



**HAL**  
open science

## A Domain Decomposition Method for the Poisson-Boltzmann Solvation Models

Chaoyu Quan, Benjamin Stamm, Yvon Maday

► **To cite this version:**

Chaoyu Quan, Benjamin Stamm, Yvon Maday. A Domain Decomposition Method for the Poisson-Boltzmann Solvation Models. *SIAM Journal on Scientific Computing*, 2019, 10.1137/18M119553X . hal-02170834

**HAL Id: hal-02170834**

**<https://hal.sorbonne-universite.fr/hal-02170834>**

Submitted on 2 Jul 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A Domain Decomposition Method for the Poisson-Boltzmann Solvation Models

Chaoyu Quan<sup>1</sup>, Benjamin Stamm<sup>2,3</sup>, and Yvon Maday<sup>\*4,5</sup>

<sup>1</sup>Sorbonne Universités, UPMC Univ Paris 06, UMR 7598, Laboratoire Jacques-Louis Lions, and Institut des Sciences du Calcul et des Données, F-75005, Paris, France (quan@ann.jussieu.fr)

<sup>2</sup>Center for Computational Engineering Science, RWTH Aachen University, Aachen, Germany (best@mathcces.rwth-aachen.de)

<sup>3</sup>Computational Biomedicine, Institute for Advanced Simulation IAS-5 and Institute of Neuroscience and Medicine INM-9, Forschungszentrum Jülich, Germany

<sup>4</sup>Sorbonne Universités, UPMC Univ Paris 06, UMR 7598, Laboratoire Jacques-Louis Lions, and Institut Universitaire de France, 75005, Paris, France (maday@ann.jussieu.fr)

<sup>5</sup>Division of Applied Mathematics, Brown University, 182 George St, Providence, RI 02912, USA

## Abstract

In this paper, a domain decomposition method for the Poisson-Boltzmann (PB) solvation model that is widely used in computational chemistry is proposed. This method, called ddLPB for short, solves the linear Poisson-Boltzmann (LPB) equation defined in  $\mathbb{R}^3$  using the van der Waals cavity as the solute cavity. The Schwarz domain decomposition method is used to formulate local problems by decomposing the cavity into overlapping balls and only solving a set of coupled sub-equations in balls. A series of numerical experiments is presented to test the robustness and the efficiency of this method including the comparisons with some existing methods. We observe exponential convergence of the solvation energy with respect to the number of degrees of freedom which allows this method to reach the required level of accuracy when coupling with quantum mechanical descriptions of the solute.

**Keywords:** Implicit solvation model, Poisson-Boltzmann equation, domain decomposition method, spherical harmonic approximation

## 1 Introduction

The properties of numerous charged bio-molecules and their complexes with other molecules are dependent on the dielectric permittivity and the ionic strength of their environment. There are various methods to model ionic solution effects on molecular systems, which can be commonly

---

\*Accepted by SIAM Journal on Scientific Computing, December 5, 2018. Benjamin Stamm acknowledges the funding from the German Academic Exchange Service (DAAD) from funds of the “Bundesministeriums für Bildung und Forschung” (BMBF) for the project Aa-Par-T (project-id 57317909). Yvon Maday and Chaoyu Quan acknowledge the funding from the PICS-CNRS and the PHC PROCOPE 2017 (Project No.37855ZK). Chaoyu Quan acknowledges the financial support of the Fondation Sciences Mathématiques de Paris.

divided into two broad categories according to whether they employ an explicit or implicit solvation model. Explicit solvation models adopt molecular representations of both the solute and the solvent molecules, which produce accurate results, but are very expensive. Implicit solvation models adopt a microscopic treatment of the solute (with possibly a few solvent molecules), but characterize the solvent in terms of its macroscopic physical properties (for example, the solvent dielectric permittivity and the ionic strength). This reduces greatly the computational cost compared to an explicit description of the solvent. For this reason, implicit solvation models based on the Poisson-Boltzmann (PB) equation [1, 2] are now widely-used, taking into account both the solvent (relative) dielectric permittivity and the ionic strength. In this paper, we call these models the PB solvation models and we mention that the ESU-CGS (electrostatic units, centimetre-gram-second) system of units [3] is used for all equations.

For the sake of simplicity, we consider the linear Poisson-Boltzmann (LPB) equation, which describes the electrostatic potential  $\psi$  of the PB solvation model in the following form (see [2])

$$-\nabla \cdot [\varepsilon(\mathbf{x})\nabla\psi(\mathbf{x})] + \bar{\kappa}(\mathbf{x})^2\psi(\mathbf{x}) = 4\pi\rho_M(\mathbf{x}), \quad \text{in } \mathbb{R}^3, \quad (1.1)$$

where  $\varepsilon(\mathbf{x})$  represents the space-dependent dielectric permittivity function,  $\bar{\kappa}(\mathbf{x})$  is the modified Debye-Hückel parameter and  $\rho_M(\mathbf{x})$  represents the known solute's charge distribution. Usually,  $\varepsilon(\mathbf{x})$  has the following form

$$\varepsilon(\mathbf{x}) = \begin{cases} \varepsilon_1 & \text{in } \Omega, \\ \varepsilon_2 & \text{in } \Omega^c := \mathbb{R}^3 \setminus \bar{\Omega}, \end{cases} \quad (1.2)$$

where  $\varepsilon_1$  and  $\varepsilon_2$  are respectively the solute dielectric permittivity and the solvent dielectric permittivity,  $\Omega$  and  $\Omega^c$  represents respectively the solute cavity and the solvent region. Furthermore,  $\bar{\kappa}(\mathbf{x})$  usually has the following form

$$\bar{\kappa}(\mathbf{x}) = \begin{cases} 0 & \text{in } \Omega, \\ \sqrt{\varepsilon_2}\kappa & \text{in } \Omega^c, \end{cases} \quad (1.3)$$

where  $\kappa$  is the Debye-Hückel screening constant. More details on the nonlinear Poisson-Boltzmann (NPB) equation and its linearization will be presented in Section 2.

Finally, we also mention two popular implicit solvation models as particular cases: the polarizable continuum model (PCM) [4, 5, 6] and the conductor-like screening model (COSMO) [7]. In the classical PCM, the solvent is represented as a polarizable continuous medium which is non-ionic, i.e.,  $\kappa = 0$ . The COSMO is a reduced version of the PCM, where the solvent is represented as a conductor-like continuum. Both the PCM and the COSMO can be seen as two particular PB solvation models.

## 1.1 Previous work

We recall three widely used methods for solving the LPB equation: the boundary element method (BEM), the finite difference method (FDM) and the finite element method (FEM), see [8] for a review. As the names indicate, the BEM is based on solving an integral equation defined on the solute-solvent interface [9], while the FDM and the FEM are implemented in some 3-dimensional big domain covering the solute molecule.

In the BEM, the LPB equation is recast as some integral equations defined on the 2-dimensional solute-solvent boundary [1, 10, 11, 12]. To solve the integral equations, a surface mesh should be

generated, for example, using the MSMS [13] or the NanoShaper [14] etc. The BEM is efficient to solve the LPB equation and some techniques can be used to accelerate the BEM solvers, including the fast multipole method [15] and the hierarchical “treecode” technique [8]. For instance, the PAFMPB solver [16, 15] developed by Lu et al. provides a fast calculation of the solvation energy, which uses the adaptive fast multipole method and achieves linear complexity with respect to (w.r.t.) the number of mesh elements. Another interesting BEM solver, called TABI-PB [17], has been developed in the past several years, which uses the “treecode” technique. However, the BEM has a limitation that it can not be easily generalized to solve the NPB equation.

To solve the general PB equation (linear or nonlinear), the FDM might be the most popular method. Here, we list some successful FDM solvers: UHBD [18], DelPhi [19], MIBPB by Wei’s group [20] and APBS by Baker, Holst, McCammon et al. [21, 22, 23]. In particular, the APBS is well-developed with many useful options and its popularity is still increasing. In addition, there are some other contributions to the FDM for the PB equation [2, 24, 25, 19, 26]. In the FDM, a big box with grid is first taken, which covers the region of interest. Then, different types of boundary conditions can be chosen, such as zero, single Debye-Hückel, multiple Debye-Hückel and focusing boundary conditions (see the APBS documentation [21, 23]). We mention that the cost of FDM can increase considerably with respect to the grid dimension, for example, when the grid dimension is  $1000^3$  as mentioned in [8].

Comparing to the BEM and the FDM, the FEM provides in general more flexibility for mesh refinement, more analysis of convergence and more selections of linear solvers [8]. A rigorous solution and approximation theory of the FEM for the PB equation has been established in [27]. Furthermore, the adaptive FEM developed by Holst et al. has tackled some of the most important issues of the PB equation [28, 29, 27, 30, 31]. In addition, the SDPBS and SMPBS web servers developed by Xie et al. for solving the size-modified PB equation have performed fast and efficiently [32, 33, 34, 35, 36].

In addition to the above methods, we mention the framework of particular domain decomposition methods for implicit solvation models (see also website [37]). In the past several years, a domain decomposition method for COSMO (called ddCOSMO) has been developed [38, 39, 40, 41]. This method is independent on mesh and grid, easy to implement, and about two orders of magnitude faster than the state of the art as demonstrated in [40].

The ddCOSMO method can be coupled with a quantum Hamiltonian [40, 41] or a polarizable force-field within molecular dynamics [42]. Numerical tests of the method show linear scaling with respect to the number of atoms and first results of these scaling properties of the ddCOSMO in a simplified setting can be found in [43, 44]. Recently, a similar discretization scheme for the classical PCM was proposed within the domain decomposition paradigm (called ddPCM) [45, 46]. Both the ddCOSMO and the ddPCM work for the solute cavity constituted by overlapping balls, such as the van der Waals (VDW) cavity and the solvent accessible surface (SAS) cavity [47, 48]. In the case of the PCM based on the “smooth” molecular surface, i.e., based on the solvent excluded surface (SES) [49, 50], another domain decomposition method has been proposed in [51], which is called the ddPCM-SES.

Inspired by the previous work mentioned above, we develop a particular domain decomposition method for the PB solvation model (called ddLPB), to solve the LPB equation in  $\mathbb{R}^3$ .

## 1.2 ddLPB

In fact, the LPB equation (1.1) consists of a Poisson equation defined in the bounded solute cavity  $\Omega$  and a homogeneous screened Poisson (HSP) equation defined in the unbounded solvent region  $\Omega^c$ , which are coupled by some jump conditions on the interface  $\Gamma := \partial\Omega$ .

To solve this problem, we first transform the Poisson equation in (1.1) into the following Laplace equation of  $\psi_r := \psi - \psi_0$ ,

$$-\Delta\psi_r = 0, \quad \text{in } \Omega, \quad (1.4)$$

where  $\psi_r$  is called the reaction potential and  $\psi_0$  satisfies  $-\Delta\psi_0 = \frac{4\pi}{\varepsilon_1}\rho_M$  in  $\mathbb{R}^3$ . Then, according to the potential theory, the electrostatic potential  $\psi|_{\Omega^c}$  can be represented as a single-layer potential (an exterior Dirichlet problem), which simultaneously gives an extended potential  $\psi_e$  satisfying the following HSP equation defined now in  $\Omega$  (an interior Dirichlet problem)

$$-\Delta\psi_e(\mathbf{x}) + \kappa^2\psi_e(\mathbf{x}) = 0, \quad \text{in } \Omega. \quad (1.5)$$

Based on the classical jump-conditions (see Section 3) of  $\psi$  on the solute-solvent boundary, a coupling condition (see Figure 1) between the Laplace equation (1.4) and the extended HSP equation (1.5) arises through an auxiliary function  $g$  defined by

$$g = \mathcal{S}_\kappa \left( \partial_{\mathbf{n}}\psi_e - \frac{\varepsilon_1}{\varepsilon_2}\partial_{\mathbf{n}}(\psi_0 + \psi_r) \right), \quad \text{on } \Gamma, \quad (1.6)$$

where  $\mathcal{S}_\kappa : H^{-\frac{1}{2}}(\Gamma) \rightarrow H^{\frac{1}{2}}(\Gamma)$  denotes the single-layer operator on  $\Gamma$  ( $\mathcal{S}_\kappa$  is defined in Section 3). Here,  $H^{-\frac{1}{2}}(\Gamma)$  and  $H^{\frac{1}{2}}(\Gamma)$  denote the usual Sobolev spaces of order  $\pm\frac{1}{2}$  on  $\Gamma$ , see [52]. The initial problem defined in  $\mathbb{R}^3$  is therefore transformed into two equations (1.4)–(1.5) coupled through  $g$  in Eq. (1.6).

Considering the fact that the solute cavity is commonly modeled as a union of overlapping balls, a particular Schwarz domain decomposition method (called ddLPB) can be used to solve Eqs (1.4)–(1.5) by respectively solving a group of coupled sub-equations in balls. The main idea of this domain decomposition method is illustrated in Figure 1. Ultimately, only a Laplace solver and a HSP solver in the unit ball need to be developed for the local Laplace sub-equations and the local HSP sub-equations. Each solver uses the spectral method for the corresponding PDE, where the spherical harmonics are taken as basis functions in the angular direction of the spherical coordinate system.

The ddLPB provides a new discretization of the LPB equation and has its own features. In fact, this method is initially designed for quantum calculations, which usually require the accurate electrostatic solvation energy and the derivatives w.r.t. the atom positions. This method does not rely on mesh nor grid, but only on the Lebedev quadrature points [53] on 2-dimensional spheres. Therefore, it will be convenient to apply the ddLPB in molecular dynamics, without remeshing molecular surface as in the BEM. The computation of forces becomes also very natural as the spheres are centered at the nuclear positions. In the numerical tests, we will further show that the ddLPB is numerically robust and efficient.

## 1.3 Outline

In Section 2, we introduce the derivation of the PB equation as well as its linearization. Then, in Section 3, we transform the original LPB equation defined in  $\mathbb{R}^3$  into the Laplace equation and

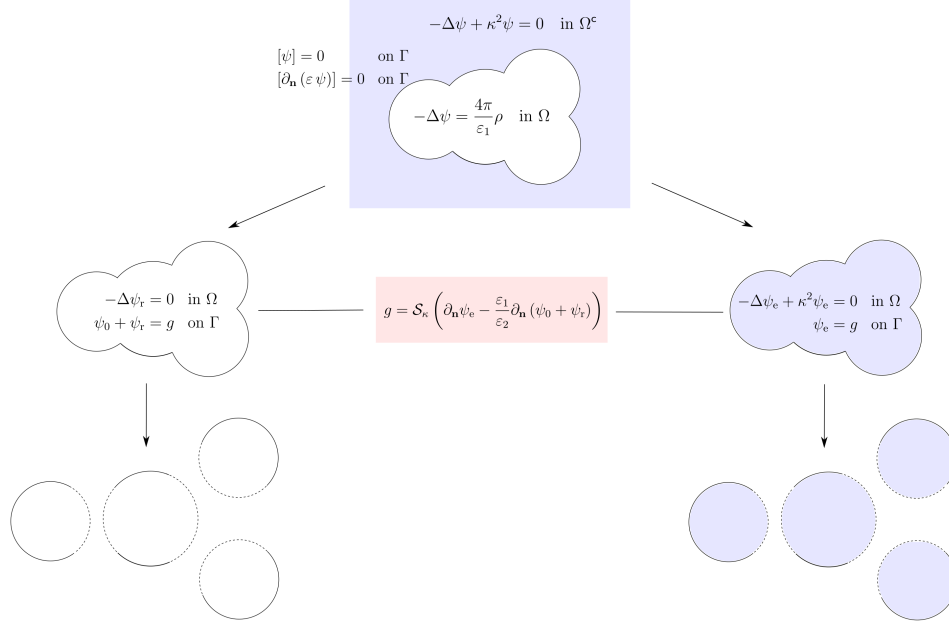


Figure 1: Schematic diagram of the ddLPB.

the HSP equation both defined in the bounded solute cavity, as briefly outlined above. We present a global strategy for solving the transformed problem. This strategy involves solving the Laplace equation and the HSP equation defined in the solute cavity, which will be presented in Section 4 using a particular domain decomposition method. In Section 5, we develop a Laplace solver and a HSP solver in the unit ball to solve the local equations. After that, in Section 6, we reformulate the coupling conditions and deduce a global linear system to be finally solved. In Section 7, we present some numerical results about the ddLPB. In the last section, we draw some conclusions.

