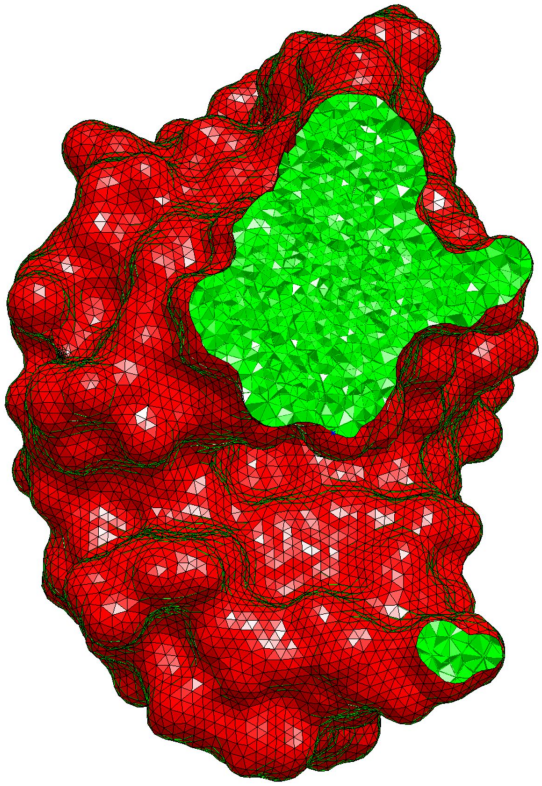
ROBUST FINITE ELEMENT SOLVER FOR MOLECULAR ELECTROSTATIC ENERGY COMPUTATIONS



Dissertation zur Erlangung des akademischen Grades des
Doktors der Naturwissenschaften (Dr. rer. nat.)

eingereicht im Fachbereich Biologie, Chemie, Pharmazie
der Freien Universität Berlin

vorgelegt in englischer Sprache von

ILKAY SAKALLI

aus Hamburg

Mai 2015

Die vorliegende Arbeit wurde unter Anleitung von Prof. Dr. Ernst Walter Knapp am Institut für Chemie und Biochemie im Fachbereich Biologie, Chemie und Pharmazie der Freien Universität Berlin durchgeführt und von der Volkswagen Stiftung finanziert.

1. Gutachter: Prof. Dr. E. W. Knapp
Macromolecular Modelling
Institute of Chemistry, Biochemistry & Pharmacy
Free University of Berlin
Germany

2. Gutachter: Prof. Dr. G. Matthias Ullmann
Structural Biology/Bioinformatics
Institute of Computational Biochemistry
University of Bayreuth
Germany

Tag der Disputation: 09.07.2015

Universe is the playground of *energy*.
(Das Universum ist der Spielplatz der Energie.)
— Jerome Anders (*1975)

This dissertation is dedicated to my lovely universe, Elif, who gives me my *energy* in life to keep on doing the things I'm doing especially while my research in scientific computing over the past years. This work got the icing on the cake by my daughter, Melina, who was born during this work and makes me smile every day, repeatedly.

In loving memory of Şevket and Hediye Serbest.

Acknowledgements

I want to acknowledge Prof. Dr. G. Matthias Ullmann for supervision of this work and my continuous supervisor Prof. Dr. Ernst Walter Knapp for his faith in me, providing the opportunity to work in his lively work group and asking the same question nearly every day "Na, wie siehts aus?" (Well, how is it [your work] going?). Although, this was sometimes a bit frustrating and I had to work hard until I had an answer to his everyday question, this made me a lot of progress and forced my work, making things a bit better, every day.

Thanks to Joachim Schöberl who has developed NETGEN [1] together with his team and gave me the opportunity to work with him in Vienna and who visited us in Berlin. It was a very inspiring and exciting time with his working group. Special thanks go to Kaliyaperumal Elankumaran who helped in the beginning of this work. I am very thankful to Tolga Can who has provided me with source code of LSMS [2] and gave me support.

All my love and thanks goes to my wife, Elif, for her continuous support and my daughter, Melina, for her supporting smile when I was writing this thesis. Further thanks go to my whole family as well as to my brother.

I would like to thank my group members for their continuous support, Gegham Galstyan for his administrative support, and many fruitful discussions about technical and chemical details.

A special mention goes to Milan Hodošček for his support in compiling, workflow in Gentoo and introduction into CHARMM development.

I thank the computing officers at the ZEDAT providing additional computing resources and the ability to manage our own cluster with approximately 150 cores and parallel computing capabilities located at our work group.

I gratefully acknowledge the financial support from the Volkswagen Stiftung and the Deutsche Forschungsgemeinschaft (DFG) 1078 and all friends in the Dahlem Research School (DRS) Molecular Science.

Ilkay Sakalli

Berlin, May 2015

Preamble

During my research at the Macromolecular Modelling Group (AG Knapp) I worked in the field of electrostatic energy computations using different methods.
This cumulative thesis summarizes my doctoral research and is divided into four parts. It is mainly based on the peer-reviewed journal paper which was recently published [3]:

> I. Sakalli, J. Schöberl, and E. W. Knapp, "mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations," *J. Chem. Theory Comput.*, vol. 10, no. 11, pp. 5095–5112, Oct. 2014. doi: 10.1021/ct5005092

I developed a new program called *mFES* (*m*olecular *F*inite *E*lement *S*olver). mFES is a publicly available program (under the GNU lesser general public licence v2.0) and is suitable to solve two problems in computational chemistry, namely evaluating:

i. solvation energies of proteins (ΔG) and
ii. pK_A values of titratable groups in proteins.

Chapter I describes the theory to solve both computational chemistry problems *(i.)* and *(ii.)*. An introduction in continuum electrostatics is given to solve the Poisson-Boltzmann equation with different methods: Finite Difference (FD), Finite Element (FE) and Boundary Element (BE) method. Subsequently, the framework of pK_A computations is introduced using thermodynamic cycles and a Monte Carlo method.

Chapter II provides a summary of the peer-reviewed article [3] solving problem *(i.)* computing the electrostatics solvation energy of proteins.

Chapter III introduces into the framework of pK_A computations using mFES to solve problem *(ii.)*. This chapter includes an introduction and discussion about using mFES for the computation of pK_A values for titratable groups while comparing to results obtained by classical FD methods.

