}} - (\mathbf{J}_{\mathrm{MS}} \cdot \mathbf{n})\mathbf{n} = \mathbf{n} \times (\mathbf{J}_{\mathrm{MS}} \times \mathbf{n})$ and that $\mathbf{J}_{\mathrm{MS}}$ must be tangential to the surface ($\mathbf{J}_{\mathrm{MS}} \cdot \mathbf{n} = 0$), $\mathbf{J}_{\mathrm{MS}} = \mathbf{n} \times (\mathbf{J}_{\mathrm{MS}} \times \mathbf{n})$ can be inserted into (19), and factoring out the cross product with $\mathbf{n}$ yields

$$\mathbf{H}^{\mathcal{I}} = \frac{1}{2}\mathbf{J}_{\mathrm{MS}} \times \mathbf{n} - \int_{S'} \big[\mathbf{J}_{\mathrm{MS}}(\mathbf{x}') \times \nabla' G(\mathbf{x}, \mathbf{x}')\big] dS', \tag{20}$$

which we can use to solve (12).

However, as described in the introduction, integrals containing the Green's function (3) or its derivatives, such as those appearing in (20), cannot be computed analytically. Additionally, the singularity when $R \to 0$ complicates their accurate approximation, potentially contaminating convergence studies. Therefore, as is done in [43], we manufacture the Green's function, using the form

$$G_{\mathrm{MS}}(\mathbf{x}, \mathbf{x}') = \left(1 - \frac{R^2}{R_m^2}\right)^d, \tag{21}$$

where $R_m = \max_{\mathbf{x}, \mathbf{x}' \in S} R$ is the maximum possible distance between two points on S, and $d \in \mathbb{N}$. The form of (21) is chosen for two reasons: 1) the even powers of R permit the integrals in (12) and (20) to be computed analytically for many choices of $\mathbf{J}_{\mathrm{MS}}$, avoiding contamination from additional error, and 2) G_{MS} increases when R decreases, as with the actual Green's function (3).

We update (12) to account for the modifications arising from $\mathbf{J}_{\mathrm{MS}}$ and G_{MS}:

$$a(\mathbf{J}_h, \boldsymbol{\Lambda}_i) = b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \boldsymbol{\Lambda}_i\big), \tag{22}$$

where

$$\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}} = \frac{1}{2}\mathbf{J}_{\mathrm{MS}} \times \mathbf{n} - \int_{S'} \big[\mathbf{J}_{\mathrm{MS}}(\mathbf{x}') \times \nabla' G_{\mathrm{MS}}(\mathbf{x}, \mathbf{x}')\big] dS',$$

and, for notational simplicity, $a(\cdot, \cdot)$ uses G_{MS} (21) in (9), instead of G (3).

3.1. Solution-Discretization Error

In (22), if the integrals in $a(\cdot, \cdot)$ are evaluated exactly, the only contribution to the discretization error is the solution-discretization error. Solving for $\mathbf{J}^h$ enables us to compute the discretization error

$$\mathbf{e}_{\mathbf{J}} = \mathbf{J}^h - \mathbf{J}_n, \tag{23}$$

where J_{n_j} denotes the component of $\mathbf{J}_{\mathrm{MS}}$ flowing from T_j^+ to T_j^-. The norm of (23) has the property $\|\mathbf{e}_{\mathbf{J}}\| \leq C_{\mathbf{J}} h^{p_{\mathbf{J}}}$, where $C_{\mathbf{J}}$ is a function of the solution derivatives, h is representative of the mesh size, and $p_{\mathbf{J}}$ is the order of accuracy. By performing a mesh-convergence study of the norm of the discretization error, we can ensure the expected order of accuracy is obtained. For the RWG basis functions, the expectation is second-order accuracy ($p_{\mathbf{J}} = 2$).

3.1.1. Solution Uniqueness

For the EFIE, the manufactured Green's function G_{MS} (21) yields a matrix that is practically singular, admitting infinite solutions $\mathbf{J}^h$. For the MFIE, the matrix arising from the first term of (9), which does not contain a Green's function, is nonsingular. Additionally, the matrix arising from the first and second terms is nonsingular when G_{MS} is used. However, the matrix arising from only the second term is practically singular.

In [43, 53], a mitigation approach is presented to select $\mathbf{J}^h$, by solving the optimization problem

$$
\begin{aligned}
\text{minimize} \quad & \|\mathbf{e_J}\|_2 \\
\text{subject to} \quad & a(\mathbf{J}_h, \mathbf{\Lambda}_i) = b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{\Lambda}_i\big).
\end{aligned}
\tag{24}
$$

The solution to (24) is

$$
\mathbf{J}^h = \mathbf{J}_n + \mathbf{Q}_1\big(\mathbf{u} - \mathbf{Q}_1^H \mathbf{J}_n\big),
$$

where $\mathbf{R}_1^H \mathbf{u} = \mathbf{P}^T \mathbf{V}$ and $\mathbf{Q}_1$, $\mathbf{R}_1$, and $\mathbf{P}$ arise from the pivoted QR factorization of $\mathbf{Z}^H$.

In formulating the optimization problem (24), we selected $\|\mathbf{e_J}\|_2$ as the cost function because it has the benefit of a closed form solution. However, in order to assess the rate of convergence, we must select a second norm that is used to report the error. $\|\mathbf{e_J}\|_\infty$ is often preferred for error reporting in code verification because it is more sensitive and therefore more rigorous. However, the mismatch between the minimized norm (L^2) and the reported norm (L^∞) may require finer meshes to enter the asymptotic region. Therefore, as a trade-off, in this paper, we additionally solve the optimization problem

$$
\begin{aligned}
\text{minimize} \quad & \|\mathbf{e_J}\|_\infty \\
\text{subject to} \quad & a(\mathbf{J}_h, \mathbf{\Lambda}_i) = b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{\Lambda}_i\big).
\end{aligned}
\tag{25}
$$

The L^∞-norm of the error arising from the solution to (25) reaches the asymptotic region faster than that arising from (24) but requires the solution to a linear programming problem, which is more expensive.

3.2. Numerical-Integration Error

In practice, the integrals in $a(\cdot, \cdot)$ (9) and $b(\cdot, \cdot)$ (10) are evaluated numerically, yielding the approximations $a^q(\cdot, \cdot)$ and $b^q(\cdot, \cdot)$. $a^q(\cdot, \cdot)$ and $b^q(\cdot, \cdot)$ are obtained by integrating over each triangular element using quadrature, and generally incur a numerical-integration error. Therefore, it is important to measure the numerical-integration error without contamination from the solution-discretization error. To do this, we build upon the solution-discretization error cancellation and elimination approaches of [44] by defining the error functionals

$$
e_a(\mathbf{u}) = a^q(\mathbf{u}, \mathbf{u}) - a(\mathbf{u}, \mathbf{u}),
\tag{26}
$$
$$
e_b(\mathbf{u}) = b^q\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{u}\big) - b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{u}\big),
\tag{27}
$$

which have the properties $|e_a| \le C_a h^{p_a}$ and $|e_b| \le C_b h^{p_b}$, where C_a and C_b are functions of the integrand derivatives, and p_a and p_b depend on the quadrature accuracy.

In Section 3.2.1, we describe our approach to canceling the solution-discretization error, and in Section 3.2.2, we describe our approach to eliminating the solution-discretization error.

3.2.1. Solution-Discretization Error Cancellation

In this paper, we cancel the solution-discretization error and measure the numerical-integration error from

$$
e_a(\mathbf{J}_{h_{\mathrm{MS}}}) = a^q(\mathbf{J}_{h_{\mathrm{MS}}}, \mathbf{J}_{h_{\mathrm{MS}}}) - a(\mathbf{J}_{h_{\mathrm{MS}}}, \mathbf{J}_{h_{\mathrm{MS}}}),
\tag{28}
$$
$$
e_b(\mathbf{J}_{h_{\mathrm{MS}}}) = b^q\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{h_{\mathrm{MS}}}\big) - b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{h_{\mathrm{MS}}}\big),
\tag{29}
$$

where $\mathbf{J}_{h_{\mathrm{MS}}}$ is the basis-function representation of $\mathbf{J}_{\mathrm{MS}}$, obtained from (11) by setting the coefficients J_j equal to the normal component of $\mathbf{J}_{\mathrm{MS}}$ at the midpoint of each edge associated with $\mathbf{\Lambda}_j(\mathbf{x})$. Due to the presence of the basis functions in the minuend and subtrahend of (28) and (29), we have canceled the solution-discretization error. This approach is more advantageous than the solution-discretization error cancellation approach of [44], as it does not require solving a potentially inconsistent system of equations.

To relate $e_a(\mathbf{J}_{h_{\mathrm{MS}}})$ to the discretization error $\mathbf{e_J}$ (23), we can solve the optimization problem

$$
\begin{aligned}
\text{minimize} \quad & \|\mathbf{e_J}\|_2 \\
\text{subject to} \quad & a^q(\mathbf{J}_h, \mathbf{J}_{h_{\mathrm{MS}}}) = a(\mathbf{J}_{h_{\mathrm{MS}}}, \mathbf{J}_{h_{\mathrm{MS}}}).
\end{aligned}
\tag{30}
$$

Similarly, to relate $e_b(\mathbf{J}_{h_{\mathrm{MS}}})$ to $\mathbf{e_J}$, we can solve the optimization problem

$$
\begin{aligned}
\text{minimize} \quad & \|\mathbf{e_J}\|_2 \\
\text{subject to} \quad & b^q\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_h\big) = b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{h_{\mathrm{MS}}}\big).
\end{aligned}
\tag{31}
$$

In (30) and (31), $\|\mathbf{e_J}\|_\infty$ could be minimized instead; however, in this paper, minimizing $\|\mathbf{e_J}\|_2$ is sufficient to reach the asymptotic region for the meshes considered.

3.2.2. Solution-Discretization Error Elimination

Canceling the solution-discretization error enables us to assess how the numerical integration performs for the integrands arising from the approximated solution. The polynomial degrees of these integrands are finite, depending on those of the basis functions and the manufactured Green's function. Therefore, the ability to perform convergence studies on quadrature rules capable of integrating higher polynomial degrees is limited. Alternatively, we can assess the performance of the numerical integration without polynomial degree limits by removing the basis-function restriction.

In this paper, we eliminate the solution-discretization error and measure the numerical-integration error from

$$
e_a(\mathbf{J}_{\mathrm{MS}}) = a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}}) - a(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}}),
\tag{32}
$$

$$
e_b(\mathbf{J}_{\mathrm{MS}}) = b^q\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\big) - b\big(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\big).
\tag{33}
$$

By eliminating the presence of the basis functions in (32) and (33), we have eliminated the solution-discretization error.

3.3. Domain-Discretization Error

For domains with curvature, the discretization of S with planar triangular elements yields the faceted approximation S_h. To address domain-discretization error, we consider two code-verification approaches: accounting for the curvature by using curved triangular elements and neglecting the curvature by using planar triangular elements.

3.3.1. Accounting for Curvature

We restrict the scope of the basis functions in this paper to planar RWG basis functions. Therefore, because they rely on basis functions, isolating the solution-discretization error in Section 3.1 and isolating the numerical-integration error by canceling the solution-discretization error in Section 3.2.1 will not account for curvature since the faceted approximation to the geometry S_h is used.

For the solution-discretization error elimination approach of Section 3.2.2, instead of integrating over a planar triangle, we can integrate over a curved triangle that conforms to S. Through this approach, we modify the planar triangular integrals to include the determinant of the transformation between the planar and curved triangles. By doing so, we can assess the curvature implementation by measuring the numerical-integration error.

3.3.2. Neglecting Curvature

To neglect curvature, the solution-discretization error cancellation approach of Section 3.2.1 enables us to isolate and measure the numerical-integration error by computing the integrals on S_h instead of S.

4. Numerical Examples

In this section, we demonstrate the effectiveness of the approaches described in Section 3. In Section 4.1, we consider domains without curvature, thereby avoiding domain-discretization error, and we isolate and measure the solution-discretization error (Section 3.1) and the numerical-integration error (Section 3.2). In Section 4.2, we consider a domain with curvature. To address the domain-discretization error, we (1) account for the curvature, which modifies the integrands, such that we measure the numerical-integration error (Section 3.3.1), and (2) neglect the curvature, such that we isolate and measure the numerical-integration error (Section 3.3.2).
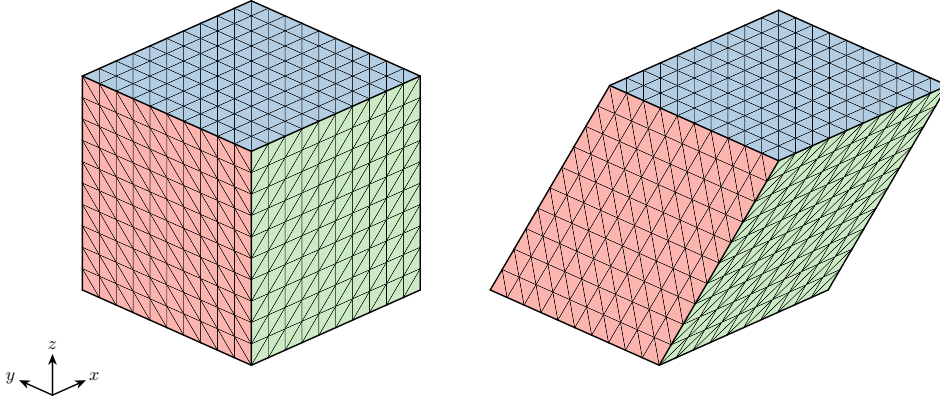
Figure 1: Meshes for the cube (left) and the rhombic prism (right), with $n_t = 1200$.
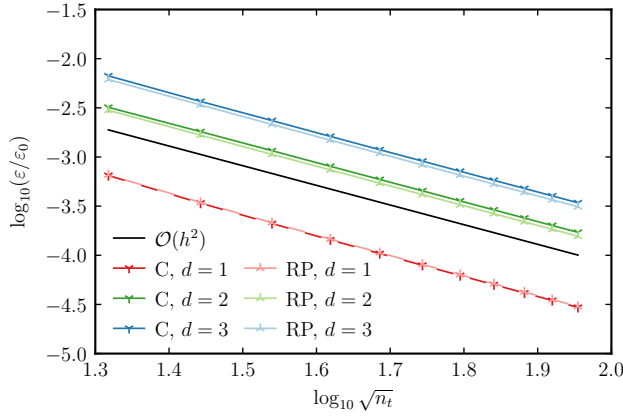
When measuring a norm of the discretization error $\|\mathbf{e_J}\|$ (23), we nondimensionalize by the constant $\varepsilon_0 = 1$ A/m. When measuring the numerical-integration error $e_a(\cdot)$ (26) or $e_b(\cdot)$ (27), we nondimensionalize by the constant $\varepsilon_0 = 1$ A^2.