## 2 PB solvation model

In this section, we introduce the well-known PB equation and its linearization, which describe the electrostatic potential in the implicit solvation model with ionic solutions.

The space  $\mathbb{R}^3$  is simply divided into the solute cavity and the solvent region, as introduced in Eqs (1.1) – (1.3). Three types of molecular surfaces are mostly used to define the solute-solvent interface: the VDW surface, the SAS and the SES. Both the VDW surface and the SAS are the boundary of the union of balls (respectively the VDW-balls and the SAS-balls), while the geometrical structure of SES is more complicated, see [50, 54] for a thorough characterization. In practice, the scaled VDW surface is often used, where each VDW-radius is multiplied by a scalar factor such as  $1.1 \sim 1.2$  which is a common approach. For the rest of this article we will limit the development to VDW-cavities. Note that without any further difficulty, the ddLPB method also work for the scaled VDW-cavity and the SAS-cavity.

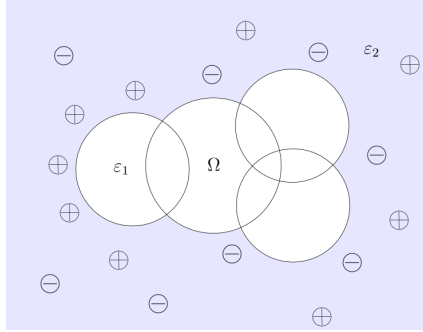


Figure 2: 2D schematic diagram of the implicit solvation model with ionic solutions, i.e., the PB solvation model.

## 2.1 Poisson-Boltzmann equation

In the PB solvation model, the solvent is represented by a polarizable and ionic continuum. The freedom of the ions to move in the solution is accounted for by Boltzmann statistics. That is to say, the Boltzmann equation is used to calculate the local ion density  $c_i$  of the  $i$ -th type of ion as follows

$$c_i = c_i^\infty \exp\left(\frac{-W_i}{k_B T}\right), \quad (2.1)$$

where  $c_i^\infty$  is the bulk ion concentration at an infinite distance from the solute molecule,  $W_i$  is the work required to move the  $i$ -th type of ion to a given position from an infinitely far distance,  $k_B$  is the Boltzmann constant,  $T$  is the temperature in Kelvins (K). The electrostatic potential  $\psi$  of a general implicit solvation model is described originally by the Poisson equation as follows

$$-\nabla \cdot \varepsilon(\mathbf{x}) \nabla \psi(\mathbf{x}) = 4\pi \rho(\mathbf{x}), \quad \text{in } \mathbb{R}^3, \quad (2.2)$$

where  $\psi(\mathbf{x}) = O(\frac{1}{|\mathbf{x}|})$  as  $|\mathbf{x}| \rightarrow \infty$ . Here,  $\varepsilon(\mathbf{x})$  represents the space-dependent dielectric permittivity and  $\rho(\mathbf{x})$  represents the charge distribution of the solvated system. Given the solute's charge distribution  $\rho_M$  and the ionic distribution  $c_i$  in (2.1), we can derive the PB equation from Eq. (2.2) as follows (see [24])

$$-\nabla \cdot [\varepsilon(\mathbf{x}) \nabla \psi(\mathbf{x})] = 4\pi \rho_M(\mathbf{x}) + \sum_i z_i e c_i^\infty \exp\left(\frac{-z_i e \psi(\mathbf{x})}{k_B T}\right) \chi_{\Omega^c}(\mathbf{x}), \quad (2.3)$$

where  $z_i e$  is the charge of the  $i$ -th type of ion,  $e$  is the elementary charge and  $\chi_{\Omega^c}$  is the characteristic function of the solvent region  $\Omega^c$ .

In the PB solvation model with a 1 : 1 electrolyte, there are two types of ions respectively with charge  $+e$  and  $-e$  (see Figure 2 for a schematic diagram). With the assumption that  $\psi$  satisfies the low potential condition, i.e.,  $\left|\frac{e\psi}{k_B T}\right| \ll 1$ , the NPB equation (2.3) can be linearized to (see [2] for this form)

$$-\nabla \cdot [\varepsilon(\mathbf{x}) \nabla \psi(\mathbf{x})] + \bar{\kappa}(\mathbf{x})^2 \psi(\mathbf{x}) = 4\pi \rho_M(\mathbf{x}), \quad (2.4)$$

where  $\psi$  is determined by the data  $\varepsilon(\mathbf{x})$ ,  $\bar{\kappa}(\mathbf{x})$  and  $\rho_M(\mathbf{x})$  that are introduced in Section 1.

**Remark 2.1.** *When the ionic solution has more than two types of ions, the nonlinear Poisson-Boltzmann equation can still be linearized to the form (2.4). In order to obtain a simpler expression of the modified Debye-Hückel parameter  $\bar{\kappa}(\mathbf{x})$ , we consider the 1 : 1 electrolyte in this paper.*

In the definition (1.2) of  $\varepsilon(\mathbf{x})$ , the solute (relative) dielectric permittivity  $\varepsilon_1$  should theoretically be set to 1 as in the vacuum (for example, in [9]). However, values different from  $\varepsilon_1 = 1$  might be used. For example, in [2], the authors claim to obtain better approximations with the empirical value  $\varepsilon_1 = 2$ . The solvent dielectric permittivity  $\varepsilon_2$  is determined by the solvent as well as the temperature, for example,  $\varepsilon_2 = 78.54$  for water at the room temperature 25°C. The modified Debye-Hückel parameter in the implicit solvation model with a 1 : 1 electrolyte is taken as

$$\bar{\kappa}(\mathbf{x}) = \begin{cases} 0 & \text{in } \Omega, \\ \sqrt{\varepsilon_2} \kappa & \text{in } \Omega^c, \end{cases} \quad (2.5)$$

where  $\kappa$  is the Debye-Hückel screening constant representing the attenuation of interactions due to the presence of ions in the solvent region, which is related to the ionic strength  $I$  of the ionic solution according to (see [2] and [55, Section 1.4] for the following formula)

$$\kappa^2 = \frac{8\pi e^2 N_A I}{1000 \varepsilon_2 k_B T}, \quad (2.6)$$

where  $N_A$  is the Avogadro constant.

Furthermore, it is usually assumed that the solute's charge distribution  $\rho_M$  is supported in  $\Omega$ . For example, for a classical description of the solute,  $\rho_M$  is given by the sum of  $M$  point charges in the following form

$$\rho_M(\mathbf{x}) = \sum_{i=1}^M q_i \delta(\mathbf{x} - \mathbf{x}_i), \quad (2.7)$$

where  $M$  is the number of solute atoms,  $q_i$  represents the (partial) charge carried on the  $i$ th atom with center  $\mathbf{x}_i$ ,  $\delta$  is the Dirac delta function. For a quantum description of the solute,  $\rho_M$  consists of a sum of classical nuclear charges and the electron charge density.

### 3 Problem transformation

In this section, we first introduce the integral representation of the LPB equation in the potential theory. Based on this, we then transform the original electrostatic problem to two coupled equations restricted to the (bounded) solute cavity.

#### 3.1 Problem setting

The LPB equation can be divided into two equations: first, the Poisson equation in the solute cavity and second, the HSP equation in the solvent region. That is to say, the problem is recast in the following form

$$\begin{cases} -\Delta\psi(\mathbf{x}) = \frac{4\pi}{\varepsilon_1} \rho_M(\mathbf{x}) & \text{in } \Omega, \\ -\Delta\psi(\mathbf{x}) + \kappa^2 \psi(\mathbf{x}) = 0 & \text{in } \Omega^c, \end{cases} \quad (3.1)$$



with two classical jump-conditions

$$\begin{cases} [\psi] = 0 & \text{on } \Gamma, \\ [\partial_{\mathbf{n}}(\varepsilon \psi)] = 0 & \text{on } \Gamma, \end{cases} \quad (3.2)$$

where  $\Gamma := \partial\Omega$  is the solute-solvent boundary,  $\mathbf{n}$  is the unit normal vector on  $\Gamma$  pointing outwards with respect to  $\Omega$  and  $\partial_{\mathbf{n}} = \mathbf{n} \cdot \nabla$  is the notation of normal derivative.  $[\psi]$  represents the jump (inside minus outside) of the potential and  $[\partial_{\mathbf{n}}(\varepsilon \psi)]$  represents the jump of the normal derivative of the electrostatic potential multiplied by the dielectric permittivity.

### 3.2 Necessary tools from the potential theory

The free-space Green's function of the operator  $-\Delta$  is given as

$$G(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^3, \quad (3.3)$$

and similarly, the free-space Green's function of the operator  $-\Delta + \kappa^2$  is given as

$$G_{\kappa}(\mathbf{x}, \mathbf{y}) = \frac{\exp(-\kappa|\mathbf{x} - \mathbf{y}|)}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^3, \quad (3.4)$$

which yields

$$-\Delta_{\mathbf{x}}G(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y}), \quad \forall \mathbf{y} \in \mathbb{R}^3, \quad (3.5)$$

and

$$-\Delta_{\mathbf{x}}G_{\kappa}(\mathbf{x}, \mathbf{y}) + \kappa^2G_{\kappa}(\mathbf{x}, \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y}), \quad \forall \mathbf{y} \in \mathbb{R}^3. \quad (3.6)$$

In the solute cavity  $\Omega$ , we define the reaction potential  $\psi_{\text{r}} := \psi - \psi_0$ , where  $\psi_0$  is the potential generated by  $\rho_{\text{M}}$  in vacuum written as

$$\psi_0 = \sum_{i=1}^M \frac{q_i}{\varepsilon_1|\mathbf{x} - \mathbf{x}_i|}, \quad (3.7)$$

satisfying  $-\Delta\psi_0 = \frac{4\pi}{\varepsilon_1}\rho_{\text{M}}$  in  $\mathbb{R}^3$ . Then,  $\psi_{\text{r}}$  is harmonic in  $\Omega$ , that is,

$$-\Delta\psi_{\text{r}} = 0, \quad \text{in } \Omega, \quad (3.8)$$

which yields the following integral equation

$$\psi_{\text{r}}(\mathbf{x}) = \tilde{\mathcal{S}}\sigma_{\text{r}}(\mathbf{x}) := \int_{\Gamma} \frac{\sigma_{\text{r}}(\mathbf{y})}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x} \in \Omega, \quad (3.9)$$

where  $\sigma_{\text{r}}$  is some function in  $H^{-\frac{1}{2}}(\Gamma)$  and  $\tilde{\mathcal{S}} : H^{-\frac{1}{2}}(\Gamma) \rightarrow H^1(\mathbb{R}^3 \setminus \Gamma)$  is the single-layer potential associated with  $G$ .

Furthermore, according to the HSP equation in (3.1), the electrostatic potential in the solvent region  $\Omega^c$  can be represented by

$$\psi|_{\Omega^c}(\mathbf{x}) = \tilde{\mathcal{S}}_{\kappa}\sigma_{\text{e}}(\mathbf{x}) := \int_{\Gamma} \frac{\exp(-\kappa|\mathbf{x} - \mathbf{y}|)}{4\pi|\mathbf{x} - \mathbf{y}|} \sigma_{\text{e}}(\mathbf{y}), \quad \forall \mathbf{x} \in \Omega^c, \quad (3.10)$$

where  $\sigma_e$  is another function in  $H^{-\frac{1}{2}}(\Gamma)$  and  $\tilde{\mathcal{S}}_\kappa : H^{-\frac{1}{2}}(\Gamma) \rightarrow H^1(\mathbb{R}^3 \setminus \Gamma)$  is the single-layer potential associated with  $G_\kappa$ . Here, we also introduce the single-layer operator  $\mathcal{S}_\kappa : H^{-\frac{1}{2}}(\Gamma) \rightarrow H^{\frac{1}{2}}(\Gamma)$  defined by

$$\mathcal{S}_\kappa \sigma_e(\mathbf{x}) := \int_\Gamma \frac{\exp(-\kappa|\mathbf{x} - \mathbf{y}|) \sigma_e(\mathbf{y})}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x} \in \Gamma, \quad (3.11)$$

which is an invertible operator (this is true according to the proof of the invertibility of the single-layer operator for the Helmholtz equation, see [56, Corollary 7.26] and [57, Theorem 3.9.1]). The invertibility of  $\mathcal{S}_\kappa$  implies that  $\sigma_e$  can be characterized as  $\sigma_e = \mathcal{S}_\kappa^{-1} \psi|_\Gamma$ .

### 3.3 Transformation

We will now transform the original problem defined in  $\mathbb{R}^3$  equivalently to two coupled equations both defined in the solute cavity.

According to the continuity of the single-layer potential  $\tilde{\mathcal{S}}_\kappa$  across the interface [57], we can artificially extend the electrostatic potential  $\psi|_{\Omega^c}$  from  $\Omega^c$  to  $\Omega$  as follows

$$\psi_e(\mathbf{x}) := \tilde{\mathcal{S}}_\kappa \sigma_e(\mathbf{x}) = \int_\Gamma \frac{\exp(-\kappa|\mathbf{x} - \mathbf{y}|) \sigma_e(\mathbf{y})}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x} \in \Omega, \quad (3.12)$$

where  $\psi_e$  is called the extended potential in this paper. As a consequence,  $\psi_e$  satisfies the same HSP equation as  $\psi|_{\Omega^c}$ , but defined on  $\Omega$ , as follows

$$-\Delta \psi_e(\mathbf{x}) + \kappa^2 \psi_e(\mathbf{x}) = 0, \quad \text{in } \Omega, \quad (3.13)$$

with the same Dirichlet boundary conditions on  $\Gamma$ . Furthermore, from [57, Theorem 3.3.1], we have a relation among  $\sigma_e$  and the normal derivatives of  $\psi_e$  and  $\psi|_{\Omega^c}$  on  $\Gamma$  as follows

$$\sigma_e = \partial_{\mathbf{n}} \psi_e|_\Omega - \partial_{\mathbf{n}} \psi|_{\Omega^c}, \quad \text{on } \Gamma. \quad (3.14)$$

As introduced above, Eq. (3.8) of  $\psi_r$  and (3.13) of  $\psi_e$  are two PDEs defined on  $\Omega$ , that are derived from the original LPB equation (3.1). As a consequence, it is sufficient to couple these two equations. According to  $[\psi] = 0$  on  $\Gamma$  and the continuity of  $\tilde{\mathcal{S}}_\kappa$  across  $\Gamma$  [57], we then deduce a first coupling condition

$$\psi_0 + \psi_r = \psi_e, \quad \text{on } \Gamma. \quad (3.15)$$

Further, combining Eq. (3.14) with the second equation of the jump conditions (3.2), i.e.,

$$\varepsilon_1 \partial_{\mathbf{n}} \psi|_\Omega - \varepsilon_2 \partial_{\mathbf{n}} \psi|_{\Omega^c} = 0, \quad \text{on } \Gamma, \quad (3.16)$$

we deduce another coupling condition

$$\sigma_e = \partial_{\mathbf{n}} \psi_e - \frac{\varepsilon_1}{\varepsilon_2} \partial_{\mathbf{n}} (\psi_0 + \psi_r), \quad \text{on } \Gamma. \quad (3.17)$$

In summary, the original problem (3.1) is transformed into the following two equations defined on  $\Omega$

$$\begin{cases} -\Delta \psi_r(\mathbf{x}) = 0 & \text{in } \Omega, \\ -\Delta \psi_e(\mathbf{x}) + \kappa^2 \psi_e(\mathbf{x}) = 0 & \text{in } \Omega, \end{cases} \quad (3.18)$$

with two coupling conditions on  $\Gamma$  given by

$$\begin{cases} \psi_0 + \psi_r = \psi_e & \text{on } \Gamma, \\ \sigma_e = \partial_{\mathbf{n}}\psi_e - \frac{\varepsilon_1}{\varepsilon_2}\partial_{\mathbf{n}}(\psi_0 + \psi_r) & \text{on } \Gamma, \end{cases} \quad (3.19)$$

where  $\sigma_e$  is the charge density generating  $\psi_e$ , as presented in (3.12). The second equation of (3.19) is also equivalent to

$$\varepsilon_2\psi_e + \mathcal{S}_\kappa(\varepsilon_1\partial_{\mathbf{n}}\psi_r - \varepsilon_2\partial_{\mathbf{n}}\psi_e) = -\varepsilon_1\mathcal{S}_\kappa(\partial_{\mathbf{n}}\psi_0), \quad \text{on } \Gamma, \quad (3.20)$$

which is derived from letting  $\mathcal{S}_\kappa$  act on both sides of the equation.