Chapter IV. introduces into implementation details of mFES and a web version $mFES^+$ web. $mFES^+$ web computes ΔG or pK_A values with mFES over a web interface. Protein files from RCSB PDB database are prepared using CHARMM. The user does not need to install any plugin and the service is ready for use without registration. By submitting an $mFES^+$ web-job, computations are performed in the background of a compute cluster. It is designed to make the life of a researcher easier by presenting results for ΔG or pK_A computations in a fresh and informative way.

mFES sources and the online version $mFES^+$ web are available at:

http://agknapp.chemie.fu-berlin.de/mfes

CONTENTS

Abbreviations

*l*PBE	linear Poisson-Boltzmann Equation
*n*PBE	non-linear Poisson-Boltzmann Equation
FD	Finite Difference
FE	Finite Element
BE	Boundary Element
vdW	van-der-Waals
MD	Molecular Dynamics
PDB	Protein Data Bank
DOF	Degree Of Freedom
IEL	Ion Exclusion Layer
SES	Solvent Excluded Surface
LES	Linear Equation System
SDH	Single Debye-Hückel
MDH	Multiple Debye-Hückel
GEM	Gauss Elimination Method
LUM	LU-Factorisation Method
JM	Jacobi Method
GSM	Gauss-Seidel Method
SOR	Successive Over-Relaxation
CGM	Conjugate Gradient Method
MCM	Monte Carlo Method
DFT	Density Functional Theory
RMSD	Root-Mean-Square Deviation

This page was intentionally left blank

I. Theory

"Look deep, deep into nature, and then you will understand everything better."
— Albert Einstein (†1951)

Introduction

Electrostatic interactions play an important role in nature. These interactions influence the binding and functioning of proteins. From a macroscopic point of view, electrostatic interactions model long-range interactions which cover more than a few Ångströms. Every living cell is influenced by electrostatic effects because cells hold relatively high concentrations of ions. In general, the ionic effect dominates the electrostatic effect in biological systems. From a microscopic point of view, strong physical interactions of molecules are electrostatically driven. Chemical bonds are dominated by electrostatic interactions because of the opposite charges of atom nuclei and electrons while H-bonds are dominated by dipole interactions (e.g. in water). Ionic bonds are strong chemical bonds and display large electronegativity differences (>1.7, e.g. Na^+Cl^-), whereas covalent bonds (most organic compounds) have less or no electronegativity difference. Furthermore, electrostatic interactions play an important role in weak bonds, e.g. hydrogen bonds and dipole-dipole or van-der-Waals interactions.

Classical Electrostatics in Molecular Systems

This thesis focuses on protein electrostatics. Therefore, one has to model proteins in their natural environment. Generally, proteins are inside a cell and are surrounded by thousands of water molecules and different ions. Because of the large number of molecules in the environment, describing a molecular system in detail is very complex and computationally expensive. This is why the protein in its environment

is approximated by a mathematical model. Water and ions are described implicitly using a dielectric medium (see Fig. 1). Although this mathematical model has an approximate character, it is known to provide a suitable description of an equilibrated solvent, and electrostatic effects are represented well enough for many applications [4].

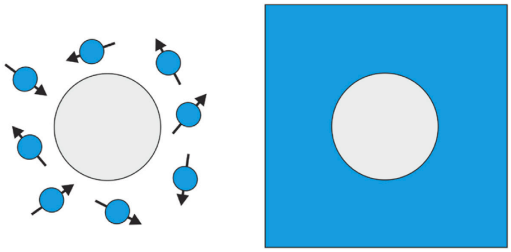


Fig. 1. Comparison of a discrete (left) with an implicit, continuum model (right). *Blue circles* with arrows represent water molecules with their dipole moment vectors. The *solid blue area* represents implicit water. The *grey circle* represents a single ion with a given radius (or atoms of the whole protein).

The introduction of a dielectric medium [5] is the main modification in implicit modelling. Dielectric media model electrostatic interactions of molecular dipoles (Fig 2), in different regions of the model. Classically, a dielectric constant of $\varepsilon_{out} = 80$ is used to model an aqueous solvent and a low dielectric constant ($\varepsilon_{in} = 2–4$) is used for the solute, a protein. If all electrostatic effects are accounted for (explicitly) in a molecular model, the dielectric constant is unity. If this precondition is not fulfilled, the dielectric constant used for the protein moiety is higher, and a value of between 2 and 4 is used to account for electronic polarization and small backbone fluctuations [6]. In this thesis, this issue is not discussed in more detail. The choice of the values for dielectric constants is controversial, and often different values are used in different applications to answer specific questions. The interested reader is referred to [5], [7]–[9].

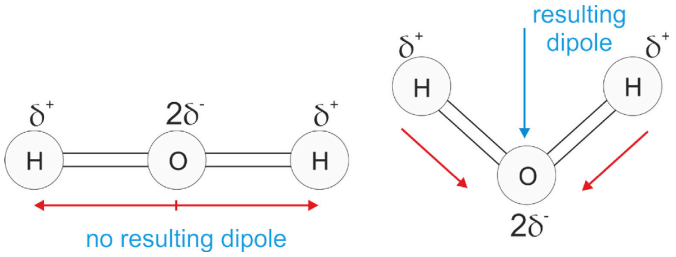


Fig 2. Directions of the dipole moment in water; left: sum of both dipoles is zero and there is no resulting dipole; right: resulting dipole is the sum of both vectors.

Testing different methods to compute electrostatic potentials is challenging, and analytical models are useful as a reference. One simple model which has an analytical solution with and without a surrounding ion concentration (see appendix B) is the solvation of an ion. This so-called Born ion model has one explicit point charge in the centre of a sphere. It is a useful model for comparison of different numerical methods [3].

Coulomb's Law

Coulomb's law was established by Charles-Augustin de Coulomb in 1785 [10]. In general Coulomb's law describes the force between two point charges which can approximate the electrostatic interaction between two atoms. Using Coulomb's law a description of the implicit solvent model in an electrostatic homogeneous medium (e.g. a vacuum) is obtained. The implicit solvent model uses point charges for atoms in a protein. In more detail, an atom is modelled as a point charge inside a sphere whose radius is the van-der-Waals (vdW) radius with the atomic nucleus in the centre. Atomic charges are given in units of Coulomb [C]. One Coulomb is defined as the charge which is flowing through a wire with a current of one Ampere in one second. Computing electrostatic energies, the interest of this work lies in potential energy differences of homogeneous and inhomogeneous dielectric media which is discussed in the following sections.

Coulomb's Law for Computations of an Electrostatic Homogeneous Medium

The simplest description for an electrostatic potential $\Phi(\vec{r})$ is defined by Coulomb's law using an electrostatic dielectric homogeneous medium (e.g. a vacuum). One atom is moved in the homogeneous dielectric medium from infinity to position $\vec{r}$, as shown in the following equation:

$$\Phi(\vec{r}) = -\int \mathbf{E} \cdot d\mathbf{s} = \frac{1}{4\pi\varepsilon}\frac{Q}{|\vec{r}|} + C\,, \tag{1}$$

where $Q = z_i \cdot e$ is the point charge of the atom with z_i being the charge number and e the elementary charge, and $\mathbf{E}$ is the electrostatic field along the pathway of the displaced point charge.