4.1. Without Curvature

We consider two domains without curvature: a cube and a rhombic prism, each with all edges of length 1 m, as shown in Figures 1 and 2 with the total number of triangles $n_t = 1200$. The acute angle of the rhombic prism is 45°. We manufacture the surface current density $\mathbf{J}_{\mathrm{MS}}(\mathbf{x}) = J_\xi(\xi, \eta)\mathbf{e}_\xi$, where

$$
J_\xi(\xi, \eta) = J_0 \begin{cases} \sin\left(\dfrac{\pi\xi}{2L_0}\right)\sin^3\left(\dfrac{\pi\eta}{L_0}\right), & \text{for } \mathbf{n} \cdot \mathbf{e}_y = 0 \\ 0, & \text{for } \mathbf{n} \cdot \mathbf{e}_y \neq 0 \end{cases}, \tag{34}
$$

$J_0 = 1$ A/m, and $L_0 = 1$ m. $\xi \in [0, 4]$ m is perpendicular to $\eta = y \in [0, 1]$ m, wrapping around the surfaces for which $\mathbf{n} \cdot \mathbf{e}_y = 0$, beginning at $x = 0$ m and $z = 1$ m for the cube and $x = z = \sqrt{2}/2$ m for the rhombic prism, as depicted in Figure 2, which shows the nets of these domains. Equation (34) is of class C^2. Figures 3 and 4 show plots of (34).

4.1.1. Solution-Discretization Error

To isolate and measure the solution-discretization error, we proceed with the assessment described in Section 3.1. With G_{MS} (21), we are able to compute the integral in (20) analytically, which yields a finite-degree polynomial integrand for $b(\mathbf{H}^\mathcal{I}, \boldsymbol{\Lambda}_i)$ (12). Because the integrands of $a(\mathbf{J}_h, \boldsymbol{\Lambda}_i)$ and $b(\mathbf{H}^\mathcal{I}, \boldsymbol{\Lambda}_i)$ in (12) are finite-degree polynomials, they can be integrated exactly with the appropriate amount of polynomial quadrature points. For G_{MS}, we consider $d = \{1, 2, 3\}$. We account for potential disparities in the magnitudes of the contributions to (9) from the first (Term 1) and second (Term 2) terms by considering them together and separately.

Figure 5 shows the L^∞ norm of the discretization error $\|\mathbf{e_J}\|_\infty$ (23) arising from only the solution-discretization error for both geometries and for Term 1 with and without Term 2. The convergence rates are all $\mathcal{O}(h^2)$ as expected.

Figure 6 shows norms of the discretization error with only Term 2. As stated in Section 3.1.1, the arising matrix is singular. Therefore, to compute a unique solution we minimize $\|\mathbf{e_J}\|_\infty$ (25) (6a and 6c) and $\|\mathbf{e_J}\|_2$ (24) (6b and 6d), and we measure $\|\mathbf{e_J}\|_\infty$ (6a and 6b) and $\|\mathbf{e_J}\|_2$ (6c and 6d). To minimize $\|\mathbf{e_J}\|_\infty$, we use HiGHS [54]. Table 1 shows the convergence rates computed between adjacent mesh pairs for these four combinations for the cube (C) and rhombic prism (RP). Unlike the other three combinations, measuring $\|\mathbf{e_J}\|_\infty$ and minimizing $\|\mathbf{e_J}\|_2$ yields a convergence rate that is not as clearly $\mathcal{O}(h^2)$ for the meshes considered. On the other hand, by minimizing $\|\mathbf{e_J}\|_\infty$, the convergence rate is clearly $\mathcal{O}(h^2)$ without requiring finer meshes.

8
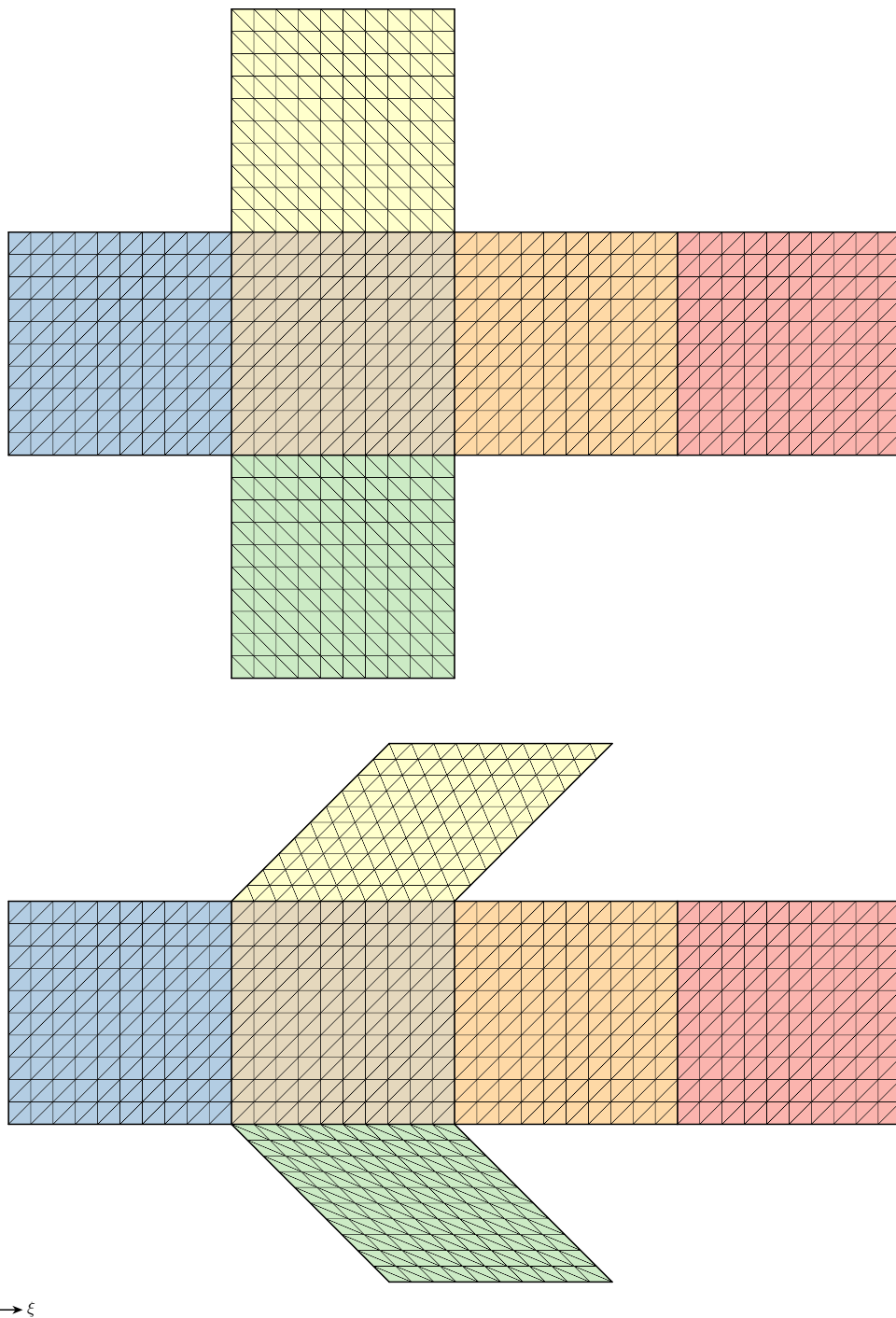
Figure 2: Meshes for the cube (top) and the rhombic prism (bottom), with $n_t = 1200$.
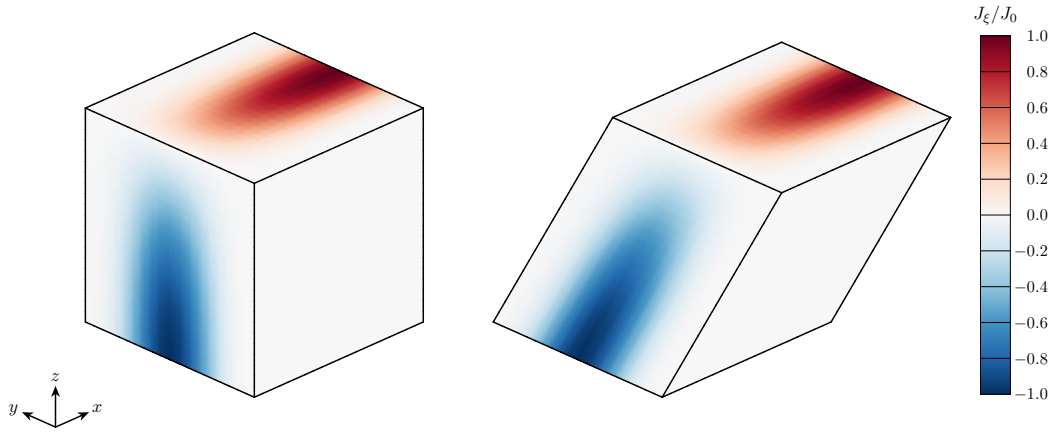
Figure 3: Manufactured surface current density $\mathbf{J}_{\mathrm{MS}}$ for the cube (left) and the rhombic prism (right).
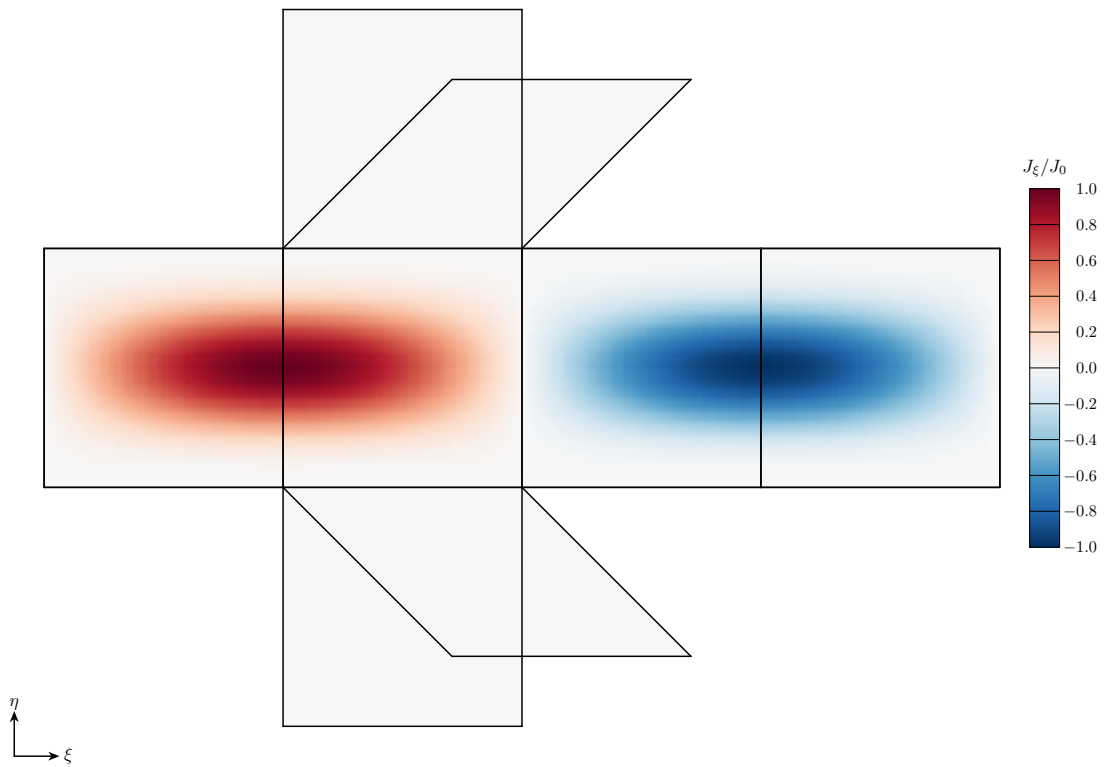
Figure 4: Manufactured surface current density $\mathbf{J}_{\mathrm{MS}}$ for cube and rhombic prism.
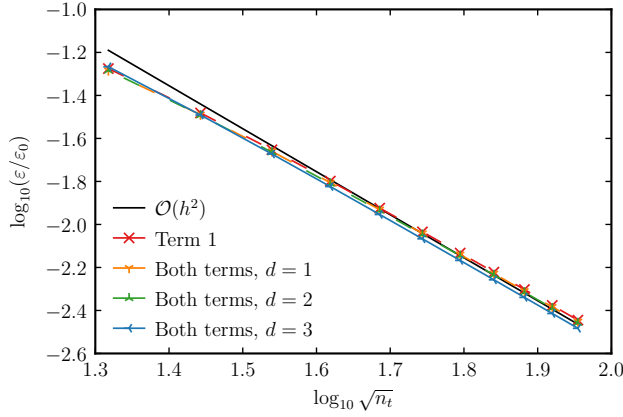
(a) Cube
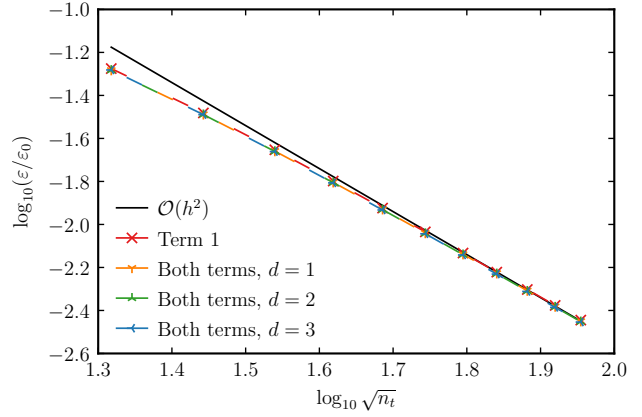
(b) Rhombic Prism

Figure 5: Solution-discretization error: $\varepsilon = \|\mathbf{e_J}\|_\infty$.