**Remark 3.1.** Eqs (3.18) – (3.20) are equivalent to the integral equation formulations (IEF) in [58] and [17]. The reason why we do the above transformation is that both the Laplace and the HSP equations in (3.18) can be solved efficiently using a particular domain decomposition method, see Section 4.2 and Section 5. The main computational cost will be spent on the coupling conditions (3.19).

**Remark 3.2.** The right hand side of Eqn (3.20) can be modified as in standard IEF-PCM by using the identity

$$-\mathcal{S}_\kappa(\partial_{\mathbf{n}}\psi_0) = (2\pi - \mathcal{D}_\kappa)\psi_0, \quad \text{on } \Gamma,$$

where  $\mathcal{D}_\kappa$  is the corresponding double layer boundary operator, see [9, 58]. This allows to obtain a right hand side that only depends on the potential and not on the field which subsequently leads to simpler expressions in the contribution to the Fock-matrix, if coupled to a quantum mechanical Hamiltonian within a polarizable embedding.

## 4 Strategy

In this section, we introduce a global strategy for solving Eqs (3.18)–(3.19) that are derived from the LPB equation (2.4). Then, we present how the domain decomposition method can be applied to solve the two PDEs defined on  $\Omega$ , taking advantage of its particular geometrical structure (i.e., a union of overlapping balls). The scheme of this section is inspired by [51, Section 4.2 and 5], our previous work for the case of non-ionic solvent.

### 4.1 Global strategy

We propose the following iterative procedure for solving Eqs (3.18)–(3.19): let  $g^0$  defined on  $\Gamma$  be an initial guess for the Dirichlet condition  $\psi_e|_\Gamma$  and set  $k = 1$ .

[1] Solve the following Dirichlet boundary problem for  $\psi_r^k$ :

$$\begin{cases} -\Delta\psi_r^k = 0 & \text{in } \Omega, \\ \psi_r^k = g^{k-1} - \psi_0 & \text{on } \Gamma, \end{cases} \quad (4.1)$$

and derive its Neumann boundary trace  $\partial_{\mathbf{n}}\psi_r^k$  on  $\Gamma$ .

[2] Solve the following Dirichlet boundary problem for  $\psi_e^k$ :

$$\begin{cases} -\Delta\psi_e^k + \kappa^2\psi_e^k = 0 & \text{in } \Omega, \\ \psi_e^k = g^{k-1} & \text{on } \Gamma, \end{cases} \quad (4.2)$$

and derive similarly its Neumann boundary trace  $\partial_{\mathbf{n}}\psi_e^k$  on  $\Gamma$ .

[3] Build the charge density  $\sigma_e^k = \partial_{\mathbf{n}}\psi_e^k - \frac{\varepsilon_1}{\varepsilon_2}\partial_{\mathbf{n}}(\psi_0 + \psi_r^k)$  and compute a new Dirichlet condition  $g^k = \mathcal{S}_\kappa\sigma_e^k$ .

[4] Compute the contribution  $E_k^s$  to the solvation energy based on  $\psi_r^k$  at the  $k$ -th iteration, set  $k \leftarrow k + 1$ , go back to Step [1] and repeat until the increment of interaction  $|E_k^s - E_{k-1}^s|$  becomes smaller than a given tolerance  $\text{To1} \ll 1$ .

**Remark 4.1.** *In order to provide a suitable initial guess of  $g^0$  (defined on  $\Gamma$ ), we consider the (unrealistic) scenario where the whole space  $\mathbb{R}^3$  is covered by the solvent medium. Then, the electrostatic potential  $\psi$  in this case is given explicitly by*

$$\psi(\mathbf{x}) = \sum_{i=1}^M \frac{4\pi q_i}{\varepsilon_2} \frac{\exp(-\kappa|\mathbf{x} - \mathbf{x}_i|)}{|\mathbf{x} - \mathbf{x}_i|}, \quad \forall \mathbf{x} \in \mathbb{R}^3, \quad (4.3)$$

see details in [55, Section 1.3.2]. As a consequence, we choose  $g^0$  as this potential restricted on  $\Gamma$ .

**Remark 4.2.** *The above global strategy is an iterative procedure, which is presented for an easier understanding. However, the final convergent solution satisfies, after discretization, a global linear system that can be solved by different linear solvers. We will address this issue in the later Section 6.2.*

## 4.2 Domain decomposition (DD) scheme

The Schwarz's domain decomposition method [59] is a good choice to solve the PDE defined on a complex domain which can be composed as a union of overlapping and possibly simple subdomains. According to the definition of  $\Omega$ , we have a natural domain decomposition as follows

$$\Omega = \bigcup_{j=1}^M \Omega_j, \quad \Omega_j = B_{r_j}(\mathbf{x}_j),$$

where each  $\Omega_j$  denotes the  $j$ -th VDW-ball with center  $\mathbf{x}_j$  and radius  $r_j$ . As a consequence, the Schwarz's domain decomposition method can be applied to solve the PDEs (4.1) and (4.2).

Similar to the ddCOSMO method [38], Eq. (4.1) is equivalent to the following coupled local equations, each restricted to  $\Omega_j$ :

$$\begin{cases} -\Delta\psi_r|_{\Omega_j} = 0 & \text{in } \Omega_j, \\ \psi_r|_{\Gamma_j} = \phi_{r,j} & \text{on } \Gamma_j, \end{cases} \quad (4.4)$$

where  $\Gamma_j = \partial\Omega_j$  and

$$\phi_{r,j} = \begin{cases} \psi_r & \text{on } \Gamma_j^i, \\ g - \psi_0 & \text{on } \Gamma_j^e. \end{cases} \quad (4.5)$$

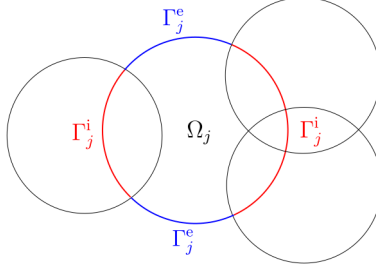


Figure 3: 2D schematic diagram of  $\Gamma_j^i$  (red) and  $\Gamma_j^e$  (blue) associated with  $\Omega_j$ .

Here, we omit the superscript due to the (outer) iteration index  $k$ .  $\Gamma_j^e$  is the external part of  $\Gamma_j$  not contained in any other ball  $\Omega_i$  ( $i \neq j$ ), i.e.,  $\Gamma_j^e = \Gamma \cap \Gamma_j$ ;  $\Gamma_j^i$  is the internal part of  $\Gamma_j$ , i.e.,  $\Gamma_j^i = \Omega \cap \Gamma_j$  (see Figure 3 for an illustration). Similarly, Eq. (4.2) is equivalent to the following coupled local equations, each restricted to  $\Omega_j$ :

$$\begin{cases} -\Delta\psi_e|_{\Omega_j} + \kappa^2\psi_e|_{\Omega_j} = 0 & \text{in } \Omega_j, \\ \psi_e|_{\Omega_j} = \phi_{e,j} & \text{on } \Gamma_j, \end{cases} \quad (4.6)$$

where

$$\phi_{e,j} = \begin{cases} \psi_e & \text{on } \Gamma_j^i, \\ g & \text{on } \Gamma_j^e. \end{cases} \quad (4.7)$$

Note that the Dirichlet conditions that appear in (4.5) and (4.7) are implicit since  $\psi_r$  (resp.  $\psi_e$ ) is not known on  $\Gamma_j^i$ . Hence, given the Dirichlet boundary condition on  $\Gamma$ , an iterative procedure must be applied to solve the coupled equations (4.4)–(4.5) (resp. (4.6)–(4.7)), such as the parallel Schwarz algorithm and the alternating Schwarz algorithm as presented in the ddCOSMO [38]. For example, the idea of the parallel algorithm is to solve each local problem based on the boundary condition of the neighboring solutions derived from the previous iteration. In this iterative procedure, the computed value of  $\psi_r|_{\Gamma_j^i}$  (resp.  $\psi_e|_{\Gamma_j^e}$ ) is updated step by step and converges to the exact value.

However, the parallel and the alternating Schwarz algorithms might not be the most efficient way to solve such a set of equations, but they are well-suited to illustrate the idea of the domain decomposition method. In fact, the global linear system obtained after discretization can be solved by different linear solvers (for example, the GMRES method). We will discuss more about this in Section 6.2. Before that, we shall develop two single-domain solvers: a Laplace solver and a HSP solver in the unit ball.

## 5 Single-domain solvers

In this section, we develop two single domain solvers in the unit ball respectively for solving Eq. (4.4) and Eq. (4.6) within the domain decomposition scheme.

### 5.1 Laplace solver

Developing a Laplace solver in a ball is not difficult and has already been presented in our previous work including the ddCOSMO [38], the ddPCM [45] and the ddPCM-SES [51]. For the sake of

completeness, we recall briefly the Laplace solver in the following content.

We want to solve the Laplace equation (4.4) defined on  $\Omega_j$ . Without loss of generality, we consider the following Laplace equation defined in the unit ball

$$\begin{cases} -\Delta u_r = 0 & \text{in } B_1(\mathbf{0}), \\ u_r = \phi_r & \text{on } \mathbb{S}^2. \end{cases} \quad (5.1)$$

Its unique solution in  $H^1(B_1(\mathbf{0}))$  can be written as

$$u_r(r, \theta, \varphi) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} [\phi_r]_{\ell}^m r^{\ell} Y_{\ell}^m(\theta, \varphi), \quad 0 \leq r \leq 1, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \varphi < 2\pi. \quad (5.2)$$

Here,  $Y_{\ell}^m$  denotes the (real orthonormal) spherical harmonic of degree  $\ell$  and order  $m$  defined on  $\mathbb{S}^2$  and

$$[\phi_r]_{\ell}^m = \int_{\mathbb{S}^2} \phi_r(\mathbf{s}) Y_{\ell}^m(\mathbf{s}) d\mathbf{s},$$

is the real coefficient of  $u_r$  corresponding to the mode  $Y_{\ell}^m$ . Then,  $u_r$  can be numerically approximated by  $\tilde{u}_r$  in the discretization space spanned by a truncated basis of spherical harmonics  $\{Y_{\ell}^m\}_{0 \leq \ell \leq \ell_{\max}, -\ell \leq m \leq \ell}$ , defined as

$$\tilde{u}_r(r, \theta, \varphi) = \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} [\tilde{\phi}_r]_{\ell}^m r^{\ell} Y_{\ell}^m(\theta, \varphi), \quad 0 \leq r \leq 1, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \varphi < 2\pi, \quad (5.3)$$

where  $\ell_{\max}$  denotes the maximum degree of spherical harmonics and

$$[\tilde{\phi}_r]_{\ell}^m = \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \phi_r(\mathbf{s}_n) Y_{\ell}^m(\mathbf{s}_n). \quad (5.4)$$

Here,  $\mathbf{s}_n \in \mathbb{S}^2$  represent Lebedev quadrature points [60],  $w_n^{\text{leb}}$  are the corresponding weights and  $N_{\text{leb}}$  is the number of Lebedev quadrature points.

## 5.2 HSP solver

We now want to solve the HSP equation (4.6) defined on  $\Omega_j$ . Without loss of generality, we consider the following HSP equation defined in the unit ball

$$\begin{cases} -\Delta u_e + \kappa^2 u_e^2 = 0 & \text{in } B_1(\mathbf{0}), \\ u_e = \phi_e & \text{on } \mathbb{S}^2. \end{cases} \quad (5.5)$$

Solving the above HSP equation in spherical coordinates by separation of variables, the radial equation corresponding to the angular dependency  $Y_{\ell}^m$  has the form

$$\frac{1}{R} \frac{d}{dr} \left( r^2 \frac{dR}{dr} \right) = \kappa^2 r^2 + \ell(\ell + 1), \quad \ell \geq 0, \quad (5.6)$$

that is,

$$r^2 \frac{d^2 R}{dr^2} + 2r \frac{dR}{dr} - (\kappa^2 r^2 + \ell(\ell + 1))R = 0, \quad (5.7)$$

which is called the modified spherical Bessel equation [61]. This equation has two linearly independent solutions as follows

$$i_\ell(r) = \sqrt{\frac{\pi}{2\kappa r}} I_{\ell+\frac{1}{2}}(\kappa r), \quad k_\ell(r) = \sqrt{\frac{2}{\pi\kappa r}} K_{\ell+\frac{1}{2}}(\kappa r), \quad (5.8)$$

where  $i_\ell$  and  $k_\ell$  are the modified spherical Bessel functions of the first and second kind associated with  $\kappa$ , see [61, Chapter 14] for details and Figure 4 for an illustration. Here,  $I_\alpha(x)$  and  $K_\alpha(x)$  with subscript  $\alpha$  are the modified Bessel functions of the first and second kind [62].

**Remark 5.1.**  $I_\alpha(x)$  and  $K_\alpha(x)$  satisfy the modified Bessel equation

$$x^2 \frac{d^2 f}{dx^2} + x \frac{df}{dx} - (x^2 + \alpha^2) f = 0. \quad (5.9)$$

In fact,  $I_\alpha$  and  $K_\alpha$  are exponentially growing and decaying functions, respectively.

Since  $k_\ell \rightarrow \infty$  as  $r \rightarrow 0$ , we are interested in the family  $i_\ell$  of the first kind. That is, we write the solution to (5.5) in the form of

$$u_e(r, \theta, \varphi) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} c_\ell^m i_\ell(r) Y_\ell^m(\theta, \varphi), \quad 0 \leq r \leq 1, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \varphi < 2\pi, \quad (5.10)$$

where  $c_\ell^m$  is the coefficient of the mode  $Y_\ell^m$ . With the same discretization as in Section 5.1, we derive the following approximate solution similar to Eq. (5.3):

$$\tilde{u}_e(r, \theta, \varphi) = \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} [\tilde{\phi}_e]_\ell^m \frac{i_\ell(r)}{i_\ell(1)} Y_\ell^m(\theta, \varphi), \quad 0 \leq r \leq 1, \quad 0 \leq \theta \leq \pi, \quad 0 \leq \varphi < 2\pi, \quad (5.11)$$

where  $[\tilde{\phi}_e]_\ell^m$  is given similar to (5.4) as follows

$$[\tilde{\phi}_e]_\ell^m = \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \phi_e(\mathbf{s}_n) Y_\ell^m(\mathbf{s}_n), \quad (5.12)$$

with the same notations  $w_n^{\text{leb}}$  and  $N_{\text{leb}}$  as above.