The dielectric constant ε is defined as $\varepsilon = \varepsilon_0 \cdot \varepsilon_r$, where ε_0 is the permittivity of free space (= vacuum) and ε_r is a dimensionless dielectric constant which is chosen such that it models the polarizability of the dielectric medium. The integration constant C is generally set to zero, defining the zero point at infinite distances. The potential energy G necessary to move a charge Q in this potential is defined as

$$G = \Phi(\vec{r}) \cdot Q\,. \tag{2}$$

The potential energy G_{all} for N charges Q_i is given by

$$G_{all}^{\text{homo}} = \frac{1}{2}\sum_{i=1}^{N} Q_i \Phi(\vec{r}_i) = \frac{1}{2}\sum_{i=1}^{N} Q_i \sum_{j=1}^{N} \frac{1}{4\pi\,\varepsilon_0}\frac{Q_j}{|\,\vec{r}_{ij}\,|}\,, \tag{3}$$

where $i \neq j$ and $|\,\vec{r}_{ij}\,|$ is the distance between the two charges Q_i and Q_j. A simple summation is possible, since the energy terms are additive due to a linear dependence on the effect of each charge in the homogeneous dielectric medium. Computing the electrostatic energy for proteins in solvent is more complicated, since the dielectric medium is inhomogeneous.

Generalization of Coulomb's Law in an Inhomogeneous Dielectric Medium

To model the electrostatic interaction in a protein solvent system the complex-shaped molecular surface of the protein, acting as solute and being surrounded by a solvent (water), is considered as inhomogeneous dielectric medium. In this case two different values of the dielectric constants are used for modelling protein and solvent dielectric implicitly. As described in the introduction, a dielectric constant of $\varepsilon_{in} = 4$ is used for the protein volume and of $\varepsilon_{out} = 80$ for the solvent, reflecting the large polarization because of water due to the large dipole of water molecules.

In general, it is not possible to compute the electrostatic potential analytically as it can be done for homogeneous dielectric media without interfaces or surfaces, eq. (3). Complex molecular models have to be computed numerically solving the Poisson-Boltzmann equation by using techniques like Finite Difference (FD), Finite Element (FE) or Boundary Element (BE) methods which will be explained beginning at page 26 ("*Discretizing the Linearized Poisson-Boltzmann Equation*"). Furthermore, grid artefacts will be discussed (see chapter "*Discretization Pitfalls*").

An exception to this is the existence of an analytical solution if the computation is done for one point charge in a dielectric medium with spherical symmetry. With vanishing ion concentration this so-called Born ion model has the analytical solution

$$\Delta G_{Born} = 694.6773088 \cdot \frac{z^2}{r} \cdot \left(\frac{1}{\varepsilon_{out}} - \frac{1}{\varepsilon_{in}} \right) \left[\frac{kJ}{mol} \right], \qquad (4)$$

where z is the charge number of the point charge with radius r. The derivation of the Born ion model for the case of non-vanishing ion concentration is presented in appendix B.

Gauss's Law and the Poisson Equation

Gauss's law is one of the four fundamental Maxwell's equations. Gauss's law describes an electrostatic flux, generated by point charges, through a closed, arbitrarily shaped surface, e.g. a molecular surface. Gauss's law for electricity has the form

$$\nabla \cdot \mathbf{D} = \rho\,, \tag{5}$$

where ρ is the charge density of point charges in a dielectric medium and $\mathbf{D}$ is the so-called electrostatic displacement field. $\mathbf{D}$ is the electrostatic field $\mathbf{E}$, which is scaled by the constant ε which is defined as $\varepsilon = \varepsilon_r \varepsilon_0$, where ε_0 is the vacuum permittivity and ε_r a constant depending on the polarizability of the medium. Thus,

$$\mathbf{D} = \varepsilon \mathbf{E}\,, \tag{6}$$

where

$$\mathbf{E} = -\vec{\nabla}\Phi\,. \tag{7}$$

One may imagine creating an electric field $\mathbf{E}$. The vector $\mathbf{E}$ points from positive to negative atomic charges. Applying the nabla operator to $\mathbf{E}$, i.e. $\nabla \cdot \mathbf{E}$, yields $-4\pi\rho$, where ρ is the charge density.

Calculating the volume integral of the charge density ρ, yields $\iiint_\Omega \rho \mathrm{d}V = Q$.

An alternative way of expressing Gauss's law in its integral form is

$$\oiint_S \mathbf{E} \cdot \mathrm{d}\mathbf{S} = \frac{1}{\varepsilon} \iiint_\Omega \rho \mathrm{d}V = \frac{Q}{\varepsilon}.$$

This equation describes the overall flux through the entire closed surface S to equal the total charge inside the entire volume.

Replacing $\mathbf{D}$ in eq. (5) and rearranging the equation leads to

$$\vec{\nabla} \bullet \mathbf{D} = -\vec{\nabla} \bullet \left(\varepsilon \vec{\nabla} \Phi \right) = \rho\,, \tag{8}$$

where Φ is the electrostatic potential. In a homogeneous dielectric medium ε is a constant. In an inhomogeneous medium the dielectric constant is a spatially dependent function, i.e. $\varepsilon(\vec{r})$. More details are given in "*Solvent Model Regions*" page 25.

For easier comparison with textbook notation, Gaussian units (*cgs* units) are used. A factor of 4π arises from Gauss's law for spherical symmetries [11], [12], leading to:

$$\nabla\varepsilon(\vec{r})\nabla\Phi(\vec{r}) = -4\pi \cdot \rho(\vec{r}). \qquad (9)$$

This last equation is known in mathematics as Laplace's equation if $\varepsilon = \varepsilon_0$ and in physics as Poisson's equation.

The Debye-Hückel Approximation

The Debye-Hückel approximation can be used to generalize the Poisson equation with regard to ion concentration. It is important to create a realistic natural environment for proteins. By introducing ions (e.g. Na^+Cl^-) into the water the electric conductivity is increased. The charge carriers move in the water depending on the concentration of the electrolyte in solution.

Modelling ion concentration realistically is difficult, because a large number of ions move rapidly and rearrange depending on the surroundings of every single ionic charge. Using the Debye-Hückel approximation, discrete ion positions are neglected in favour of using an implicit ion description. The Debye-Hückel approximation describes ions as spheres, with constant charges, which generate a radial electrostatic potential (see Fig. 3).

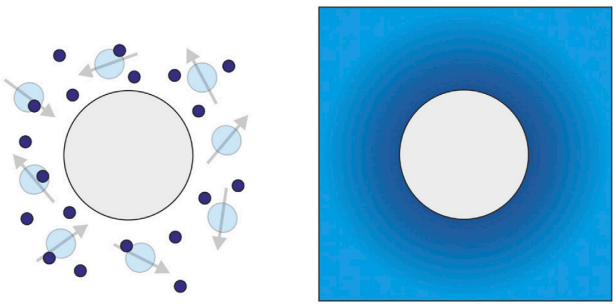


Fig 3. Discrete (left) and continuum model (right) for non-vanishing ion concentration. The *grey circle* represents a single ion of given radius (or atoms of a whole protein); left: *circles with arrows* represent water molecules with their dipole-moment vectors, *points* are discrete ions; right: the *blue area* represents implicit water with implicit ions.

