An Improved Fast Local Level Set Method for Three-dimensional Inverse Gravimetry

Wangtao Lu * Shingyu Leung † Jianliang Qian ‡

March 12, 2014

Abstract

We propose an improved fast local level set method for the inverse problem of gravimetry by developing two novel algorithms: one is of linear complexity designed for computing the Frechet derivative of the nonlinear domain inverse problem, and the other is designed for carrying out numerical continuation rapidly so as to obtain fictitious full measurement data from partial measurement. The new algorithm for the Frechet derivative is based on the properties of the Laplacian kernel. Since the kernel is symmetric and translationally invariant, we design certain affine transformations to speed up the computational process in evaluating the Frechet derivative; since it decays rapidly away from diagonal, we carry out low-rank matrix approximation to reduce storage requirements. These properties are eventually translated into an algorithm of linear complexity and linear storage requirement for computing the derivative. The new algorithm for the numerical continuation method is based on the property of the single layer density function used in representing the potential. It is smooth and periodic on an artificial hypersurface enclosing the target domain; therefore the spectral expansion is allowed to approximate this density function, which eventually leads to rapid algorithms for carrying out the numerical continuation in both 2-D and 3-D cases. 2-D and 3-D numerical examples exhibit the performance and effectiveness of the proposed new algorithms; they also illustrate that this improved level-set method is capable of computing high-resolution inversions and handling 3-D large-scale inverse gravimetry problems.

1 Introduction

This paper introduces an efficient fast local level set method for inverse problems of gravimetry. Let Ω be a domain in $\mathbb{R}^m$ with connected complement $\mathbb{R}^m \setminus \Omega$ and let $\mathcal{U}(\cdot;\ \mu)$ be the potential of a measure μ with respect to the kernel for the Laplacian operator. The integral form of inverse problems of gravimetry is posed as the following [9]: find a measure μ with support contained in Ω from its exterior potential $\mathcal{U}(\cdot;\ \mu)$ in $\mathbb{R}^m \setminus \Omega$. Often times to alleviate the severe non-uniqueness in

*Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Email: **wangtaol@math.msu.edu**

†Department of Mathematics, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR. Email: **masyleung@ust.hk**

‡Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Email: **qian@math.msu.edu**

inverse gravimetry, one assumes that the measure is a volume mass distribution, $\mu = f\chi(D)$, where D is an open subset of Ω and f is a density function. Accordingly, we obtain the inverse problem of finding $D \subset \Omega$ and f from the exterior potential induced by the volume mass distribution; furthermore, in most practical situations f is a given constant so that we end up with the domain problem of volume potential: find the characteristic domain D of the volume mass distribution μ from its exterior potential $\mathcal{U}(\cdot;\mu)$ in $\mathbb{R}^m \setminus \Omega$. In this work, we propose an improved fast level set method for the domain problem of volume potential by developing two novel algorithms: one is of linear complexity designed for computing the Frechet derivative of the nonlinear domain inverse problem and the other is designed for carrying out numerical continuation rapidly so as to obtain fictitious full measurement data from partial measurement.

A fast local level set method for the domain problem of volume potential has been proposed in [10], and the method has also been extended to a more general class of inverse gravimetric problems [11]. The theoretical foundation for the effectiveness of the level set method in dealing with the domain problem of volume potential is based on the following uniqueness result: if D is star-shaped with respect to its center of gravity or is convex in a certain coordinate direction, then the exterior potential uniquely determines D. Starting from this result, the work in [10] utilized the level set function to represent the unknown domain D and further designed a geometry-motivated efficient algorithm for computing the Frechet derivative needed in the level set evolution; since the resulting algorithm for computing the Frechet derivative is of computational complexity $O(N^{2-\frac{2}{m}})$ and storage requirement $O(N^{2-\frac{1}{m}})$, where N is the total number of mesh points and m is the dimensionality of the spatial domain Ω, the algorithm has linear complexity when $m = 2$ and has super-linear complexity when $m = 3$. Because both complexity and storage requirement are superlinear in the three-dimensional case, the resulting algorithm is not so efficient in computing high resolution inversions and handling three-dimensional large-scale inverse gravimetry problems. Consequently, in this work we propose for computing the Frechet derivative a new algorithm which is of linear complexity and linear storage requirement and works for both 2-D and 3-D cases.

Underlying this new algorithm for the Frechet derivative is a mathematical insight concerning the properties of the Laplacian kernel: it is symmetric and translational invariant and it decays rapidly away from the diagonal. The symmetry and translational invariance allows us to design certain affine transformations to speed up the computational process in evaluating the Frechet derivative; the decaying property allows us to carry out low-rank matrix approximation to reduce storage requirement. These properties are eventually translated into an algorithm of linear complexity and linear storage requirement for computing the derivative.

One of the essential difficulties in geophysical inverse problems is limited aperture or partial measurement. Specifically, in inverse gravimetric problems gravity potential related quantities are only measured on a part of the whole boundary of the computational domain. To deal with this difficult issue, numerical continuation is frequently used to obtain fictitious measurements over a closed surface enclosing the unknown domain; see [10, 6]. The underlying idea of numerical continuation is the following: first represent the potential measurement due to the unknown domain as an equivalent single-layer potential due to the unknown single-layer density on a closed surface Γ enclosing the unknown domain; second, reconstruct the single-layer density on Γ from the partial measurement by solving a Fredholm integral equation of the first kind; third, simulate fictitious full measurement on an artificial boundary due to the unknown domain. The numerical continuation approach used in [10] is not efficient as it requires a large number quadrature points to discretize the

single-layer density function on an artificial hypersurface. Observing that the single-layer density function is smooth and defined on a closed hypersurface so that it can be considered to be periodic, we propose to carry out a 2-D or 3-D spectral expansion to discretize the single-layer density function, which eventually lead to a rapid algorithm for numerical continuation.

1.1 Related work

Due to its severe ill-posedness [9], some regularization techniques were developed to convert the inverse gravimetry problem into a conditionally well-posed problem [1]. The level set method [14] is a very suitable and powerful tool for interface or shape-optimization problems due to its great ability in interface merging and topological changes. This method was first used for inverse obstacle problems in [18], and has then been applied to analyze a variety of inverse problems; see [7, 12, 20, 8, 2, 3, 7], and the references therein. More recently, a level set method was applied to identification of a characteristic function of a domain in the source term of the Poisson equation from the Cauchy data on the whole boundary of the reference domain Ω [19].

The rest of the paper is organized as follows. In section 2, we summarize the local level set method for inverse gravimetry problems as developed in [10]. In section 3, we propose a linear complexity algorithm for computing the Frechet derivative. In section 4, we propose an improved numerical continuation method for both 2-D and 3-D cases. In section 5, we carry out a number of numerical experiments for both 2-D and 3-D cases to show the performance and the effectiveness of proposed new algorithms.

2 Level-set method for inverse gravimetry

2.1 Inverse gravimetry

In a given domain $\Omega \subset \mathbb{R}^m$ for $m = 2, 3$, the gravity force generated by a mass distribution μ is defined by

$$\nabla_{\boldsymbol{r}} u(\boldsymbol{r};\ \mu) = \int_\Omega \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p}) d\mu(\boldsymbol{p}), \tag{2.1}$$

where $\boldsymbol{r} = [x, y, z]^T$ if $m = 3$ or $\boldsymbol{r} = [x, y]^T$ if $m = 2$, $\boldsymbol{p} \in \Omega$, K is the Green's function of the m-dimensional Laplace equation, i.e.,

$$K(\boldsymbol{r}, \boldsymbol{p}) = \frac{1}{4\pi|\boldsymbol{r} - \boldsymbol{p}|} \quad \text{when} \quad m = 3, \quad \text{and} \quad -\frac{1}{2\pi} \log|\boldsymbol{r} - \boldsymbol{p}| \quad \text{when} \quad m = 2,$$

and, $\{x, y, z\}$ is the standard Cartesian coordinate system in $\mathbb{R}^3$. We assume that μ is a volume distribution, $\mu = \chi_D$ where D is some bounded open set satisfying $D \subset\subset \Omega$; it is the most geophysically realistic assumption. The inverse problem of gravimetry can be posed as follows: solve the equation

$$\nabla u(\cdot;\ \chi_D)|_{\Gamma_0} = \boldsymbol{g}, \tag{2.2}$$

for the unknown domain D, where the gravimetry data $\boldsymbol{g} \in L^2(\Gamma_0)$ is given on a part of the boundary: $\Gamma_0 \subset \partial\Omega$.

The following continuation lemma and uniqueness theorem demonstrate a sufficient condition for the uniqueness of the unknown domain D [9].

Lemma 2.1. [9] *Let Ω be a convex domain with analytic (regular) boundary, and Γ_0 be a nonempty hyper surface contained in $\partial\Omega$. If measure μ_j $(j = 1, 2)$ are non-negative and $|\nabla\mathcal{U}_1| = |\nabla\mathcal{U}_2|$ on Γ_0, then $\mathcal{U}_1 = \mathcal{U}_2$ on $\mathbb{R}^m \setminus \Omega$. Here $\mathcal{U}_j = \mathcal{U}(\cdot;\ \mu_j),\ j = 1, 2$.*

Theorem 2.1. [9] *Suppose that either 1) D_1 and D_2 are star-shaped with respect to their centers of gravity or 2) D_1 and D_2 are convex in z. If $\mathcal{U}(\cdot;\ \chi_{D_1}) = \mathcal{U}(\cdot;\ \chi_{D_2})$ on $\mathbb{R}^m \setminus \Omega$, then $D_1 = D_2$.*

In light of the above theorems, we first use numerical continuation, if needed, to continue the partial measurement towards the unknown domain D so as to construct a set of gravimetry data on the whole boundary of a certain artificial domain enclosing D, and then the uniqueness Theorem 2.1 applies.

2.2 The level set method

In this method, we use a level set function ϕ^* defined in $\mathbb{R}^m$ to represent the unknown domain D, which is Lipschitz continuous satisfying

$$\phi^*(\boldsymbol{p}) > 0, \quad \text{for} \quad \boldsymbol{p} \in D, \tag{2.3a}$$
$$\phi^*(\boldsymbol{p}) = 0, \quad \text{for} \quad \boldsymbol{p} \in \partial D, \tag{2.3b}$$
$$\phi^*(\boldsymbol{p}) < 0, \quad \text{for} \quad \boldsymbol{p} \in \bar{D}^c. \tag{2.3c}$$

Then we define the following operator according to the gravity force relation (2.1):

$$A(\phi^*)(\cdot) = \nabla u(\cdot;\ \chi_D)|_{\Gamma_0}, \tag{2.4a}$$
$$\nabla u(\boldsymbol{r};\ \chi_D) = \int_D \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p}) d\boldsymbol{p} = \int_\Omega \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p}) H(\phi^*(\boldsymbol{p})) d\boldsymbol{p}, \tag{2.4b}$$

where H is the Heaviside function.

Due to its severe instability, the domain inverse problem for solving the unknown domain D can be formulated based on the Tikhonov regularization theory, as solving the following minimizing problem:

$$\min F(\phi) = \min ||A(\phi) - \boldsymbol{g}||^2_{L^2(\Gamma_0)}, \tag{2.5}$$

for the level set function ϕ.

A necessary condition for ϕ being a minimizer is that the Frechet derivative of functional F with respect to ϕ is 0, i.e.,

$$0 = \frac{\partial F}{\partial \phi}(\boldsymbol{p}) = \int_{\Gamma_0} 2G(\boldsymbol{r}; \phi)^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p}) d\boldsymbol{r} \delta(\phi(\boldsymbol{p})), \tag{2.6a}$$
$$0 = \frac{1}{|\nabla\phi|} \frac{\partial \phi}{\partial \boldsymbol{\nu}}, \quad \text{on} \quad \partial\Omega, \tag{2.6b}$$

where $\boldsymbol{\nu}$ is the unit normal vector to $\partial\Omega$, the mismatch term G is defined by

$$G(\boldsymbol{r}; \phi) = \int_\Omega \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{q}) H(\phi(\boldsymbol{q})) d\boldsymbol{q} - \boldsymbol{g}(\boldsymbol{r}), \tag{2.7}$$

and the natural boundary condition for ϕ is imposed. Thus applying the method of steepest descent, we end up with the following evolution equation

$$\frac{\partial \phi}{\partial t} = -\frac{\partial F}{\partial \phi}, \tag{2.8a}$$

$$0 = \frac{1}{|\nabla \phi|}\frac{\partial \phi}{\partial \boldsymbol{\nu}}, \quad \text{on} \quad \partial\Omega, \tag{2.8b}$$

where $\phi = \phi(\boldsymbol{r}, t)$ with t being the artificial evolution time. Finally, we take $\phi^* = \phi(\boldsymbol{r}, \infty)$ so that ∂D is defined by the zero level set of ϕ^*: $\partial D = \{\boldsymbol{p} : \phi^*(\boldsymbol{p}) = 0\}$.

To avoid numerical instabilities, the Heaviside function in the integrand (2.4) is approximated by the following ϵ-Heaviside function $H_\epsilon(\phi)$, i.e.,

$$H_\epsilon(\phi) = \begin{cases} 0, & \phi < -\epsilon, \\ \frac{1}{2} + \frac{\phi}{2\epsilon} + \frac{1}{2\pi}\sin\left(\frac{\pi\phi}{\epsilon}\right), & -\epsilon \le \phi \le \epsilon, \\ 1, & \phi > \epsilon. \end{cases} \tag{2.9}$$

The delta function $\delta(\phi)$ in (2.6) can be approximated by $\delta_\epsilon(\phi) = \chi_{T_\epsilon}|\nabla\phi|$, see [20], with support $T_\epsilon = \{\boldsymbol{p} \in \Omega : |\phi(\boldsymbol{p})| \le \epsilon\} \subset\subset \Omega$ for some $\epsilon > 0$. Without loss of generality, in the rest of the paper, we let Ω be the m-dimensional unit cube for $m = 2,\ 3$.

The numerical algorithm [10] for realizing the above level-set based formulation is summarized as follows:

Algorithm 2.1. *Isakov-Leung-Qian Algorithm [10]:*

1. *Initialize the level set function ϕ.*
2. *Compute the mismatch G along the boundary Γ_0 according to (2.7).*
3. *Compute the level set derivative of the energy according to (2.6).*
4. *Evolve the level set function according to the gradient flow (2.8).*
5. *Reinitialize the level set function to maintain the signed distance property.*
6. *Repeat 2-5 until it converges.*

In Algorithm 2.1, the most time and storage consuming steps are step 2 and 3, aiming at computing the Frechet derivative in Eq. (2.6a). Directly applying trapezoidal rules to discretize involved integrals in Eqs. (2.6a) and (2.7) requires a complexity of $\mathcal{O}(N^{2-1/m})$ where $N = n^m$ is the total number of mesh points and n is the number of mesh points in each direction for $m = 2, 3$. A geometry-motivated algorithm was developed to reduce the complexity to $\mathcal{O}(N^{2-2/m})$ in [10]; however, it achieves superlinear complexity of $\mathcal{O}(N^{4/3})$ for the 3-D case. Motivated by this, we develop a linear complexity algorithm to compute Frechet derivative Eq. (2.6a) in the following. For the other steps, please see [10] for details; related techniques have been widely used in the level set community in various applications [14, 15, 16, 17, 4].

3 Fast algorithm for computing Frechet derivative

We focus on the 3-D case only in this section. Firstly, we develop a linear complexity algorithm for computing the Frechet derivative based on the properties of the kernel ∇K. To further accelerate the computational process and to reduce the cost of storage, we develop a compression algorithm. One will find that the new algorithm can be naturally extended to the 2-D case.