**Remark 5.2.** According to the fact that (see [61, page 708])

$$i_\ell(r) \approx \frac{(\kappa r)^\ell}{(2\ell+1)!!}, \quad \text{when } r > 0 \text{ is very small,}$$

we have  $\frac{i_\ell(r)}{i_\ell(1)} \rightarrow r^\ell$  as  $\kappa \rightarrow 0$ . Therefore, if  $\phi_r = \phi_e$ , then  $\tilde{u}_e \rightarrow \tilde{u}_r$  as  $\kappa \rightarrow 0$ . This means that the solution to Eq. (5.5) tends to the solution to Eq. (5.1), which makes sense.

## 6 Discretization

The global strategy in Section 4 in combination with the domain decomposition schemes for solving Eq. (4.1) and Eq. (4.2) is an iterative procedure. This implies that the proposed algorithm can be parallelized since only a group of local problems on each  $\Omega_j$  are solved. However, as mentioned in Remark 4.2, we will solve the problem in a global way, meaning that we finally solve a global linear system derived from discretization. To present this, we first introduce a reformulation of the coupling conditions and then present the global linear system for its discretization.

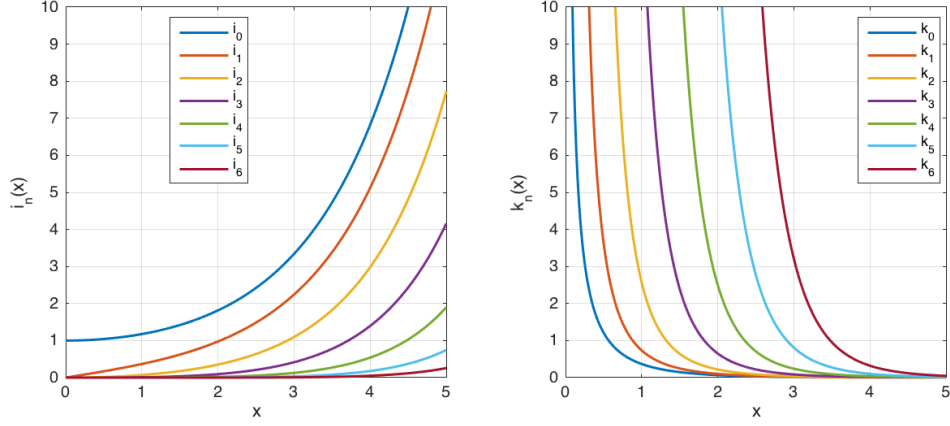


Figure 4: The modified spherical Bessel functions of the first kind ( $i_n$ , left) and the second kind ( $k_n$ , right) with  $\kappa = 1$ .

## 6.1 Reformulation

Let  $\chi_i$  be the characteristic function of  $\Omega_i$ , i.e.,

$$\chi_i(\mathbf{x}) := \begin{cases} 1 & \text{if } \mathbf{x} \in \Omega_i \\ 0 & \text{if } \mathbf{x} \notin \Omega_i \end{cases} \quad (6.1)$$

and then let

$$w_{ji}(\mathbf{x}) := \frac{\chi_i(\mathbf{x})}{|\mathcal{N}(j, \mathbf{x})|} = \frac{\chi_i(\mathbf{x})}{\sum_{i \neq j} \chi_i(\mathbf{x})}, \quad \forall \mathbf{x} \in \Gamma_j, \quad (6.2)$$

where  $\mathcal{N}(j, \mathbf{x})$  represents the index set of all balls containing  $\mathbf{x}$ . Here, we make the convention that in the case of  $|\mathcal{N}(j, \mathbf{x})| = 0$  (i.e.,  $\mathbf{x} \in \Gamma_j^e$ ), we define  $w_{ji}(\mathbf{x}) = 0$ ,  $\forall i$ . Furthermore,  $\forall \mathbf{x} \in \Gamma_j$ , we define

$$\chi_j^e(\mathbf{x}) := \begin{cases} 1 & \text{if } \mathbf{x} \in \Gamma_j^e, \\ 0 & \text{if } \mathbf{x} \in \Gamma_j^i, \end{cases} \quad (6.3)$$

which is equivalent to

$$\chi_j^e(\mathbf{x}) = 1 - \sum_{i \neq j} w_{ji}(\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma_j. \quad (6.4)$$

### 6.1.1 DD scheme

There are two local coupling conditions in the DD scheme, i.e., Eq. (4.5) and (4.7), respectively for coupling the local Laplace equation (4.4) and the local HSP equation (4.6). Based on the above-defined notations, Eq. (4.5) can be recast as

$$\psi_r|_{\Gamma_j}(\mathbf{x}) - \sum_{i \neq j} w_{ji}(\mathbf{x}) \psi_r|_{\Omega_i}(\mathbf{x}) = \chi_j^e(\mathbf{x}) (g(\mathbf{x}) - \psi_0(\mathbf{x})), \quad \forall \mathbf{x} \in \Gamma_j. \quad (6.5)$$



Similarly, Eq. (4.7) can be recast as

$$\psi_e|_{\Gamma_j}(\mathbf{x}) - \sum_{i \neq j} w_{ji}(\mathbf{x}) \psi_e|_{\Omega_i}(\mathbf{x}) = \chi_j^e(\mathbf{x})g(\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma_j. \quad (6.6)$$

### 6.1.2 Boundary coupling condition

There is a global coupling condition on  $\Gamma$ , i.e., Eq. (3.20), between the global Laplace equation and the global HSP equation defined in  $\Omega$  (see (3.18)), involving the nonlocal operator  $\mathcal{S}_\kappa$ :

$$g(\mathbf{x}) = \psi_e|_{\Gamma}(\mathbf{x}) = \mathcal{S}_\kappa \left( \partial_{\mathbf{n}} \psi_e - \frac{\varepsilon_1}{\varepsilon_2} \partial_{\mathbf{n}} (\psi_0 + \psi_r) \right) (\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma. \quad (6.7)$$

The single-layer operator  $\mathcal{S}_\kappa$  involves an integral over the whole solute-solvent boundary  $\Gamma$  which seems difficult to compute at a first glance. We introduce a technique to compute the integral of  $\mathcal{S}_\kappa$  efficiently. For each sphere  $\Gamma_i$ , we define a local single-layer potential  $\tilde{\mathcal{S}}_{\kappa, \Gamma_i}$  as follows

$$\tilde{\mathcal{S}}_{\kappa, \Gamma_i} \sigma(\mathbf{x}) := \int_{\Gamma_i} \frac{\exp(-\kappa|\mathbf{x} - \mathbf{y}|) \sigma(\mathbf{y})}{4\pi|\mathbf{x} - \mathbf{y}|}, \quad \forall \mathbf{x} \in \mathbb{R}^3, \quad (6.8)$$

where  $\sigma$  is an arbitrary function in  $H^{-\frac{1}{2}}(\Gamma_i)$ . As a consequence,  $\forall \sigma \in H^{-\frac{1}{2}}(\Gamma)$ , we have

$$\mathcal{S}_\kappa \sigma = \sum_{i=1}^M \tilde{\mathcal{S}}_{\kappa, \Gamma_i} (\chi_i^e \sigma), \quad (6.9)$$

where  $\chi_i^e \sigma$  extends  $\sigma|_{\Gamma_i^e}$  by zero to the whole sphere  $\Gamma_i$ . The above equation implies that the integral over  $\Gamma$  can be divided into a group of integrals respectively over each sphere  $\Gamma_i$ . Therefore, Eq. (6.7) can be recast as

$$g(\mathbf{x}) = \sum_{i=1}^M \tilde{\mathcal{S}}_{\kappa, \Gamma_i} \left[ \chi_i^e \left( \partial_{\mathbf{n}} \psi_e - \frac{\varepsilon_1}{\varepsilon_2} \partial_{\mathbf{n}} (\psi_0 + \psi_r) \right) \right] (\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma, \quad (6.10)$$

which is used for updating the boundary potential in the global strategy in Section 4.

## 6.2 Linear system

In this part, we first present the discretization of the above reformulation and then introduce the global linear system derived from this discretization.

### 6.2.1 Local truncation in balls

In Section 5, without the loss of generalization, we have presented the discretization of the solutions to the Laplace equation and the HSP equation defined in the unit ball.

Based on this, for each sphere  $\Gamma_j$ , we first approximate  $\psi_r|_{\Gamma_j}$  and  $\psi_e|_{\Gamma_j}$  respectively by a linear combination of spherical harmonics  $\{Y_\ell^m\}$  with  $0 \leq \ell \leq \ell_{\max}$  and  $-\ell \leq m \leq \ell$  as follows

$$\psi_r|_{\Gamma_j}(\mathbf{x}_j + r_j \mathbf{s}) = \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} [X_r]_{j\ell m} Y_\ell^m(\mathbf{s}), \quad \mathbf{s} \in \mathbb{S}^2, \quad (6.11)$$

and

$$\psi_e|_{\Gamma_j}(\mathbf{x}_j + r_j \mathbf{s}) = \sum_{\ell=0}^{\ell_{\max}} \sum_{m=-\ell}^{\ell} [X_e]_{j\ell m} Y_{\ell}^m(\mathbf{s}), \quad \mathbf{s} \in \mathbb{S}^2, \quad (6.12)$$

where  $[X_r]_{j\ell m}$  and  $[X_e]_{j\ell m}$  are unknown coefficients of the mode  $Y_{\ell}^m$  respectively associated with  $\psi_r|_{\Gamma_j}$  and  $\psi_e|_{\Gamma_j}$ . Here, for any point  $\mathbf{x} \in \Gamma_j$ , we actually use its spherical coordinates  $(r_j, \mathbf{s})$  s.t.  $\mathbf{x} = \mathbf{x}_j + r_j \mathbf{s}$ . According to the Laplace solver and the HSP solver presented in Section 5, we deduce directly

$$\psi_r|_{\Omega_i}(\mathbf{x}_i + r \mathbf{s}) = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [X_r]_{i\ell' m'} \left(\frac{r}{r_i}\right)^{\ell'} Y_{\ell'}^{m'}(\mathbf{s}), \quad 0 \leq r \leq r_i, \quad \mathbf{s} \in \mathbb{S}^2, \quad (6.13)$$

and

$$\psi_e|_{\Omega_i}(\mathbf{x}_i + r \mathbf{s}) = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [X_e]_{i\ell' m'} \frac{i_{\ell'}(r)}{i_{\ell'}(r_i)} Y_{\ell'}^{m'}(\mathbf{s}), \quad 0 \leq r \leq r_i, \quad \mathbf{s} \in \mathbb{S}^2, \quad (6.14)$$

where for any point  $\mathbf{x} \in \Omega_i$ , we take its spherical coordinates  $(r, \mathbf{s})$  s.t.  $\mathbf{x} = \mathbf{x}_i + r \mathbf{s}$ . Also, for each sphere  $\Gamma_i$ , we can compute the normal derivative of  $\psi_r$  on  $\Gamma_i^e$  as follows

$$\partial_{\mathbf{n}} \psi_r(\mathbf{x}_i + r_i \mathbf{s}) = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [X_r]_{i\ell' m'} \left(\frac{\ell'}{r_i}\right) Y_{\ell'}^{m'}(\mathbf{s}), \quad \mathbf{x}_i + r_i \mathbf{s} \in \Gamma_i^e, \quad (6.15)$$

and the normal derivative of  $\psi_e$  on  $\Gamma_i^e$

$$\partial_{\mathbf{n}} \psi_e(\mathbf{x}_i + r_i \mathbf{s}) = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [X_e]_{i\ell' m'} \frac{i'_{\ell'}(r_i)}{i_{\ell'}(r_i)} Y_{\ell'}^{m'}(\mathbf{s}), \quad \mathbf{x}_i + r_i \mathbf{s} \in \Gamma_i^e, \quad (6.16)$$

where  $i'_{\ell'}$  represents the derivative of  $i_{\ell'}$ .

**Remark 6.1.** We compute  $i'_{\ell'}(r_i)$  according to the following equation (see [61, page 707] for the derivation)

$$(2n+1)i'_n(r_i) = nr_i i_{n-1}(r_i) + (n+1)r_i i_{n+1}(r_i), \quad (6.17)$$

and in analogy, we compute  $k'_{\ell'}(r_i)$  used in the Appendix A as follows

$$-(2n+1)k'_n(r_i) = nr_i k_{n-1}(r_i) + (n+1)r_i k_{n+1}(r_i). \quad (6.18)$$

## 6.2.2 Discretization

So far, we have written the Ansatz for the unknowns  $\psi_r|_{\Gamma_j}$ ,  $\psi_e|_{\Gamma_j}$ , respectively  $\psi_r|_{\Omega_i}$ ,  $\psi_e|_{\Omega_i}$  with normal derivatives  $\partial_{\mathbf{n}} \psi_r$  and  $\partial_{\mathbf{n}} \psi_e$ , following Eqs (6.11) – (6.16), which depend on the unknowns  $X_r$  and  $X_e$ . This allows us to discretize the coupling conditions (6.5), (6.6) and (6.10), to derive a final linear system.