Many physical effects play a role in a correct description of the overall influence of the ions. Therefore, the validity of the Debye-Hückel model is limited. This extension holds true for low ion concentrations up to 0.2 M (molar concentration), where Coulomb interactions between the ions are weak.

To derive the Debye-Hückel approximation, the Poisson equation, eq. (9), with variable $\epsilon(\vec{r})$ and $\rho(\vec{r})$ is used as starting point:

$$\nabla\varepsilon(\vec{r})\nabla\Phi(\vec{r}) = -4\pi\cdot\rho(\vec{r}),$$

where the charge density $\rho(\vec{r})$ is split into two parts [13]:

$$\rho(\vec{r}) = \rho^{i}(\vec{r}) + \rho^{f}(\vec{r}). \tag{10}$$

In the above sum, $\rho^{f}(\vec{r})$ denotes the fixed charges of known point charges and $\rho^{i}(\vec{r})$ denotes an unknown non-linear ion contribution to the effective charge density. This is determined by a factor $\kappa^{2}(\vec{r})$ which will be derived and discussed in the following.

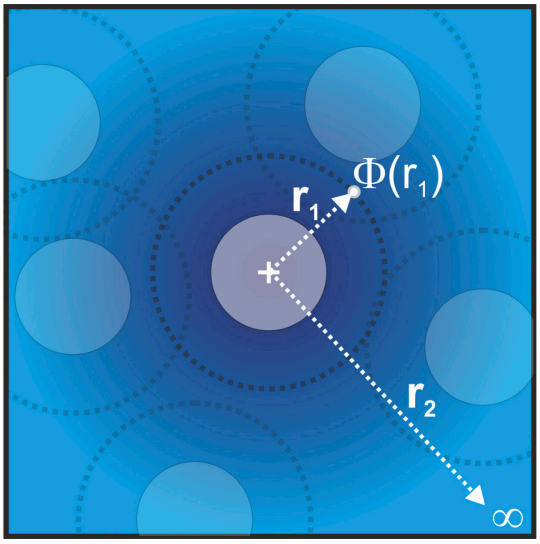


Fig 4. The central ion acts as a representative for every discrete ion. The centre of the ion is denoted by the white cross and the van-der-Waals radius of the central ion is denoted by the grey circle. The ion density of this central ion is given by the ion distribution $n_j(\vec{r}_1)$. The ion distribution at infinity with $\vec{r}_2 \to \infty$ (electrostatic potential at infinity is zero) is given by n_j^0.

In general, the effect of the ions is included into Poisson's equation by deriving an expression for the ion charge $\rho^i(\vec{r})$ in eq. (10). To achieve this, an ion distribution function $n_j(\vec{r})$ is used, (Fig 4). The central ion acts as a representative for every ion in the solvent.

The electrostatic potential energy, g_j, to move an ion from infinity ($r_2 = \infty$ with $\Phi(\vec{r}_2) = 0$) to a point r_1 (r_1 with $\Phi(\vec{r}_1) = y$) toward the central ion is defined by Coulomb's law, eq. (2), $g_j = z_j \cdot e \cdot \Phi(\vec{r}_1)$, where z_j is the charge number and e the elementary charge. As mentioned before, the Debye-Hückel approximation is valid for low ion concentrations up to 0.2 M. This is equal to the assumption that $g_j = z_j \cdot e \cdot \Phi(\vec{r}) \ll k_B \cdot T$, where T is the temperature in Kelvin and k_B the Boltzmann constant.

The overall ion distribution n_j^0 (ion distribution at infinity) can be expressed by

$$n_j^0 = m_j \cdot N_A, \tag{11}$$

where m_j is the mass of ion type j and N_A the Avogadro constant given in $[1/mol]$.

Two types of ions are assumed with the same radius and opposite net unit charges. The ion distribution, n_j, of the central ion with radius r_j (see Fig 4) is Boltzmann-distributed (i.e. temperature-dependent) and can be computed using the laws of thermodynamics:

$$n_j = n_j^0 \cdot \exp\left(-\frac{g_j}{k_B \cdot T}\right). \tag{12}$$

The overall ion charge density is related to the sum of all ion distributions multiplied by their charge $(= z_j \cdot e)$. Since two oppositely charged types of ion are assumed to be present in the solvent, the charge density $\rho^i(\vec{r})$ is

$$\rho^i(\vec{r}) = \sum_j^N n_j \cdot z_j \cdot e = n_j^+ \cdot z_j \cdot e - n_j^- \cdot z_j \cdot e = \left[\exp\left(-\frac{+g_j}{k_B \cdot T}\right) - \exp\left(-\frac{-g_j}{k_B \cdot T}\right)\right] \cdot z_j \cdot e \cdot n_j^0, \tag{13}$$

where j denotes the ion type and N the number of ion types. As mentioned before, the Debye-Hückel approximation models 1:1 electrolytes. Therefore, $N = 2$ and n_j^+ denotes the ion with positive net charge and n_j^- the ion with negative net charge.

Now the function $\sinh(\mathrm{x}) = \frac{1}{2}\left(-\mathrm{e}^{-x} + \mathrm{e}^{x}\right)$ is used to rewrite eq. (13):

$$\begin{aligned} \rho^i(\vec{r}) &= -2 \cdot \frac{1}{2}\left[-\exp\left(-\frac{g_j}{k_B \cdot T}\right) + \exp\left(+\frac{g_j}{k_B \cdot T}\right)\right] \cdot n_j^0 \cdot z_j \cdot e \\ &= -2 \cdot n_j^0 \cdot z_j \cdot e \cdot \sinh\left(\frac{z_j \cdot e \cdot \Phi(\vec{\mathrm{r}})}{k_B \cdot T}\right). \end{aligned} \tag{14}$$

Inserting this expression into eq. (9) and making use of eq. (10) results in

$$\begin{aligned} \nabla \varepsilon(\vec{r}) \nabla \Phi(\vec{r}) &= -4\pi \cdot \rho(\vec{\mathrm{r}}) = -4\pi\left(\rho^i(\vec{r}) + \rho^f(\vec{r})\right) \\ &= 8\pi \cdot n_j^0 \cdot z_j \cdot e \cdot \sinh\left(\frac{z_j \cdot e \cdot \Phi(\vec{\mathrm{r}})}{k_B \cdot T}\right) - 4\pi \cdot \rho^f(\vec{r}) \end{aligned} \tag{15}$$

A linearization of this equation can be achieved by using the Taylor expansion, $\sinh(\mathrm{x}) = \mathrm{x} + \frac{x^3}{6} + \frac{x^5}{120} + O\left(\mathrm{x}^7\right)$, and truncating the series expansion after the first term yielding

$$\begin{aligned} \nabla\varepsilon(\vec{r})\nabla\Phi(\vec{r}) &= -4\pi\cdot\rho(\vec{\mathrm{r}}) = -4\pi\left(\rho^i(\vec{r}) + \rho^f(\vec{r})\right) \\ &= 8\pi\cdot n_j^0\cdot z_j^2\cdot e^0\,\frac{\Phi(\vec{\mathrm{r}})}{k_B\cdot T} - 4\pi\cdot\rho^f(\vec{r}) \end{aligned} \tag{16}$$