3.1 Linear complexity algorithm

Suppose the gravity field $\boldsymbol{g}$ is measured on the whole boundary of Ω. Since Ω is assumed to be a unit cube, $\Gamma_0 = \partial\Omega$ is the union of six faces of the unit cube: bottom, top, front, back, left and right faces, as shown in Fig. 3.1. Each face is further split into eight triangles. Since rigid transformations

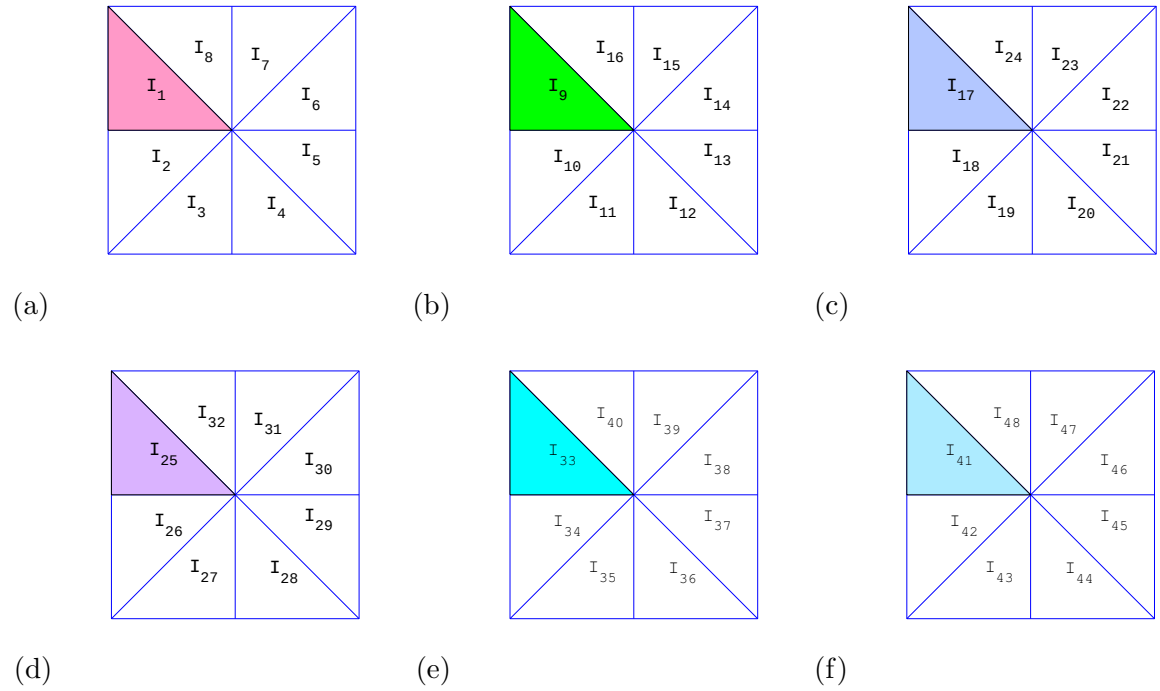


Figure 3.1: Six faces of the unit cube: (a): Bottom ($z = 0$): I_1 to I_8; (b): Top ($z = 1$): I_9 to I_{16}; (c): Front ($x = 1$): I_{17} to I_{24}; (d): Back ($x = 0$): I_{25} to I_{32}; (e): Left ($y = 0$): I_{33} to I_{40}; (f): Right ($y = 1$): I_{41} to I_{48}.

can be used to transform one triangle to another between any two of the 48 triangles, there exists an orthogonal transformation f_i and a translation operator c_i such that $g_i(I_1) := c_i \circ f_i(I_1) = I_i$ for $1 \le i \le 48$. For example,

$$g_4(\boldsymbol{r}) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{r}, \quad g_{41}(\boldsymbol{r}) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \boldsymbol{r}, \quad \boldsymbol{r} \in I_1.$$

Since g_i may not be unique, we choose g_i such that it can be extended to a one-to-one mapping from $\overline{\Omega}$ onto $\overline{\Omega}$. Therefore, the following choice of g_4,

$$g_4(\boldsymbol{r}) = \begin{bmatrix} -1/2 \\ 1/2 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{r},$$

will be excluded. Without confusing, we will overload the notation to denote the extension of map g_i by g_i.

Since the Frechet derivative in Eq. (2.6a) vanishes outside T_ϵ, we only need to compute the Frechet derivative at point $\boldsymbol{p} \in T_\epsilon$. The surface integral over $\Gamma_0 = \partial\Omega$ in Eq. (2.6a) satisfies

$$\begin{aligned}
&\int_{\partial\Omega} G(\boldsymbol{r};\phi)^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{p}) d\boldsymbol{r} = \sum_{i=1}^{48} \int_{I_i} G(\boldsymbol{r};\phi)^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{p}) d\boldsymbol{r} \\
&= \sum_{i=1}^{48} \int_{I_i} \int_{\Omega} \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{q})^T H(\phi(\boldsymbol{q})) d\boldsymbol{q} \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{p}) d\boldsymbol{r} - \sum_{i=1}^{48} \int_{I_i} \boldsymbol{g}(\boldsymbol{r})^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{p}) d\boldsymbol{r} \\
&=: Y_1(\boldsymbol{p}) - Y_2(\boldsymbol{p}),
\end{aligned} \tag{3.1}$$

for $\boldsymbol{p} \in T_\epsilon$. To simplify Y_1 and Y_2, we need the following relation,

$$\begin{aligned}
\nabla_{\boldsymbol{q}} K(\boldsymbol{q},\boldsymbol{p})|_{\boldsymbol{q}=g_i(\boldsymbol{r})} &= -\frac{g_i(\boldsymbol{r}) - \boldsymbol{p}}{4\pi |g_i(\boldsymbol{r}) - \boldsymbol{p}|^3} \\
&= -\frac{f_i(\boldsymbol{r}) - c_i^{-1}(\boldsymbol{p})}{4\pi |f_i(\boldsymbol{r}) - c_i^{-1}(\boldsymbol{p})|^3} \\
&= -\frac{f_i(\boldsymbol{r} - g_i^{-1}(\boldsymbol{p}))}{4\pi |f_i(\boldsymbol{r} - g_i^{-1}(\boldsymbol{p}))|^3} \quad (\text{since } f_i \circ g_i^{-1} = c_i^{-1}) \\
&= f_i\left(-\frac{\boldsymbol{r} - g_i^{-1}(\boldsymbol{p})}{4\pi |\boldsymbol{r} - g_i^{-1}(\boldsymbol{p})|^3}\right) \quad (\text{since } f_i \text{ is orthogonal and linear}) \\
&= f_i\left(\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p}))\right).
\end{aligned} \tag{3.2}$$

By (3.2) and by change of variables, we have

$$\begin{aligned}
Y_1(\boldsymbol{p}) &= \sum_{i=1}^{48} \int_{I_1} \int_{\Omega} f_i(\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{q})))^T H(\phi(\boldsymbol{q})) d\boldsymbol{q} f_i(\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p}))) |\det(\boldsymbol{J}_{\boldsymbol{r}} g_i(\boldsymbol{r}))| d\boldsymbol{r} \\
&= \sum_{i=1}^{48} \int_{I_1} \int_{\Omega} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{q}))^T H(\phi(\boldsymbol{q})) d\boldsymbol{q} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p})) d\boldsymbol{r} \\
&\quad (\text{since the Jacobian } \det(\boldsymbol{J}_{\boldsymbol{r}} g_i(\boldsymbol{r})) = \pm 1 \text{ and } f_i \text{ is orthogonal}) \\
&= \int_{I_1} \left\{ \sum_{i=1}^{48} \int_{\Omega} \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{q})^T H \circ \phi \circ g_i(\boldsymbol{q}) d\boldsymbol{q} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p})) \right\} d\boldsymbol{r}, \quad \boldsymbol{p} \in T_\epsilon,
\end{aligned} \tag{3.3}$$

and similarly,

$$Y_2(\boldsymbol{p}) = \sum_{i=1}^{48} \int_{I_1} \boldsymbol{g} \circ g_i(\boldsymbol{r})^T f_i(\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p})))|\det(\boldsymbol{J}_{\boldsymbol{r}} g_i(\boldsymbol{r}))| d\boldsymbol{r}$$
$$= \int_{I_1} \left\{ \sum_{i=1}^{48} [f_i^{-1} \circ \boldsymbol{g} \circ g_i(\boldsymbol{r})]^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, g_i^{-1}(\boldsymbol{p})) \right\} d\boldsymbol{r}, \quad \boldsymbol{p} \in T_\epsilon. \tag{3.4}$$

Here, $\boldsymbol{J}_{\boldsymbol{r}}$ denotes the Jacobian matrix with respect to $\boldsymbol{r}$.

In fact, the above two integrands as functions of $\boldsymbol{r}$ are smooth in I_1. The reason is as follows: denoting the unknown domain represented by ϕ by $\tilde{D}$, it is reasonable to assume that $\tilde{D}$ is not far away from the correct solution D so that $\tilde{D} \subset\subset \Omega$; consequently, for any $\boldsymbol{q} \notin g_i^{-1}(\tilde{D})$ (or $g_i(\boldsymbol{q}) \notin \tilde{D}$), we have

$$H \circ \phi \circ g_i(\boldsymbol{q}) = 0$$

so that the integrand of the integral over Ω inside the brackets of Eq. (3.3), as a function of $\boldsymbol{q}$, has a compact support inside $\overline{g_i^{-1}(\tilde{D})} \subset\subset \Omega$; and for any $\boldsymbol{p} \in T_\epsilon$, recalling that $T_\epsilon \subset\subset \Omega$, we have

$$g_i^{-1}(\boldsymbol{p}) \in g_i^{-1}(T_\epsilon) \subset\subset \Omega$$

so that $g_i^{-1}(\boldsymbol{p}) \neq \boldsymbol{r}$ for any $\boldsymbol{r} \in I_1$. Therefore, Gauss-Legendre quadrature rules can be applied to approximate the above two surface integrals over triangle I_1, that is,

$$Y_1(\boldsymbol{p}) \approx \sum_{k=1}^{N_0} w_k \sum_{i=1}^{48} \int_{\Omega} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, \boldsymbol{q})^T H \circ \phi \circ g_i(\boldsymbol{q}) d\boldsymbol{q} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p})), \tag{3.5}$$

$$Y_2(\boldsymbol{p}) \approx \sum_{k=1}^{N_0} w_k \sum_{i=1}^{48} f_i^{-1} \circ \boldsymbol{g} \circ g_i(\boldsymbol{r}_k)^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p})), \tag{3.6}$$

where $\nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, \cdot) = \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \cdot)|_{\boldsymbol{r}=\boldsymbol{r}_k}$, and $\{\boldsymbol{r}_k\}_{k=1}^{N_0}$ and w_k are the associated nodes and weights of the N_0-point Gaussian quadrature rule in domain I_1.

In Eq. (3.5), we apply the trapezoidal rule to approximate the volume integral over Ω, yielding

$$Y_1(\boldsymbol{p}) \approx \sum_{k=1}^{N_0} w_k \sum_{i=1}^{48} h^3 \sum_{l=1}^{N} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, \boldsymbol{q}_l)^T H \circ \phi \circ g_i(\boldsymbol{q}_l) \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p})), \tag{3.7}$$

where Ω is uniformly discretized by $N = n \times n \times n$ mesh points $\{\boldsymbol{q}_l\}_{l=1}^{N} = \{[ih, jh, kh]^T | 0 \leq i, j, k \leq n-1\}$, and $h = 1/(n-1)$ is the grid size in each direction. Although $\nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, \boldsymbol{q}_l)$ may be unbounded which only happens when $\boldsymbol{q}_l$ is located on the boundary $\partial\Omega$, this situation does not affect the computation since the term $H \circ \phi \circ g_i$ vanishes on the boundary $\partial\Omega$. To stay on the safe side, we approximate

$$\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{q}) = -\frac{1}{4\pi} \frac{\boldsymbol{r} - \boldsymbol{q}}{|\boldsymbol{r} - \boldsymbol{q}|^3} \approx -\frac{1}{4\pi} \frac{\boldsymbol{r} - \boldsymbol{q}}{|\boldsymbol{r} - \boldsymbol{q}|^3 + \epsilon_0},$$

for extremely small $\epsilon_0 > 0$ (e.g., $\epsilon_0 = 10^{-20}$).

To evolve ϕ at the mesh points $\{\boldsymbol{q}_l\}_{l=1}^N$ in Ω, we need its Frechet derivative at those points. Since the Frechet derivative vanishes outside T_ϵ, we only need to compute the derivative at mesh points lying in T_ϵ. Assuming that there are M mesh points in T_ϵ, denoted by $\{\boldsymbol{p}_j\}_{j=1}^M$, there exists an $l_j \in [1, N]$ so that $\boldsymbol{p}_j = \boldsymbol{q}_{l_j}$ for $1 \le j \le M$. We evaluate Y_1 and Y_2 at $\boldsymbol{p} = \boldsymbol{p}_j$,

$$Y_1(\boldsymbol{p}_j) \approx \sum_{k=1}^{N_0} w_k \sum_{i=1}^{48} h^3 \sum_{l=1}^{N} \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, \boldsymbol{q}_l)^T H \circ \phi \circ g_i(\boldsymbol{q}_l) \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p}_j)), \tag{3.8}$$

$$Y_2(\boldsymbol{p}_j) \approx \sum_{k=1}^{N_0} w_k \sum_{i=1}^{48} f_i^{-1} \circ \boldsymbol{g} \circ g_i(\boldsymbol{r}_k)^T \nabla_{\boldsymbol{r}} K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p}_j)), \tag{3.9}$$

for $j = 1, \cdots, M$. Both equations can be written in a matrix form, i.e.,

$$\boldsymbol{Y}_1 \approx h^3 \sum_{i=1}^{48} \sum_{\alpha \in \{x,y,z\}} \boldsymbol{D}_\alpha^{i,T} \boldsymbol{W} \boldsymbol{D}_\alpha \boldsymbol{b}^i, \quad \text{and} \quad \boldsymbol{Y}_2 \approx h^3 \sum_{i=1}^{48} \sum_{\alpha \in \{x,y,z\}} \boldsymbol{D}_\alpha^{i,T} \boldsymbol{W} \boldsymbol{a}_\alpha^i, \tag{3.10}$$

where

$$\begin{aligned}
&\boldsymbol{a}_\alpha^i = \left[\left(f_i^{-1} \circ \boldsymbol{g} \circ g_i(\boldsymbol{r}_k)\right)_\alpha\right] = [(\boldsymbol{a}_\alpha^i)_k]_{N_0 \times 1}, \\
&\boldsymbol{b}^i = [H \circ \phi \circ g_i(\boldsymbol{q}_l)] = [(\boldsymbol{b}^i)_l]_{N \times 1}, \\
&\boldsymbol{W} = \mathrm{diag}\{w_1, \cdots, w_{N_0}\}, \\
&\boldsymbol{D}_\alpha = [\partial_\alpha K(\boldsymbol{r}_k, \boldsymbol{q}_l)] = [(\boldsymbol{D}_\alpha)_{kl}]_{N_0 \times N}, \\
&\boldsymbol{D}_\alpha^i = \left[\partial_\alpha K(\boldsymbol{r}_k, g_i^{-1}(\boldsymbol{p}_j)\right] = [(\boldsymbol{D}_\alpha^i)_{kj}]_{N_0 \times M}.
\end{aligned}$$

Here, $(\cdot)_\alpha$ denotes its α-component, $\alpha = x, y, z$, $\boldsymbol{Y}_1 = [Y_1(\boldsymbol{p}_j)]_{M \times 1}$, and $\boldsymbol{Y}_2 = [Y_2(\boldsymbol{p}_j)]_{M \times 1}$. Since the mesh points $\{\boldsymbol{q}_l\}_{l=1}^N$ remain invariant under any transformation g_i, we assert that $\boldsymbol{D}_\alpha^i$ consists of certain columns of $\boldsymbol{D}_\alpha$.