(a) $\varepsilon = \|\mathbf{e_J}\|_\infty$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$
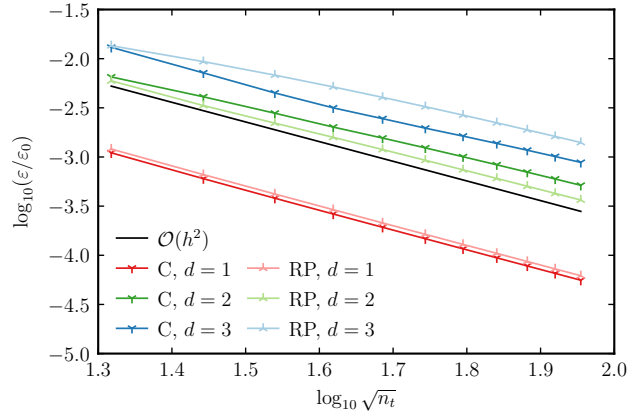
(b) $\varepsilon = \|\mathbf{e_J}\|_\infty$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$
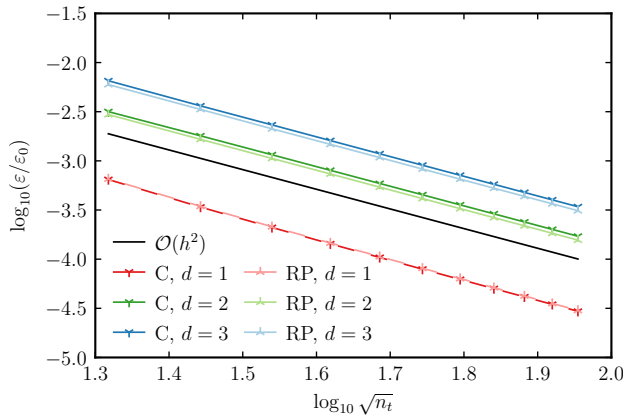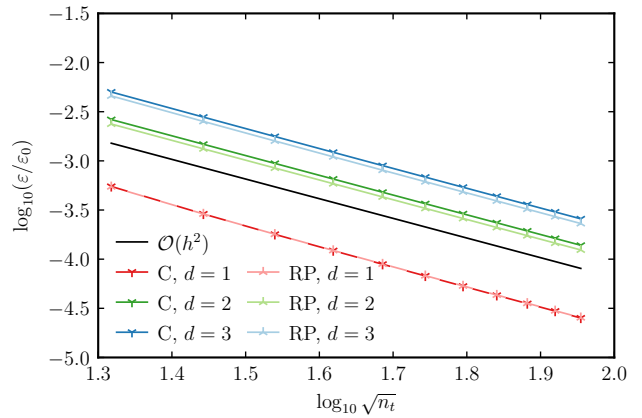
(c) $\varepsilon = \|\mathbf{e_J}\|_2$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$

(d) $\varepsilon = \|\mathbf{e_J}\|_2$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$

Figure 6: Solution-discretization error: $\varepsilon = \|\mathbf{e_J}\|$ for different metric and minimization norms.

11

| | $\varepsilon = \|\mathbf{e_J}\|_\infty$ | | | | $\varepsilon = \|\mathbf{e_J}\|_2$ | | | |
| | $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$ | | $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$ | | $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$ | | $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$ | |
Mesh	C	RP	C	RP	C	RP	C	RP
1–2	2.0800	2.0653	2.0811	1.2935	2.0447	2.0229	2.0454	2.0763
2–3	2.0141	2.0529	2.1055	1.4193	1.9948	2.0323	2.0359	2.0499
3–4	2.0303	2.0193	1.9159	1.5150	2.0141	1.9999	2.0283	2.0372
4–5	2.0196	2.0163	1.6421	1.5847	2.0064	2.0093	2.0229	2.0297
5–6	2.0061	2.0242	1.6677	1.6372	2.0060	2.0102	2.0190	2.0246
6–7	2.0133	2.0158	1.5800	1.6779	2.0057	2.0097	2.0162	2.0211
7–8	2.0113	2.0167	1.6282	1.7104	2.0057	2.0122	2.0140	2.0184
8–9	2.0037	2.0122	1.6664	1.7369	1.9965	2.0076	2.0123	2.0163
9–10	2.0086	2.0117	1.6974	1.7589	2.0039	2.0067	2.0110	2.0146
10–11	2.0053	2.0118	1.7231	1.7776	2.0039	2.0094	2.0099	2.0133

Table 1: Solution-discretization error: convergence rates computed between adjacent mesh pairs for $d = 3$ in Figure 6.

(a) $\varepsilon = \|\mathbf{e_J}\|_\infty$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$

(b) $\varepsilon = \|\mathbf{e_J}\|_\infty$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$

(c) $\varepsilon = \|\mathbf{e_J}\|_2$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_\infty$

(d) $\varepsilon = \|\mathbf{e_J}\|_2$, $\min_{\mathbf{J}^h} \|\mathbf{e_J}\|_2$
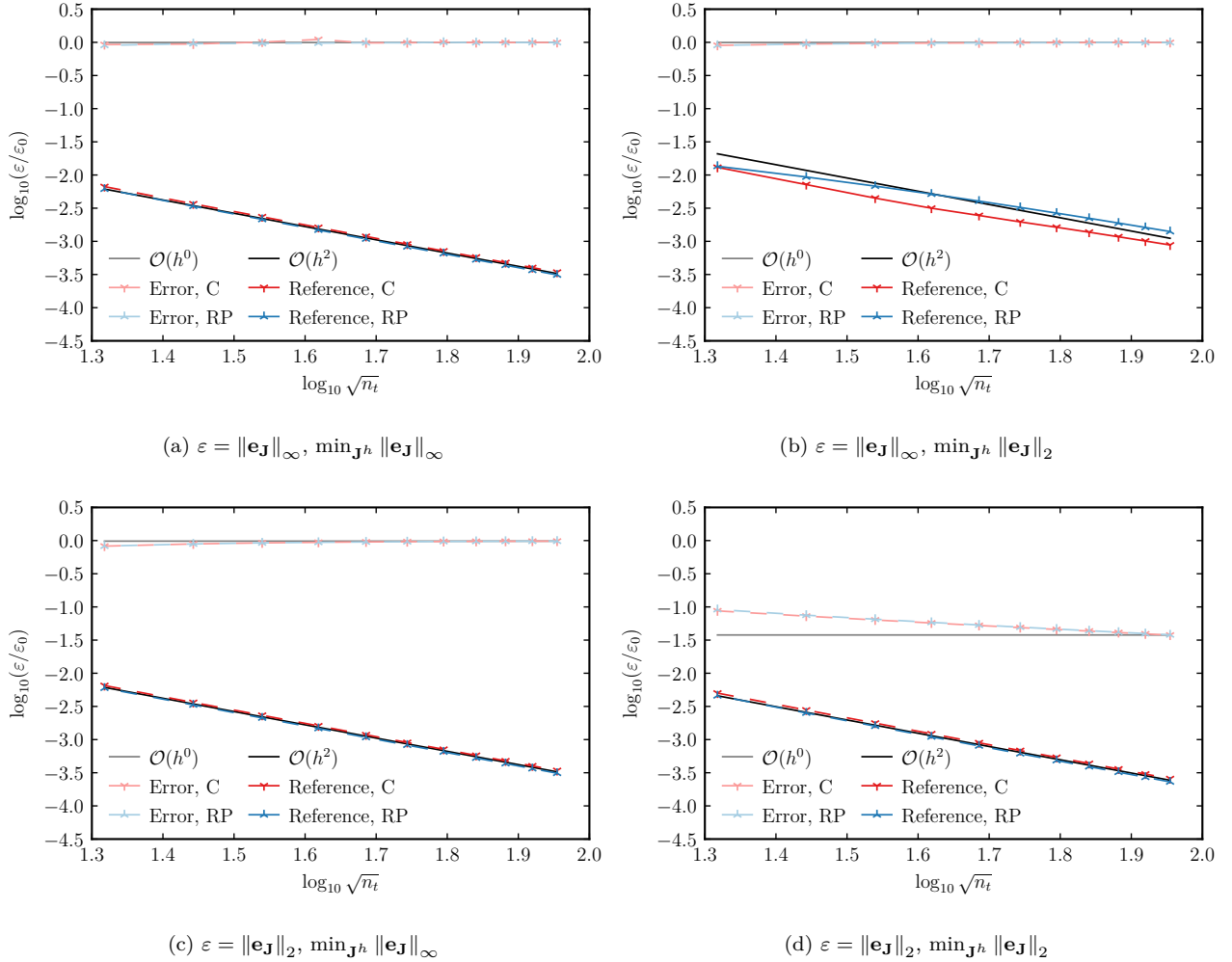
Figure 7: Solution-discretization error: $\varepsilon = \|\mathbf{e_J}\|$ for different metric and minimization norms in the presence of coding error for $d = 3$.

(a) Maximum polynomial degree: 4   (b) Maximum polynomial degree: 3
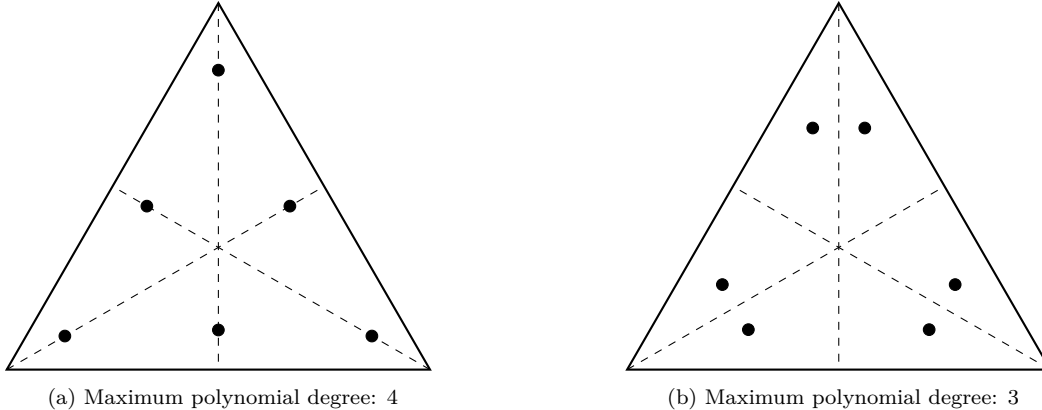
Figure 8: 6-point quadrature rules.

Number of points	1	3	4	6	7	12	13
Maximum integrand degree	1	2	3	4	5	6	7
Convergence rate	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$	$\mathcal{O}(h^8)$

Table 2: Polynomial triangle quadrature properties.

Figure 7 shows the four metric and minimization combinations shown in Figure 6 for Term 2 and $d = 3$ in the presence of a coding error in which the magnitudes of the diagonal elements of the matrix are increased by 1%. None of the cases with the coding errors have convergence rates that are $\mathcal{O}(h^2)$; therefore, each of these combinations detects the coding error.

4.1.2. Numerical-Integration Error

To isolate and measure the numerical-integration error, we continue by performing the assessments described in Section 3.2. The numerical integration is performed using polynomial quadrature rules for triangles. For multiple quadrature point amounts, Table 2 lists the maximum polynomial degree of the integrand the points can integrate exactly [55, 56], as well as the convergence rates of the errors for inexact integrations of nonsingular integrands. These properties correspond to the optimal point locations and weights. For nonsingular integrands, the slowest expected quadrature convergence rate is $\mathcal{O}(h^2)$. Figure 8a shows the optimal 6-point quadrature rule, which can exactly integrate polynomials up to degree 4, whereas Figure 8b shows a suboptimal 6-point quadrature rule [57], which can exactly integrate polynomials up to degree 3.

Figure 9 shows the numerical-integration error $e_a(\mathbf{J}_{h_{\mathrm{MS}}})$ (28) when the solution-discretization error is canceled for $d = \{1, 2, 3\}$ in (21), both geometries, and both terms. In the legend entries, the first number is the amount of quadrature points used to compute the integral over S, whereas the second is the amount used to compute the integral over S'. Each of the solutions converges at the expected rate listed in Table 2.

Figure 10 shows the numerical-integration error $e_b(\mathbf{J}_{h_{\mathrm{MS}}})$ (29) when the solution-discretization error is canceled. In the legend entries, the number is the amount of quadrature points used to compute the integral. Each of the solutions converges at the expected rate. For the finest meshes considered, the round-off error arising from the double-precision calculations exceeds the numerical-integration error.

To relate the $e_a(\mathbf{J}_{h_{\mathrm{MS}}})$ (28) and $e_b(\mathbf{J}_{h_{\mathrm{MS}}})$ (29) to the discretization error $\mathbf{e_J}$ (23), we can solve the optimization problems (30) and (31). These discretization errors are respectively shown in Figures 11 and 12 and converge at the same rates as those in Figures 9 and 10, until the round-off error exceeds the discretization error.