Introducing the "ionic strength" I as defined by Lewis and Randall [14]:

$$I = \frac{1}{2}\cdot\sum_j^N z_j^2\cdot n_j^0 \tag{17}$$

and the inverse Debye length parameter κ

$$\kappa^2(\vec{r}) = \frac{8\pi\cdot e^2\cdot I}{k_B\cdot T} \tag{18}$$

the linear Poisson-Boltzmann equation (*l*PBE) is obtained:

$$\nabla\left(\varepsilon(\vec{\mathrm{r}})\nabla\vec{\Phi}\right) - \kappa^2(\vec{\mathrm{r}})\Phi(\vec{\mathrm{r}}) = -4\pi\cdot\rho^f(\vec{r}). \tag{19}$$

With this linearization the superposition property is obtained. This equation is discussed in more detail in the next section, "*The Poisson-Boltzmann Equation*"

The Poisson-Boltzmann Equation

It is crucial to have a mathematical model which reproduces the real situation. The Poisson-Boltzmann equation (PBE) is a second-order partial differential equation, which describes the electrostatics of biomolecular systems in solution. For low ion concentrations, a linear description of the PBE the linear Poisson-Boltzmann equation (*l*PBE) is used. One part of the PBE is the Poisson equation, see eq. (9), which describes the electrostatics of proteins in solution without ions by considering atoms as point charges. Debye and Hückel improved this in 1923 by adding the Boltzmann part to the Poisson equation (see previous section). The extension makes it possible to add ions to the implicit solvent model by using statistical thermodynamic averaging techniques. Ions are omnipresent and can significantly influence results. The non-linear Poisson-Boltzmann equation (*n*PBE) possesses the following form:

Poisson equation		Boltzmann part		Charge density
$\nabla(\varepsilon(\vec{r})\nabla\Phi(\vec{r}))$	$-$	$\kappa^2(\vec{r})\cdot\sinh(\Phi(\vec{r}))$	$=$	$-4\pi\cdot\rho^f(\vec{r})\,.$

Applying the approximation $\sinh(x)\approx x$, the linearized PBE (*l*PBE) is obtained:

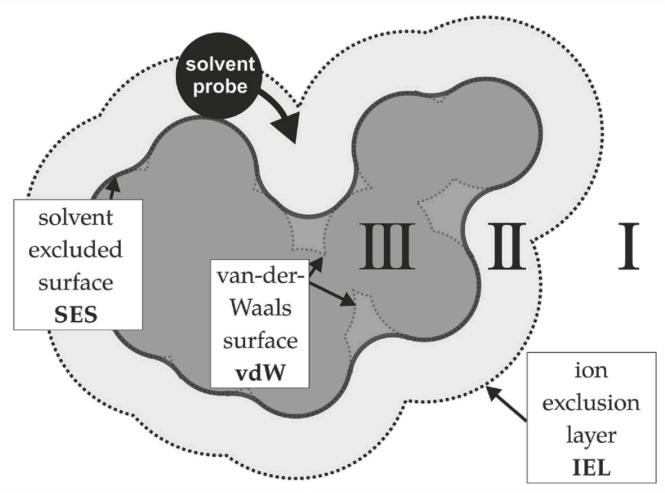
Poisson equation		Boltzmann part		Charge density
$\nabla(\varepsilon(\vec{r})\nabla\Phi(\vec{r}))$	$-$	$\kappa^2(\vec{r})\cdot\Phi(\vec{r})$	$=$	$-4\pi\cdot\rho^f(\vec{r})\,.$

Most textbooks on electrostatics use Gaussian (*cgs*) units. To obtain Gaussian units, κ^2 is multiplied with 4π. For the correct units of energy, ρ^f has to be multiplied with $e/(k_B\cdot T)$ (see appendix A for details).

In the linear approximation of the PBE, the electrostatic potentials can be superimposed. Computations of pK_A are one example of using the *l*PBE, because energy terms are split up and combined again in the final results.

Solvent Model Regions

It was shown how to derive the PBE from first principles to describe the implicit solvent model, which includes also ions. Due to the fact that ions are located in the environment and are separated from the molecular surface by the effective ion radius, different model regions are defined. Surfaces separate those regions around the protein molecule in which different values of ε and κ are used (see Fig 5 and Chapter II). Generating the model to solve the PBE with different methods is described in the next section, "*Discretizing the Linearized Poisson-Boltzmann Equation*."



For the dielectric constants, the following values are used:

$$\varepsilon(\vec{r}) = \begin{cases} \varepsilon_{in} = 4 & \text{if } \vec{r} \in \text{III} \\ \varepsilon_{out} = 80 & \text{if } \vec{r} \in \text{I} \vee \text{II} \end{cases}$$

The ion concentration is accounted for by:

$$\kappa(\vec{r}) = \begin{cases} 0 & \text{if } \vec{r} \in \text{III} \vee \text{II} \\ \kappa(\vec{r}) & \text{if } \vec{r} \in \text{I} \end{cases}$$

Fig 5. Regions of the implicit solvent model; the different surfaces are the ion exclusion layer (*IEL*), the solvent excluded surface (*SES*), and the van-der-Waals surface (*vdW*). The regions are numbered I, II and III.

Discretizing the Linearized Poisson-Boltzmann Equation

There are different methods to solve the *l*PBE numerically [15]. An introduction into some commonly used methods, namely the Finite Difference (FD) method, the Boundary Element (BE) method, and the Finite Element (FE) method, is given below. This work focuses on the comparison of FE and FD methods. BE methods are still under development, while FD methods are well-established and have been in for many years. It will be shown how to use the FE method to perform electrostatic computations for proteins with mFES (*m*olecular *F*inite *E*lement *S*olver), a new FE program which has been developed in this work. First, the discretization into a linear equation system is shown for all three methods. Then, a method to solve these linear equation systems numerically is described.

The Finite Difference Method

Using the FD method, a protein is discretized on an equidistant grid with a lattice constant h given in [Å]. In general, every Cartesian direction in space may have a different h; in the following, h is assumed to be equal in all Cartesian directions (simple cubic grid). In recent applications, it is possible to refine the grid constant adaptively until its length attains a minimum value [16]. This feature is the so-called focusing technique used in the FD method [6], [17], [18] (Fig 6). Another approach to make the computation of huge molecular systems feasible is a parallel focusing method (Fig 7).