First, we replace the variable  $\mathbf{x} \in \Gamma_j$  of Eq. (6.5) by  $\mathbf{x} = \mathbf{x}_j + r_j \mathbf{s}$  with  $\mathbf{s} \in \mathbb{S}^2$  and derive the equation for each sphere  $\Gamma_j$  as follows

$$\begin{aligned} & \psi_r|_{\Gamma_j}(\mathbf{x}_j + r_j \mathbf{s}) - \sum_{i \neq j} w_{ji}(\mathbf{x}_j + r_j \mathbf{s}) \psi_r|_{\Omega_i}(\mathbf{x}_j + r_j \mathbf{s}) \\ &= \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}) (g(\mathbf{x}_j + r_j \mathbf{s}) - \psi_0(\mathbf{x}_j + r_j \mathbf{s})), \end{aligned} \quad (6.19)$$

which induces the following local equation by multiplying by  $Y_\ell^m$  and integrating over  $\mathbb{S}^2$  on both sides,  $\forall j, \ell, m$ ,

$$\begin{aligned} & \left\langle \psi_{\mathbf{r}}|_{\Gamma_j}(\mathbf{x}_j + r_j \cdot) - \sum_{i \neq j} w_{ji}(\mathbf{x}_j + r_j \cdot) \psi_{\mathbf{r}}|_{\Omega_i}(\mathbf{x}_j + r_j \cdot), Y_\ell^m(\cdot) \right\rangle_{\mathbb{S}^2} \\ &= \langle \chi_j^e(\mathbf{x}_j + r_j \cdot) (g(\mathbf{x}_j + r_j \cdot) - \psi_0(\mathbf{x}_j + r_j \cdot)), Y_\ell^m(\cdot) \rangle_{\mathbb{S}^2}. \end{aligned} \quad (6.20)$$

Here,  $\langle \cdot, \cdot \rangle_{\mathbb{S}^2}$  represents the integral over the unit sphere  $\mathbb{S}^2$ , which is numerically approximated using the Lebedev quadrature rule with  $N_{\text{leb}}$  points. We therefore denote such a numerical integration over  $\mathbb{S}^2$  by the notation  $\langle \cdot, \cdot \rangle_{\mathbb{S}^2, N_{\text{leb}}}$ . Eq. (6.20) can be rewritten in the form of a linear system

$$[\mathbf{A}X_{\mathbf{r}}]_{j\ell m} = [G_X]_{j\ell m} + [G_0]_{j\ell m}, \quad \forall j, \ell, m. \quad (6.21)$$

Here,  $\mathbf{A}$  is a square matrix of dimension  $M(\ell_{\max} + 1)^2 \times M(\ell_{\max} + 1)^2$  and the  $j\ell m$ -th row of  $\mathbf{A}X_{\mathbf{r}}$  is given by substituting (6.11) and (6.13) into (6.20) as follows

$$\begin{aligned} [\mathbf{A}X_{\mathbf{r}}]_{j\ell m} &= \left\langle \psi_{\mathbf{r}}|_{\Gamma_j}(\mathbf{x}_j + r_j \cdot) - \sum_{i \neq j} w_{ji}(\mathbf{x}_j + r_j \cdot) \psi_{\mathbf{r}}|_{\Omega_i}(\mathbf{x}_j + r_j \cdot), Y_\ell^m(\cdot) \right\rangle_{\mathbb{S}^2, N_{\text{leb}}} \\ &= [X_{\mathbf{r}}]_{j\ell m} - \sum_{i \neq j} \sum_{\ell', m'} \\ &\quad \left( \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} w_{ji}(\mathbf{x}_j + r_j \mathbf{s}_n) \left( \frac{r_{ijn}}{r_i} \right)^{\ell'} Y_{\ell'}^{m'}(\mathbf{s}_{ijn}) Y_\ell^m(\mathbf{s}_n) \right) [X_{\mathbf{r}}]_{i\ell' m'}, \end{aligned} \quad (6.22)$$

where  $(r_{ijn}, \mathbf{s}_{ijn})$  is the spherical coordinate associated with  $\Gamma_i$  of the point  $\mathbf{x}_j + r_j \mathbf{s}_n$  s.t.

$$\mathbf{x}_j + r_j \mathbf{s}_n = \mathbf{x}_i + r_{ijn} \mathbf{s}_{ijn}, \quad \text{with } \mathbf{s}_{ijn} \in \mathbb{S}^2.$$

Furthermore, the  $j\ell m$ -th element of the column vector  $G_X$  is given as

$$\begin{aligned} [G_X]_{j\ell m} &= \langle \chi_j^e(\mathbf{x}_j + r_j \cdot) g(\mathbf{x}_j + r_j \cdot), Y_\ell^m(\cdot) \rangle_{\mathbb{S}^2, N_{\text{leb}}} \\ &= \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}_n) g(\mathbf{x}_j + r_j \mathbf{s}_n) Y_\ell^m(\mathbf{s}_n), \end{aligned} \quad (6.23)$$

which depends on the unknowns  $X_{\mathbf{r}}$  and  $X_e$  through  $g$  given by Eq. (6.10). The notation  $X$  denotes the column of all unknowns, i.e.,

$$X = \begin{pmatrix} X_{\mathbf{r}} \\ X_e \end{pmatrix} \in \mathbb{R}^{2M(\ell_{\max} + 1)^2}. \quad (6.24)$$

Similarly, the  $j\ell m$ -th element of the column vector  $G_0$  is given as

$$[G_0]_{j\ell m} = - \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}_n) \psi_0(\mathbf{x}_j + r_j \mathbf{s}_n) Y_\ell^m(\mathbf{s}_n), \quad (6.25)$$

which can be computed a priori, since it is independent of  $X$ .

Similar to the linear system (6.21) and according to Eq. (6.6) for each sphere  $\Gamma_j$ , we have another linear system in the form of matrices

$$[\mathbf{B}X_e]_{j\ell m} = [G_X]_{j\ell m}, \quad \forall j, \ell, m, \quad (6.26)$$

where the square matrix  $\mathbf{B}$  satisfies

$$[\mathbf{B}X_e]_{j\ell m} = [X_e]_{j\ell m} - \sum_{i \neq j} \sum_{\ell', m'} [X_e]_{i\ell' m'} \left( \sum_{n=1}^{N_{\text{ieb}}} w_n^{\text{leb}} w_{ji}(\mathbf{x}_j + r_j \mathbf{s}_n) \frac{i_{\ell'}(r_{ijn})}{i_{\ell'}(r_i)} Y_{\ell'}^{m'}(\mathbf{s}_{ijn}) Y_{\ell}^m(\mathbf{s}_n) \right) \quad (6.27)$$

and  $[G_X]_{j\ell m}$  is given by (6.23).

So far, we have derived two linear systems of the form

$$\begin{cases} \mathbf{A} X_r = G_X + G_0, \\ \mathbf{B} X_e = G_X, \end{cases} \quad (6.28)$$

where  $X_r$  and  $X_e$  are the column vectors of unknowns  $[X_r]_{j\ell m}$  and  $[X_e]_{j\ell m}$  (respectively associated with the potentials  $\psi_r$  and  $\psi_e$ ). However, the column vector  $G_X$  depending on both  $X_r$  and  $X_e$  is not specified yet. To do this, the coupling condition (6.10) in terms of  $g$  should be used (which has not been used yet). Combining Eq. (6.10) with (6.23), we deduce the following form of  $G_X$ ,

$$G_X = F_0 - \mathbf{C}_1 X_r - \mathbf{C}_2 X_e, \quad (6.29)$$

where the column vector  $F_0$  is associated with  $\partial_{\mathbf{n}}\psi_0$ , two dense square matrices  $\mathbf{C}_1$  and  $\mathbf{C}_2$  are respectively associated with  $\partial_{\mathbf{n}}\psi_r$  and  $\partial_{\mathbf{n}}\psi_e$ . Considering the complexity of the formulas of  $F_0$ ,  $\mathbf{C}_1$ ,  $\mathbf{C}_2$ , we present them in Append. A.

**Remark 6.2.** *The number of the intersection of one atom with others is bounded from above. From the definition (6.2) of  $w_{ji}$ , we have  $w_{ji}(\mathbf{x}_j + r_j \mathbf{s}_n) = 0$  if  $r_{ijn} \geq r_i$ . Therefore,  $\mathbf{A}$  and  $\mathbf{B}$  are both sparse matrices for a large molecule.*

### 6.3 Linear solver

We finally obtain a global linear system written as

$$\begin{pmatrix} \mathbf{A} + \mathbf{C}_1 & \mathbf{C}_2 \\ \mathbf{C}_1 & \mathbf{B} + \mathbf{C}_2 \end{pmatrix} \begin{pmatrix} X_r \\ X_e \end{pmatrix} = \begin{pmatrix} G_0 + F_0 \\ F_0 \end{pmatrix}, \quad (6.30)$$

where both  $\mathbf{A}$  and  $\mathbf{B}$  are sparse for a large molecule, but not  $\mathbf{C}_1$  nor  $\mathbf{C}_2$ .

To solve this linear system (6.30), the LU factorization method and the GMRES method can be used directly [63, 64], where the first one gives an exact solution and the second one gives an approximate solution. However, the global strategy introduced in Section 4 provides another iterative strategy as follows

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \begin{pmatrix} X_r^k \\ X_e^k \end{pmatrix} = - \begin{pmatrix} \mathbf{C}_1 & \mathbf{C}_2 \\ \mathbf{C}_1 & \mathbf{C}_2 \end{pmatrix} \begin{pmatrix} X_r^{k-1} \\ X_e^{k-1} \end{pmatrix} + \begin{pmatrix} G_0 + F_0 \\ F_0 \end{pmatrix}, \quad (6.31)$$

where  $k$  denotes the (outer) iteration number as in Section 4,  $X_r^k$  and  $X_e^k$  are respectively the values of  $X_r$  and  $X_e$  computed at the  $k$ -th iteration. At the  $k$ -th iteration, we first update the right-hand side of Eq. (6.31), based on the previously-computed  $X_r^{k-1}$  and  $X_e^{k-1}$ . Then, we use the GMRES method to solve the linear system associated with  $X_r^k$  and  $X_e^k$ . To distinguish the GMRES iterations, we call the above iteration with index  $k$  as the outer iteration.

## 7 Numerical results

For an implicit solvation model, one important issue is to compute the solute-solvent interaction energy, to which the electrostatic contribution plays an important role. In fact, the electrostatic solvation energy  $E^s$  is computed from the reaction potential  $\psi_r$  according to the following formula (see [24] for the derivation of this formula)

$$E^s = \frac{1}{2} \int_{\mathbb{R}^3} \rho_M(\mathbf{r}) \psi_r(\mathbf{r}) d\mathbf{r}, \quad (7.1)$$

where the solute’s charge density  $\rho_M$  is given in Eq. (2.7) and  $\psi_r$  is obtained by solving the LPB equation. The outer iteration stops if  $\text{inc}_k < \text{To1}$ , where  $\text{To1}$  is the stopping tolerance and

$$\text{inc}_k := \frac{|E_k^s - E_{k-1}^s|}{|E_k^s|}, \quad (7.2)$$

where  $E_k^s$  denotes the electrostatic solvation energy computed at the  $k$ -th iteration. In the following content, we study the electrostatic solvation energy computed numerically by the ddLPB, which has been implemented in both Matlab and Fortran. Our Fortran code is based on the ddCOSMO and ddPCM codes written by Lipparini et al. (see GitHub link [65]).

By default, we take the dielectric permittivity in the solute cavity as in vacuum, that is,  $\varepsilon_1 = 1$ , and take the solvent to be water with the dielectric permittivity  $\varepsilon_2 = 78.54$  at the room temperature  $T = 298.15\text{K}$  ( $25^\circ\text{C}$ ). Further, we set the Debye-Hückel screening constant to  $\kappa = 0.1040 \text{ \AA}^{-1}$ , for an ionic strength  $I = 0.1$  molar. The solute cavity is chosen as the VDW-cavity. The atomic centers, charges and VDW radii are obtained from the PDB files [66] and the PDB2PQR package [67, 22] with the PEOEPB force field. By default, the stopping tolerance introduced in Section 6.3 is set to  $\text{To1} = 10^{-4}$ , while the GMRES tolerance in the ddLPB is set to  $10^{-8}$ . In the following content, these default parameters are used if they are not specified.

### 7.1 Kirkwood model

To test the ddLPB, we start from the Kirkwood model which has the explicit analytical solution, see [68, 69, 70]. In this model, there is only one sphere but with multiple charges distributed in this sphere. We consider the following six cases with the sphere radii all set to  $2\text{\AA}$ .

- Case 0 (Born model). One positive unit charge placed at  $(0, 0, 0)$ .
- Case 1. Two positive unit charges placed at  $(1, 0, 0)$  and  $(-1, 0, 0)$ .
- Case 2. Two positive unit charges placed at  $(1, 0, 0)$  and  $(-1, 0, 0)$ , and two negative unit charges placed at  $(0, 1, 0)$  and  $(0, -1, 0)$ .

Table 1: Electrostatic solvation energies (kcal/mol) of different Kirkwood models computed by the ddLPB with different discretization parameters  $\ell_{\max}$  and  $N_{\text{leb}}$ , where  $\kappa = 0$ . RE represents the relative error of the ddLPB result with respect to the exact result.

$\ell_{\max}$	$N_{\text{leb}}$	Case 0		Case 1		Case 2	
		$E^{\text{s}}$	RE	$E^{\text{s}}$	RE	$E^{\text{s}}$	RE
3	26	-81.9589	0	-349.5532	1.3762e-04	-64.0454	2.0606e-02
5	50	-81.9589	0	-349.4132	2.6294e-04	-62.5341	3.4772e-03
7	86	-81.9589	0	-349.5064	3.7195e-06	-62.7587	1.0199e-04
9	146	-81.9589	0	-349.5043	2.2890e-06	-62.7508	2.3904e-05
11	194	-81.9589	0	-349.5052	2.8612e-07	-62.7524	1.5936e-06
Exact		-81.9589		-349.5051		-62.7523	
$\ell_{\max}$	$N_{\text{leb}}$	Case 3		Case 4		Case 5	
		$E^{\text{s}}$	RE	$E^{\text{s}}$	RE	$E^{\text{s}}$	RE
3	26	-141.3629	4.5417e-02	-2991.4727	9.8768e-04	-3114.9078	2.7422e-03
5	50	-133.2890	1.4292e-02	-2988.0565	1.5543e-04	-3126.2105	8.7643e-04
7	86	-135.3437	9.0296e-04	-2988.5869	2.2051e-05	-3124.0588	1.8755e-04
9	146	-135.1534	5.0436e-04	-2988.4893	1.0607e-05	-3123.5037	9.8288e-06
11	194	-135.2189	1.9967e-05	-2988.5196	4.6846e-07	-3123.5193	1.4823e-05
Exact		-135.2216		-2988.5210		-3123.4730	

- Case 3. Two positive unit charges placed at  $(1.2, 0, 0)$  and  $(-1.2, 0, 0)$ , and two negative unit charges symmetrically placed at  $(0, 1.2, 0)$  and  $(0, -1.2, 0)$ .
- Case 4. Six Positive unit charges placed at  $(0.4, 0, 0)$ ,  $(0, 0.8, 0)$ ,  $(0, 0, 1.2)$ ,  $(0, 0, -0.4)$ ,  $(-0.8, 0, 0)$  and  $(0, -1.2, 0)$ .
- Case 5. Six positive unit charges placed at  $(0.2, 0.2, 0.2)$ ,  $(0.5, 0.5, 0.5)$ ,  $(0.8, 0.8, 0.8)$ ,  $(-0.2, 0.2, -0.2)$ ,  $(0.5, -0.5, 0.5)$  and  $(-0.8, -0.8, -0.8)$ .

The first case is also called the Born model and the other five cases are recommended in [70] to be tested for PB solvers. Table 1 lists the electrostatic solvation energies computed by the ddLPB as well as the relative errors, which are computed according to the following definition

$$\text{RE} := \frac{|E^{\text{s}} - E_{\text{ex}}^{\text{s}}|}{|E_{\text{ex}}^{\text{s}}|}. \quad (7.3)$$

Here,  $E^{\text{s}}$  denotes the electrostatic solvation energy computed by the ddLPB, while  $E_{\text{ex}}^{\text{s}}$  denotes the exact result. In the table, for a fixed  $\ell_{\max}$ , the corresponding  $N_{\text{leb}}$  ensures the accuracy of computing the scalar products of two arbitrary spherical harmonics in the function basis.

## 7.2 Convergence w.r.t. discretization parameters

We study the relationship between the electrostatic solvation energy  $E^{\text{s}}$  and the discretization parameters,  $\ell_{\max}$  and  $N_{\text{leb}}$ . For the sake of simplicity, we test a small molecule, namely formaldehyde with 4 atoms (see Table 2 for the geometry data). First, we compute an “exact” electrostatic solvation energy with large discretization parameters  $\ell_{\max} = 25$  and  $N_{\text{leb}} = 4334$ . This implies that we use 676 basis functions and 4334 integration points on each VDW sphere.

Table 2: Charges, centers ( $x, y, z$ ) and radii ( $\text{\AA}$ ) of the 4 atoms of formaldehyde.