Clearly, both the computational complexity and storage requirement for evaluating $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ in Eq.(3.10) are $\mathcal{O}(N_0(N+M)) = \mathcal{O}(N_0 N)$. Since N_0 is fixed, the new method achieves linear complexity.

3.2 A low-rank-matrix based compression algorithm

If N_0 is comparable to N, the $N_0 \times N$ matrix $\boldsymbol{D}_\alpha$ and its $N_0 \times M$ submatrix $\boldsymbol{D}_\alpha^i$ for $\alpha = x, y, z$, can still occupy high volume of storage and slow down the computational procedure when Ω is heavily refined. The kernel function $\nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{q})$ is inversely proportional to the square of distance $|\boldsymbol{r} - \boldsymbol{q}|$ for $\boldsymbol{r} \in I_1$ and $\boldsymbol{q} \in \Omega$. Therefore, elements in $\boldsymbol{D}_\alpha$ decay rapidly as $\boldsymbol{q}$ goes away from $\boldsymbol{r}$. We expect that if each $\boldsymbol{D}_\alpha$ is partitioned into submatrices so that entries indexed by $\boldsymbol{q}$ and $\boldsymbol{r}$ in each submatrix are of almost the same magnitude in terms of $\frac{1}{|\boldsymbol{r}-\boldsymbol{q}|}$, then a low-rank structure can be revealed for each submatrix.

Take $\boldsymbol{D}_x$ as an example. As the z-component of $\boldsymbol{r} \in I_1$ is zero, the larger z-component of $\boldsymbol{q}$, say $(\boldsymbol{q})_z$, is, the smaller the related element in $\boldsymbol{D}_x$ becomes. This suggests an initial partition of $\boldsymbol{D}_x$ into n submatrices $\{\boldsymbol{D}_x^k\}_{k=0}^{n-1}$ so that entries of each $N_0 \times n^2$ matrix $\boldsymbol{D}_x^k$ corresponding to $(\boldsymbol{q})_z = kh$

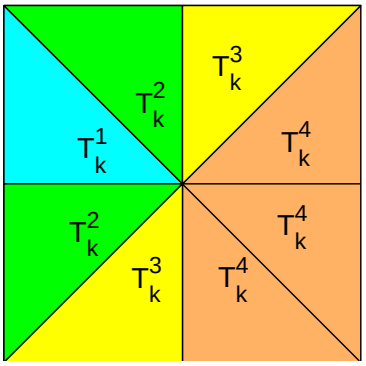


Figure 3.2: Four domains $\{T_k^l\}_{l=1}^4$ at $z = kh$.

decay as k increases. The cross section of Ω at $z = kh$ consists of eight triangles as shown in Fig. 3.2, where T_k^l is defined by

$$E_k^1 = \{(x, y, kh)|((x, y, 0) \in I_1\}, \quad E_k^2 = \{(x, y, kh)|(x, y, 0) \in I_2 \cup I_8\},$$
$$E_k^3 = \{(x, y, kh)|(x, y, 0) \in I_3 \cup I_7\}, E_k^4 = \{(x, y, kh)|(x, y, 0) \in I_4 \cup I_5 \cup I_6\},$$
$$T_k^l = \overline{E_k^l} \setminus \overline{E_k^{l-1}}, \quad l = 1, 2, 3, 4. \quad (\text{Here, } E_k^0 = \emptyset), \quad \text{for} \quad k = 0, \cdots, n-1.$$

Triangles with different colors have different levels of distance from I_1. Since T_k^1 is the closest to I_1, followed by T_k^2, T_k^3 and T_k^4, we further partition $\boldsymbol{D}_x^k$ into four submatrices $\{\boldsymbol{D}_x^{k,l}\}_{l=1}^4$ so that entries in $\boldsymbol{D}_x^{k,l}$ correspond to $\boldsymbol{q} \in T_k^l$. Although we can further proceed with the partition of $\boldsymbol{D}_x^{k,l}$ so that the resulting blocks yield uniform rank distribution, our numerical experiments show that the above two-step partition provides enough savings in time and storage; therefore no further partitions are executed in this paper.

To summarize, we partition $\boldsymbol{D}_x$ into $4n$ submatrices $\{\boldsymbol{D}_x^{k,l}|0 \le k \le n-1, 1 \le l \le 4\}$, where $\boldsymbol{D}_x^{k,l}$ is of size $N_0 \times N^{k,l}$ with $N^{k,l} = \mathcal{O}(n^2)$. Entries in each $\boldsymbol{D}_x^{k,l}$ decay rapidly as k or l increases and so does the numerical rank of $\boldsymbol{D}_x^{k,l}$. Consequently, it becomes feasible to use a truncated singular value decomposition (SVD) to decompose $\boldsymbol{D}_x^{k,l}$ into a sum of low-rank matrices such that only the information of those low-rank matrices is needed and the related matrix-vector multiplications can be carried out rapidly.

We start from the $N_0 \times N^{0,1}$ matrix $\boldsymbol{D}_x^{0,1}$. Firstly, we compute its SVD,

$$\boldsymbol{D}_x^{0,1} = \boldsymbol{U}^{0,1}\boldsymbol{S}^{0,1}(\boldsymbol{V}^{0,1})^T,$$

where both $\boldsymbol{U}^{0,1}$ and $V^{0,1}$ are unitary matrices of size $N_0 \times N_0$ and $N^{0,1} \times N_0$, respectively, and the diagonal matrix $S = \text{diag}\{\sigma_1, \cdots, \sigma_{N_0}\}$ with singular values in descending order. Then we set a threshold ϵ_{SVD} for the decomposition, set those singular values less than ϵ_{SVD} to be zero, and

obtain

$$\boldsymbol{D}_x^{0,1} \approx \boldsymbol{U}_+^{0,1}\boldsymbol{S}_+^{0,1}(\boldsymbol{V}_+^{0,1})^T,$$

where $\boldsymbol{S}_+^{0,1} = \mathrm{diag}\{\sigma_1, \cdots, \sigma_{\tau_{0,1}}\}$ with the number of $\tau_{0,1}$ singular values greater than ϵ_{SVD}, and $\boldsymbol{U}_+^{0,1}$ and $\boldsymbol{V}_+^{0,1}$ are the first $\tau_{0,1}$ columns of $\boldsymbol{U}^{0,1}$ and $\boldsymbol{V}^{0,1}$, respectively. If $\tau_{0,1}$ is much smaller than $\min\{N_0, N^{0,1}\}$, we choose to store $\boldsymbol{U}_+^{0,1}, \boldsymbol{S}_+^{0,1}$ and $\boldsymbol{V}_+^{0,1}$ instead and we perform the matrix-vector multiplication of $\boldsymbol{D}_x^{0,1}$ and any column vector $\boldsymbol{c} \in \mathbb{R}^{N^{0,1}\times 1}$ by

$$\boldsymbol{D}_x^{0,1}\boldsymbol{c} \approx \boldsymbol{U}_+^{0,1}(\boldsymbol{S}_+^{0,1}((\boldsymbol{V}_+^{0,1})^T\boldsymbol{c})). \tag{3.11}$$

Consequently, both the computational complexity and storage requirement are reduced from $\mathcal{O}(N_0N^{0,1})$ to $\mathcal{O}(\tau^{0,1}(N_0 + N^{0,1}))$. Cases for the other matrices $\boldsymbol{D}_x^{k,l}$ can be analyzed similarly.

Now, for any $\boldsymbol{b}^i$ as introduced in Eq. (3.10), we have

$$\boldsymbol{D}_x\boldsymbol{b}^i = \sum_{k=0}^{n-1}\sum_{l=1}^{4}\boldsymbol{D}_x^{k,l}\boldsymbol{b}_{k,l}^i \approx \sum_{k=0}^{n-1}\sum_{l=1}^{4}\boldsymbol{U}_+^{k,l}(\boldsymbol{S}_+^{k,l}((\boldsymbol{V}_+^{k,l})^T\boldsymbol{b}_{k,l}^i)), \tag{3.12}$$

where $\boldsymbol{b}^i$ is partitioned into $4n$ sub-vectors $\boldsymbol{b}_{k,l}^i$ with its size being consistent with $\boldsymbol{D}_x^{k,l}$, and the three matrices $\boldsymbol{U}_+^{k,l}, \boldsymbol{S}_+^{k,l}$ and $\boldsymbol{V}_+^{k,l}$ are assumed to be the truncated SVD of $\boldsymbol{D}_x^{k,l}$ using the fixed threshold ϵ_{SVD}. Denote by $\tau_{k,l}$, the number of dominant singular values of $\boldsymbol{D}_x^{k,l}$ greater than ϵ_{SVD}. We can deduce from Eq. (3.12) that both the computational complexity and storage requirement are reduced from $\mathcal{O}(N_0N)$ to

$$\mathcal{O}\left(\sum_{k=0}^{n-1}\sum_{l=1}^{4}\tau_{k,l}(N_0 + N/(4n))\right) \le \mathcal{O}(\tau_{\max}(4nN_0 + N)), \tag{3.13}$$

where $\tau_{\max} = \max_{k,l}\tau_{k,l}$. Although it is expensive to compute truncated SVDs of $\boldsymbol{D}_x^{k,l}$, this computational procedure is executed only once as a preprocessing step and we store those matrices $[\boldsymbol{U}_+^{k,l}, \boldsymbol{S}_+^{k,l}, \boldsymbol{V}_+^{k,l}]$; the total complexity is

$$\mathcal{O}\left(\sum_{k=0}^{n-1}\sum_{l=1}^{4}\mathcal{O}(N_0^2N^{k,l})\right) = \mathcal{O}(N_0^2N).$$

We can reload them whenever they are needed. Cases for the other two matrices $\boldsymbol{D}_y$ and $\boldsymbol{D}_z$ can be analyzed similarly.

For the other three matrices $\boldsymbol{D}_\alpha^i, \alpha = x, y, z$ of size $N_0 \times M$, we need to compute, for any column vector $\boldsymbol{d} \in \mathbb{R}^{N_0\times 1}$, the product of $(\boldsymbol{D}_\alpha^i)^T$ and $\boldsymbol{d}$. Since a direct computation requires $\mathcal{O}(N_0M)$ floating operations, we may apply the same strategy to process $\boldsymbol{D}_\alpha^i$ as used for $\boldsymbol{D}_\alpha$. However, because of the same underlying structure, we claim that we can directly extract relevant information for $\boldsymbol{D}_\alpha^i$ from those obtained for $\boldsymbol{D}_\alpha$. To illustrate this, we take $\boldsymbol{D}_x^i$ as an example, which is a submatrix of $\boldsymbol{D}_x$. Applying the same two-step partition as before, we obtain submatrices of $\boldsymbol{D}_x^i$: $\{(\boldsymbol{D}_x^i)^{k,l}|0 \le k \le n-1, 1 \le l \le 4\}$, where $(\boldsymbol{D}_x^i)^{k,l}$ consists of column vectors of $\boldsymbol{D}_x^{k,l}$ with the column index vector denoted by $\boldsymbol{I}_{k,l}$. To obtain a formula similar to Eq. (3.12), we need the truncated SVD of $(\boldsymbol{D}_x^i)^{k,l}$.

We start from $(\boldsymbol{D}_x^i)^{0,1}$. Since matrix multiplication rules show that its SVD is

$$(\boldsymbol{D}_x^i)^{0,1} = \boldsymbol{U}^{0,1}\boldsymbol{S}^{0,1}(\boldsymbol{V}_{\boldsymbol{I}_{0,1}}^{0,1})^T,$$

where $\boldsymbol{V}_{\boldsymbol{I}_{0,1}}^{0,1}$ consists of row vectors of $\boldsymbol{V}^{0,1}$ with the row index vector $\boldsymbol{I}_{0,1}$, its truncated SVD using the threshold ϵ_{SVD} is

$$(\boldsymbol{D}_x^i)^{0,1} \approx \boldsymbol{U}_+^{0,1}\boldsymbol{S}_+^{0,1}(\boldsymbol{V}_{\boldsymbol{I}_{0,1}}^{0,1})_+^T,$$

where $(\boldsymbol{V}_{\boldsymbol{I}_{0,1}}^{0,1})_+$ is the first $\tau_{0,1}$ columns of $\boldsymbol{V}_{\boldsymbol{I}_{0,1}}^{0,1}$. Cases for other matrices $(\boldsymbol{D}_x^i)^{k,l}$ can be analyzed similarly to get their corresponding approximate SVDs: $[\boldsymbol{U}_+^{k,l}, \boldsymbol{S}_+^{k,l}, (\boldsymbol{V}_{\boldsymbol{I}_{k,l}}^{k,l})_+]$, where $(\boldsymbol{V}_{\boldsymbol{I}_{kl}}^{k,l})_+$ is the matrix composed by certain row vectors of $\boldsymbol{V}_+^{k,l}$ with the row index vector $\boldsymbol{I}_{kl}$. Finally, we obtain ${\boldsymbol{D}_x^i}^T\boldsymbol{d}$ by carrying out the following multiplication:

$$\boldsymbol{d}^T\boldsymbol{D}_x^i = \sum_{k=0}^{n-1}\sum_{l=1}^{4}\boldsymbol{d}^T(\boldsymbol{D}_x^i)^{k,l} \approx \sum_{k=0}^{n-1}\sum_{l=1}^{4}((\boldsymbol{d}^T\boldsymbol{U}_+^{k,l})\boldsymbol{S}_+^{k,l})(\boldsymbol{V}_{\boldsymbol{I}_{kl}}^{k,l})_+^T, \tag{3.14}$$

Clearly, the computational complexity and storage requirements are reduced from $\mathcal{O}(N_0M)$ to

$$\mathcal{O}\left(\sum_{k=0}^{n-1}\sum_{l=1}^{4}\tau_{k,l}(N_0 + |\boldsymbol{I}_{kl}|)\right) \leq \mathcal{O}(\tau_{\max}(4nN_0 + M)),$$

where $|\boldsymbol{I}_{kl}|$ denotes the length of vector $\boldsymbol{I}_{kl}$. We emphasize that we only need to store the index vectors $\boldsymbol{I}_{kl}$ instead of $\boldsymbol{D}_x^i$. Cases for the other matrices $\boldsymbol{D}_y^i$ and $\boldsymbol{D}_z^i$ can be analyzed similarly.