Figures 13 and 14 show the numerical-integration errors $e_a(\mathbf{J}_{\mathrm{MS}})$ (32) and $e_b(\mathbf{J}_{\mathrm{MS}})$ (33) when the solution-discretization error is eliminated for $d = \{1, 2, 3\}$ in (21), both geometries, and both terms. Each of the solutions converges at the expected rate listed in Table 2, except for $d = 1$ in Figures 13a and 13b, where the convergence rates for the 1×1 quadrature combination are $\mathcal{O}(h^4)$ instead of $\mathcal{O}(h^2)$. Once more, for the finer meshes, the round-off error exceeds the discretization error.
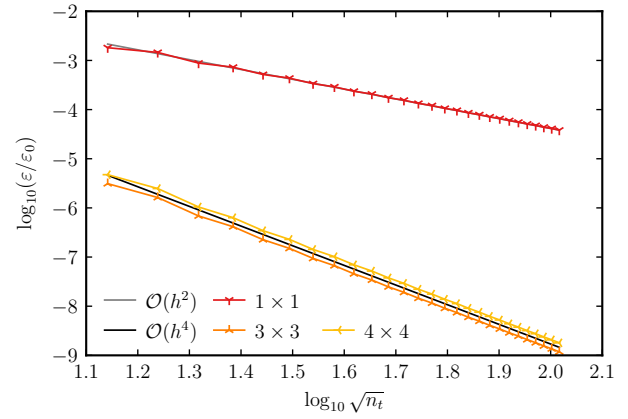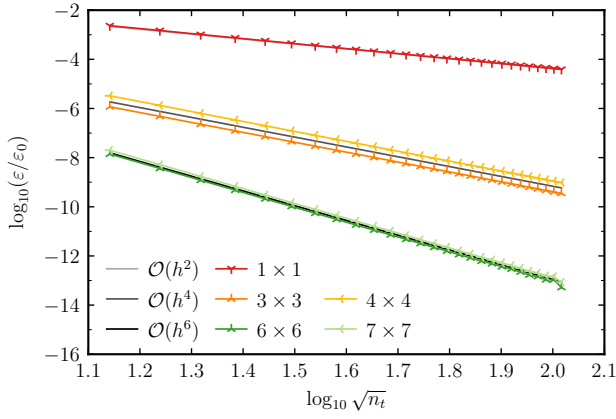
13
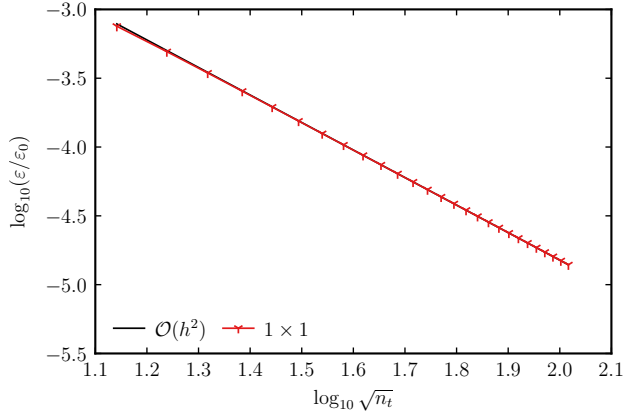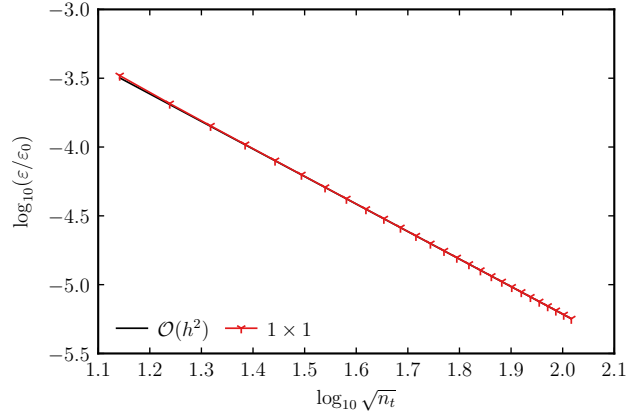
(a) Cube, $d = 1$
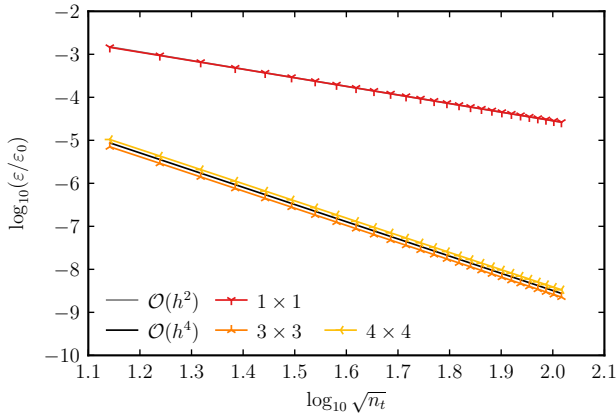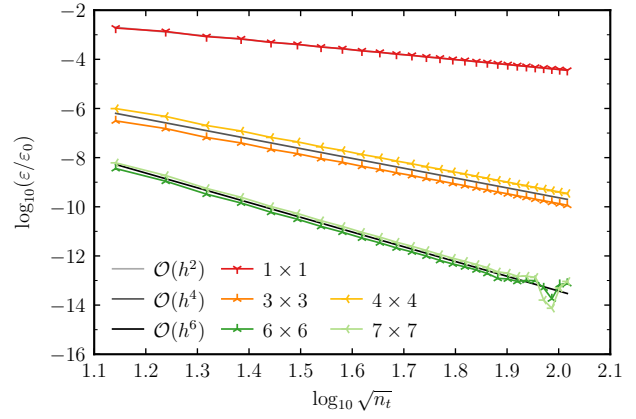
(b) Rhombic Prism, $d = 1$
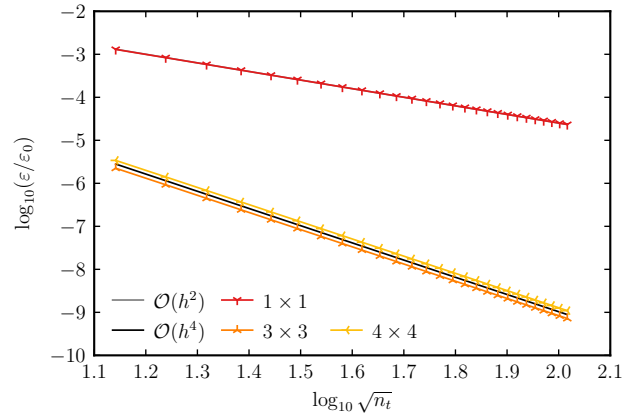
(c) Cube, $d = 2$
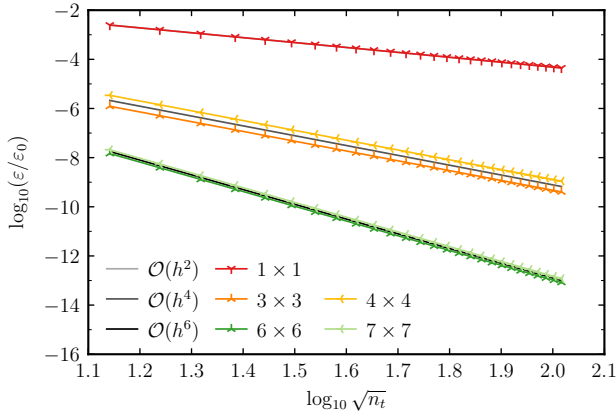
(d) Rhombic Prism, $d = 2$
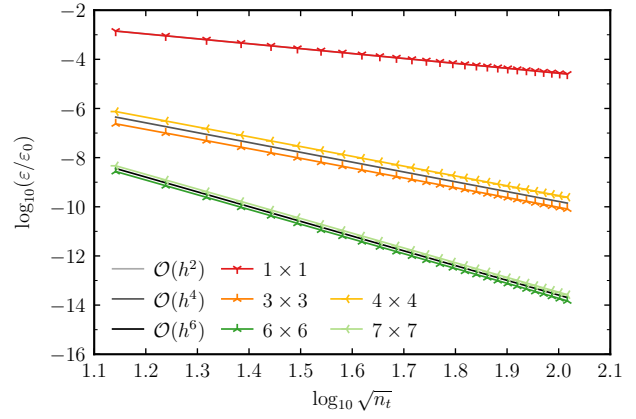
(e) Cube, $d = 3$
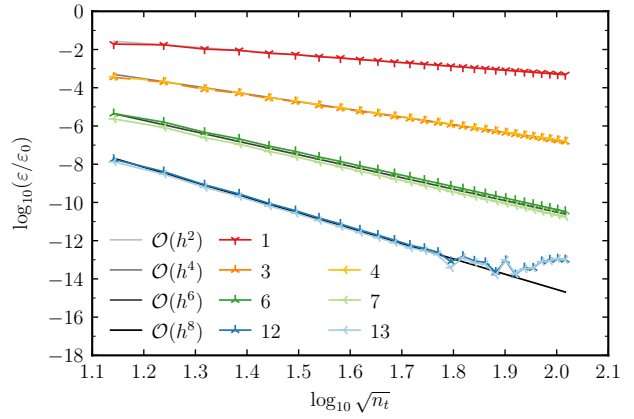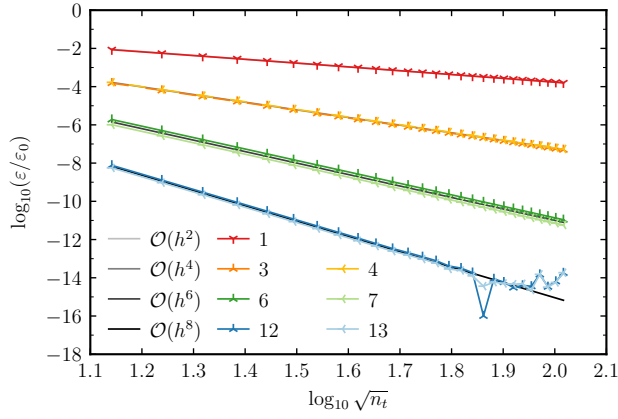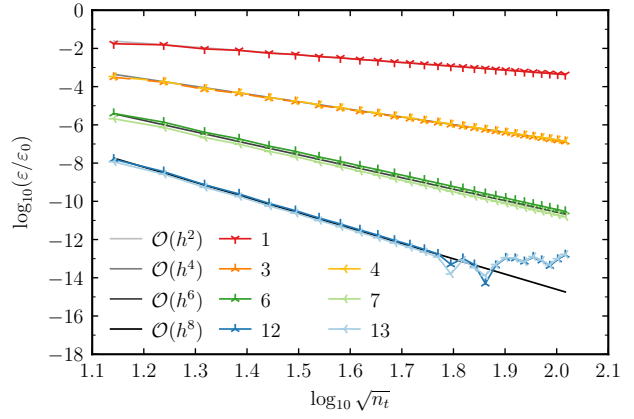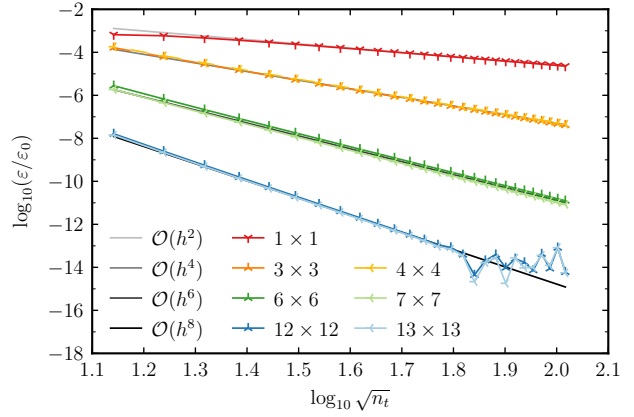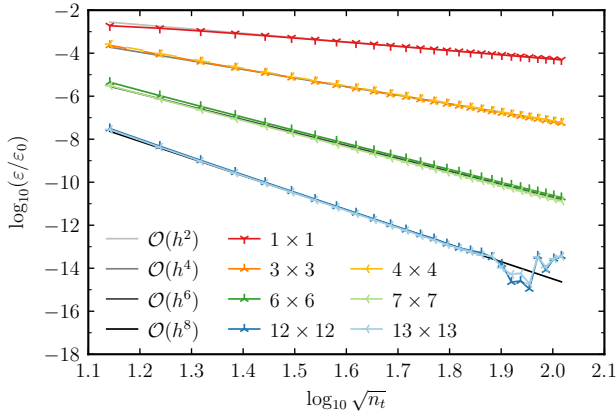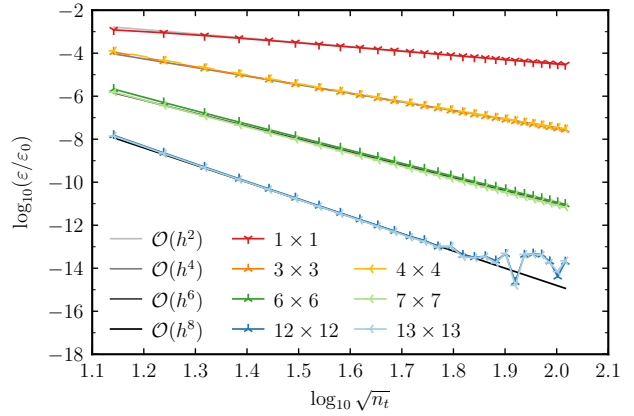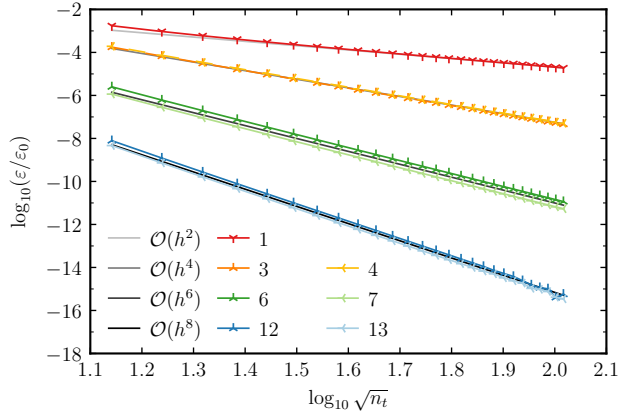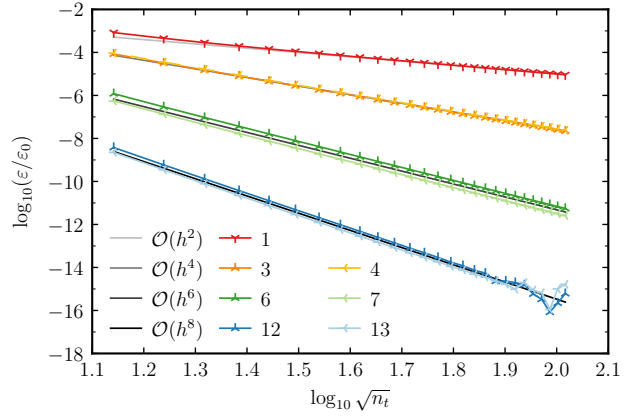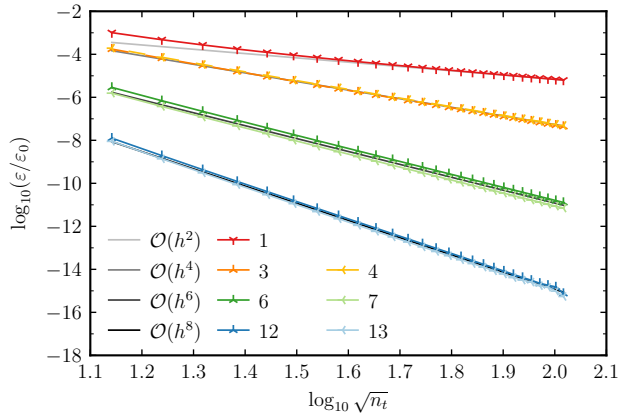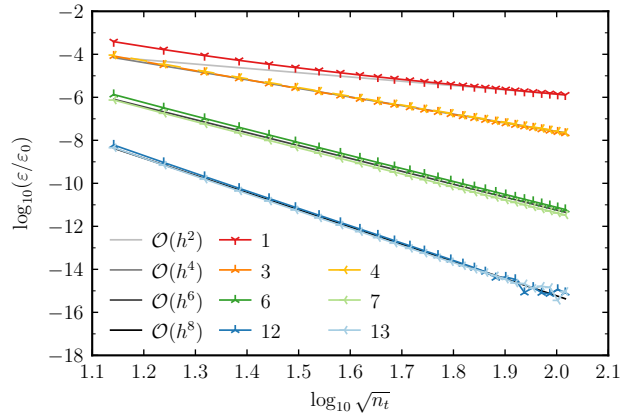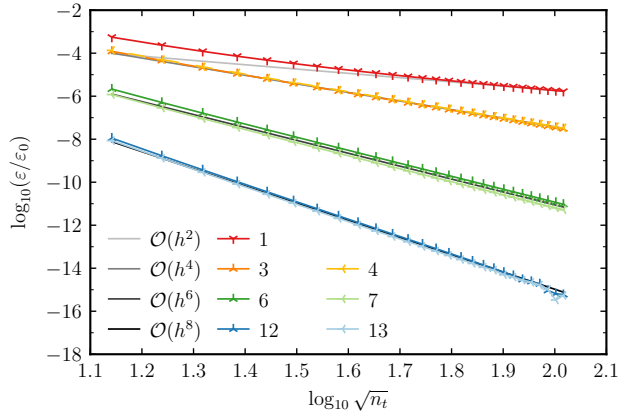
(f) Rhombic Prism, $d = 3$