To simplify the discretization steps, eq. (19) is rewritten as

$$\nabla(\varepsilon(\vec{r})\nabla\vec{\Phi}) = \kappa^2\Phi(\vec{r}) - 4\pi\cdot\rho^f(\vec{r}) = \kappa^2\Phi(\vec{r}) - \beta(\vec{r})\,. \qquad (20)$$

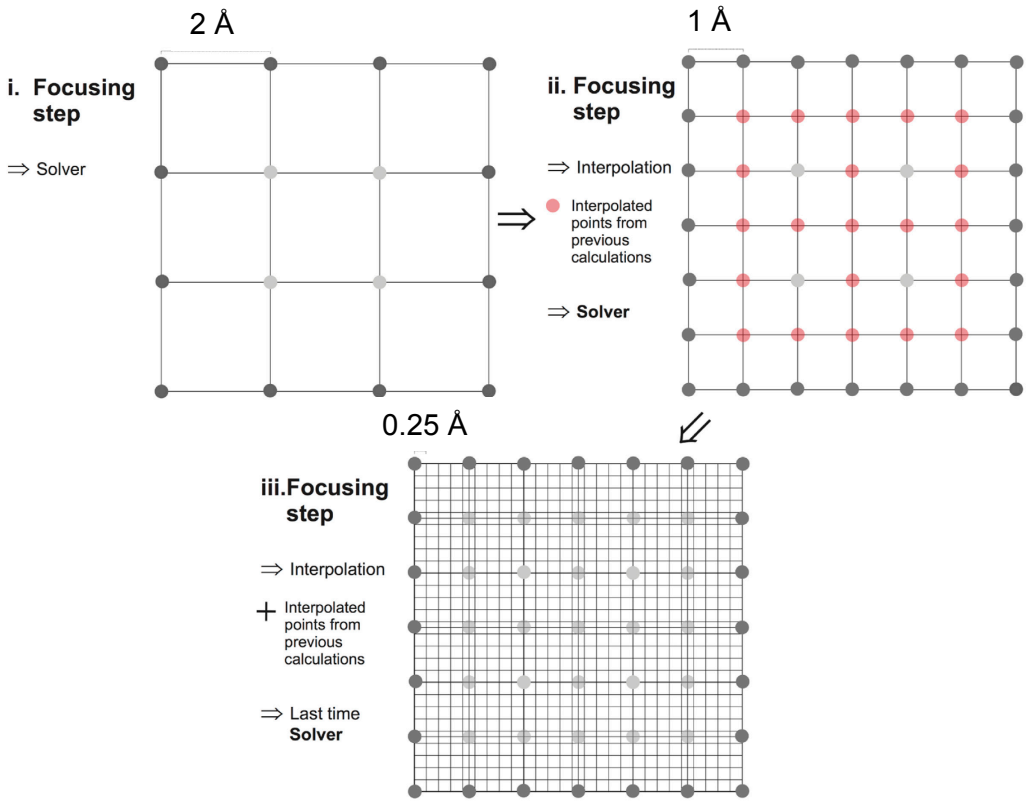


Fig 6. Example of a three-step focusing procedure using the FD method. In the first step, a grid constant of 2 Å is used. The solution of the first step provides the starting point for the new boundary of the second focusing step with a finer resolution of 1 Å. A last focusing step is done with a grid constant of 0.25 Å. *n* may be very high at the last focusing step.

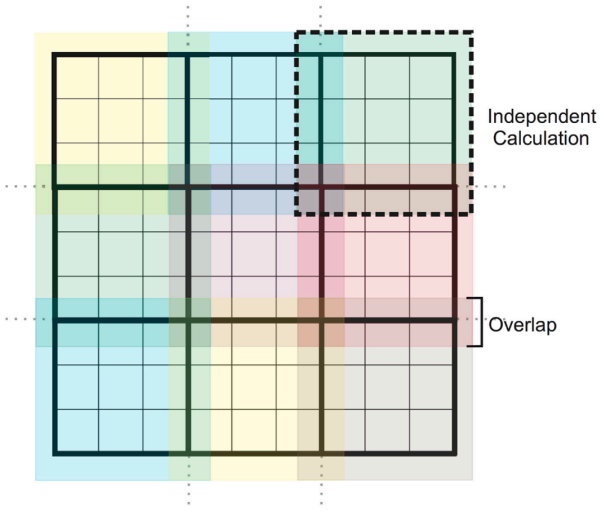


Fig 7. Parallel focusing technique combining the overlapping partial solutions obtained by the FD method. Independent solutions are overlapping by a given overlap region.

Furthermore, the following notations are used to simplify the formulation of the equations:

$$\Phi(\vec{r}_{i,j,k}) \equiv \Phi_{i,j,k} \; ; \; \beta(\vec{r}_{i,j,k}) \equiv \beta_{i,j,k} \; ; \; \varepsilon(\vec{r}_{i+\frac{1}{2},j,k}) \equiv \varepsilon_{i+\frac{1}{2},j,k} \; ; \; \Delta_i \Phi_{i+\frac{1}{2},j,k} \equiv \Phi_{i+1,j,k} - \Phi_{i,j,k} \, .$$

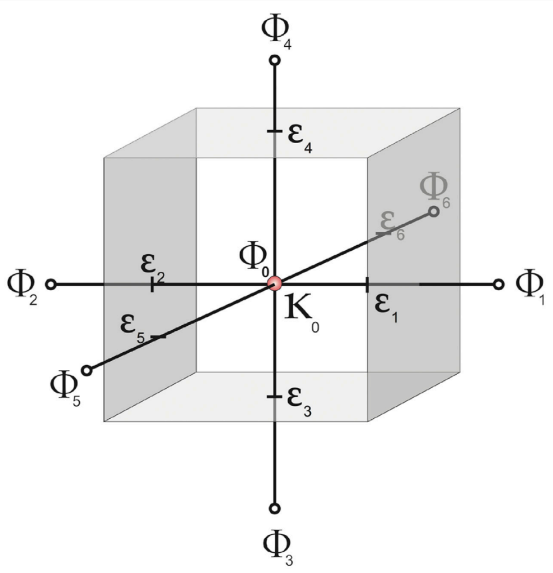


Fig 8. One cell of an equidistant cubic grid. Φ_0 is computed using neighbouring grid potentials and dielectric constants at half-grid points, see eq. (22).

The dielectric constant $\varepsilon_{i+\frac{1}{2},j,k}$ is defined between the two grid points (i, j, k) and (i+1, j, k) forming a second half step grid (see Fig 8). On a grid, the charge density corresponding to a point charge q at position (i, j, k) is $\rho = \frac{q}{h^3}$ and, thus, $\beta_{i,j,k} = 4\pi\rho = \frac{4\pi q}{h^3}$. If the charge is not located exactly at a grid point, the charge is redistributed among the nearest grid points by interpolation.