To sum up, once relevant truncated SVDs are computed in the preprocessing step with a linear complexity $\mathcal{O}(N_0^2N)$, we conclude that by Eqs. (3.12) and (3.14), the computational complexity of evaluating $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ in Eq. (3.10) is no more than

$$\mathcal{O}(\tau_{\max}(8nN_0 + M + N)) = \mathcal{O}(nN_0 + N + M).$$

The details of complexity and storage requirement for computing the Frechet derivative in (2.6a) by different algorithms are listed in Table 3.1. To illustrate the performance of the new algorithm

	Algorithm in [10]	New algorithm (excluding the preprocessing step)
Complexity	$\mathcal{O}(N^{4/3} + N^{2/3}M)$	$\mathcal{O}(nN_0 + N + M)$
Storage	$\mathcal{O}(N^{5/3})$	$\mathcal{O}(nN_0 + N + M)$

Table 3.1: Complexity and storage for different algorithms for 3-D case.

explicitly, for the specific example studied in section 5.2.1, we check the running times (excluding the preprocessing step) in computing Frechet derivatives in only one iteration of Algorithm 2.1 for different values of N. The relation is shown in Fig. 3.3, where both axes are scaled logarithmically, and both the algorithm in [10] and the new algorithm are tested. To produce Fig. 3.3, we take N_0 to be 36; n varies from 21 to 129 with a step size 4 for the '+' line, and varies from 21 to 65 with the same step size 4 for the 'o' line. We can see from Fig. 3.3 that: the new algorithm is much faster and achieves linear complexity; the running time of the linear algorithm for the finest mesh with $N = 129^3$ is almost the same as that of the superlinear algorithm for the coarsest mesh with $N = 21^3$.

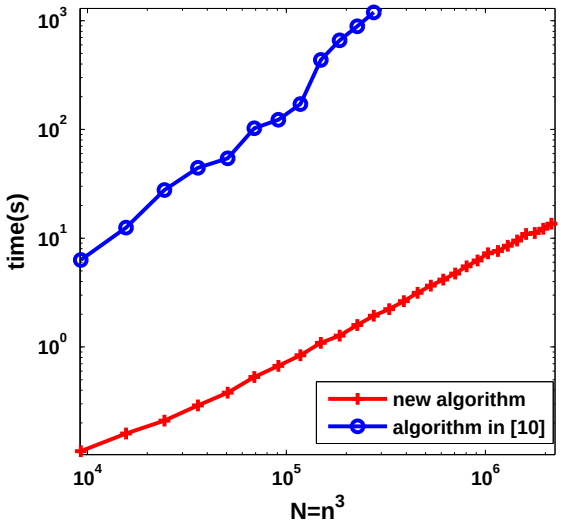


Figure 3.3: Performances for computing Frechet derivatives by the linear algorithm as proposed here and the superlinear algorithm in [10] in the 3-D case.

3.3 Implementation

We discuss some implementation details by focusing mainly on the following two aspects.

3.3.1 Gaussian quadrature rule for the triangle I_1

Since the domain I_1 is a triangle, there is no standard Gaussian quadrature rule available; therefore, we derive such a quadrature rule from the standard Gaussian quadrature on a rectangle.

For any two dimensional analytic function $f(\boldsymbol{r})$ defined on the bottom face $[0,1]\times[0,1]\times\{z=0\}$, its double integral can be approximated by

$$\int_0^1 dx \int_0^1 f(x,y,0)dy \approx \sum_{k=1}^{\tilde{m}}\sum_{l=1}^{\tilde{m}} f(x_k,x_l,0)\tilde{w}_k\tilde{w}_l, \tag{3.15}$$

where $\{x_k\}_{k=1}^{\tilde{m}}$ and $\{\tilde{w}_k\}_{k=1}^{\tilde{m}}$ are the associated nodes and weights of the standard $\tilde{m}$-points Gauss-Legendre quadrature rules over the interval $[0,1]$ in ascending order. It is well-known that the Gauss-Legendre rule achieves exponential convergence for analytical functions. To make sure that the triangle-based quadrature yields the same level of accuracy as the rectangle-based quadrature, the nodes $\{\boldsymbol{r}_i\}_{i=1}^{N_0}$ and weights $\{w_i\}_{i=1}^{N_0}$ as used in Eqs. (3.5) and (3.6), have to be chosen in the following way,

$$\int_0^1 dx \int_0^1 f(x,y,0)dy = \sum_{j=1}^{8}\int_{I_j} f(\boldsymbol{r})dxdy$$

$$= \sum_{j=1}^{8} \int_{I_j} f(\boldsymbol{r}) dxdy = \sum_{j=1}^{8} \int_{I_1} f(g_j(\boldsymbol{r})) dxdy \approx \sum_{j=1}^{8} \sum_{i=1}^{N_0} f(g_j(\boldsymbol{r}_i)) w_i, \tag{3.16}$$

where the approximation must agree with the result in Eq. (3.15). Consequently, $\boldsymbol{r}_i$ and w_i satisfy

$$\sum_{k=1}^{\tilde{m}} \sum_{l=1}^{\tilde{m}} f(x_k, x_l, 0) \tilde{w}_k \tilde{w}_l = \sum_{j=1}^{8} \sum_{i=1}^{N_0} f(g_j(\boldsymbol{r}_i)) w_i, \tag{3.17}$$

and a special solution to (3.17) for $\boldsymbol{r}_i$ and w_i is given as follows.

For simplicity, we assume that $\tilde{m}$ is odd so that $x_{(\tilde{m}+1)/2} = 0.5$. The nodes $\{\boldsymbol{r}_i\}_{i=1}^{N_0}$ can be directly selected from the set $\mathcal{A} := \{[x_k, x_l, 0]^T | 1 \le k, l \le \tilde{m}\}$ by choosing those lying in $\overline{I_1}$ as the desired nodes. We have

$$\{\boldsymbol{r}_i\}_{i=1}^{N_0} = \mathcal{A}_1 := \{[x_k, x_l, 0]^T | 1 \le k \le (\tilde{m}+1)/2, 1 \le l \le k\},$$

where the elements of $\mathcal{A}_1$ are arranged in such an order that $\boldsymbol{r}_i = [x_k, x_l, 0]^T$ for some $i = i(k,l)$, yielding $N_0 = |\mathcal{A}_1| = (\tilde{m}+1)(\tilde{m}+3)/8$. To find $w_{i(k,l)}$ for any point $\boldsymbol{r}_{i(k,l)} = [x_k, x_l, 0]^T \in \mathcal{A}_1$, we equate the coefficients of $f(x_k, x_l, 0)$ in both sides of Eq.(3.17), yielding

$$\{w_i\}_{i=1}^{N_0} = \mathcal{W}_1 := \{\tilde{w}_k \tilde{w}_l / p_{kl} | 1 \le k \le (\tilde{m}+1)/2, 1 \le l \le k\},$$

Here, p_{kl} is the number of pairs $(j, \boldsymbol{r}_i)$ that satisfy $g_j(\boldsymbol{r}_i) = \boldsymbol{r}_{i(k,l)}$ for $1 \le j \le 8$ and $\boldsymbol{r}_i \in \mathcal{A}$. By the definitions of g_j, it is easy to find that p_{kl} is equal to the number of triangles among the eight sharing the same $\boldsymbol{r}_{i(k,l)}$. Thus,

$$p_{kl} = \begin{cases} 2 & 1 \le k = l < (\tilde{m}+1)/2, \\ 2 & k = (\tilde{m}+1)/2, 1 \le l < (\tilde{m}+1)/2, \\ 8 & k = (\tilde{m}+1)/2, l = (\tilde{m}+1)/2, \\ 1 & \text{otherwise.} \end{cases}$$

3.3.2 How small can the number of nodes N_0 be?

In section 3.2, we use Gauss-Legendre rules, instead of trapezoidal rules, to approximate surface integrals in (3.3) and (3.4) over I_1. The Gauss-Legendre rule requires the measurement of $\boldsymbol{g}(g_j(\boldsymbol{r}_i))$ for $1 \le i \le N_0$ and $1 \le j \le 48$. The measured points $\{g_j(\boldsymbol{r}_i) | 1 \le j \le 8, \boldsymbol{r}_i \in \mathcal{A}_1\}$ actually constitute the 2-D Legendre mesh points in all six faces. The trapezoidal rule requires the measurement of $\boldsymbol{g}(\boldsymbol{r})$ at certain uniform mesh points in $\{\boldsymbol{q}_l\}_{l=1}^{N}$ lying on all six faces. Since the Gauss-Legendre quadrature rule enjoys exponential accuracy when approximating a surface integral with analytic integrand, we check a specific example to show the relation between the accuracy and the number of nodes N_0.

Let the unknown domain D be a sphere centered at $\boldsymbol{c}_0 = (1/2, 1/2, 1/2)^T$ with radius $r_0 = 0.4$. We have

$$\boldsymbol{g}(\boldsymbol{r}) = \int_D \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p}) d\boldsymbol{p} = \int_0^{r_0} dr \int_0^{\pi} d\theta \int_0^{2\pi} d\phi \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{p})|_{\boldsymbol{p}=\boldsymbol{p}(r,\theta,\phi)},$$

where $\boldsymbol{p}(r, \theta, \phi) = \boldsymbol{c}_0 + (r \sin\theta \cos\phi, r \sin\theta \sin\phi, r \cos\phi)$. A multi-Gaussian quadrature rule can be applied to compute $\boldsymbol{g}(\boldsymbol{r})$ for any $\boldsymbol{r} \in \partial\Omega$ due to the smooth integrand. Now we compute

$\boldsymbol{g}(\boldsymbol{r})$ at two different sets of points on the bottom face $z = 0$: one set consists of uniform points $\mathcal{U}_n = \{(ih, jh, 0)^T | 0 \le i, j \le n-1\}$, and the other consists of Legendre points $\mathcal{L}_{\tilde{m}} = \{[x_k, x_l, 0]^T | 1 \le k, l \le \tilde{m}\}$. To verify whether $\mathcal{L}_{\tilde{m}}$ suffices for constructing the gravimetry data, we check whether $\boldsymbol{g}(\boldsymbol{r})$ for $\boldsymbol{r} \in \mathcal{U}_n$ can be recovered from $\{\boldsymbol{g}(\boldsymbol{r}_0) | \boldsymbol{r}_0 \in \mathcal{L}_{\tilde{m}}\}$ with sufficient accuracy.

We employ the 2-D Lagrange interpolation formula to approximate $\boldsymbol{g}(\boldsymbol{r})$ by

$$\boldsymbol{g}_{\tilde{m}}(\boldsymbol{r}) = \sum_{\boldsymbol{r}_0 \in \mathcal{L}_{\tilde{m}}} I_{\boldsymbol{q}_0}(\boldsymbol{r}) \boldsymbol{g}(\boldsymbol{r}_0),$$

where $\boldsymbol{r} = [x, y, 0]^T, 0 \le x, y \le 1$, and $I_{\boldsymbol{r}_0}(\boldsymbol{r})$ is the Lagrange cardinal basis polynomial of x and y satisfying $I_{\boldsymbol{r}_0}(\boldsymbol{r}) = 0$ for all $\boldsymbol{r} \in \mathcal{L}_{\tilde{m}}$ except when $\boldsymbol{r} = \boldsymbol{r}_0$ in which case $I_{\boldsymbol{r}_0}(\boldsymbol{r}_0) = 1$. We define the relative error

$$\mathcal{E}_{\tilde{m},n} = \frac{\max_{\boldsymbol{r} \in \mathcal{U}_n} |\boldsymbol{g}_{\tilde{m}}(\boldsymbol{r}) - \boldsymbol{g}(\boldsymbol{r})|}{\max_{\boldsymbol{r} \in \mathcal{U}_n} |\boldsymbol{g}(\boldsymbol{r})|},$$

and take n to be 129. The relation between $\mathcal{E}_{\tilde{m},n}$ and the number of one-dimensional Legendre nodes $\tilde{m}$ is shown in Fig. 3.4, where the vertical axis represents the relative error $\mathcal{E}_{\tilde{m},n}$ in logarithmic

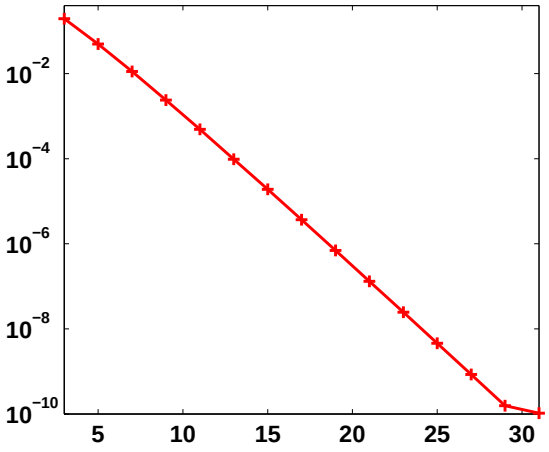


Figure 3.4: The relation between $\mathcal{E}_{\tilde{m},129}$ (the vertical axis) and $\tilde{m}$ (the horizontal axis).

scale and the horizontal axis represents $\tilde{m}$ varying from 3 to 31 with the step size 2. We can see that the error decays exponentially. In particular, when $\tilde{m} = 15$, the relative error $\mathcal{E}_{15,129} \sim 9^{-5}$ which may be considered to be accurate enough since the practical data may be polluted with higher level of noise. Numerical examples studied in this paper will show that taking $\tilde{m} = 15$ and $N_0 = (\tilde{m} + 1)(\tilde{m} + 3)/8 = 36$ can give satisfactory results.

4 Numerical continuation

In the previous section, we concluded that if the gravity field $\boldsymbol{g}$ is fully measured at the 2-D Legendre mesh points in each of the six faces of Ω, a linear complexity algorithm can be developed. The

Legendre measurement data set requires much less information than the uniform measurement data set does. However, in practical surveys, measurement data are always collected at uniform mesh points lying on only one of the six faces of the unit cube Ω. It was illustrated in [10] that a partial measurement data set may not be able to illuminate the targeted object completely. Therefore, a proper numerical continuation method to reconstruct a full measurement data from a given partial measurement data was used in [10]. Underlying this method is the fact that the gravity field can be represented by a single-layer potential due to some unknown single-layer density function on a closed surface Γ enclosing the target domain D. Since the density function is smooth and periodic on Γ, it is feasible to apply the spectral expansion to approximate the density function. Motivated by this, following [10], we develop an improved numerical continuation method for both 2-D and 3-D cases. Before detailing the method, we give a brief sketch of the basic idea.

4.1 Brief sketch of numerical continuation

As shown in Fig. 4.1, suppose we are given measurement data $\boldsymbol{g}$ on a portion of $\Gamma_0 = \partial\Omega$, denoted

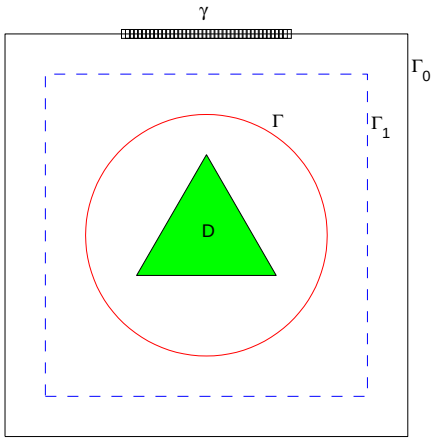


Figure 4.1: Illustration of numerical continuation.

by γ, that is,

$$\nabla_{\boldsymbol{r}} u(\boldsymbol{r}; \chi_D) = \int_D \nabla_{\boldsymbol{r}} K(\boldsymbol{r}, \boldsymbol{q}) d\boldsymbol{q} = \boldsymbol{g}(\boldsymbol{r}), \quad \boldsymbol{r} \in \gamma. \tag{4.1}$$

Clearly, u satisfies the Laplace equation outside the to-be-determined domain D. Thus, we represent u by a single layer potential for the Laplace operator defined on some surface Γ enclosing D, i.e.,

$$u(\boldsymbol{r}) = \int_\Gamma K(\boldsymbol{r}, \boldsymbol{q}) \psi(\boldsymbol{q}) ds(\boldsymbol{q}), \tag{4.2}$$

where f is the potential density function to be determined on Γ. For example, we can choose Γ to be a circle for the 2-D case or a sphere for the 3-D case. Combining Eqs. (4.1) and (4.2), we have

$$\int_\Gamma \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{q})\psi(\boldsymbol{q})ds(\boldsymbol{q}) = \boldsymbol{g}(\boldsymbol{r}), \tag{4.3}$$

where $\boldsymbol{r} \in \gamma$.