Charge	$x$	$y$	$z$	VDW-radius
0.08130	0.00000	0.00000	-0.61750	2.11805
-0.20542	0.00000	0.00000	0.75250	1.92500
0.06206	0.00000	0.93500	-1.15750	1.58730
0.06206	0.00000	-0.93500	-1.15750	1.58730

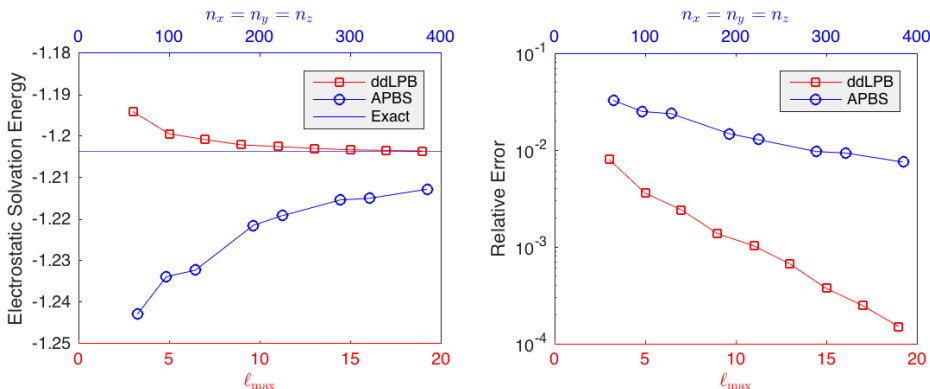


Figure 5: The electrostatic solvation energies (kcal/mol, left) of formaldehyde and the relative errors (right). On the left-hand side, the blue line represents the “exact” electrostatic solvation energy; the red curve illustrates the energies computed by the ddLPB w.r.t.  $\ell_{\max}$ ; the blue curve illustrates the energies computed by the APBS w.r.t. the number of grid points  $n_x = n_y = n_z$  in each axis direction.

The red curve in Figure 5 illustrates how  $E^s$  computed by ddLPB varies w.r.t. the maximum degree of spherical harmonics  $\ell_{\max}$ , where  $N_{\text{leb}} = 4334$ . It is observed that the ddLPB provides systematically improvable approximations when  $\ell_{\max}$  increases and we observed even exponential convergence of the energy w.r.t.  $\ell_{\max}$ . This allows to efficiently obtain an accuracy that is needed when the solvation model is coupled to a quantum mechanical description of the solute. Further, we also run the APBS software for comparison, where the box size is fixed to be  $20 \times 20 \times 20$  ( $\text{\AA}^3$ ) and the grid dimension  $n_x \times n_y \times n_z$  varies (with  $n_x = n_y = n_z$  in  $x$ -,  $y$ -,  $z$ -axis). In the input file, the molecular surface is chosen to be the VDW surface, by setting `srad = 0`. We use the multi-grid solver by setting `mg-manual`. The other parameters are set as follows: `gcent = mol 1`, `bcfl = mdh`, `chgm = spl4`, `sdens = 10`, `srfm = mol`, `swin = 0.3`. In Figure 5, the blue curve illustrates how  $E^s$  computed by the APBS varies w.r.t.  $n_x$ .

Figure 6 illustrates how  $E^s$  (computed by the ddLPB) varies w.r.t. the number of Lebedev quadrature points  $N_{\text{leb}}$ , where  $\ell_{\max} = 25$ . In fact, when  $N_{\text{leb}}$  is greater than 1000,  $E^s$  varies very slightly (less than 0.04%), despite that it does not decay monotonically.

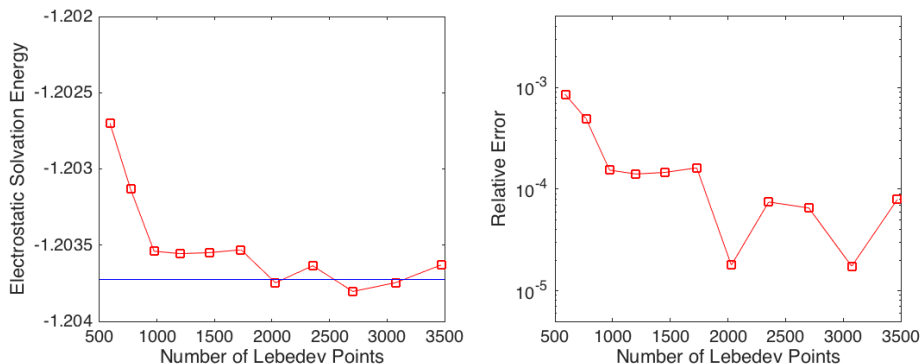


Figure 6: The electrostatic solvation energies (kcal/mol, left) of formaldehyde and the relative errors (right) w.r.t.  $N_{\text{leb}}$ , computed by the ddLPB. On the left-hand side, the blue line represents the “exact” electrostatic solvation energy.

### 7.3 Varying the Debye-Hückel screening constants

We now study the relationship between the electrostatic solvation energy and the Debye-Hückel screening constant  $\kappa$ . On the continuous level, the solution of the Poisson-Boltzmann equation tends to the one of COSMO in the limit  $\kappa \rightarrow \infty$ . Physically speaking, this is reasonable since the solvent becomes a perfect conductor as the ionic strength tends to  $\infty$  and screens any charge from the solute. On the other hand, the solution of PB equation tends to the solution of PCM in the limit  $\kappa \rightarrow 0$ .

It is therefore natural to compare the ddLPB with the ddCOSMO [38] and the ddPCM [45]. In Figure 7, we plot the electrostatic solvation energies of formaldehyde (with 4 atoms) and letn (PDB ID, with 141 atoms) w.r.t. the Debye-Hückel screening constant. The same discretization parameters  $\ell_{\text{max}} = 11$  and  $N_{\text{leb}} = 590$  are used for all three methods ddCOSMO, ddPCM and ddLPB.

For the limit  $\kappa \rightarrow \infty$ , the ddLPB result tends to the ddCOSMO result. This is consistent with the theory since the discretized equations for the ddLPB with  $\kappa$  coincide with those for the ddCOSMO even after discretization. Indeed, the ddCOSMO method can be seen as a particular ddLPB method in the case of  $\kappa = \infty$ .

When  $\kappa = 0$ , we observe a difference between the ddLPB and ddPCM results. The reason is that the ddPCM discretizes the IEF-PCM [9] directly whereas the ddLPB uses a discretization of a different integral formulation. Since the continuous models are equivalent, the difference tends to zero when higher discretization parameters are used.

### 7.4 Comparison with APBS

In this part, we further compare the ddLPB solver with the widely-used software, APBS. The same parameters are used as in Section 7.2, except the grid spacings and dimensions. The test is performed on a MacBook Pro with a 2.5 GHz Intel Core i7 processor and we consider the protein letn with 141 atoms as test case.

Table 3 illustrates the ddLPB and APBS results for different discretization parameters, includ-



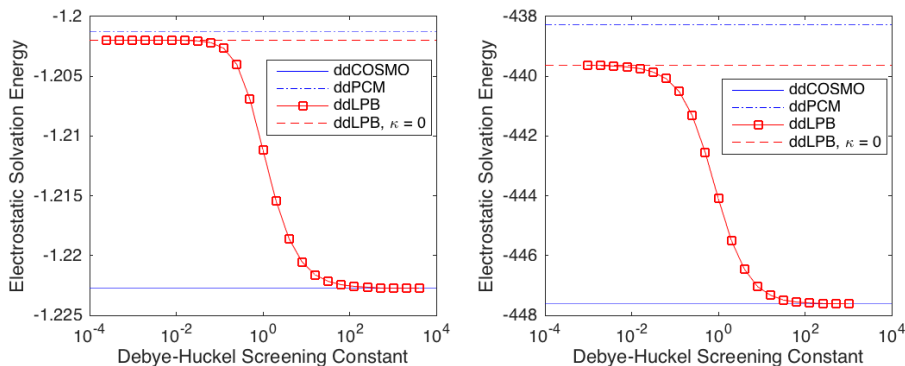


Figure 7: The electrostatic solvation energy (kcal/mol) of formaldehyde (left) and 1etn (right) w.r.t. the Debye-Hückel screening constant  $\kappa$ . Solid blue line: ddCOSMO result; dashed blue line: ddPCM result; red curve: ddLPB result,  $0 < \kappa < \infty$ ; dashed red line: ddLPB result,  $\kappa = 0$ .

ing the electrostatic solvation energy, the relative error, the number of iterations, the run time and the memory. To compute the relative error, two “exact” electrostatic solvation energies are computed respectively using an exponential fitting for ddLPB and a linear extrapolation for APBS, as illustrated in Figure 8.

In fact, we have observed an exponential convergence of the energy w.r.t.  $\ell_{\max}$  for formaldehyde in Section 7.2. It appears therefore consistent to apply an exponential data fitting for the ddLPB-energies. More precisely, we use the function `fit` in Matlab where the fit type is set to  $y = a + b \cdot \exp(-c \cdot x)$ . The “exact” electrostatic solvation energy  $E_{\text{ddlpb}}^*$  is obtained as the coefficient  $a$  when the fitting function is figured out. For the APBS, we use the linear extrapolation procedure introduced in [17]. We first plot the APBS energies w.r.t.  $h_g$  and then draw a line crossing the leftmost two energies at  $h_g = 0.1$  and  $h_g = 0.12$ . As a consequence, as  $h_g$  tends to zero, this line crosses the  $y$ -axis at an “exact” electrostatic solvation energy  $E_{\text{apbs}}^*$ .

In Table 3, we can observe that the ddLPB usually cost less memory than the APBS due to the nuclear-centered spectral-type basis functions similar to atomic orbitals. Furthermore, from the relative errors obtained by extrapolation, one observes that the ddLPB results are more accurate in this example, as also observed in Figure 5 which can be explained that APBS is a first order method while ddLPB shows exponential convergence for the energy.

## 7.5 Computational cost

To study the computational cost of the ddLPB, we test a set of 24 proteins with the following PDB IDs: 1ajj, 1ptq, 1vjw, 1bor, 1fxd, 1sh1, 1hpt, 1fca, 1bpi, 1r69, 1bbl, 1vii, 2erl, 451c, 2pde, 1cbn, 1frd, 1uxc, 1mbg, 1neq, 1a2s, 1svr, 1o7b, 1a63. The discretization parameters of the ddLPB are set to  $\ell_{\max} = 5$  and  $N_{\text{leb}} = 50$ . For the comparison, we also run the TABI-PB solver [17], which however works on the SES, not the VDW surface. To have approximately the same degree of freedom, we set the density of vertices in the TABI-PB to be 5 per  $\text{\AA}^2$ . In the TABI-PB, the order of Taylor approximation is set to 1, the MAC parameter is set to 0.4 and the GMRES tolerance (relative residual error) is set to  $10^{-4}$ . Both the TABI-PB and the ddLPB use the default parameters given at the beginning of Section 7, that is,  $\varepsilon_1 = 1$ ,  $\varepsilon_2 = 78.54$  and  $\kappa = 0.1040 \text{\AA}^{-1}$ .

Table 3: ddLPB (top) and APBS results (bottom) for the protein 1etn.  $N_{\text{iter}}$  represents the number of outer iterations in the ddLPB.  $h_g$  and  $N_g$  represent the grid spacing and the grid dimension in the APBS.  $E_{\text{ddlpb}}^*$  and  $E_{\text{apbs}}^*$  represent the “exact” electrostatic solvation energies computed respectively from an exponential fitting for ddLPB and a linear extrapolation for APBS, as illustrated in Fig. 8.

$\ell_{\text{max}}$	$N_{\text{leb}}$	$E^{\text{s}}$ (kcal/mol)	RE	$N_{\text{iter}}$	Run time (s)	Memory (MB)
3	26	-429.3457	2.5091e-02	9	1	7
5	50	-433.3903	1.5907e-02	11	2	30
7	86	-437.6449	6.2460e-03	11	8	91
9	146	-438.9725	3.2314e-03	11	22	241
11	194	-439.1880	2.7421e-03	14	63	461
13	266	-439.1629	2.7991e-03	18	198	861
15	350	-440.1754	5.0001e-04	12	206	893
17	434	-440.1017	6.7735e-04	13	362	1398
$E_{\text{ddlpb}}^*$		-440.3956				

$h_g$	$N_g$	$E^{\text{s}}$ (kcal/mol)	RE	$N_{\text{iter}}$	Run time (s)	Memory (MB)
0.5	65 <sup>3</sup>	-476.3143	7.9224e-02	–	1	63
0.4	97 <sup>3</sup>	-464.0155	5.1358e-02	–	3	203
0.26	129 <sup>3</sup>	-457.4919	3.6577e-02	–	8	475
0.2	193 <sup>3</sup>	-452.8998	2.6172e-02	–	29	1606
0.16	225 <sup>3</sup>	-450.5827	2.0922e-02	–	46	2572
0.13	289 <sup>3</sup>	-448.8017	1.6887e-02	–	113	5608
0.12	321 <sup>3</sup>	-448.0838	1.5260e-02	–	190	7819
0.1	385 <sup>3</sup>	-446.9613	1.2717e-02	–	525	14044
$E_{\text{apbs}}^*$		-441.3488				

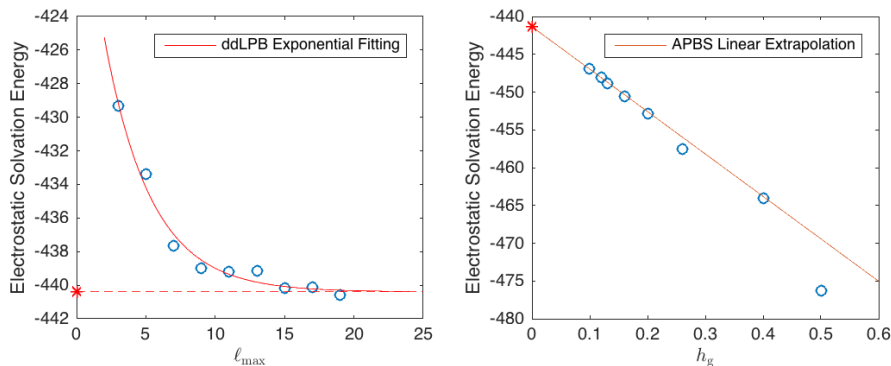


Figure 8: Estimation of the electrostatic solvation energy of 1etn (141 atoms), based on the exponential data fitting for the ddLPB results (left) and the linear extrapolation for the APBS results (right) in Table 3. Here,  $h_g$  represents the grid spacing in the APBS. On the left, the red curve plots the fitting function  $y = -440.3956 + 27.4584 \exp(-0.2974x)$ , which tends to  $E_{\text{ddlpb}}^* = -440.3956$  at the infinity. The dashed horizontal line is the asymptote of the curve. On the right, the red line crosses the leftmost two energies and intersects the  $y$ -axis at the star maker  $E_{\text{apbs}}^* = -441.3488$ .

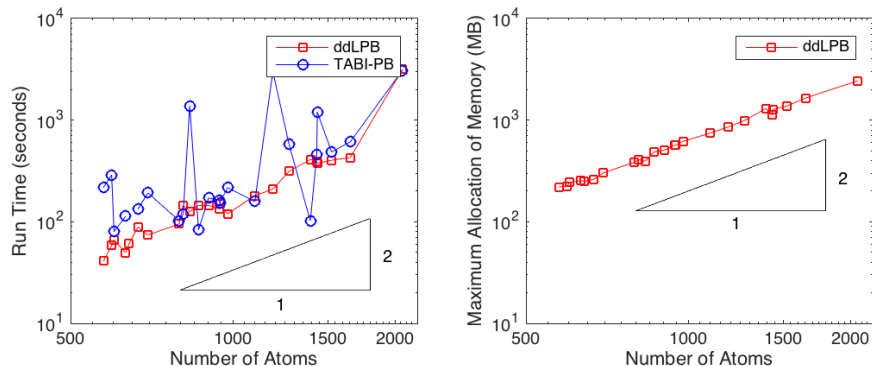


Figure 9: Run time (left) and maximum allocation of memory (right) of the ddLPB and the TABI-PB w.r.t. the number of atoms, for different proteins in the test set. The detailed memory information of the TABI-PB is not available and the memory is finally released.