Figure 9: Numerical-integration error: $\varepsilon = |e_a(\mathbf{J}_{h_{\mathrm{MS}}})|$ (28) for different amounts of quadrature points.
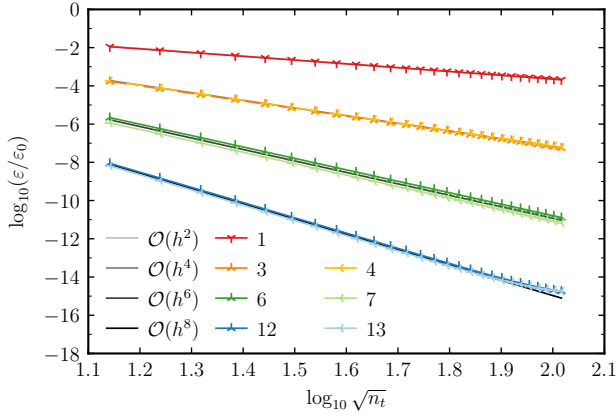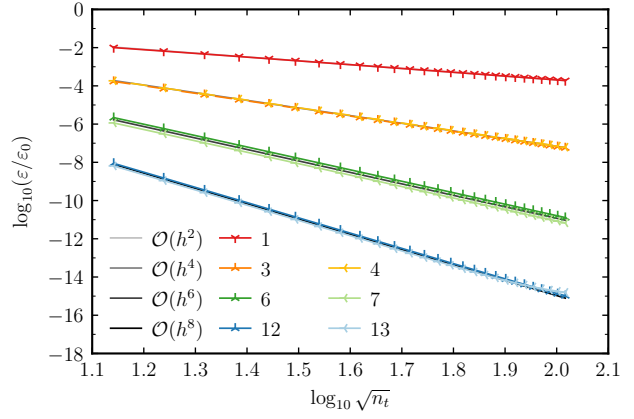
(a) Cube, $d = 1$
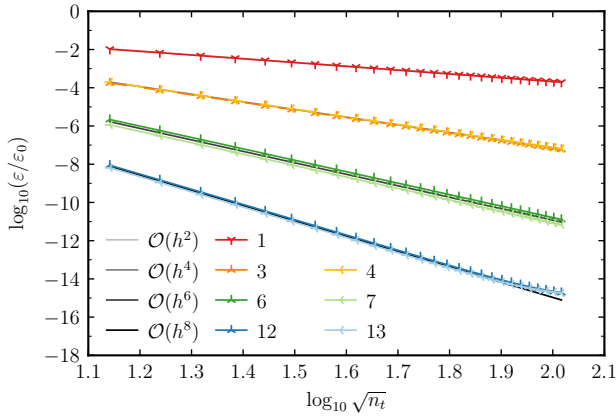
(b) Rhombic Prism, $d = 1$
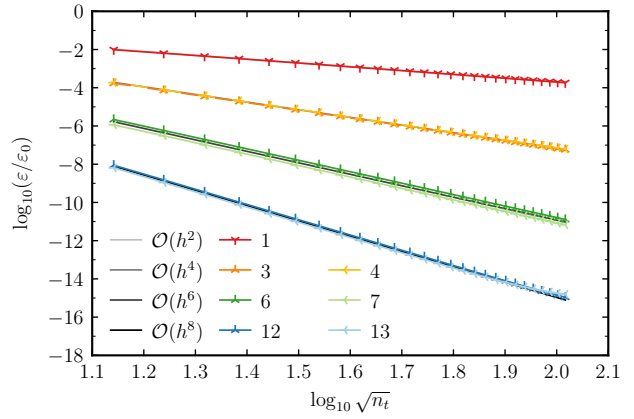
(c) Cube, $d = 2$
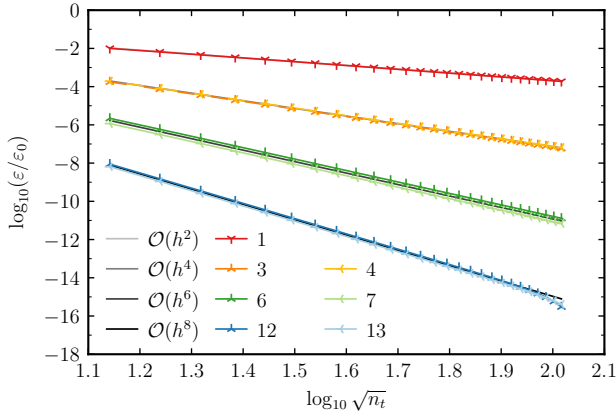
(d) Rhombic Prism, $d = 2$
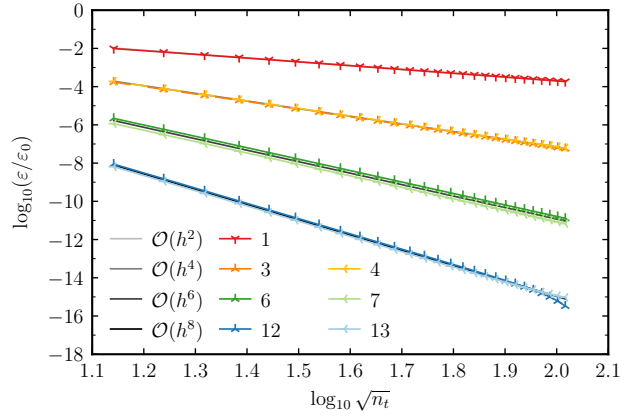
(e) Cube, $d = 3$

(f) Rhombic Prism, $d = 3$

Figure 10: Numerical-integration error: $\varepsilon = |e_b(\mathbf{J}_{h_{\mathrm{MS}}})|$ (29) for different amounts of quadrature points.
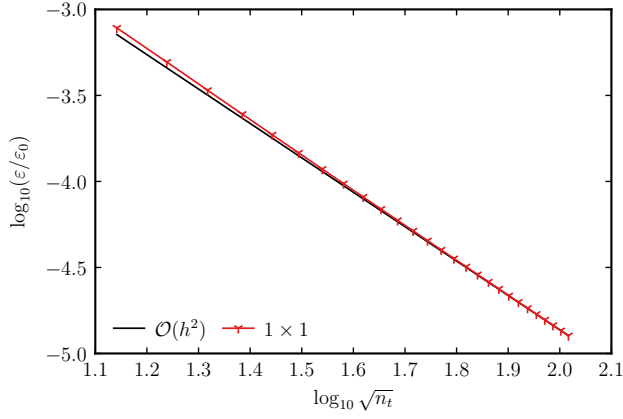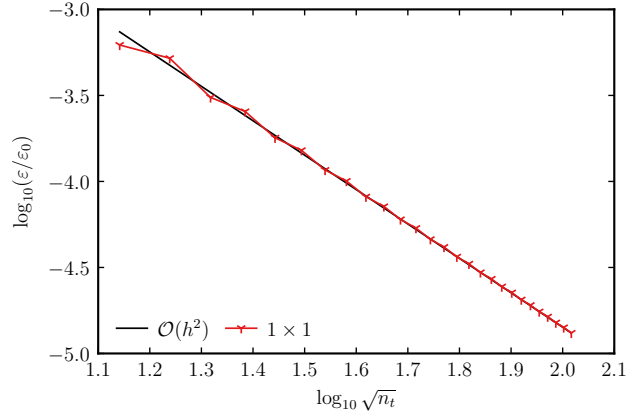
(a) Cube, $d = 1$
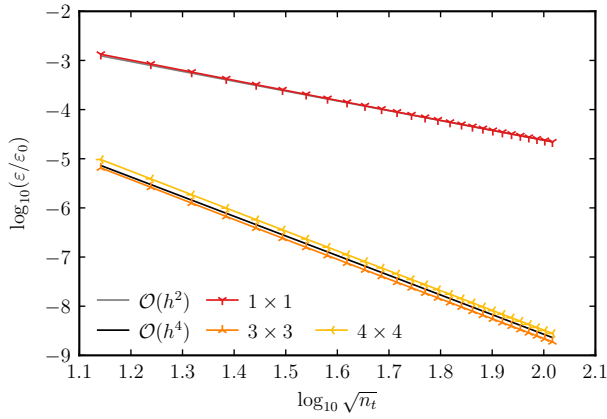
(b) Rhombic Prism, $d = 1$
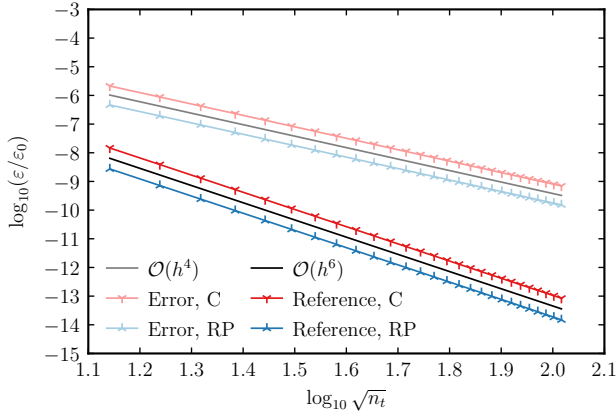
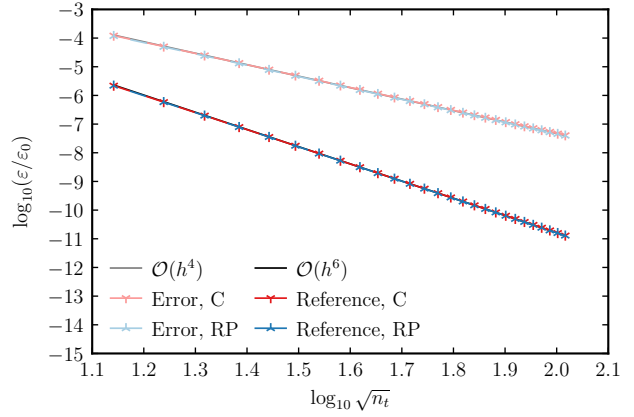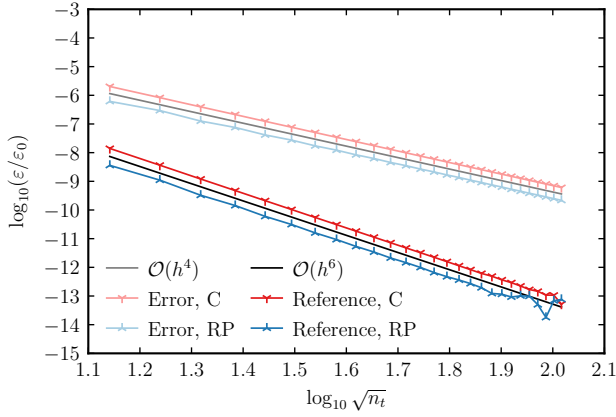(c) Cube, $d = 2$

(d) Rhombic Prism, $d = 2$

(e) Cube, $d = 3$

(f) Rhombic Prism, $d = 3$

Figure 11: Numerical-integration error: $\varepsilon = \|\mathbf{e_J}\|_\infty$ from (30) for different amounts of quadrature points.
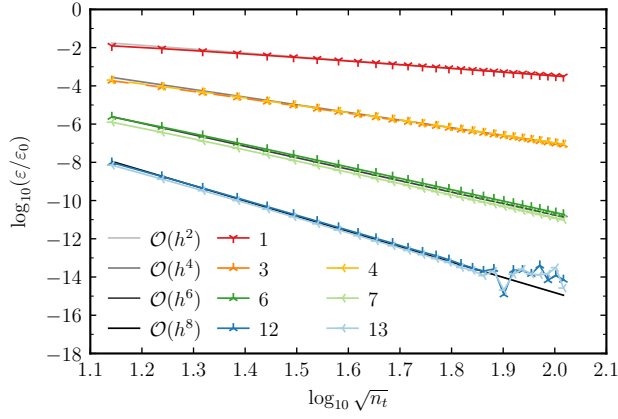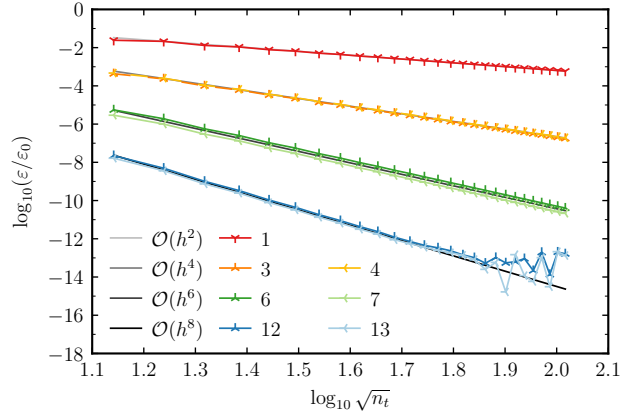
(a) Cube, $d = 1$
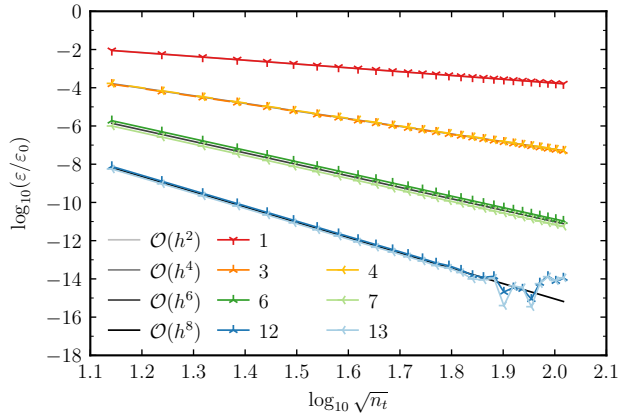
(b) Rhombic Prism, $d = 1$

(c) Cube, $d = 2$

(d) Rhombic Prism, $d = 2$

(e) Cube, $d = 3$

(f) Rhombic Prism, $d = 3$

Figure 12: Numerical-integration error: $\varepsilon = \|\mathbf{e_J}\|_\infty$ from (31) for different amounts of quadrature points.
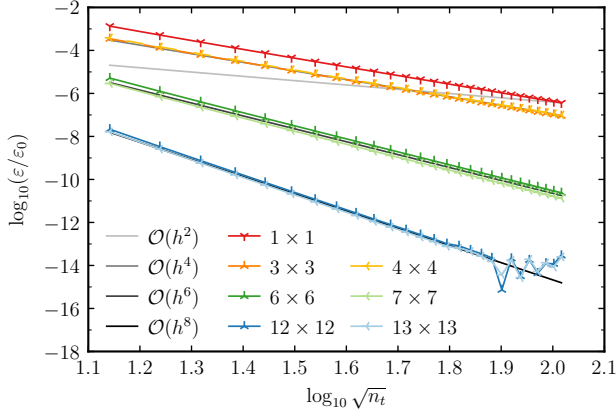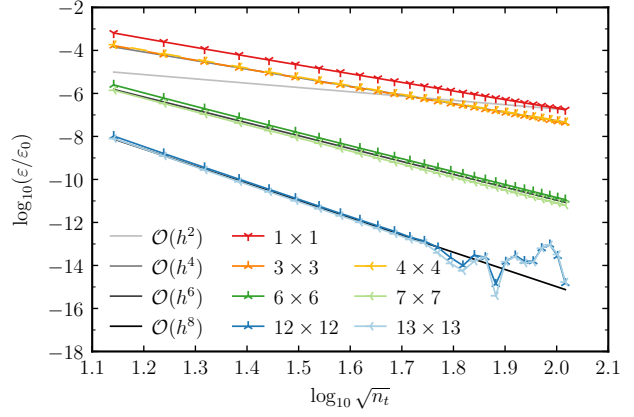
(a) Cube, $d = 1$
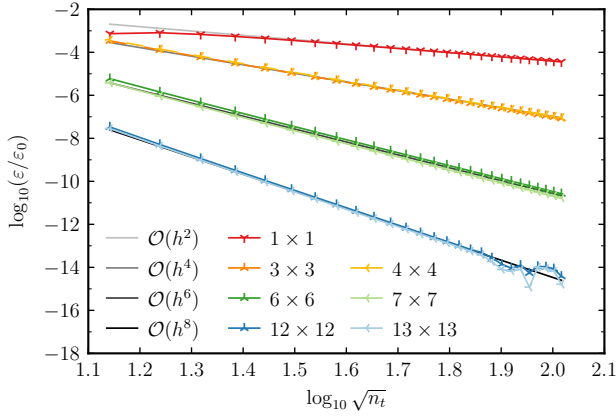
(b) Rhombic Prism, $d = 1$