As is well-known, Eq.(4.3) is an ill-posed Fredholm integral equation of the first kind. To find ψ, we apply a proper quadrature rule to discretize the above integral over Γ, collocate $\boldsymbol{r}$ at the given measurement points on γ, and end up with the following linear system

$$\boldsymbol{L}\boldsymbol{\psi} = \boldsymbol{g}, \tag{4.4}$$

where $\boldsymbol{\psi}$ denotes the column vector for the values of $\psi(\boldsymbol{q})$ at the discretization points on Γ. Since $\boldsymbol{L}$ is not necessarily square and may be rank deficient, it is suitable to use its pseudo inverse to find $\boldsymbol{\psi}$. Thus, we compute the truncated SVD of $\boldsymbol{L}$ using threshold $\tilde{\epsilon}_{SVD}$, that is,

$$\boldsymbol{L} \approx \tilde{\boldsymbol{U}}\tilde{\boldsymbol{S}}\tilde{\boldsymbol{V}}^T, \tag{4.5}$$

where $\tilde{\boldsymbol{S}}$ is the diagonal matrix with the dominant singular values greater than $\tilde{\epsilon}_{SVD}$, and $\tilde{\boldsymbol{U}}$ and $\tilde{\boldsymbol{V}}$ are two unitary matrices. We thus obtain $\boldsymbol{\psi} = \boldsymbol{L}^+\boldsymbol{g}$ where $\boldsymbol{L}^+ = \tilde{\boldsymbol{V}}\tilde{\boldsymbol{S}}^{-1}\tilde{\boldsymbol{U}}^T$.

Once $\boldsymbol{\psi}$ is found, we can compute $\nabla_{\boldsymbol{r}} u(\boldsymbol{r};\chi_D)$ on some surface Γ_1 outside Γ by (4.3) and by the same quadrature rule to discretize the above integral over Γ for $\boldsymbol{r}$ belonging to the desired set of measurement points; for instance, we may take $\Gamma_1 = \partial\Omega$. Numerical results shown in [10] exhibit good performance of this method, where the trapezoidal rule is used to approximate the involved integral over Γ. The drawback of employing the trapezoidal rule is that a large number, denoted by $\tilde{N}$, of points are used in discretizing Γ to achieve the desired accuracy, giving rise to a matrix $\boldsymbol{L}$ with $\tilde{N}$ columns. Empirically, we take $\tilde{N} = \mathcal{O}(n^{m-1})$ so it is expensive to compute truncated SVD of $\boldsymbol{L}$ for large n. Therefore, we are motivated to develop faster and more accurate quadrature rules in the following for both 2-D and 3-D cases.

4.2 Two-dimensional case

For the 2-D case, suppose Γ is a circle centered at $\boldsymbol{c} = (1/2, 1/2)$ with radius r, represented by

$$x = 1/2 + r\cos\theta, \tag{4.6a}$$
$$y = 1/2 + r\sin\theta, \tag{4.6b}$$

with $0 < r < 1/2$ and $0 \le \theta \le 2\pi$. Thus, we can transform Eq.(4.3) into

$$\int_0^{2\pi} \tilde{K}(x,y,\theta)\psi(\theta)d\theta = -\frac{2\pi}{r}\boldsymbol{g}(x,y), \tag{4.7}$$

where, for simplicity, we denote

$$\psi(\theta) = \psi(1/2 + r\cos\theta, 1/2 + r\sin\theta),$$

$$\tilde{K}(x,y,\theta) = \frac{(x-1/2-r\cos\theta, y-1/2-r\sin\theta)}{(x-1/2-r\cos\theta)^2+(y-1/2-r\sin\theta)^2}.$$

As is well-known, since the integrand in (4.7) is analytic and 2π-periodic, the trapezoidal quadrature formula converges exponentially to the exact value as the number of discretization points increases. Observing that the measurement boundary γ and the artificial boundary Γ may not be well separated leading to a highly oscillatory function $\tilde{K}$ with respect to θ, trapezoidal rules with only a few discretization points may not give a very accurate approximation to the integral in (4.7), whereas more discretization points may result in a larger matrix $\boldsymbol{L}$. To resolve the problem, we use the spectral expansion as in [13].

We assume that $\psi(\theta)$ is smooth enough and 2π-periodic, since we may locate Γ to be away from the to-be-determined domain D. We approximate ψ by its trigonometric interpolation polynomial,

$$\psi(\theta) \approx \sum_{l=-\tilde{N}/2}^{\tilde{N}/2-1} \hat{\psi}_l e^{il\theta}, \tag{4.8}$$

where $\tilde{N}$ is a given even integer, and the Fourier coefficients $\hat{\psi}_l$ are approximated by

$$\hat{\psi}_l = \frac{1}{2\pi}\int_0^{2\pi} \psi(\theta)e^{-il\theta}d\theta \tag{4.9a}$$

$$\approx \frac{1}{\tilde{N}}\sum_{k=0}^{\tilde{N}-1} \psi(\theta_k)e^{-il\theta_k}, \quad l=-\tilde{N}/2,\cdots,\tilde{N}/2-1, \tag{4.9b}$$

where $\theta_k = 2k\pi/\tilde{N},\ k=0,\cdots,\tilde{N}-1$. Therefore, Eq.(4.7) becomes

$$\sum_{k=0}^{\tilde{N}-1}\left\{\sum_{l=-\tilde{N}/2}^{\tilde{N}/2-1}\tilde{K}_l(x,y)e^{-il\theta_k}\right\}\psi(\theta_k) = -\frac{2\tilde{N}\pi}{r}\boldsymbol{g}(x,y), \tag{4.10}$$

where the Fourier coefficient $\tilde{K}_l$ for each l is defined by

$$\tilde{K}_l(x,y) = \int_0^{2\pi}\tilde{K}(x,z,\theta)e^{il\theta}d\theta.$$

We still employ the trapezoidal rule to approximate $\tilde{K}_l$ for each l,

$$\tilde{K}_l(x,y) \approx \frac{2\pi}{\tilde{M}}\sum_{k=0}^{\tilde{M}-1}\tilde{K}(x,y,\tilde{\theta}_k)e^{il\tilde{\theta}_k},$$

where $\tilde{M}$ is the number of discretization points and $\tilde{\theta}_k = 2\pi k/\tilde{M},\ k=0,\cdots,\tilde{M}-1$. However, because of the possibly high oscillation of the function $\tilde{K}$, $\tilde{M}$ has to be chosen to be larger than $\tilde{N}$. The FFT (fast Fourier transform) can be employed to speed up the evaluation of those coefficients.

Now, we collocate (x,y) at the given measured points on γ, obtain a linear system as (4.4), where matrix $\boldsymbol{L}$ has much fewer columns. Therefore, by computing the pseudoinverse of $\boldsymbol{L}$, we can find $\boldsymbol{\psi}$ so as to reconstruct a fictitious full measurement data on Γ_1 as desired.

4.3 Three-dimensional case

For the 3-D case, suppose now Γ is a sphere centered at $\boldsymbol{c} = (1/2, 1/2, 1/2)$ with radius r, i.e., $\Gamma = \boldsymbol{c} + r\mathbb{S}^2$, where $\mathbb{S}^2$ is the unit sphere $\{\boldsymbol{r} \in \mathbb{R}^3 || \boldsymbol{r}| = 1\}$ and $0 < r < 1/2$. To approximate the smooth density function ψ defined on Γ, we need spherical harmonics $\{Y_l^m\}$, which are special solutions of the 3-D Laplace equation. Before detailing the quadrature rules, we introduce some properties of the spherical harmonics [5].

For a 3-D Helmholtz equation

$$\Delta u + k^2 u = 0, \tag{4.11}$$

with wavenumber k, its Green's function is

$$\Phi_k(\boldsymbol{r}, \boldsymbol{q}) = \frac{e^{ik|\boldsymbol{r}-\boldsymbol{q}|}}{4\pi|\boldsymbol{r}-\boldsymbol{q}|}, \quad \boldsymbol{r} \neq \boldsymbol{q}.$$

Let j_l and y_l be the spherical Bessel function and spherical Neumann function of order l, respectively, defined by

$$j_l(t) = \sum_{p=0}^{\infty} \frac{(-1)^p t^{l+2p}}{2^p p! 1 \cdot 3 \cdots (2l+2p+1)}, \tag{4.12}$$

and

$$y_l(t) = -\frac{(2l)!}{2^l l!} \sum_{p=0}^{\infty} \frac{(-1)^p t^{2p-l-1}}{2^p p!(-2l+1)(-2l+3)\cdots(-2l+2p-1)}, \tag{4.13}$$

and let $h_l^{(1)}$ be the first kind spherical Hankel function of order l defined by $h_l^{(1)} = j_l + iy_l$. We denote the normalized vector of $\boldsymbol{r}$ by $\hat{\boldsymbol{r}}$, i.e., $\hat{\boldsymbol{r}} = \boldsymbol{r}/|\boldsymbol{r}|$. We have the following lemmas [5].

Lemma 4.1. Eq.(2.30) in [5]: *Let Y_l^m, $m = -l, \cdots, l$ be any system of $2l+1$ orthogonal spherical harmonics of order l defined on $\mathbb{S}^2$. Then, for all $\hat{\boldsymbol{r}}, \hat{\boldsymbol{q}} \in \mathbb{S}^2$ we have*

$$\sum_{m=-l}^{l} Y_l^m(\hat{\boldsymbol{r}}) Y_l^{-m}(\hat{\boldsymbol{q}}) = \frac{2l+1}{4\pi} P_l(\hat{\boldsymbol{r}} \cdot \hat{\boldsymbol{q}}), \tag{4.14}$$

where P_l is the Legendre polynomial of order l defined on $[-1, 1]$.

This is the so-called *additional theorem*.

Lemma 4.2. Eq.(2.44) in [5]: *For any $\boldsymbol{r}$ satisfying $|\boldsymbol{r}| > r$, we have*

$$\frac{1}{ikr^2} \int_{|\boldsymbol{q}|=r} \Phi_k(\boldsymbol{r}, \boldsymbol{q}) Y_l^m(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) = j_l(kr) h_l^{(1)}(k|\boldsymbol{r}|) Y_l^m(\hat{\boldsymbol{r}}). \tag{4.15}$$

Considering (4.12) and (4.13), passing $k \to 0$ in Eq.(4.15) in Lemma 4.2 gives

$$\int_{|\boldsymbol{q}|=r} K(\boldsymbol{r}, \boldsymbol{q}) Y_l^m(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) = \frac{1}{2l+1} \frac{r^{l+2}}{|\boldsymbol{r}|^{l+1}} Y_l^m(\hat{\boldsymbol{r}}), |\boldsymbol{r}| > r. \tag{4.16}$$

Taking the gradient with respect to $\boldsymbol{r}$, we have

$$\int_{|\boldsymbol{q}|=r} \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{q}) Y_l^m(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) = \frac{r^{l+2}}{2l+1}\left(\frac{\nabla_{\boldsymbol{r}} Y_l^m(\hat{\boldsymbol{r}})}{|\boldsymbol{r}|^{l+1}} - (l+1)\frac{Y_l^m(\hat{\boldsymbol{r}})}{|\boldsymbol{r}|^{l+2}}\hat{\boldsymbol{r}}\right). \tag{4.17}$$

We are ready to discretize Eq. (4.3) now. By change of variables, Eq. (4.3) becomes

$$\int_{|\boldsymbol{q}|=r} \nabla_{\boldsymbol{r}} K(\boldsymbol{r},\boldsymbol{c}+\boldsymbol{q})\psi(\boldsymbol{c}+r\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) = \int_{|\boldsymbol{q}|=r} \nabla_{\bar{\boldsymbol{r}}} K(\bar{\boldsymbol{r}},\boldsymbol{q})\psi(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) = \boldsymbol{g}(\boldsymbol{r}), \quad \boldsymbol{r}\in\gamma, \tag{4.18}$$

where for convenience, $\psi(\hat{\boldsymbol{q}}) = \psi(\boldsymbol{c}+r\hat{\boldsymbol{q}})$ and $\bar{\boldsymbol{r}} = \boldsymbol{r}-\boldsymbol{c}$.

Since ψ is smooth on $\mathbb{S}^2$, we define a projection operator $Q_{\tilde{N}}$ that maps ψ onto a linear space $H_{\tilde{N}-1}$ of all spherical harmonics of order less than $\tilde{N}$,

$$Q_{\tilde{N}}\psi(\hat{\boldsymbol{q}}) := \frac{\pi}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} \alpha_j \psi(\boldsymbol{q}_{jk}) \sum_{l=0}^{\tilde{N}-1} \sum_{m=-l}^{l} Y_l^{-m}(\boldsymbol{q}_{jk}) Y_l^m(\hat{\boldsymbol{q}}),$$

where $\tilde{N}$ is a given integer,

$$\begin{aligned} \boldsymbol{q}_{jk} &= (\sin\theta_j\cos\phi_k, \sin\theta_j\sin\phi_k, \cos\theta_j), \\ \theta_j &= \cos^{-1}(t_j), \quad j = 1,\cdots,\tilde{N}, \\ \phi_k &= 2k\pi/\tilde{N}, \quad k = 0,\cdots,2\tilde{N}-1, \end{aligned}$$

$\{t_j\}_{j=1}^{\tilde{N}}$ and $\{\alpha_j\}_{j=1}^{\tilde{N}}$ are the associated nodes and weights of the standard $\tilde{N}$-points Gauss-Legendre quadrature rules. As shown in [5] and the references therein, $Q_{\tilde{N}}\psi(\hat{\boldsymbol{q}})$ can approximate ψ with exponentially convergent rate.