Table 4: Details on the ddLPB and the TABI-PB for the test set of proteins, including the degree of freedom and the number of iterations.  $M$  denotes the number of atoms.

PDB	$M$	Degree of freedom		Iteration		PDB	$M$	Degree of freedom		Iteration	
		ddLPB	TABI	ddLPB	TABI			ddLPB	TABI	ddLPB	TABI
1ajj	602	43344	41420	15	10	2erl	633	45576	46642	10	11
1ptq	795	57240	56898	12	9	451c	1435	103320	89286	13	45
1vjw	946	68112	56084	12	13	2pde	667	48024	48430	18	12
1bor	832	59904	58076	16	100	1cbn	642	46224	–	12	–
1fxd	978	70416	62844	9	14	1frd	1652	118944	90772	11	25
1sh1	696	50112	55604	14	15	1uxc	809	58248	58164	19	10
1hpt	945	68040	65572	12	10	1mbg	902	64944	64190	14	12
1fca	864	62208	49744	14	9	1neq	1187	85464	98964	11	100
1bpi	1393	103320	64482	13	7	1a2s	1272	91584	93574	15	22
1r69	1099	79128	62648	11	11	1svr	1432	103104	100546	15	16
1bbl	576	41472	53654	10	17	1o7b	1525	109800	103920	13	17
1vii	596	42912	50874	14	24	1a63	2065	148680	144796	13	74

Figure 9 illustrates the run time and the maximum allocation of memory of the ddLPB and the TABI-PB for different proteins in the test set. Table 4 provides more information including the degree of freedom and the number of iterations. As a general observation, we see that the number of iteration to reach the tolerance is more stable in ddLPB. Notice that for the protein 1cbn, the MSMS in the TABI-PB fails to generate a suitable mesh and the TABI-PB stops without returning the energy. Further, for the protein 1bor and 1neq, the GMRES in the TABI-PB reaches the maximum number of iterations 100, before reaching the residual tolerance  $10^{-4}$ . The ddLPB reaches the tolerance  $\text{To1} = 10^{-4}$  within 20 outer iterations for all proteins in the test set.

**Remark 7.1.** *At this moment, we haven't yet employed acceleration techniques in the ddLPB implementation, while the TABI-PB features the "treecode" acceleration technique [17].*

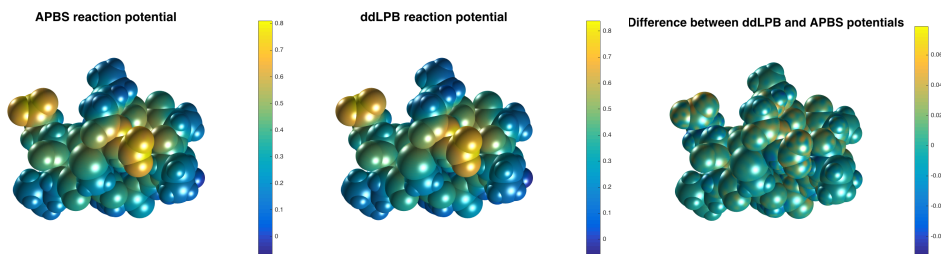


Figure 10: Reaction potentials ( $e/\text{\AA} = 561.91 \text{ kT/e}$ ) on the VDW surface of 1etn with 141 atoms: APBS result (left), ddLPB result (middle) and difference (right). The following parameters are used: grid dimension  $193 \times 193 \times 193$  and grid spacing  $0.2 \times 0.2 \times 0.2$  for the APBS,  $\ell_{\max} = 11$  and  $N_{\text{leb}} = 194$  for the ddLPB. The other parameters are the default ones as used previously.

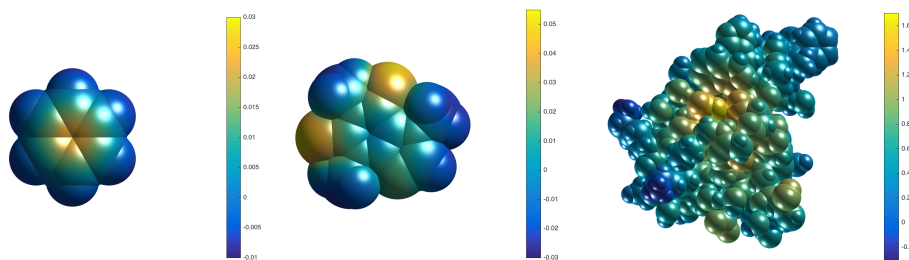


Figure 11: Reaction potential ( $e/\text{\AA}$ ) on the VDW surfaces of benzene (left), caffeine (middle) and 1ajj (right) respectively with 4, 24 and 602 atoms. The following parameters are used:  $\ell_{\max} = 11$  and  $N_{\text{leb}} = 194$  for benzene and caffeine,  $\ell_{\max} = 7$  and  $N_{\text{leb}} = 86$  for the protein 1ajj. The other parameters are the default ones as used previously.

## 7.6 Graphical illustration

Finally, we provide some graphical illustrations of the reaction potential  $\psi_r$  on the VDW surface. In Figure 10, we illustrate the reaction potentials of 1etn computed by the APBS and the ddLPB, and their potential difference on the VDW surface. It is observed that the difference is usually small over the surface. In Figure 11, we present the reaction potentials of two small molecules benzene and caffeine, and the protein 1ajj. We notice that the reaction potential of benzene has rotational symmetry, which matches its geometrical structure.

## 8 Conclusion

In this paper, we proposed a domain decomposition method for the Poisson-Boltzmann solvation model that shows exponential convergence in the energy w.r.t to the number of basis functions employed. This allows to reach a precision which enable this method to couple it with models on the level of quantum mechanics.

The original problem defined in  $\mathbb{R}^3$  is first transformed into two coupled equations defined in the bounded solute cavity, based on potential theory. Then, the Schwarz domain decomposition

method was used to solve these two problems by decomposing the solute cavity into balls. In consequence, we developed two direct single-domain solvers respectively for solving the Laplace equation and the HSP equation defined in the unit ball, which becomes easy to tackle by using the spherical harmonics in the angular direction. Taking into account the coupling conditions allowed then to obtain a global linear system. A series of numerical results have been presented to show the performance of the ddLPB method.

In the future, we will focus on accelerating the ddLPB to make it suitable to very large molecules based on linear scaling acceleration techniques such as the Fast Multipole Method (FMM). In addition, we will embed the ddLPB solver in software packages which simulate the computation of the solute on the level of theory of quantum mechanics or molecular dynamics.

## 9 Acknowledgements

We would like to thank F. Lipparini for discussion and guidance with the implementation in Fortran. We also thank the other members in our DD-family [37] for the usual fruitful discussions, including E. Cancès, L. Lagardère, J.P. Piquemal and B. Mennucci.

## References

- [1] Byung Jun Yoon and AM Lenhoff. A boundary element method for molecular electrostatics with electrolyte effects. *Journal of Computational Chemistry*, 11(9):1080–1086, 1990.
- [2] Anthony Nicholls and Barry Honig. A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation. *Journal of Computational Chemistry*, 12(4):435–445, 1991.
- [3] David Griffiths. *Introduction to elementary particles*. John Wiley & Sons, 2008.
- [4] Roberto Cammi and Benedetta Mennucci. *Continuum Solvation Models in Chemical Physics: From Theory to Applications*. John Wiley, 2007.
- [5] Jacopo Tomasi, Benedetta Mennucci, and Roberto Cammi. Quantum mechanical continuum solvation models. *Chemical Reviews*, 105(8):2999–3094, 2005.
- [6] Benedetta Mennucci. Continuum solvation models: What else can we learn from them? *Journal of Physical Chemistry Letters*, 1(10):1666–1674, 2010.
- [7] Andreas Klamt and GJGJ Schüürmann. COSMO: a new approach to dielectric screening in solvents with explicit expressions for the screening energy and its gradient. *Journal of the Chemical Society, Perkin Transactions 2*, (5):799–805, 1993.
- [8] BZ Lu, YC Zhou, MJ Holst, and JA McCammon. Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications. *Communications in Computational Physics*, 3(5):973–1009, 2008.
- [9] Eric Cancès and Benedetta Mennucci. New applications of integral equations methods for solvation continuum models: ionic solutions and liquid crystals. *Journal of Mathematical Chemistry*, 23(3-4):309–326, 1998.

- [10] Alexander H Boschitsch, Marcia O Fenley, and Huan-Xiang Zhou. Fast boundary element method for the linear Poisson-Boltzmann equation. *Journal of Physical Chemistry B*, 106(10):2741–2754, 2002.
- [11] Michael D Altman, Jaydeep P Bardhan, Jacob K White, and Bruce Tidor. Accurate solution of multi-region continuum biomolecule electrostatic problems using the linearized Poisson-Boltzmann equation with curved boundary elements. *Journal of Computational Chemistry*, 30(1):132–153, 2009.
- [12] Chandrajit Bajaj, Shun-Chuan Chen, and Alexander Rand. An efficient higher-order fast multipole boundary element solution for Poisson-Boltzmann-based molecular electrostatics. *SIAM Journal on Scientific Computing*, 33(2):826–848, 2011.
- [13] Michel F Sanner, Arthur J Olson, and Jean-Claude Spehner. Reduced surface: an efficient way to compute molecular surfaces. *Biopolymers*, 38(3):305–320, 1996.
- [14] Yongjie Zhang, Guoliang Xu, and Chandrajit Bajaj. Quality meshing of implicit solvation models of biomolecular structures. *Computer Aided Geometric Design*, 23(6):510–530, 2006.
- [15] Bo Zhang, Bo Peng, Jingfang Huang, Nikos P Pitsianis, Xiaobai Sun, and Benzhuo Lu. Parallel AFMPB solver with automatic surface meshing for calculation of molecular solvation free energy. *Computer Physics Communications*, 190:173–181, 2015.
- [16] Benzhuo Lu, Xiaolin Cheng, Jingfang Huang, and J Andrew McCammon. AFMPB: an adaptive fast multipole Poisson–Boltzmann solver for calculating electrostatics in biomolecular systems. *Computer physics communications*, 181(6):1150–1160, 2010.
- [17] Weihua Geng and Robert Krasny. A treecode-accelerated boundary integral Poisson–Boltzmann solver for electrostatics of solvated biomolecules. *Journal of Computational Physics*, 247:62–78, 2013.
- [18] Jeffrey D Madura, James M Briggs, Rebecca C Wade, Malcolm E Davis, Brock A Luty, Andrew Ilin, Jan Antosiewicz, Michael K Gilson, Babak Bagheri, L Ridgway Scott, et al. Electrostatics and diffusion of molecules in solution: simulations with the University of Houston Brownian Dynamics program. *Computer Physics Communications*, 91(1-3):57–95, 1995.
- [19] Lin Li, Chuan Li, Subhra Sarkar, Jie Zhang, Shawn Witham, Zhe Zhang, Lin Wang, Nicholas Smith, Marharyta Petukh, and Emil Alexov. DelPhi: a comprehensive suite for DelPhi software and associated resources. *BMC Biophysics*, 5(1):9, 2012.
- [20] Duan Chen, Zhan Chen, Changjun Chen, Weihua Geng, and Guo-Wei Wei. MIBPB: A software package for electrostatic analysis. *Journal of Computational Chemistry*, 32(4):756–770, 2011.
- [21] Nathan A Baker, David Sept, Simpson Joseph, Michael J Holst, and J Andrew McCammon. Electrostatics of nanosystems: application to microtubules and the ribosome. *Proceedings of the National Academy of Sciences*, 98(18):10037–10041, 2001.
- [22] Todd J Dolinsky, Paul Czodrowski, Hui Li, Jens E Nielsen, Jan H Jensen, Gerhard Klebe, and Nathan A Baker. PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic Acids Research*, 35(suppl\_2):W522–W525, 2007.

- [23] Elizabeth Jurrus, Dave Engel, Keith Star, Kyle Monson, Juan Brandi, Lisa E Felberg, David H Brookes, Leighton Wilson, Jiahui Chen, Karina Liles, et al. Improvements to the APBS biomolecular solvation software suite. *Protein Science*, 27(1):112–128, 2018.
- [24] Federico Fogolari, Alessandro Brigo, and Henriette Molinari. The Poisson-Boltzmann equation for biomolecular electrostatics: a tool for structural biology. *Journal of Molecular Recognition*, 15(6):377–392, 2002.
- [25] Zhong-hua Qiao, Zhi-lin Li, and Tao Tang. A finite difference scheme for solving the nonlinear Poisson-Boltzmann equation modeling charged spheres. *Journal of Computational Mathematics*, pages 252–264, 2006.
- [26] G Fisicaro, L Genovese, O Andreussi, N Marzari, and S Goedecker. A generalized Poisson and Poisson-Boltzmann solver for electrostatic environments. *Journal of Chemical Physics*, 144(1):014103, 2016.
- [27] Long Chen, Michael J Holst, and Jinchao Xu. The finite element approximation of the nonlinear Poisson–Boltzmann equation. *SIAM Journal on Numerical Analysis*, 45(6):2298–2320, 2007.
- [28] Michael Holst, Nathan Baker, and Feng Wang. Adaptive multilevel finite element solution of the Poisson–Boltzmann equation I. Algorithms and examples. *Journal of Computational Chemistry*, 21(15):1319–1342, 2000.
- [29] Nathan Baker, Michael Holst, and Feng Wang. Adaptive multilevel finite element solution of the Poisson–Boltzmann equation II. Refinement at solvent-accessible surfaces in biomolecular systems. *Journal of Computational Chemistry*, 21(15):1343–1352, 2000.
- [30] Michael Holst, James Andrew Mccammon, Zeyun Yu, YC Zhou, and Yunrong Zhu. Adaptive finite element modeling techniques for the Poisson-Boltzmann equation. *Communications in Computational Physics*, 11(1):179–214, 2012.
- [31] Burak Aksoylu, Stephen D. Bond, Eric C. Cyr, and Michael Holst. Goal-oriented adaptivity and multilevel preconditioning for the Poisson-Boltzmann equation. *Journal of Scientific Computing*, 52(1):202–225, 2012.
- [32] Dexuan Xie. New solution decomposition and minimization schemes for Poisson–Boltzmann equation in calculation of biomolecular electrostatics. *Journal of Computational Physics*, 275:294–309, 2014.
- [33] Jinyong Ying and Dexuan Xie. A new finite element and finite difference hybrid method for computing electrostatics of ionic solvated biomolecule. *Journal of Computational Physics*, 298:636–651, 2015.
- [34] Yi Jiang, Yang Xie, Jinyong Ying, Dexuan Xie, and Zeyun Yu. SDPBS web server for calculation of electrostatics of ionic solvated biomolecules. *Molecular Based Mathematical Biology*, 3(1), 2015.
- [35] Jinyong Ying and Dexuan Xie. A hybrid solver of size modified Poisson–Boltzmann equation by domain decomposition, finite element, and finite difference. *Applied Mathematical Modelling*, 58:166–180, 2018.