(c) Cube, $d = 2$

(d) Rhombic Prism, $d = 2$

(e) Cube, $d = 3$

(f) Rhombic Prism, $d = 3$

Figure 13: Numerical-integration error: $\varepsilon = |e_a(\mathbf{J}_{\mathrm{MS}})|$ (32) for different amounts of quadrature points.
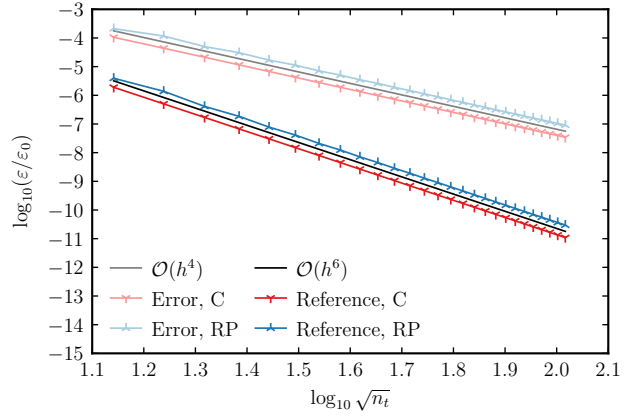
(a) Cube, $d = 1$

(b) Rhombic Prism, $d = 1$

(c) Cube, $d = 2$

(d) Rhombic Prism, $d = 2$

(e) Cube, $d = 3$
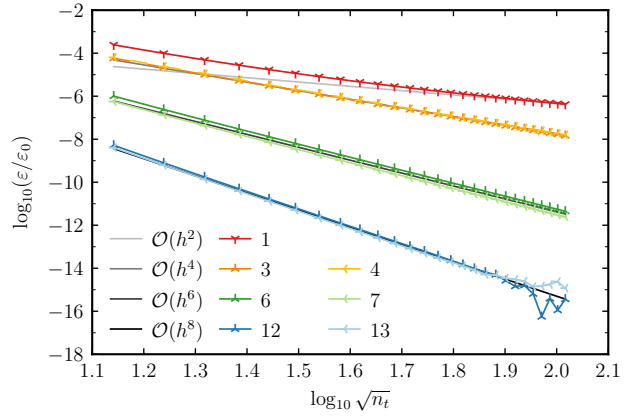
(f) Rhombic Prism, $d = 3$

Figure 14: Numerical-integration error: $\varepsilon = |e_b(\mathbf{J}_{\mathrm{MS}})|$ (33) for different amounts of quadrature points.

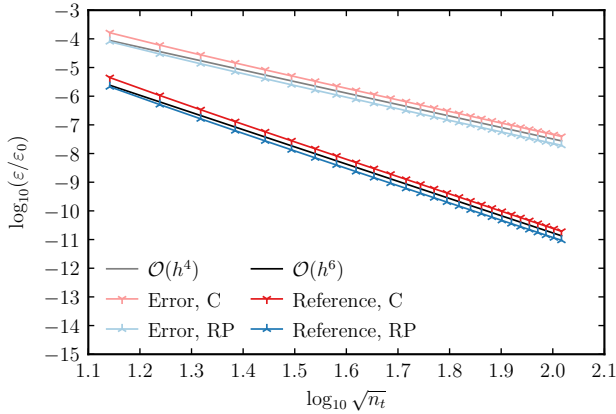(a) $\varepsilon = \left| e_a \left(\mathbf{J}_{h_{\mathrm{MS}}} \right) \right|$

(b) $\varepsilon = \left| e_b \left(\mathbf{J}_{h_{\mathrm{MS}}} \right) \right|$

(c) $\varepsilon = \|\mathbf{e_J}\|_\infty,\ a^q\left(\mathbf{J}_h, \mathbf{J}_{h_{\mathrm{MS}}} \right) = a\left(\mathbf{J}_{h_{\mathrm{MS}}}, \mathbf{J}_{h_{\mathrm{MS}}} \right)$

(d) $\varepsilon = \|\mathbf{e_J}\|_\infty,\ b^q\left(\mathbf{H}^{\mathcal{I}}_{\mathrm{MS}}, \mathbf{J}_h \right) = b\left(\mathbf{H}^{\mathcal{I}}_{\mathrm{MS}}, \mathbf{J}_{h_{\mathrm{MS}}} \right)$

(e) $\varepsilon = |e_a(\mathbf{J}_{\mathrm{MS}})|$
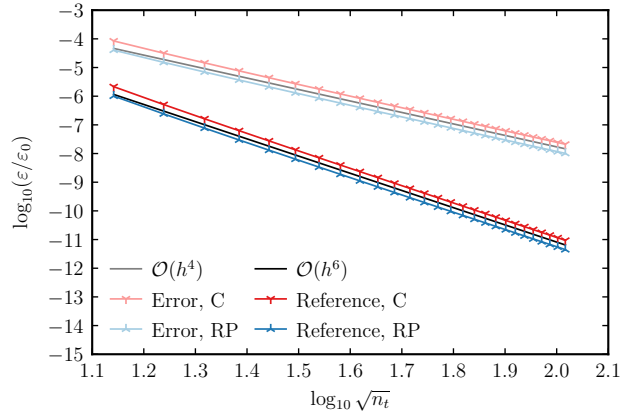
(f) $\varepsilon = |e_b(\mathbf{J}_{\mathrm{MS}})|$

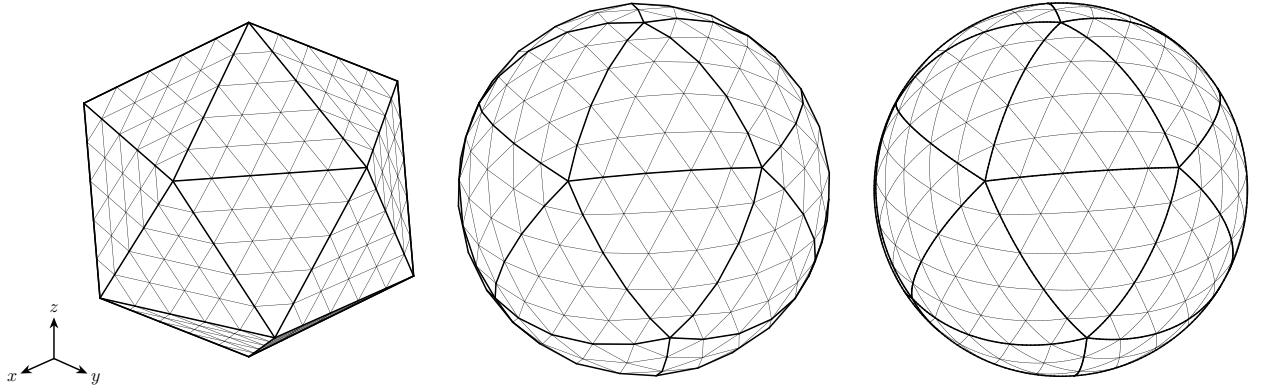Figure 15: Numerical-integration error: ε in the presence of a coding error for $d = 3$.

Figure 16: Meshes for the regular icosahedron (left), the faceted sphere (center), and the smooth sphere (right), with $n_t = 500$.
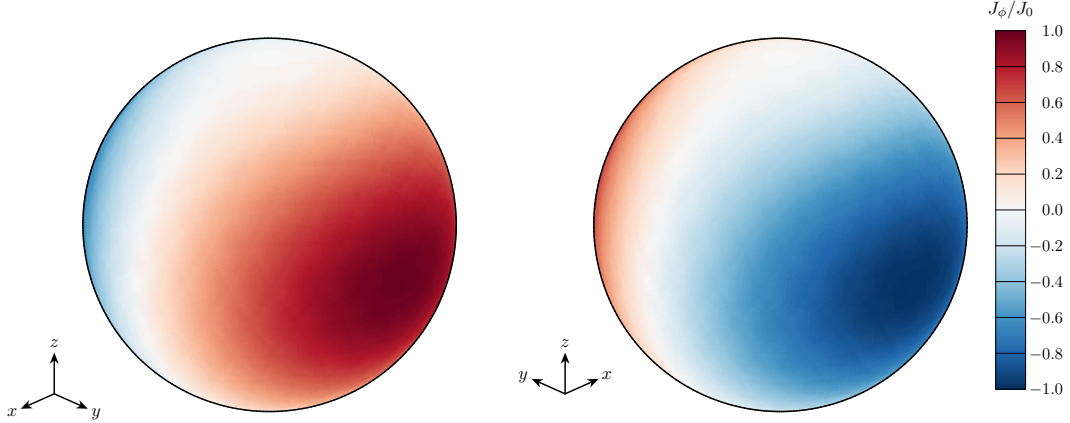

Figure 17: Manufactured surface current density $\mathbf{J}_{\mathrm{MS}}$ for the sphere.

To test the ability to detect a coding error, we replace the optimal 6-point quadrature rule that can exactly integrate polynomials up to degree 4 (Figure 8a) with a suboptimal rule (Figure 8b) that can integrate polynomials up to degree 3. With this coding error, for $d = 3$, Figures 15a–15f respectively show the approaches presented in Figures 9–14. The convergence rates are $\mathcal{O}(h^4)$ for the cases with the coding errors, compared to the expected $\mathcal{O}(h^6)$ rates without. Therefore, each of these methods detects the coding error.

4.2. With Curvature

To account for curvature in the domain, we consider a spherical domain with a radius of 1 m. The mesh for the sphere is obtained by generating a regular icosahedron inscribed in the sphere, subdividing the twenty triangular faces, and projecting the nodes radially onto the sphere. To account for the curvature of the sphere, instead of discretizing it with planar triangles, we can discretize it with spherical triangles. These three phases – the icosahedron, the faceted sphere, and the smooth sphere – are depicted in Figure 16 with the total number of triangles $n_t = 500$. The icosahedron is oriented in a manner such it has vertices at $z = \pm 1$ m.

We manufacture the surface current density $\mathbf{J}_{\mathrm{MS}}(\mathbf{x}) = J_\phi(\theta, \phi)\mathbf{e}_\phi$, where

$$J_\phi(\theta, \phi) = J_0 \sin^2 \theta \sin \phi, \tag{35}$$

21

and $x = r\sin\theta\cos\phi$, $y = r\sin\theta\sin\phi$, $z = r\cos\theta$, and $r = \sqrt{x^2 + y^2 + z^2}$. Equation (35) is of class C^∞. Figure 17 shows plots of (35).

4.2.1. Domain-Discretization Error

To address the domain-discretization error, we perform the assessments described in Section 3.3. For $d = \{1, 2, 3\}$ in (21), Figure 18 shows $e_a(\mathbf{J}_{\mathrm{MS}})$ (32), as described in Section 3.3.1. For the left column (18a, 18c, 18e), $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$ in (32) is computed on meshes composed of spherical triangles. For the right column (18b, 18d, 18f), $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$ is approximated by $a_h^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$ and computed on meshes composed of planar triangles. By accounting for the curvature, $e_a(\mathbf{J}_{\mathrm{MS}})$ converges at the expected rates listed in Table 2 (until the round-off error exceeds the domain-discretization error) for $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, whereas its rate is limited to $\mathcal{O}(h^2)$ for $a_h^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$.

Figure 19 shows $e_b(\mathbf{J}_{\mathrm{MS}})$ (33) for $d = \{1, 2, 3\}$. For the left column (19a, 19c, 19e), $b^q(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}})$ in (33) is computed on meshes composed of spherical triangles. For the right column (19b, 19d, 19f), $b^q(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}})$ is approximated by $b_h^q(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}})$ and computed on meshes composed of planar triangles. By accounting for the curvature, $e_b(\mathbf{J}_{\mathrm{MS}})$ converges at the expected rates listed in Table 2 (until the round-off error exceeds the domain-discretization error) for $b^q(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}})$, whereas its rate is limited to $\mathcal{O}(h^2)$ for $b_h^q(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}})$.

Finally, we can neglect curvature as described in Section 3.3.2. Figure 20 shows $e_a(\mathbf{J}_{h_{\mathrm{MS}}})$ (28) in the left column (20a, 20c, 20e), and $\mathbf{e_J}$ (23) arising from (30) in the right column (20b, 20d, 20f). This assessment isolates the numerical-integration error, which converges at the expected rates listed in Table 2.

5. Conclusions

In this paper, we presented code-verification approaches for the MoM implementation of the magnetic-field integral equation that account for the three sources of numerical error: domain discretization, solution discretization, and numerical integration.

We isolated and measured the solution-discretization error by integrating exactly over the domain. To integrate exactly, we manufactured the Green's function, and presented optimization approaches to select a unique solution when the manufactured Green's function makes the matrix practically singular.

We isolated and measured the numerical-integration error by canceling and eliminating the solution-discretization error. Canceling the solution-discretization error uses the basis functions, whereas eliminating does not.
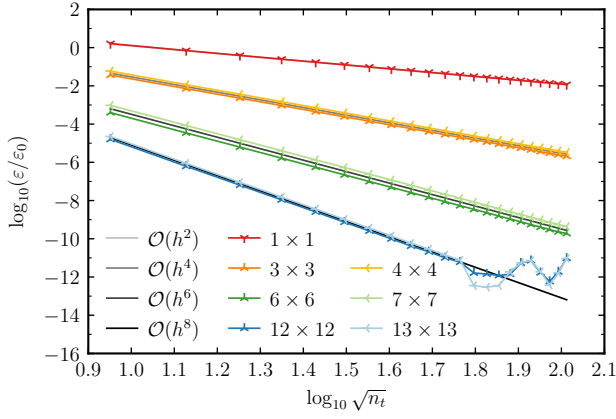
To address the domain-discretization error, we accounted for curvature and we neglected the curvature. We accounted for curvature by modifying the integrand to effectively integrate over curved triangular elements, such that we could measure the numerical-integration error. We neglected the curvature by using planar triangular elements to isolate and measure the numerical-integration error.