Thus, the left hand side of Eq. (4.18) can be approximated by

$$\begin{aligned} &\int_{|\boldsymbol{q}|=r} \nabla_{\bar{\boldsymbol{r}}} K(\bar{\boldsymbol{r}},\boldsymbol{q}) Q_{\tilde{N}}\psi(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) \\ &= \frac{\pi}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} \alpha_j \psi(\boldsymbol{q}_{jk}) \sum_{l=0}^{\tilde{N}-1} \sum_{m=-l}^{l} Y_l^{-m}(\boldsymbol{q}_{jk}) \int_{|\boldsymbol{q}|=r} \nabla_{\bar{\boldsymbol{r}}} K(\bar{\boldsymbol{r}},\boldsymbol{q}) Y_l^m(\hat{\boldsymbol{q}}) ds(\boldsymbol{q}) \\ &= \frac{\pi}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} \alpha_j \psi(\boldsymbol{q}_{jk}) \sum_{l=0}^{\tilde{N}-1} \sum_{m=-l}^{l} Y_l^{-m}(\boldsymbol{q}_{jk}) \frac{r^{l+2}}{2l+1}\left(\frac{\nabla_{\bar{\boldsymbol{r}}} Y_l^m(\hat{\bar{\boldsymbol{r}}})}{|\bar{\boldsymbol{r}}|^{l+1}} - (l+1)\frac{Y_l^m(\hat{\bar{\boldsymbol{r}}})}{|\bar{\boldsymbol{r}}|^{l+2}}\hat{\bar{\boldsymbol{r}}}\right) \\ &\quad \text{(by (4.17))} \\ &= \frac{\pi}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} \alpha_j \psi(\boldsymbol{q}_{jk}) \sum_{l=0}^{\tilde{N}-1} \frac{r^{l+2}}{(2l+1)|\bar{\boldsymbol{r}}|^{l+1}} \\ &\left(\nabla_{\bar{\boldsymbol{r}}} \sum_{m=-l}^{l} Y_l^m(\hat{\bar{\boldsymbol{r}}}) Y_l^{-m}(\boldsymbol{q}_{jk}) - \frac{(l+1)\hat{\bar{\boldsymbol{r}}}}{|\bar{\boldsymbol{r}}|} \sum_{m=-l}^{l} Y_l^m(\hat{\bar{\boldsymbol{r}}}) Y_l^{-m}(\boldsymbol{q}_{jk})\right) \end{aligned}$$

$$= \frac{\pi}{\tilde{N}} \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} \alpha_j \psi(\boldsymbol{q}_{jk}) \sum_{l=0}^{\tilde{N}-1} \frac{r^{l+2}}{(2l+1)|\bar{\boldsymbol{r}}|^{l+1}} \frac{2l+1}{4\pi} \left(\nabla_{\bar{\boldsymbol{r}}} P_l(\theta_{\boldsymbol{r}}^{jk}) - \frac{(l+1)\hat{\bar{\boldsymbol{r}}}}{|\bar{\boldsymbol{r}}|} P_l(\theta_{\boldsymbol{r}}^{jk}) \right)$$

(where $\theta_{\boldsymbol{r}}^{jk} = \hat{\bar{\boldsymbol{r}}} \cdot \boldsymbol{q}_{jk}$, by (4.14)) in Lemma 4.2

$$= \sum_{j=1}^{\tilde{N}} \sum_{k=0}^{2\tilde{N}-1} c_{jk}(\boldsymbol{r}) \psi(\boldsymbol{q}_{jk}) \approx \boldsymbol{g}(\boldsymbol{r}), \quad \boldsymbol{r} \in \gamma, \tag{4.19}$$

where the coefficients c_{jk} are

$$c_{jk}(\boldsymbol{r}) = \frac{\alpha_j}{4\tilde{N}} \sum_{l=0}^{\tilde{N}-1} \left[P_l'(\theta_{\boldsymbol{r}}^{jk})(\boldsymbol{q}_{jk} - \theta_{\boldsymbol{r}}^{jk} \hat{\bar{\boldsymbol{r}}}) - (l+1) P_l(\theta_{\boldsymbol{r}}^{jk}) \hat{\bar{\boldsymbol{r}}} \right] \left(\frac{r}{|\bar{\boldsymbol{r}}|} \right)^{l+2},$$

and P_l' is the derivative of P_l.

We now collocate $\boldsymbol{r}$ in Eq. (4.19) at the given measurement points on γ and assemble the linear system as Eq. (4.4). Then we compute the pseudoinverse of $\boldsymbol{L}$ with the threshold $\tilde{\epsilon}_{SVD}$ to find $\boldsymbol{\psi}$. Once $\boldsymbol{\psi}$ is found, we can compute the gravity field $\boldsymbol{g}(\boldsymbol{r})$ on the artificial hypersurface Γ_1 for $\boldsymbol{r}$ belonging to the desired set of measurement points to reconstruct the fictitious full measurement data.

5 Numerical Examples

We study a number of synthetic numerical examples for both 2-D and 3-D cases to demonstrate the performance of the improved local level set method.

5.1 Two-dimensional cases

Partial measurement at uniform mesh points is given on a part of the boundary $\partial\Omega$. Once the fictitious full measurement data is constructed by the improved numerical continuation method, we follow [10] to carry out the inversion process.

Throughout the 2-D examples, the initial level set function is chosen to be a circle as

$$\phi(\boldsymbol{r}) = r - ||\boldsymbol{r} - \boldsymbol{c}||_2, \tag{5.1}$$

which is centered at point $\boldsymbol{c} = (0.5, 0.5)^T$ with radius $r = 0.45$. Two different sets of partial measurements are considered:

1) the measurement is made on $\gamma = \{(x, y)^T : x \in [0.25, 0.75], y = 1\}$;

2) the measurement is made on $\gamma = \{(x, y)^T : x \in [0, 1], y = 1\}$.

The computational domain Ω is uniformly discretized by $N = n \times n$ points for $n = 129$ with grid size $1/(n-1) = 1/128$ in each direction. We choose those uniform mesh points lying on γ to be the measured points. To apply the improved numerical continuation method, we take the hypersurface Γ to be a circle centered at point $\boldsymbol{c}$ with radius 0.375 and the artificial boundary surface Γ_1 to be the boundary $\partial\Omega$. We discretize Γ uniformly by a fixed number $\tilde{N} = 30$ of points.

5.1.1 An elliptic target

Let the target domain D be the ellipse $a_1^{-2}(x-x_c)^2 + a_2^{-2}(y-y_c)^2 < 1$ where $a_1 = 0.3$, $a_2 = 0.15$ and $x_c = y_c = 0.5$. As shown in [9], page 99, we have the exact formula for computing the gravity field $\boldsymbol{g}$:

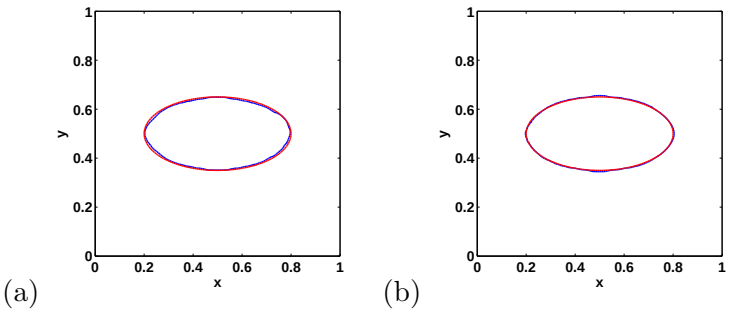
$$\boldsymbol{g}(\boldsymbol{r}) = u(\boldsymbol{r}; \chi_D) = -\frac{a_1 a_2}{e_0^2}\,(x - \tilde{x}, -y + \tilde{y})\,, \tag{5.2}$$

where $e_0 = \sqrt{a_1^2 - a_2^2}$, $\tilde{x}$ and $\tilde{y}$ are (signed) solutions to

$$(x-x_c)^2 - (y-y_c)^2 - e_0^2 = (\tilde{x}-x_c)^2 - (\tilde{y}-y_c)^2, \quad (x-x_c)(y-y_c) = (\tilde{x}-x_c)(\tilde{y}-y_c),$$

with $\text{sign}(\tilde{x} - x_c) = \text{sign}(x - x_c)$ and $\text{sign}(\tilde{y} - x_c) = \text{sign}(y - y_c)$.

Firstly, we assume that the gravity field $\boldsymbol{g}$ is measured through Eq. (5.2) at the two aforementioned measurement sets γ with 0% noise (clean measurement). Numerical results are shown in Fig. 5.1, where the elliptic target is plotted in red centered at $(x, y) = (0.5, 0.5)^T$ and numerical



(a) (b)

Figure 5.1: (Elliptic target) Numerical results based on applying the improved numerical continuation method to the partial measurement data with 0% noise made on (a): $\{(x, y)^T : x \in [0.25, 0.75], y = 1\}$; (b): $\{(x, y)^T : x \in [0, 1], y = 1\}$ (Red: exact solution. Blue: Numerical solution).

solutions are plotted in blue. To produce Fig. 5.1, we take the threshold $\tilde{\epsilon}_{SVD}$ to be $10^{-7}\sigma_{\max}$ to compute the pseudoinverse of matrix $\boldsymbol{L}$ in Eq. (4.4), where $\sigma_{\max}$ is the largest singular value of $\tilde{\boldsymbol{S}}$; we perform 400 iterations in Algorithm 2.1 to generate a steady state solution. We can see from Fig. 5.1 that numerical inversions match with the exact solution very well.

Secondly, to check the stability of the improved numerical continuation method, we consider a data set polluted with noise. We assume that the gravity field $\boldsymbol{g}$ is measured through Eq. (5.2) at the measurement set $\gamma = \{(x, y)^T : x \in [0, 1], y = 1\}$ and is further polluted with 10% Gaussian noise. Numerical result is shown in Fig. 5.2, where the elliptic target is plotted in red centered at $(x, y) = (0.5, 0.5)$ and the numerical solution is plotted in blue. To produce Fig. 5.2, we set the threshold $\tilde{\epsilon}_{SVD}$ to be $3 \times 10^{-2}\sigma_{\max}$ in constructing the pseudoinverse of matrix $\boldsymbol{L}$ in Eq. (4.4); we perform 400 iterations to generate a steady state solution. We can see from Fig. 5.2 that the numerical inversion matches with the exact solution reasonably well.

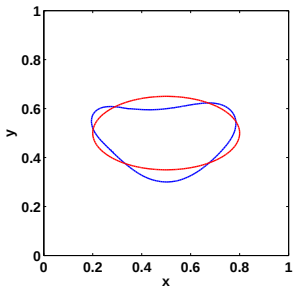
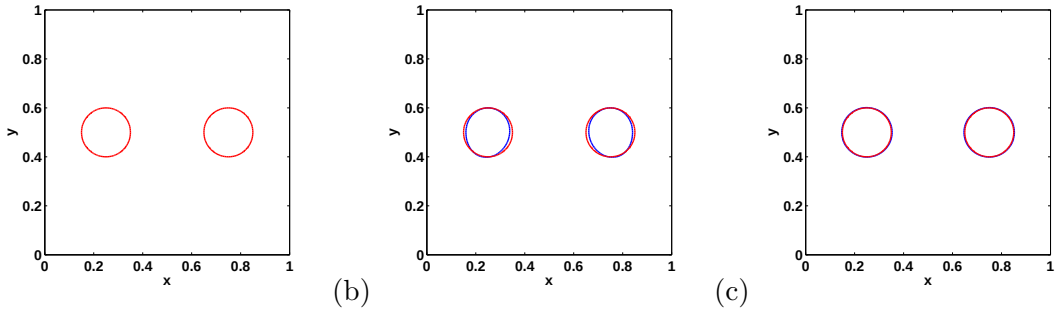


Figure 5.2: (Ellipsoidal target) Numerical result based on applying the improved numerical continuation method to partial measurement data with 10% noise made on $\{(x, y) : x \in [0, 1], y = 1\}$ (Red: exact solution. Blue: numerical solution).

5.1.2 A disjointed target

We next study a disjointed target. The target domain D consists of two disks D_1 and D_2 with the same radius $r = 0.1$ and centered at $\boldsymbol{c}_1 = (0.25, 0.5)^T$ and $\boldsymbol{c}_2 = (0.75, 0.5)^T$, respectively, as shown in Fig. 5.3(a). In fact, since the exact gravity field for either $u(\cdot; \chi_{D_i})$ can be obtained by



(a) (b) (c)

Figure 5.3: (Two disks) (a): Exact solution; Numerical results on applying the improved numerical continuation method to partial measurement data with 0% noise made on (b): $\{(x, y) : x \in [0.25, 0.75], y = 1\}$; (c): $\{(x, y) : x \in [0, 1], y = 1\}$ (Red: exact solution. Blue: numerical solution).

Eq. (5.2), the exact formula for the gravity field $\boldsymbol{g} = u(\cdot; \chi_{D_1}) + u(\cdot; \chi_{D_2})$ is still available. We omit the details here.

Firstly, we assume that the gravity field $\boldsymbol{g}$ is measured through the exact formula at the two aforementioned sets γ with 0% noise (clean measurement). Numerical results are shown in Figs. 5.3(b-c), where the two-disk target is plotted in red and numerical solutions are plotted in blue. To produce Figs. 5.3(b-c), we set the threshold $\tilde{\epsilon}_{\text{SVD}}$ to be $10^{-7}\sigma_{\max}$ to compute the pseudoinverse of matrix $\boldsymbol{L}$ in Eq. (4.4), and we perform 800 iterations to generate a steady state solution. We can see from Fig. 5.3 that numerical inversions match with the exact solution very well.

Secondly, we assume that the gravity field $\boldsymbol{g}$ is measured through the exact formula at the top side $\gamma = \{(x, y)^T : x \in [0, 1], y = 1\}$ and is further polluted with 10% Gaussian noise. Numerical

result is shown in Fig. 5.4, where the two-disk target is plotted in red and the numerical solution

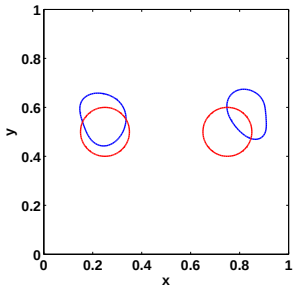


Figure 5.4: (Two disks) Numerical result on applying numerical continuation method to partial measurement data with 10% noise made on $\{(x, y) : x \in [0, 1], y = 1\}$ (Red: exact solution. Blue: numerical solution).

is plotted in blue. To produce Fig. 5.4, we set the threshold $\tilde{\epsilon}_{\text{SVD}}$ to be $3 \times 10^{-3}\sigma_{\max}$ in computing the pseudoinverse of matrix $\boldsymbol{L}$ in (4.4); we use 800 iterations to generate a steady state solution. We can see from Fig. 5.4 that the numerical inversion matches with the exact solution reasonably well.

5.2 Three-dimensional Cases

For 3-D cases, we require that the full measurement data for the gravity field $\boldsymbol{g}$ be given at the 2-D Legendre mesh points rather than the uniform mesh points, on each of the six faces of Ω. When applying the improved numerical continuation method, we assume that the partial measurement data for the gravity field is provided at the 2-D Legendre mesh points on the top face of Ω: $[0, 1] \times [0, 1] \times \{z = 1\}$ only, and we reconstruct the fictitious full measurement data at the 2-D Legendre mesh points on each of the six faces of Ω again.

Throughout the 3-D examples, we initialize the level set function as

$$\phi(\boldsymbol{r}) = r - ||\boldsymbol{r} - \boldsymbol{c}||,$$

representing a sphere centered at $\boldsymbol{c} = (0.5, 0.5, 0.5)$ with radius $r = 0.35$. We discretize the computational domain Ω by $N = n \times n \times n$ uniform mesh points with the grid size $h = 1/(n - 1)$ in each direction in three different cases: 1) $n = 33$, 2) $n = 65$, and 3) $n = 129$. We choose $\tilde{m}^2 = 15 \times 15$ 2-D Guass-Legendre mesh points to discretize each face so that $N_0 = 36$.

To produce the synthetic partial or full measurement Legendre data set for the gravity field $\boldsymbol{g}$, we compute $\boldsymbol{g}(\boldsymbol{r})$ for $\boldsymbol{r}$ belonging to the desired set of measurement points by Eq.(2.4b); the volume integral over Ω is approximated by the N-points trapezoidal rule so that different values of N give different levels of accuracy for the measurement data. To apply the improved numerical continuation method, we take the hypersurface Γ to be a circle centered at $\boldsymbol{c}$ with radius $r = 0.4$, and take the artificial boundary surface Γ_1 to be the boundary $\partial\Omega$. When constructing the matrix $\boldsymbol{L}$ by Eq. (4.19), we take $\tilde{N} = 26$ so that the matrix $\boldsymbol{L}$ has $2 \times \tilde{N}^2 = 1352$ columns.