- [36] Yang Xie, Jinyong Ying, and Dexuan Xie. SMPBS: Web server for computing biomolecular electrostatics using finite element solvers of size modified Poisson-Boltzmann equation. *Journal of Computational Chemistry*, 38(8):541–552, 2017.
- [37] Benjamin Stamm, Filippo Lipparini, Eric Cancès, Yvon Maday, Jean-Philip Piquemal, Benedetta Mennucci, Louis Lagardère, and Quan Chaoyu. ddCOSMO & ddPCM, 2015.
- [38] Eric Cancès, Yvon Maday, and Benjamin Stamm. Domain decomposition for implicit solvation models. *Journal of Chemical Physics*, 139(5):054111, 2013.
- [39] Filippo Lipparini, Benjamin Stamm, Eric Cancès, Yvon Maday, and Benedetta Mennucci. Fast domain decomposition algorithm for continuum solvation models: Energy and first derivatives. *Journal of Chemical Theory and Computation*, 9(8):3637–3648, 2013.
- [40] Filippo Lipparini, Louis Lagardère, Giovanni Scalmani, Benjamin Stamm, Eric Cancès, Yvon Maday, Jean-Philip Piquemal, Michael J Frisch, and Benedetta Mennucci. Quantum calculations in solution for large to very large molecules: A new linear scaling QM/continuum approach. *Journal of Physical Chemistry Letters*, 5(6):953–958, 2014.
- [41] Filippo Lipparini, Giovanni Scalmani, Louis Lagardère, Benjamin Stamm, Eric Cancès, Yvon Maday, Jean-Philip Piquemal, Michael J Frisch, and Benedetta Mennucci. Quantum, classical, and hybrid QM/MM calculations in solution: General implementation of the ddCOSMO linear scaling strategy. *Journal of Chemical Physics*, 141(18):184108, 2014.
- [42] Filippo Lipparini, Louis Lagardère, Christophe Raynaud, Benjamin Stamm, Eric Cancès, Benedetta Mennucci, Michael Schnieders, Pengyu Ren, Yvon Maday, and Jean-Philip Piquemal. Polarizable molecular dynamics in a polarizable continuum solvent. *Journal of Chemical Theory and Computation*, 11(2):623–634, 2015.
- [43] Gabrielle Ciaramella and Martin J Gander. Analysis of the parallel Schwarz method for growing chains of fixed-sized subdomains: Part I. *SIAM Journal on Numerical Analysis*, 55(3):1330–1356, 2017.
- [44] G Ciaramella and MJ Gander. Analysis of the parallel schwarz method for growing chains of fixed-sized subdomains: Part ii. *SIAM Journal on Numerical Analysis*, 56(3):1498–1524, 2018.
- [45] Benjamin Stamm, Eric Cancès, Filippo Lipparini, and Yvon Maday. A new discretization for the polarizable continuum model within the domain decomposition paradigm. *Journal of Chemical Physics*, 144(5):054101, 2016.
- [46] Paolo Gatto, Filippo Lipparini, and Benjamin Stamm. Computation of forces arising from the polarizable continuum model within the domain-decomposition paradigm. *The Journal of Chemical Physics*, 147(22):224108, 2017.
- [47] Byungkook Lee and Frederic M Richards. The interpretation of protein structures: estimation of static accessibility. *Journal of Molecular Biology*, 55(3):379–IN4, 1971.
- [48] Frederic M Richards. Areas, volumes, packing and protein structure. *Annual Review of Biophysics and Bioengineering*, 6:151–176, 1977.



- [49] M. L. Connolly. Analytical molecular surface calculation. *Journal of Applied Crystallography*, 16(5):548–558, Oct 1983.
- [50] Chaoyu Quan and Benjamin Stamm. Mathematical analysis and calculation of molecular surfaces. *Journal of Computational Physics*, 322:760 – 782, 2016.
- [51] Chaoyu Quan, Benjamin Stamm, and Yvon Maday. A domain decomposition method for the polarizable continuum model based on the solvent excluded surface. *Mathematical Models and Methods in Applied Sciences*, 2018.
- [52] Robert A Adams and John JF Fournier. *Sobolev spaces*, volume 140. Academic Press, 2003.
- [53] VI Lebedev and DN Laikov. A quadrature formula for the sphere of the 131st algebraic order of accuracy. In *Doklady. Mathematics*, volume 59, pages 477–481. MAIK Nauka/Interperiodica, 1999.
- [54] Chaoyu Quan and Benjamin Stamm. Meshing molecular surfaces based on analytical implicit representation. *Journal of Molecular Graphics and Modelling*, 71:200–210, 2017.
- [55] Gene Lamm. The Poisson-Boltzmann equation. *Reviews in Computational Chemistry*, 19:147–333, 2003.
- [56] L. Banjai. Boundary element methods, October 2007.
- [57] Stefan A. Sauter and Christoph Schwab. *Elliptic Boundary Integral Equations*, pages 101–181. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [58] E Cancès, B Mennucci, and J Tomasi. A new integral equation formalism for the polarizable continuum model: Theoretical background and applications to isotropic and anisotropic dielectrics. *Journal of Chemical Physics*, 107(8):3032–3041, 1997.
- [59] Alfio Quarteroni and Alberto Valli. *Domain decomposition methods for partial differential equations*. Number CMCS-BOOK-2009-019. Oxford University Press, 1999.
- [60] Daniel J Haxton. Lebedev discrete variable representation. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 40(23):4443, 2007.
- [61] G Arfken, H Weber, and FE Harris. *Mathematical Methods for Physicists: A Comprehensive Guide 7th Edition* (London: Academic). 2012.
- [62] Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, volume 55. Courier Corporation, 1964.
- [63] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.
- [64] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.
- [65] Filippo Lipparini and Paolo Gatto. ddPCM. <https://github.com/filippolipparini/ddPCM>, 2015.
- [66] HM Berman, J Westbrook, Z Feng, G Gilliland, TN Bhat, H Weissig, IN Shindyalov, and PE Bourne. The Protein Data Bank. *Nucleic Acids Research*, URL: [www.rcsb.org](http://www.rcsb.org), 28:235–242, 2000.

- [67] Todd J Dolinsky, Jens E Nielsen, J Andrew McCammon, and Nathan A Baker. PDB2PQR: an automated pipeline for the setup of Poisson–Boltzmann electrostatics calculations. *Nucleic Acids Research*, 32(suppl\_2):W665–W667, 2004.
- [68] John G Kirkwood. Theory of solutions of molecules containing widely separated charges with special application to zwitterions. *The Journal of Chemical Physics*, 2(7):351–361, 1934.
- [69] Weihua Geng. A boundary integral Poisson-Boltzmann solvers package for solvated bimolecular simulations. *Molecular Based Mathematical Biology*, 3(1), 2015.
- [70] Duc D Nguyen, Bao Wang, and Guo-Wei Wei. Accurate, robust, and reliable calculations of Poisson–Boltzmann binding energies. *Journal of Computational Chemistry*, 38(13):941–948, 2017.

## A Computation of $\mathbf{C}_1$ , $\mathbf{C}_2$ and $F_0$

For each  $\Gamma_i^e$ , we first define a square matrix  $\mathbf{P}_{\chi_i^e}$  of dimension  $(\ell_{\max} + 1)^2 \times (\ell_{\max} + 1)^2$  for each  $\chi_i^e$ , the  $(\ell_0 m_0, \ell' m')$ -th element of which is defined as

$$[\mathbf{P}_{\chi_i^e}]_{\ell_0 m_0}^{\ell' m'} := \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_i^e(\mathbf{x}_i + r_i \mathbf{s}_n) Y_{\ell_0}^{m_0}(\mathbf{s}_n) Y_{\ell'}^{m'}(\mathbf{s}_n), \quad (\text{A.1})$$

where  $0 \leq \ell_0 \leq \ell_{\max}$ ,  $-\ell_0 \leq m_0 \leq \ell_0$ ,  $0 \leq \ell' \leq \ell_{\max}$ ,  $-\ell' \leq m' \leq \ell'$ . Based on Eq. (6.15), we can approximate  $\chi_i^e \partial_{\mathbf{n}} \psi_{\text{r}}$  (defined on  $\Gamma_i$ ) by a linear combination of spherical harmonics  $\{Y_{\ell_0}^{m_0}\}$  with  $0 \leq \ell_0 \leq \ell_{\max}$ ,  $-\ell_0 \leq m_0 \leq \ell_0$  as follows

$$\chi_i^e \partial_{\mathbf{n}} \psi_{\text{r}}(r_i, \mathbf{s}) = \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{\text{r}, \ell_0 m_0} Y_{\ell_0}^{m_0}(\mathbf{s}), \quad \mathbf{s} \in \mathbb{S}^2, \quad (\text{A.2})$$

where the coefficient  $c_{\text{r}, \ell_0 m_0}$  is computed by the Lebedev quadrature rule as follows

$$c_{\text{r}, \ell_0 m_0} = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [\mathbf{P}_{\chi_i^e}]_{\ell_0 m_0}^{\ell' m'} \frac{\ell'}{r_i} [X_{\text{r}}]_{i \ell' m'}. \quad (\text{A.3})$$

**Remark A.1.** By writing  $\chi_i^e \partial_{\mathbf{n}} \psi_{\text{r}}$  as a linear combination of spherical harmonics, the single-layer potential  $\tilde{\mathcal{S}}_{\kappa, \Gamma_i}$  can act on it conveniently.

For an arbitrary Lebedev point  $\mathbf{x}_j + r_j \mathbf{s}_n = \mathbf{x}_i + r_{ijn} \mathbf{s}_{ijn} \in \Gamma_j^e$ , we can then compute as follows

$$\begin{aligned} & \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} \chi_i^e \partial_{\mathbf{n}} \psi_{\text{r}} \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{\text{r}, \ell_0 m_0} \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} Y_{\ell_0}^{m_0} \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{\text{r}, \ell_0 m_0} \left( \frac{i_{\ell_0}'(r_i)}{i_{\ell_0}(r_i)} - \frac{k_{\ell_0}'(r_i)}{k_{\ell_0}(r_i)} \right)^{-1} \frac{k_{\ell_0}(r_{ijn})}{k_{\ell_0}(r_i)} Y_{\ell_0}^{m_0}(\mathbf{s}_{ijn}) \\ &= \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [\mathbf{Q}]_{i \ell' m'}^{j n} \frac{\ell'}{r_i} [X_{\text{r}}]_{i \ell' m'}, \end{aligned} \quad (\text{A.4})$$

where  $\mathbf{Q}$  is a matrix of dimension  $M(\ell_{\max} + 1)^2 \times MN_{\text{leb}}$ , with the  $(i\ell'm', jn)$ -th element defined by

$$[\mathbf{Q}]_{i\ell'm'}^{jn} := \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} [\mathbf{P}_{\chi_i^e}]_{\ell_0 m_0}^{\ell' m'} \left( \frac{i'_{\ell_0}(r_i)}{i_{\ell_0}(r_i)} - \frac{k'_{\ell_0}(r_i)}{k_{\ell_0}(r_i)} \right)^{-1} \frac{k_{\ell_0}(r_{ijn})}{k_{\ell_0}(r_i)} Y_{\ell_0}^{m_0}(\mathbf{s}_{ijn}). \quad (\text{A.5})$$

Therefore, we have the  $(j\ell m, i\ell'm')$ -th element of  $\mathbf{C}_1$  as follows

$$[\mathbf{C}_1]_{j\ell m}^{i\ell' m'} = \frac{\varepsilon_1}{\varepsilon_2} \left( \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}_n) Y_{\ell}^m(\mathbf{s}_n) [\mathbf{Q}]_{i\ell' m'}^{jn} \frac{\ell'}{r_i} \right). \quad (\text{A.6})$$

Similarly, based on Eq. (6.16), we can approximate  $\chi_i^e \partial_{\mathbf{n}} \psi_e$  (defined on  $\Gamma_i$ ) by another linear combination of spherical harmonics  $\{Y_{\ell_0}^{m_0}\}$  with  $0 \leq \ell_0 \leq \ell_{\max}$ ,  $-\ell_0 \leq m_0 \leq \ell_0$  as follows

$$\chi_i^e \partial_{\mathbf{n}} \psi_e(r_i, \mathbf{s}) = \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{e, \ell_0 m_0} Y_{\ell_0}^{m_0}(\mathbf{s}), \quad \mathbf{s} \in \mathbb{S}^2, \quad (\text{A.7})$$

where

$$c_{e, \ell_0 m_0} = \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [\mathbf{P}_{\chi_i^e}]_{\ell_0 m_0}^{\ell' m'} \frac{i'_{\ell'}(r_i)}{i_{\ell'}(r_i)} [X_e]_{i\ell' m'}. \quad (\text{A.8})$$

For an arbitrary Lebedev point  $\mathbf{x}_j + r_j \mathbf{s}_n = \mathbf{x}_i + r_{ijn} \mathbf{s}_{ijn} \in \Gamma_j^e$ , we can then compute

$$\begin{aligned} & \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} \chi_i^e \partial_{\mathbf{n}} \psi_e \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{e, \ell_0 m_0} \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} Y_{\ell_0}^{m_0} \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{e, \ell_0 m_0} \left( \frac{i'_{\ell_0}(r_i)}{i_{\ell_0}(r_i)} - \frac{k'_{\ell_0}(r_i)}{k_{\ell_0}(r_i)} \right)^{-1} \frac{k_{\ell_0}(r_{ijn})}{k_{\ell_0}(r_i)} Y_{\ell_0}^{m_0}(\mathbf{s}_{ijn}) \\ &= \sum_{\ell'=0}^{\ell_{\max}} \sum_{m'=-\ell'}^{\ell'} [\mathbf{Q}]_{i\ell' m'}^{jn} \frac{i'_{\ell'}(r_i)}{i_{\ell'}(r_i)} [X_e]_{i\ell' m'}. \end{aligned} \quad (\text{A.9})$$

This yields that

$$[\mathbf{C}_2]_{j\ell m}^{i\ell' m'} = - \left( \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}_n) Y_{\ell}^m(\mathbf{s}_n) [\mathbf{Q}]_{i\ell' m'}^{jn} \frac{i'_{\ell'}(r_i)}{i_{\ell'}(r_i)} \right). \quad (\text{A.10})$$

In addition, since  $\partial_{\mathbf{n}} \psi_0$  is known, for an arbitrary Lebedev point  $\mathbf{x}_j + r_j \mathbf{s}_n = \mathbf{x}_i + r_{ijn} \mathbf{s}_{ijn} \in \Gamma_j^e$ , we can compute the following column vector  $S$

$$\begin{aligned} [S]_{ijn} &= \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} \chi_i^e \partial_{\mathbf{n}} \psi_0 \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{0, \ell_0 m_0} \left( \tilde{\mathcal{S}}_{\kappa, \Gamma_i} Y_{\ell_0}^{m_0} \right) (\mathbf{x}_j + r_j \mathbf{s}_n) \\ &= \sum_{\ell_0=0}^{\ell_{\max}} \sum_{m_0=-\ell_0}^{\ell_0} c_{0, \ell_0 m_0} \left( \frac{i'_{\ell_0}(r_i)}{i_{\ell_0}(r_i)} - \frac{k'_{\ell_0}(r_i)}{k_{\ell_0}(r_i)} \right)^{-1} \frac{k_{\ell_0}(r_{ijn})}{k_{\ell_0}(r_i)} Y_{\ell_0}^{m_0}(\mathbf{s}_{ijn}), \end{aligned} \quad (\text{A.11})$$

where

$$c_{0,\ell_0 m_0} = \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_i^e(\mathbf{x}_i + r_i \mathbf{s}_n) \partial_{\mathbf{n}} \psi_0(\mathbf{x}_i + r_i \mathbf{s}_n) Y_{\ell_0}^{m_0}(\mathbf{s}_n). \quad (\text{A.12})$$

This yields that

$$[F_0]_{j\ell m} = -\frac{\varepsilon_1}{\varepsilon_2} \left( \sum_{n=1}^{N_{\text{leb}}} w_n^{\text{leb}} \chi_j^e(\mathbf{x}_j + r_j \mathbf{s}_n) Y_{\ell}^m(\mathbf{s}_n) \sum_{i=1}^M [S]_{ijn} \right). \quad (\text{A.13})$$