We considered cases with and without coding errors to demonstrate the efficacy of these approaches.
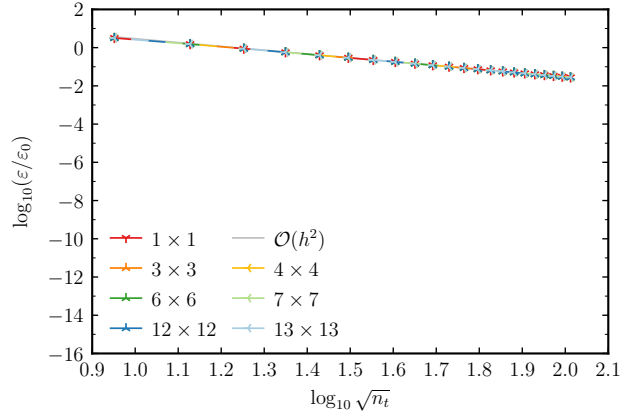
To account for the actual Green's function, this work can be complemented with unit tests that assess the evaluation of the (nearly) singular integrals that arise from the actual Green's function.
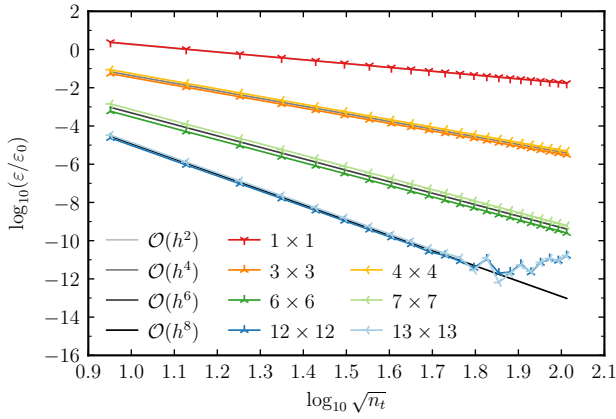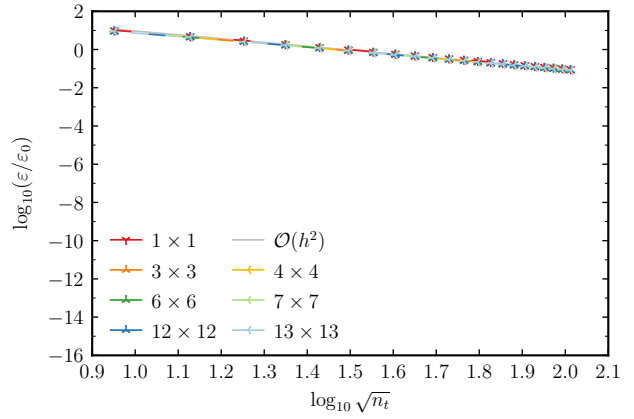
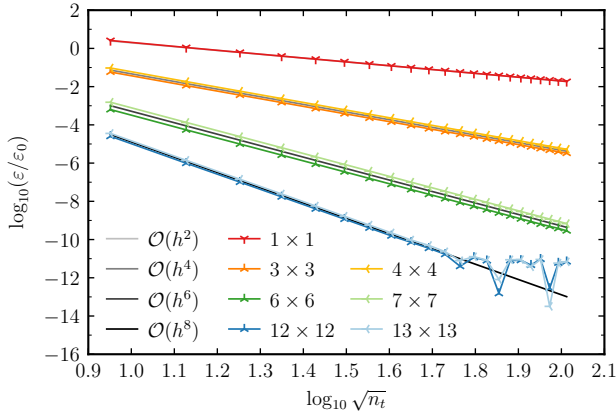(a) $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 1$

(b) $a^q_h(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 1$
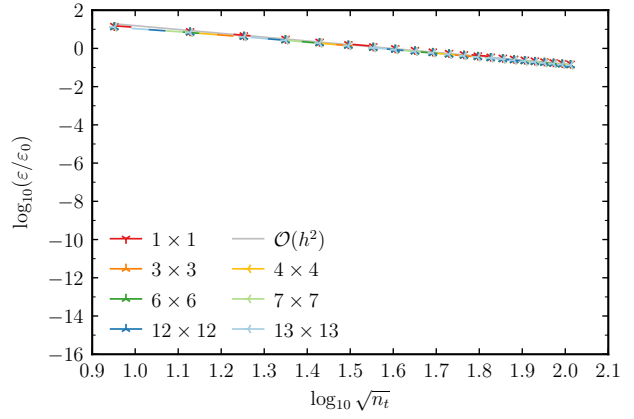
(c) $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 2$

(d) $a^q_h(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 2$
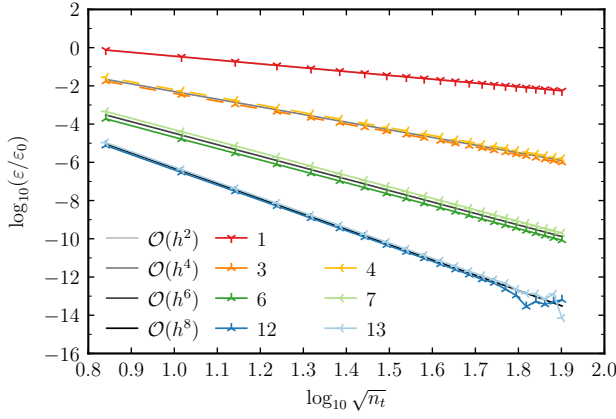
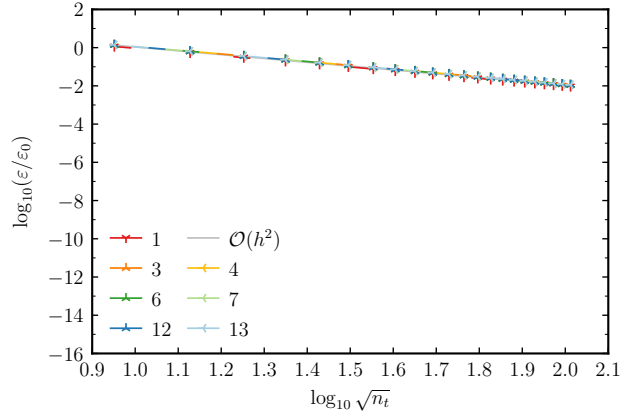(e) $a^q(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 3$

(f) $a^q_h(\mathbf{J}_{\mathrm{MS}}, \mathbf{J}_{\mathrm{MS}})$, $d = 3$

Figure 18: Domain-Discretization Error: $\varepsilon = |e_a(\mathbf{J}_{\mathrm{MS}})|$ for different amounts of quadrature points.

(a) $b^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=1$

(b) $b_h^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=1$

(c) $b^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=2$

(d) $b_h^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=2$
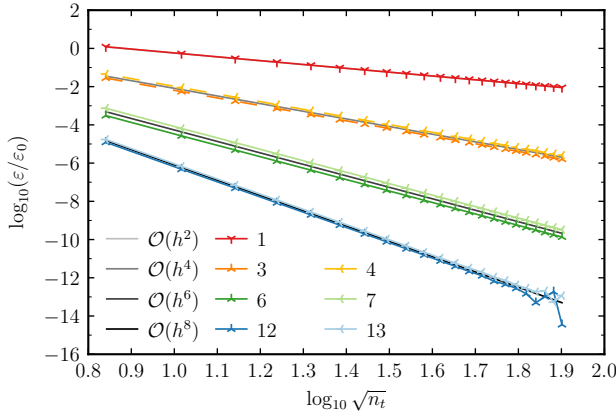
(e) $b^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=3$
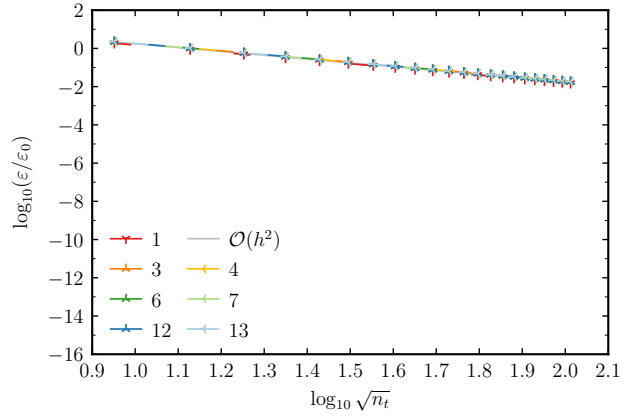
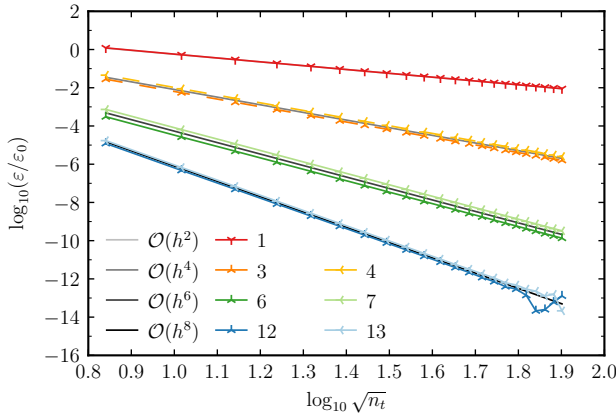(f) $b_h^q\left(\mathbf{H}_{\mathrm{MS}}^{\mathcal{I}}, \mathbf{J}_{\mathrm{MS}}\right), d=3$

Figure 19: Domain-Discretization Error: $\varepsilon = |e_b(\mathbf{J}_{\mathrm{MS}})|$ for different amounts of quadrature points.
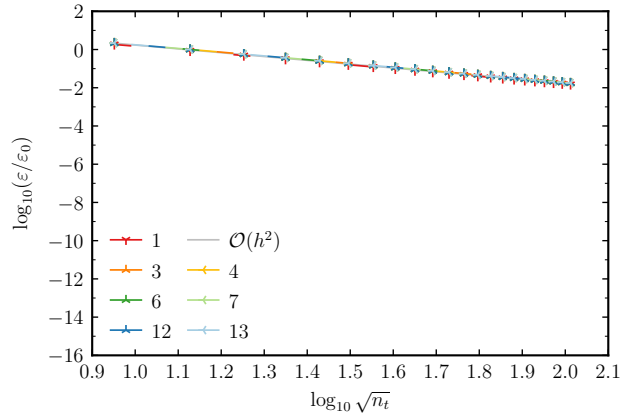
24
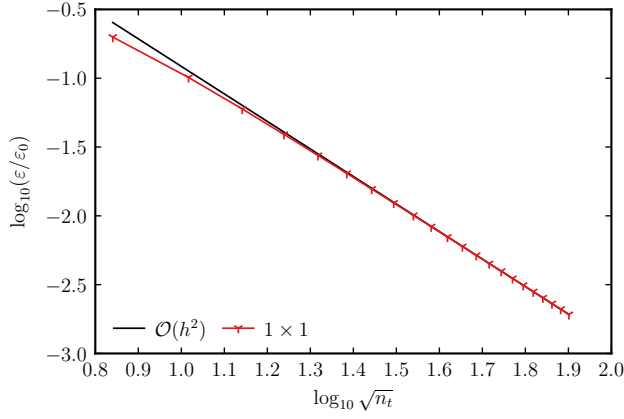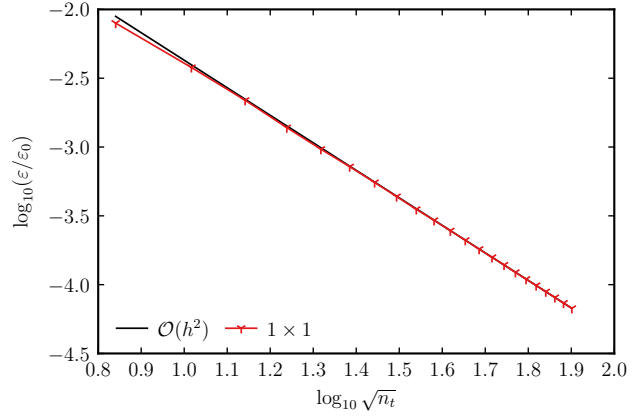
(a) $\varepsilon = e_a\big(\mathbf{J}_{h_{\mathrm{MS}}}\big)$, $d = 1$

(b) $\varepsilon = \|\mathbf{e_J}\|_\infty$ from (30), $d = 1$
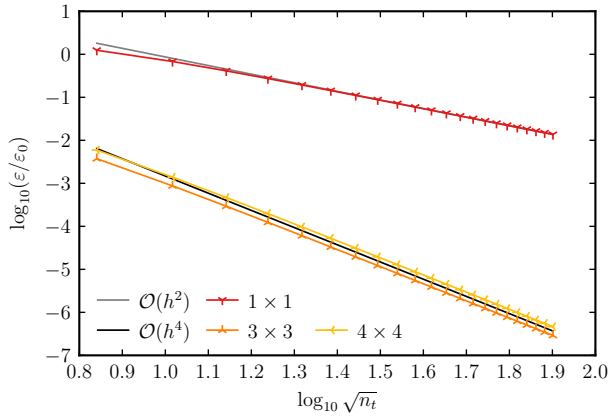
(c) $\varepsilon = e_a\big(\mathbf{J}_{h_{\mathrm{MS}}}\big)$, $d = 2$
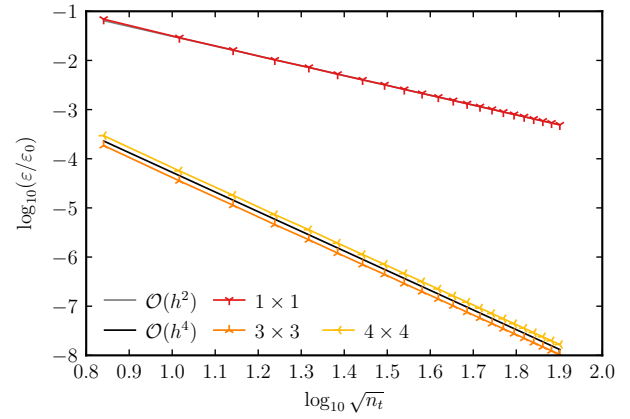
(d) $\varepsilon = \|\mathbf{e_J}\|_\infty$ from (30), $d = 2$
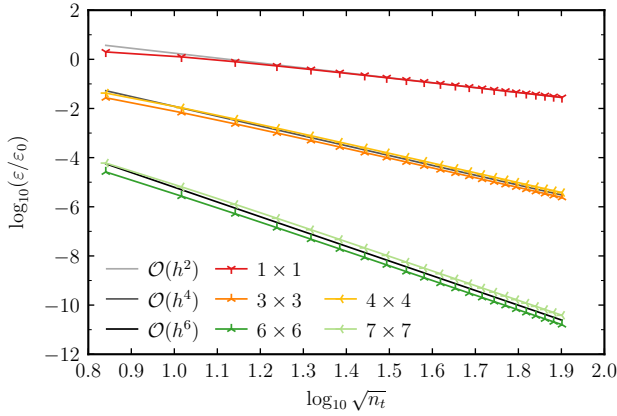
(e) $\varepsilon = e_a\big(\mathbf{J}_{h_{\mathrm{MS}}}\big)$, $d = 3$
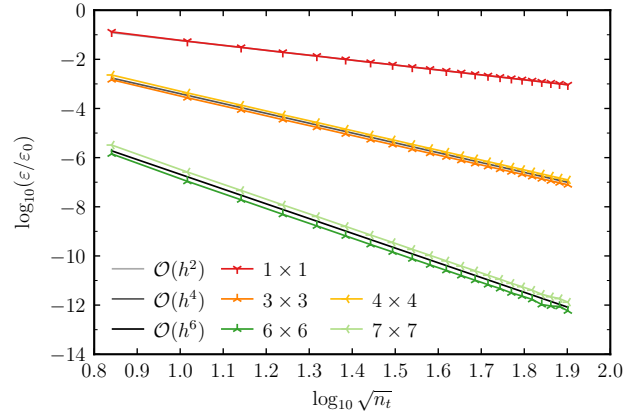
(f) $\varepsilon = \|\mathbf{e_J}\|_\infty$ from (30), $d = 3$

Figure 20: Domain-Discretization Error: ε for different amounts of quadrature points.