5.2.1 3-D Ellipsoidal Target

The first example is an ellipsoid target. Let D be the 3-D ellipsoid:

$$\frac{(x-0.5)^2}{0.4^2}+\frac{(y-0.5)^2}{0.3^2}+\frac{(z-0.5)^2}{0.2^2}<1.$$

Its shapes in different resolutions are shown in Figs. 5.5(a-c), for $n = 33,\ 65$ and 129, respectively. We consider three different measurement data sets:

- Case 1: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is unpolluted (clean full measurement data);
- Case 2: the measurement surface $\Gamma_0 = [0,1]\times[0,1]\times\{z=1\}$; the gravimetry data is clean (clean partial measurement data);
- Case 3: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is polluted with 5% Gaussian noise (noisy full measurement data).

Numerical results for Case 1, clean full measurement data, are shown in Figs. 5.5 (d-f) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.5 (d-f), we perform 640 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 2, clean partial measurement data, are shown in Figs. 5.5 (g-i) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.5 (g-i), we first construct the fictitious full measurement data by the improved numerical continuation method and then perform 640 iterations in Algorithm 2.1 to obtain the steady state level set solution. To compute the pseudo-inverse of matrix $\boldsymbol{L}$, we use its 406, 461 and 461 largest singular values in the three different cases: $n = 33,\ 65$ and 129, respectively. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 3, noisy partial measurement data, are shown in Figs. 5.6(a-c) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.6(a-c), we perform 640 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution reasonably well.

5.2.2 Two spheres

The second example consists of two disjointed spheres with the same radius 0.1 and centered at two points $(0.3, 0.3, 0.3)$ and $(0.7, 0.7, 0.7)$, respectively. The two spheres in different resolutions are shown in Figs. 5.7(a-c), for $n = 33,\ 65$, and 129, respectively. We consider four measurement data sets:

- Case 1: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is unpolluted (clean full measurement data);
- Case 2: the measurement surface $\Gamma_0 = [0,1]\times[0,1]\times\{z=1\}$; the gravimetry data is unpolluted (clean partial measurement data);

- Case 3: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is polluted with 5% Gaussian noise (noisy full measurement data);

- Case 4: the measurement surface $\Gamma_0 = [0,1] \times [0,1] \times \{z = 1\}$; the gravimetry data is polluted with 5% Gaussian noise (noisy partial measurement data).

Numerical results for Case 1, clean full measurement data, are shown in Figs. 5.7 (d-f) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.7 (d-f), we perform 2400 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 2, clean partial measurement data, are shown in Figs. 5.7 (g-i) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.7 (g-i), we first construct the fictitious full measurement data by the improved numerical continuation method and then perform 2400 iterations in Algorithm 2.1 to obtain the steady state level set solution. To compute the pseudo-inverse of matrix $\boldsymbol{L}$, we use its 297, 413 and 429 largest singular values in the three different cases: $n = 33,\ 65$ and 129, respectively. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 3, noisy full measurement data, are shown in Figs. 5.8(a-c) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.8(a-c), we perform 2400 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution reasonably well.

Numerical results for Case 4, noisy partial measurement data, are shown in Figs. 5.8(d-f) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.8 (d-f), we first construct the fictitious full measurement data by the improved numerical continuation method and then perform 2400 iterations in Algorithm 2.1 to obtain the steady state level set solution. To compute the pseudo-inverse of matrix $\boldsymbol{L}$, we use its 46 largest singular values in all the three cases: $n = 33,\ 65$ and 129. We can see that numerical inversions match the exact solution reasonably well.

5.2.3 Two cubes

The last example consists of two disjointed cubes described by

$$\max\{|x - 0.3|, |y - 0.3|, |z - 0.3|\} \leq 0.1,$$

and

$$\max\{|x - 0.7|, |y - 0.7|, |z - 0.7|\} \leq 0.1, .$$

respectively. The two cubes in different resolutions are shown in Figs. 5.9(a-c), for $n = 33,\ 65$, and 129, respectively. We consider four different measurement data sets:

- Case 1: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is unpolluted (clean full measurement data);

- Case 2: the measurement surface $\Gamma_0 = [0,1] \times [0,1] \times \{z = 1\}$; the gravimetry data is unpolluted (clean partial measurement data);

- Case 3: the measurement surface $\Gamma_0 = \partial\Omega$; the gravimetry data is polluted with 5% Gaussian noise (noisy full measurement data);

- Case 4: the measurement surface $\Gamma_0 = [0,1] \times [0,1] \times \{z = 1\}$; the gravimetry data is polluted with 5% Gaussian noise (noisy partial measurement data).

Numerical results for Case 1, clean full measurement data are shown in Figs. 5.9 (d-f) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.9 (d-f), we perform 800 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 2, clean partial measurement data are shown in Figs. 5.9 (g-i) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.9 (g-i), we first construct the fictitious full measurement data by the improved numerical continuation method and then perform 800 iterations in Algorithm 2.1 to obtain the steady state level set solution. To compute the pseudo-inverse of matrix $\boldsymbol{L}$, we use its 313 largest singular values in all the three different cases: $n = 33,\ 65$ and 129. We can see that numerical inversions match the exact solution very well.

Numerical results for Case 3, noisy full measurement data are shown in Figs. 5.10(a-c) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.10(a-c), we perform 800 iterations in Algorithm 2.1 to obtain the steady state level set solution. We can see that numerical inversions match the exact solution reasonably well.

Numerical results for Case 4, noisy partial measurement data are shown in Figs. 5.10(d-f) for the three cases: $n = 33,\ 65$ and 129, respectively. To produce Figs. 5.10 (d-f), we first construct the fictitious full measurement data by the improved numerical continuation method and then perform 800 iterations in Algorithm 2.1 to obtain the steady state level set solution. To compute the pseudo-inverse of matrix $\boldsymbol{L}$, we use its 46 largest singular values in all the three cases: $n = 33,\ 65$ and 129. We can see that numerical inversions match the exact solution reasonably well.

5.2.4 Resolution and accuracy

We make some comments on the relation between resolution and accuracy to end this section. Since in practical surveys, the most common situation is that measurement data are collected on the top face $z = 1$, we use those numerical results for partial measurement data, such as Cases 2 and 4 in the two-sphere and two-cube examples, to illustrate the relation between the resolution and accuracy.

With the new algorithms at our disposal, the improved level set method is capable of dealing with fine meshes and computing high-resolution solutions. We can see that relevant numerical results for clean partial measurement data attain almost the same level of accuracy as those for clean full measurement data in all three different resolutions. This indicates that the partial measurement data with the help of the numerical continuation method suffices to infer the anomalies in different resolutions, provided that the partial measurement data are accurate enough. In fact, higher resolution requires higher accuracy in data.

On the other hand, for noisy partial measurement data, a finer mesh seems to give a worse, not better, numerical inversion. We assert that this "unreasonable" phenomenon is caused by the gravimetry problem itself, not by our algorithms. The reason is as follows: when the synthetic measurement data are unpolluted, they have different levels of accuracy for different resolutions; however, when they are polluted with a 5% Gaussian noise, the pollution can degrade the data accuracy in different resolutions to the same low level of accuracy; the inverse gravimetry problem is severely ill-posed and therefore for a data set with a low level of accuracy, a finer mesh may make

the accumulation of numerical errors more pronounced. Therefore, when the data do not meet the required accuracy, high-resolution inversions are not so useful. However, development in contemporary scientific instruments makes it possible to collect gravity data with high precision, which renders it necessary to develop rapid algorithms to carry out high-resolution inversion to delineate gravity anomalies with sharp resolution. Therefore, the improved level-set method proposed here will be valuable in carrying out such tasks rapidly and efficiently.

6 Conclusion

We proposed an improved fast local level set method for the inverse problem of gravimetry to recover open sets from their exterior volume potentials. To achieve this purpose, we developed two novel algorithms: one is of linear complexity designed for computing the Frechet derivative of the nonlinear domain inverse problem, and the other is designed for carrying out numerical continuation rapidly so as to obtain fictitious full measurement data from partial measurement. We studied a number of numerical experiments to exhibit the effectiveness of the proposed new algorithms. The improved level-set method is capable of dealing with fine meshes and computing high-resolution inversions, indicating its great potential in handling 3-D large-scale inverse gravimetry problems.

Acknowledgements

Leung's research is partially supported by the Hong Kong RGC under Grant GRF603011. Qian's research is partially supported by NSF.

References

[1] H. Bertete-Aguirre, E. Cherkaev, and M. Oristaglio. Non-smooth gravity problem with total variation penalization functional. *Geophys. J. Int.*, 149:499–507, 2002.

[2] M. Burger. A level set method for inverse problems. *Inverse Problems*, 17:1327–1356, 2001.

[3] M. Burger and S. Osher. A survey on level set methods for inverse problems and optimal design. *European J. Appl. Math.*, 16:263–301, 2005.

[4] T. Cecil, S. J. Osher, and J. Qian. Simplex free adaptive tree fast sweeping and evolution methods for solving level set equations in arbitrary dimension. *J. Comput. Phys.*, 213:458–473, 2006.

[5] D. Colton and R. Kress. *Inverse Acoustic and Electromagnetic Scattering Theory (3rd Edition)*. Springer, 2013.

[6] T. DeLillo, V. Isakov, N. Valdivia, and L. Wang. The detection of surface vibrations from interior acoustical pressure. *Inverse Problems*, 19:507–524, 2003.

[7] O. Dorn and D. Lesselier. Level set methods for inverse scattering. *Inverse Problems*, 22:R67–R131, 2006.

[8] S. Hou, K. Solna, and H.-K. Zhao. Imaging of location and geometry for extended targets using the response matrix. *J. Comput. Phys.*, 199:317–388, 2004.

[9] V. Isakov. *Inverse source problems.* American Mathematical Society, Providence, Rhode Island, 1990.

[10] V. Isakov, S. Leung, and J. Qian. A fast local level set method for inverse gravimetry. *Comm. in Computational Physics*, 10:1044–1070, 2011.

[11] V. Isakov, S. Leung, and J. Qian. A three-dimensional inverse gravimetry problem for ice with snow caps. *Inverse Problems and Imaging*, 7:523–544, 2013.

[12] A. Litman, D. Lesselier, and F. Santosa. Reconstruction of a 2-D binary obstacle by controlled evolution of a level-set. *Inverse Problems*, 14:685–706, 1998.

[13] W. Lu and Y. Y. Lu. Efficient boundary integral equation method for photonic crystal fibers. *Journal of Lightwave Technology*, 30(11):1610–1616, 2012.

[14] S. J. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79:12–49, 1988.

[15] J. Qian, L.-T. Cheng, and S. J. Osher. A level set based Eulerian approach for anisotropic wave propagations. *Wave Motion*, 37:365–379, 2003.

[16] J. Qian and S. Leung. A level set method for paraxial multivalued traveltimes. *J. Comput. Phys.*, 197:711–736, 2004.

[17] J. Qian and S. Leung. A local level set method for paraxial multivalued geometric optics. *SIAM J. Sci. Comp.*, 28:206–223, 2006.

[18] F. Santosa. A level-set approach for inverse problems involving obstacles. *Control, Optimizat. Calculus Variat.*, 1:17–33, 1996.

[19] K. van den Doel, U. Ascher, and A. Leitao. Multiple level sets for piecewise constant surface reconstruction in highly ill-posed problems. *J. Sci. Comput.*, 43:44–66, 2010.

[20] H.-K. Zhao, T. Chan, B. Merriman, and S. J. Osher. A variational level set approach for multiphase motion. *J. Comput. Phys.*, 127:179–195, 1996.

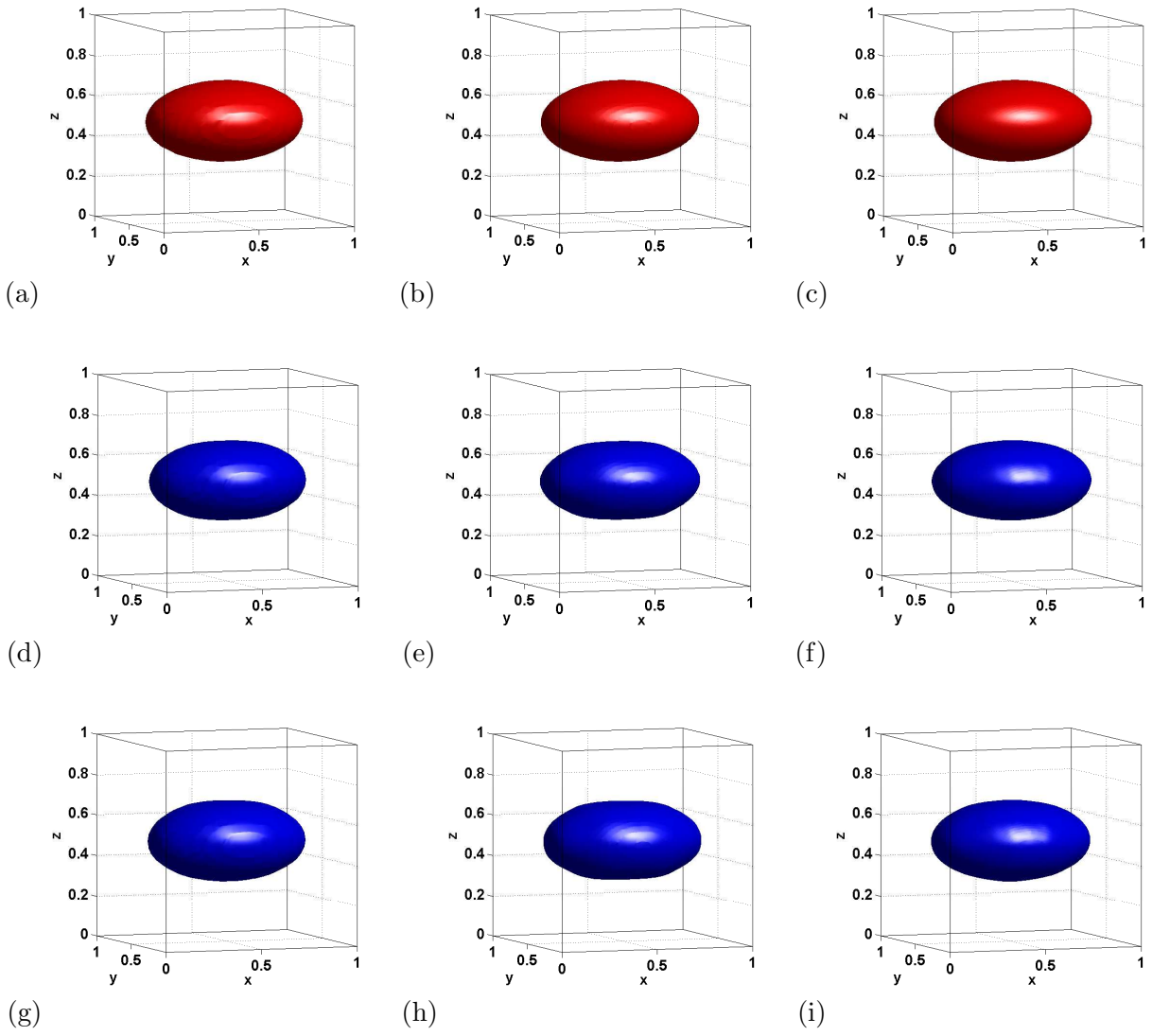
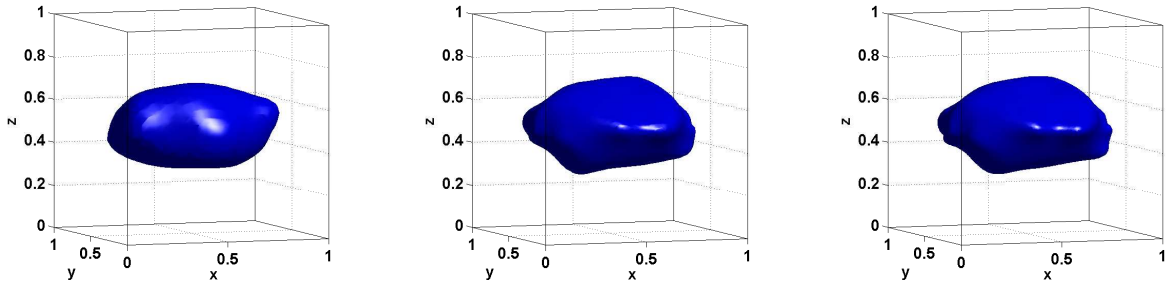


Figure 5.5: (3D ellipsoidal target) (a-c): exact solution in different resolutions; (d-f): numerical inversions for the clean full measurement data; (g-i): numerical inversions for the clean partial measurement data given on top face $z = 1$. Number of n used in: (a,d,g): 33; (b,e,h): 65; (c,f,i): 129.