Using grid constant h, the discretized derivatives are:

$$(\varepsilon\Delta_i\Phi)_{i+\frac{1}{2},j,k} - (\varepsilon\Delta_i\Phi)_{i-\frac{1}{2},j,k} + (\varepsilon\Delta_i\Phi)_{i,j+\frac{1}{2},k} - (\varepsilon\Delta_i\Phi)_{i,j-\frac{1}{2},k} + (\varepsilon\Delta_i\Phi)_{i,j,k+\frac{1}{2}} - (\varepsilon\Delta_i\Phi)_{i,j,k-\frac{1}{2}}$$
$$= h^2(\bar{\kappa}^2_{i,j,k}\Phi_{i,j,k} - \beta_{i,j,k}).$$

This series can be rewritten as

$$\varepsilon_{i+\frac{1}{2},j,k}(\Phi_{i+1,j,k}-\Phi_{i,j,k})-\varepsilon_{i-\frac{1}{2},j,k}(\Phi_{i,j,k}-\Phi_{i-1,j,k})+\varepsilon_{i,j+\frac{1}{2},k}(\Phi_{i,j+1,k}-\Phi_{i,j,k})$$
$$-\varepsilon_{i,j-\frac{1}{2},k}(\Phi_{i,j,k}-\Phi_{i,j-1,k})+\varepsilon_{i,j,k+\frac{1}{2}}(\Phi_{i,j,k+1}-\Phi_{i,j,k})-\varepsilon_{i,j,k-\frac{1}{2}}(\Phi_{i,j,k}-\Phi_{i,j,k-1})$$
$$=h^2(\overline{\kappa}^2_{i,j,k}\Phi_{i,j,k}-\beta_{i,j,k}).$$

Omitting those indices that are merely i, j or k yields

$$\varepsilon_{i+\frac{1}{2}}(\Phi_{i+1}-\Phi)-\varepsilon_{i-\frac{1}{2}}(\Phi-\Phi_{i-1})+\varepsilon_{j+\frac{1}{2}}(\Phi_{j+1}-\Phi)$$
$$-\varepsilon_{j-\frac{1}{2}}(\Phi-\Phi_{j-1})+\varepsilon_{k+\frac{1}{2}}(\Phi_{k+1}-\Phi)-\varepsilon_{k-\frac{1}{2}}(\Phi-\Phi_{k-1})-h^2\,\overline{\kappa}^2\Phi=-h^2\beta.$$

Solving this equation with respect to Φ yields

$$-\Phi(h^2\overline{\kappa}^2+\varepsilon_{i+\frac{1}{2}}+\varepsilon_{i-\frac{1}{2}}+\varepsilon_{j+\frac{1}{2}}+\varepsilon_{j-\frac{1}{2}}+\varepsilon_{k+\frac{1}{2}}+\varepsilon_{k-\frac{1}{2}})$$

$$=-(\varepsilon_{i+\frac{1}{2}}\Phi_{i+1}+\varepsilon_{i-\frac{1}{2}}\Phi_{i-1}+\varepsilon_{j+\frac{1}{2}}\Phi_{j+1}+\varepsilon_{j-\frac{1}{2}}\Phi_{j-1}+\varepsilon_{k+\frac{1}{2}}\Phi_{k+1}+\varepsilon_{k-\frac{1}{2}}\Phi_{k-1})-h^2\beta\,.$$

Using the equation $h^2\cdot\beta_{i,j,k}=h^2\cdot\frac{4\pi q}{h^3}=\frac{4\pi q}{h}$ a solution for every $\Phi_{i,j,k}$ is

$$\Phi=\frac{\frac{4\pi q}{h}+\varepsilon_{i+\frac{1}{2}}\Phi_{i+1}+\varepsilon_{i-\frac{1}{2}}\Phi_{i-1}+\varepsilon_{j+\frac{1}{2}}\Phi_{j+1}+\varepsilon_{j-\frac{1}{2}}\Phi_{j-1}+\varepsilon_{k+\frac{1}{2}}\Phi_{k+1}+\varepsilon_{k-\frac{1}{2}}\Phi_{k-1}}{\varepsilon_{i+\frac{1}{2}}+\varepsilon_{i-\frac{1}{2}}+\varepsilon_{j+\frac{1}{2}}+\varepsilon_{j-\frac{1}{2}}+\varepsilon_{k+\frac{1}{2}}+\varepsilon_{k-\frac{1}{2}}+(h\overline{\kappa})^2}. \qquad (21)$$

With the following abbreviations

$$6\hat{\varepsilon}_{ijk}=\varepsilon_{i+\frac{1}{2}}+\varepsilon_{i-\frac{1}{2}}+\varepsilon_{j+\frac{1}{2}}+\varepsilon_{j-\frac{1}{2}}+\varepsilon_{k+\frac{1}{2}}+\varepsilon_{k-\frac{1}{2}}+(h\overline{\kappa})^2$$

and $$\hat{\Phi}_{a\pm1}=\varepsilon_{\alpha+\frac{1}{2}}\Phi_{\alpha+1}+\varepsilon_{\alpha-\frac{1}{2}}\Phi_{\alpha-1},\ \alpha\in\{i,j,k\}$$

and one obtains

$$\Phi=\frac{\frac{4\pi q}{h}+\left(\hat{\Phi}_{i\pm1}+\hat{\Phi}_{j\pm1}+\hat{\Phi}_{k\pm1}\right)}{6\hat{\varepsilon}_{i,j,k}}=\frac{4\pi\cdot\rho^f\cdot h^2+\left(\hat{\Phi}_{i\pm1}+\hat{\Phi}_{j\pm1}+\hat{\Phi}_{k\pm1}\right)}{6\hat{\varepsilon}_{i,j,k}}. \qquad (22)$$

Using eq. (22) it is possible to build a linear equation system $\mathbf{A}\vec{x} = \vec{b}$, where $\mathbf{A}$ is the coefficient matrix which models the protein environment, $\vec{x}$ is the unknown potential vector that has to be determined, and $\vec{b}$ is a charge vector with known, fixed atom point charges ρ^f which are interpolated to the grid positions by trilinear interpolation. The number of all equations for a given grid with n grid points in each Cartesian direction is n^3 (Fig 9, [15]).

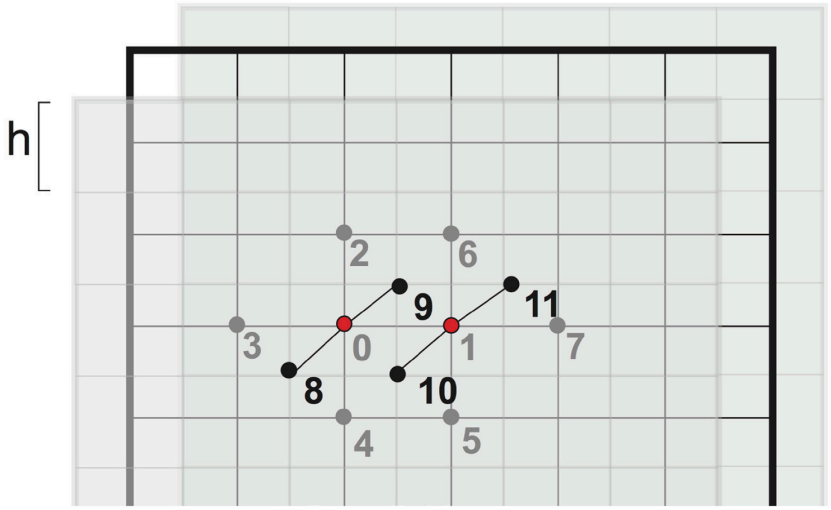


Fig 9. General discretization procedure to build a linear equation system using the FD method in three dimensions. An exemplary numbering of neighbouring grid points is given which does not have to be continuous; *h* is the grid constant.