References

[1] R. D. Graglia, On the numerical integration of the linear shape functions times the 3-D Green's function or its gradient on a plane triangle, IEEE Transactions on Antennas and Propagation 41 (10) (1993) 1448–1455. `doi:10.1109/8.247786`.

[2] D. Wilton, S. Rao, A. Glisson, D. Schaubert, O. Al-Bundak, C. Butler, Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains, IEEE Transactions on Antennas and Propagation 32 (3) (1984) 276–281. `doi:10.1109/TAP.1984.1143304`.

[3] S. Rao, D. Wilton, A. Glisson, Electromagnetic scattering by surfaces of arbitrary shape, IEEE Transactions on Antennas and Propagation 30 (3) (1982) 409–418. `doi:10.1109/TAP.1982.1142818`.

[4] M. A. Khayat, D. R. Wilton, Numerical evaluation of singular and near-singular potential integrals, IEEE Transactions on Antennas and Propagation 53 (10) (2005) 3180–3190. `doi:10.1109/TAP.2005.856342`.

[5] P. W. Fink, D. R. Wilton, M. A. Khayat, Simple and efficient numerical evaluation of near-hypersingular integrals, IEEE Antennas and Wireless Propagation Letters 7 (2008) 469–472. `doi:10.1109/LAWP.2008.2000788`.

[6] M. A. Khayat, D. R. Wilton, P. W. Fink, An improved transformation and optimized sampling scheme for the numerical evaluation of singular and near-singular potentials, IEEE Antennas and Wireless Propagation Letters 7 (2008) 377–380. `doi:10.1109/LAWP.2008.928461`.

[7] F. Vipiana, D. R. Wilton, Optimized numerical evaluation of singular and near-singular potential integrals involving junction basis functions, IEEE Transactions on Antennas and Propagation 59 (1) (2011) 162–171. `doi:10.1109/TAP.2010.2090464`.

[8] F. Vipiana, D. R. Wilton, Numerical evaluation via singularity cancellation schemes of near-singular integrals involving the gradient of Helmholtz-type potentials, IEEE Transactions on Antennas and Propagation 61 (3) (2013) 1255–1265. `doi:10.1109/TAP.2012.2227922`.

[9] M. M. Botha, A family of augmented Duffy transformations for near-singularity cancellation quadrature, IEEE Transactions on Antennas and Propagation 61 (6) (2013) 3123–3134. `doi:10.1109/TAP.2013.2252137`.

[10] J. Rivero, F. Vipiana, D. R. Wilton, W. A. Johnson, Hybrid integration scheme for the evaluation of strongly singular and near-singular integrals in surface integral equations, IEEE Transactions on Antennas and Propagation 67 (10) (2019). `doi:10.1109/TAP.2019.2920333`.

[11] F. Vipiana, D. R. Wilton, W. A. Johnson, Advanced numerical schemes for the accurate evaluation of 4-D reaction integrals in the method of moments, IEEE Transactions on Antennas and Propagation 61 (11) (2013) 5559–5566. `doi:10.1109/TAP.2013.2277864`.

[12] A. G. Polimeridis, F. Vipiana, J. R. Mosig, D. R. Wilton, DIRECTFN: Fully numerical algorithms for high precision computation of singular integrals in Galerkin SIE methods, IEEE Transactions on Antennas and Propagation 61 (6) (2013) 3112–3122. `doi:10.1109/TAP.2013.2246854`.

[13] D. R. Wilton, F. Vipiana, W. A. Johnson, Evaluation of 4-D reaction integrals in the method of moments: Coplanar element case, IEEE Transactions on Antennas and Propagation 65 (5) (2017) 2479–2493. `doi:10.1109/TAP.2017.2677916`.

[14] J. Rivero, F. Vipiana, D. R. Wilton, W. A. Johnson, Evaluation of 4-D reaction integrals via double application of the divergence theorem, IEEE Transactions on Antennas and Propagation 67 (2) (2019) 1131–1142. `doi:10.1109/TAP.2018.2882589`.

[15] B. A. Freno, W. A. Johnson, B. F. Zinser, D. F. Wilton, F. Vipiana, S. Campione, Characterization and integration of the singular test integrals in the method-of-moments implementation of the electric-field integral equation, Engineering Analysis with Boundary Elements 124 (2021) 185–193. `doi:10.1016/j.enganabound.2020.12.015`.

[16] P. J. Roache, Verification and Validation in Computational Science and Engineering, Hermosa Publishers, 1998.

[17] P. Knupp, K. Salari, Verification of Computer Codes in Computational Science and Engineering, Chapman & Hall/CRC, 2002. `doi:10.1201/9781420035421`.

[18] W. L. Oberkampf, C. J. Roy, Verification and Validation in Scientific Computing, Cambridge University Press, 2010. `doi:10.1017/cbo9780511760396`.

[19] P. J. Roache, Code verification by the method of manufactured solutions, Journal of Fluids Engineering 124 (1) (2001) 4–10. `doi:10.1115/1.1436090`.

[20] H. Nishikawa, Analytical formulas for verification of aerodynamic force and moment computations, Journal of Computational Physics 466 (2022). `doi:10.1016/j.jcp.2022.111408`.

[21] C. J. Roy, C. C. Nelson, T. M. Smith, C. C. Ober, Verification of Euler/Navier–Stokes codes using the method of manufactured solutions, International Journal for Numerical Methods in Fluids 44 (6) (2004) 599–620. `doi:10.1002/fld.660`.

[22] R. B. Bond, C. C. Ober, P. M. Knupp, S. W. Bova, Manufactured solution for computational fluid dynamics boundary condition verification, AIAA Journal 45 (9) (2007) 2224–2236. `doi:10.2514/1.28099`.

[23] S. Veluri, C. Roy, E. Luke, Comprehensive code verification for an unstructured finite volume CFD code, in: 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, American Institute of Aeronautics and Astronautics, 2010. `doi:10.2514/6.2010-127`.

[24] T. Oliver, K. Estacio-Hiroms, N. Malaya, G. Carey, Manufactured solutions for the Favre-averaged Navier–Stokes equations with eddy-viscosity turbulence models, in: 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, American Institute of Aeronautics and Astronautics, 2012. `doi:10.2514/6.2012-80`.

[25] L. Eça, C. M. Klaij, G. Vaz, M. Hoekstra, F. Pereira, On code verification of RANS solvers, Journal of Computational Physics 310 (2016) 418–439. `doi:10.1016/j.jcp.2016.01.002`.

[26] A. Hennink, M. Tiberga, D. Lathouwers, A pressure-based solver for low-Mach number flow using a discontinuous Galerkin method, Journal of Computational Physics 425 (2022). `doi:10.1016/j.jcp.2020.109877`.

[27] B. A. Freno, B. R. Carnes, V. G. Weirs, Code-verification techniques for hypersonic reacting flows in thermochemical nonequilibrium, Journal of Computational Physics 425 (2021). `doi:10.1016/j.jcp.2020.109752`.

[28] É. Chamberland, A. Fortin, M. Fortin, Comparison of the performance of some finite element discretizations for large deformation elasticity problems, Computers & Structures 88 (11) (2010) 664 – 673. `doi:10.1016/j.compstruc.2010.02.007`.

[29] S. Étienne, A. Garon, D. Pelletier, Some manufactured solutions for verification of fluid–structure interaction codes, Computers & Structures 106-107 (2012) 56–67. `doi:10.1016/j.compstruc.2012.04.006`.

[30] A. Veeraragavan, J. Beri, R. J. Gollan, Use of the method of manufactured solutions for the verification of conjugate heat transfer solvers, Journal of Computational Physics 307 (2016) 308–320. `doi:10.1016/j.jcp.2015.12.004`.

[31] P. T. Brady, M. Herrmann, J. M. Lopez, Code verification for finite volume multiphase scalar equations using the method of manufactured solutions, Journal of Computational Physics 231 (7) (2012) 2924–2944. `doi:10.1016/j.jcp.2011.12.040`.

[32] S. Lovato, S. L. Toxopeus, J. W. Settels, G. H. Keetels, G. Vaz, Code verification of non-Newtonian fluid solvers for single- and two-phase laminar flows, Journal of Verification, Validation and Uncertainty Quantification 6 (2) (2021). `doi:10.1115/1.4050131`.

[33] R. G. McClarren, R. B. Lowrie, Manufactured solutions for the p_1 radiation-hydrodynamics equations, Journal of Quantitative Spectroscopy and Radiative Transfer 109 (15) (2008) 2590–2602. `doi:10.1016/j.jqsrt.2008.06.003`.

[34] P. Tranquilli, L. Ricketson, L. Chacón, A deterministic verification strategy for electrostatic particle-in-cell algorithms in arbitrary spatial dimensions using the method of manufactured solutions, Journal of Computational Physics 448 (2022). `doi:10.1016/j.jcp.2021.110751`.

[35] J. R. Ellis, C. D. Hall, Model development and code verification for simulation of electrodynamic tether system, Journal of Guidance, Control, and Dynamics 32 (6) (2009) 1713–1722. `doi:10.2514/1.44638`.

[36] A. J. Amar, B. F. Blackwell, J. R. Edwards, One-dimensional ablation using a full Newton's method and finite control volume procedure, Journal of Thermophysics and Heat Transfer 22 (1) (2008) 71–82. `doi:10.2514/1.29610`.

[37] A. J. Amar, B. F. Blackwell, J. R. Edwards, Development and verification of a one-dimensional ablation code including pyrolysis gas flow, Journal of Thermophysics and Heat Transfer 23 (1) (2009) 59–71. `doi:10.2514/1.36882`.

[38] A. Amar, N. Calvert, B. Kirk, Development and verification of the charring ablating thermal protection implicit system solver, in: 49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, 2011. `doi:10.2514/6.2011-144`.

[39] B. A. Freno, B. R. Carnes, N. R. Matula, Nonintrusive manufactured solutions for ablation, Physics of Fluids 33 (1) (2021). `doi:10.1063/5.0037245`.

[40] B. A. Freno, B. R. Carnes, V. E. Brunini, N. R. Matula, Nonintrusive manufactured solutions for non-decomposing ablation in two dimensions, Journal of Computational Physics 463 (2022). `doi:10.1016/j.jcp.2022.111237`.

[41] R. G. Marchand, The method of manufactured solutions for the verification of computational electromagnetic codes, PhD dissertation, Stellenbosch (Mar. 2013).

[42] R. G. Marchand, D. B. Davidson, Verification of the method-of-moment codes using the method of manufactured solutions, IEEE Transactions on Electromagnetic Compatibility 56 (4) (2014) 835–843. `doi:10.1109/TEMC.2014.2325826`.

[43] B. A. Freno, N. R. Matula, W. A. Johnson, Manufactured solutions for the method-of-moments implementation of the electric-field integral equation, Journal of Computational Physics 443 (2021). `doi:10.1016/j.jcp.2021.110538`.

[44] B. A. Freno, N. R. Matula, J. I. Owen, W. A. Johnson, Code-verification techniques for the method-of-moments implementation of the electric-field integral equation, Journal of Computational Physics 451 (2022). `doi:10.1016/j.jcp.2021.110891`.

[45] K. F. Warnick, Numerical Analysis for Electromagnetic Integral Equations, Artech House, 2008.

[46] R. Graglia, D. Wilton, A. Peterson, Higher order interpolatory vector bases for computational electromagnetics, IEEE Transactions on Antennas and Propagation 45 (3) (1997) 329–342. `doi:10.1109/8.558649`.

[47] B. A. Freno, W. A. Johnson, B. F. Zinser, S. Campione, Symmetric triangle quadrature rules for arbitrary functions, Computers & Mathematics with Applications 79 (2020). `doi:10.1016/j.camwa.2019.12.021`.

[48] R. F. Harrington, Time-Harmonic Electromagnetic Fields, Wiley–IEEE Press, 2001. `doi:10.1109/9780470546710`.

[49] W. C. Chew, Waves and Fields in Inhomogenous Media, Wiley–IEEE Press, 1995. `doi:10.1109/9780470547052`.

[50] C. A. Balanis, Advanced Engineering Electromagnetics, John Wiley & Sons, Inc., 2012.

[51] A. Peterson, L. Kempel, Solution of the MFIE using curl-conforming basis functions, in: IEEE Antennas and Propagation Society International Symposium, Vol. 1, 2002, pp. 70–73. `doi:10.1109/APS.2002.1016253`.

[52] Ö. Ergül, L. Gürel, The use of curl-conforming basis functions for the magnetic-field integral equation, IEEE Transactions on Antennas and Propagation 54 (7) (2006) 1917–1926. `doi:10.1109/TAP.2006.877159`.

[53] B. A. Freno, N. R. Matula, Code verification for practically singular equations, Journal of Computational Physics 470 (2022). `doi:10.1016/j.jcp.2022.111581`.

[54] Q. Huangfu, J. A. J. Hall, Parallelizing the dual revised simplex method, Mathematical Programming Computation 10 (2018). `doi:10.1007/s12532-017-0130-5`.

[55] J. N. Lyness, D. Jespersen, Moderate degree symmetric quadrature rules for the triangle, IMA Journal of Applied Mathematics 15 (1) (1975) 19–32. `doi:10.1093/imamat/15.1.19`.

[56] D. A. Dunavant, High degree efficient symmetrical Gaussian quadrature rules for the triangle, International Journal for Numerical Methods in Engineering 21 (6) (1985) 1129–1148. `doi:10.1002/nme.1620210612`.

[57] S.-A. Papanicolopulos, Computation of moderate-degree fully-symmetric cubature rules on the triangle using symmetric polynomials and algebraic solving, Computers & Mathematics with Applications 69 (7) (2015) 650–666. `doi:10.1016/j.camwa.2015.02.014`.