(a) (b) (c)

Figure 5.6: (3D ellipsoidal target) (a-c): numerical inversions for the noisy full measurement data with 5% Gaussian noise. Number of n used in: (a): 33; (b): 65; (c): 129.

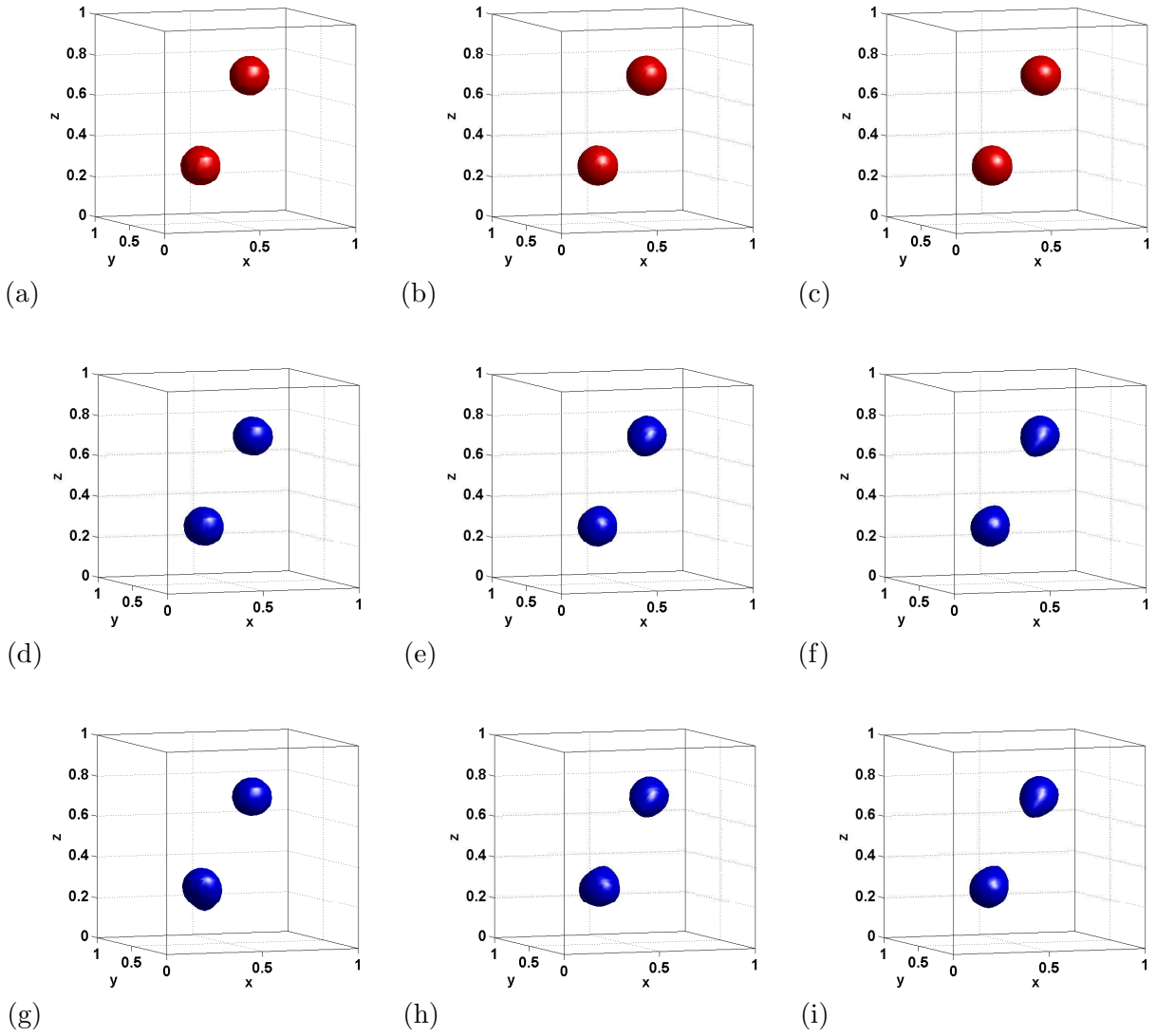


Figure 5.7: (Two spheres) (a-c): exact solution in different resolutions; (d-f): numerical inversions for the clean full measurement data; (g-i): numerical inversions for the clean partial measurement data given on top face $z = 1$. Number of n used in: (a,d,g): 33; (b,e,h): 65; (c,f,i): 129.

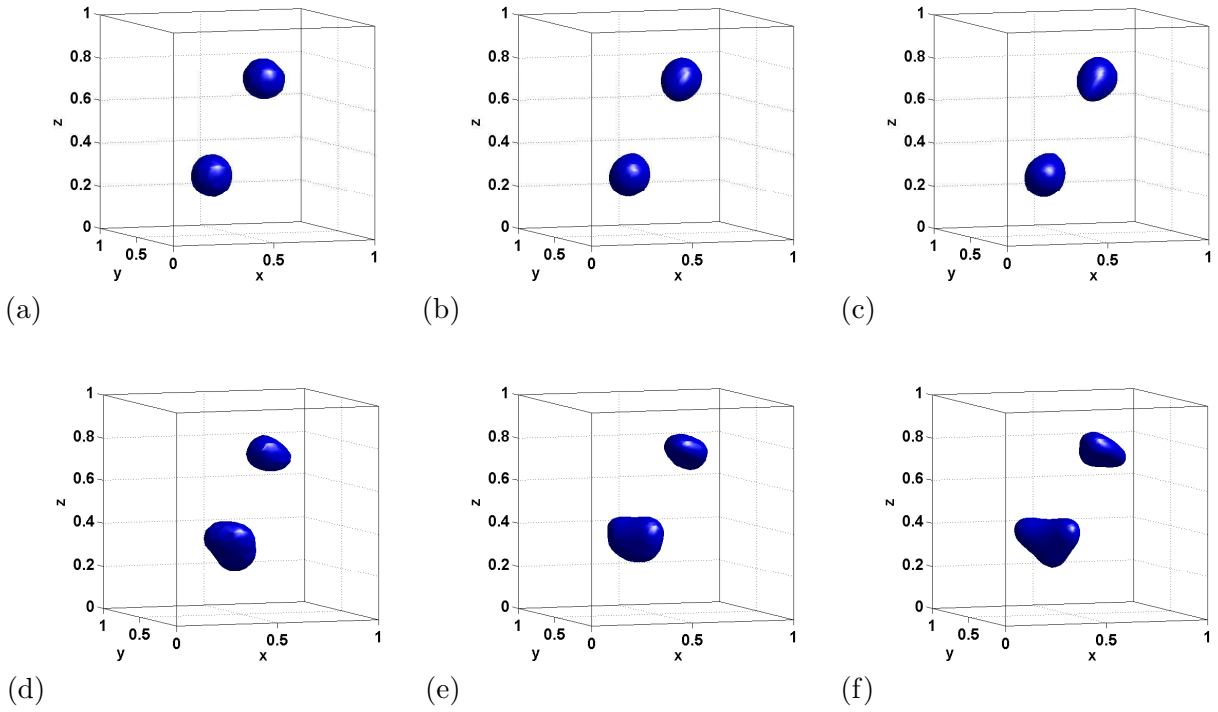


Figure 5.8: (Two spheres) (a-c): numerical inversions for the noisy full measurement data with 5% Gaussian noise; (d-f): numerical inversions for the noisy partial measurement data with 5% Gaussian noise given on top face $z = 1$. Number of n used in: (a,d): 33; (b,e): 65; (c,f): 129.

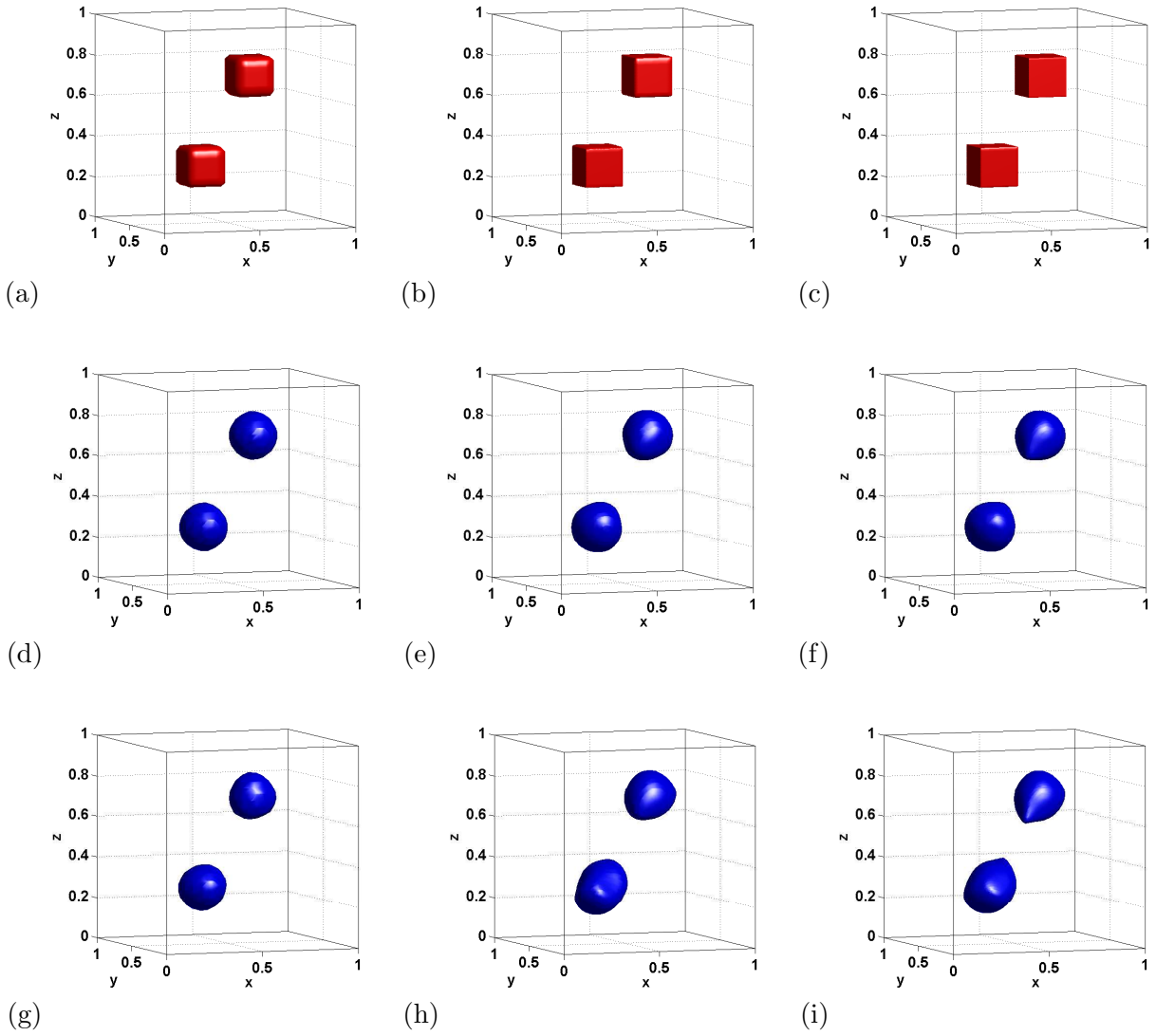


Figure 5.9: (Two cubes) (a-c): exact solution in different resolutions; (d-f): numerical inversions for the clean full measurement data; (g-i): numerical inversions for the clean partial measurement data given on top face $z = 1$. Number of n used in: (a,d,g): 33; (b,e,h): 65; (c,f,i): 129.

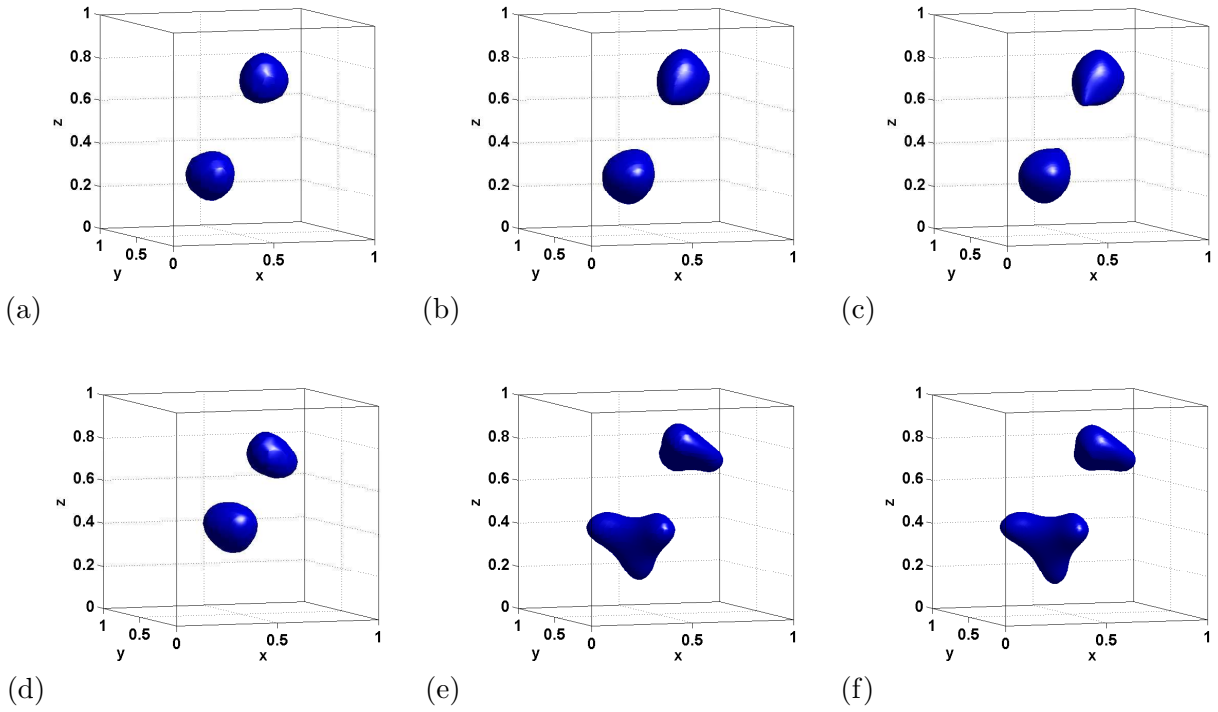


Figure 5.10: (Two cubes) (a-c): numerical inversions for the noisy full measurement data with 5% Gaussian noise; (d-f): numerical inversions for the noisy partial measurement data with 5% Gaussian noise given on top face $z = 1$. Number of n used in: (a,d): 33; (b,e): 65; (c,f): 129.