In two dimensions, every row in the coefficient matrix $\mathbf{A}$ describes the connectivity for one grid point. This means that each row has a length of n grid points. In three dimensions, every grid point is connected with six neighbouring grid points (in Fig 9, grid point 0 is connected with 3, 1, 2, 4, 8, 9 and grid point 1 with 0, 7, 6, 5, 10, 11). If a grid point is not connected with another point, the corresponding value in the coefficient matrix is zero. The coefficient matrix is sparse. This allows to use fast algorithms to solve the *l*PBE. However, it is still necessary to reorder the matrix elements so that non-vanishing elements are close to the diagonal to generate a band-like matrix.

The Boundary Element Method

This section explains how the molecular model is discretized using the Boundary Element (BE) method [19]–[21]. The BE method uses triangles (or polygons with more vertices) to model molecular surfaces without the need of filling up space with other elements like tetrahedrons. In the following section, the molecular system is split into region Ω_{in} (region III in Fig 5) which is the region of the protein interior and Ω_{out} (regions I+II in Fig 5) which denotes the exterior solvent. A BE description for a one-surface model without cavities or an ion exclusion layer, IEL is shown in the following. It is possible to add more surfaces to the BE method which is explicated by Altman et al. [22].

Using the Poisson equation, the second derivative of potential $\Phi(\vec{r})$ inside Ω_{in} follows the superposition principle and writes as

$$\Delta\Phi_{in}(\vec{r}) = -\sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} \delta(\vec{r} - \vec{r}_{in,k}), \tag{23}$$

where $\delta(..)$ is the Dirac delta function, q_k is the charge of atom k, N_q is the number of charges placed in region Ω_{in}, $\vec{r}_{in,k}$ is the position of charge k, and ε_{in} is the dielectric constant used for the protein moiety.

The Poisson equation is augmented with the Boltzmann term in region Ω_{out} to include the effect of ionic strength:

$$\Delta\phi_{out}(\vec{r}) = \kappa^2 \phi_{out}(\vec{r}).$$

The Debye screening parameter κ has been introduced by eq. 18,

$$\kappa^2 = \frac{8\pi \cdot e^2 \cdot I}{\varepsilon_{out} \cdot k_B T},$$

where I is the ionic strength, e the elementary charge, k_B the Boltzmann constant, T the temperature given in Kelvin, and ε_{out} the dielectric constant used for the solvent.

At the dielectric boundary between the exterior and the interior region, i.e. molecular surface (solvent excluded surface, SES), continuity exists [23]. For all points on the surface, $\vec{r} \in S$:

$$\Phi_{in}(\vec{r}) = \phi_{out}(\vec{r}) \text{ and } \varepsilon_{in} \frac{\partial \Phi_{in}}{\partial \vec{n}}(\vec{r}) = \varepsilon_{out} \frac{\partial \phi_{out}}{\partial \vec{n}}(\vec{r}), \tag{24}$$

where $\vec{n}$ is the normal vector pointing outwards from the molecular surface towards the solvent.

In 1991, Juffer et al. [23] derived the general solution for $\Phi(\vec{r})$ using Green's theorem[1]:

$$\Phi(\vec{r}) = \oint_{\Omega_{in}} \left[G_0(\vec{r},\vec{r}_{in}) \cdot \frac{\partial \Phi}{\partial \vec{n}}(\vec{r}_{in}) - \frac{\partial G_0}{\partial \vec{n}}(\vec{r},\vec{r}_{in}) \cdot \Phi(\vec{r}_{in}) \right] \mathrm{d}\Omega_{in} + \sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} G_0(\vec{r},\vec{r}_{in,k}) \; ; \; \vec{r} \in \Omega_{in} \text{ (protein)},$$

$$\phi(\vec{r}) = \oint_{\Omega_{out}} \left[-G_\kappa(\vec{r},\vec{r}_{out}) \cdot \frac{\partial \phi}{\partial \vec{n}}(\vec{r}_{out}) + \frac{\partial G_\kappa}{\partial \vec{n}}(\vec{r},\vec{r}_{out}) \cdot \phi(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} \; ; \; \vec{r} \in \Omega_{out} \text{ (solvent)}.$$

G_κ and G_0 are the Greens functions solving the singular Poisson equation in presence and absence of ions, respectively

$$G_\kappa(\vec{r},\vec{r}_{out}) = \frac{\exp(-\kappa \,|\, \vec{r} - \vec{r}_{out} \,|)}{4\pi \cdot |\, \vec{r} - \vec{r}_{out} \,|}, \; G_0(\vec{r},\vec{r}_{in}) = \frac{1}{4\pi \cdot |\, \vec{r} - \vec{r}_{in} \,|}. \tag{25}$$

The numerator of G_0 is unity because $\kappa = 0$ if $\vec{r} \in \Omega_{in}$ (see page 25, "*Solvent Model Regions*"). In the following, let $\vec{r}$ approach the surface S from both sides, S_{in} and S_{out}. This results in

$$\begin{aligned} \lim_{\vec{r} \to S_{in}} \Phi_{in}(\vec{\mathrm{r}}) = \frac{1}{2}\Phi_{in}(\vec{r}) + \oint_{\Omega_{in}} \left[G_0(\vec{r},\vec{r}_{in}) \frac{\partial \Phi_{in}}{\partial \vec{n}}(\vec{r}_{in}) - \frac{\partial G_0}{\partial \vec{n}}(\vec{r},\vec{r}_{in}) \Phi_{in}(\vec{r}_{in}) \right] \mathrm{d}\Omega_{in} \\ + \sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} G_0(\vec{r},\vec{r}_{in,k}) \end{aligned} \tag{26}$$

[1] $\int_V \left(G_0 \nabla^2 \hat{\Phi} - \hat{\Phi} \nabla^2 G_0 \right) \mathrm{d}V = \oint_\Omega \left(G_0 \frac{\partial \hat{\Phi}}{\partial \vec{n}} - \hat{\Phi} \frac{\partial G_0}{\partial \vec{n}} \right) d\Omega \; ; \; \hat{\Phi} = \phi$ if $\Omega = \Omega_{out}$ or $\hat{\Phi} = \Phi_{in}$ if $\Omega = \Omega_{in}$.

$$\lim_{\vec{r}\to S_{out}} \phi_{out}(\vec{r}) = \frac{1}{2}\phi_{out}(\vec{r}) + \oint_{\Omega_{out}} \left[-G_k(\vec{r},\vec{r}_{out}) \frac{\partial \phi_{out}}{\partial \vec{n}}(\vec{r}_{out}) + \frac{\partial G_k}{\partial \vec{n}}(\vec{r},\vec{r}_{out}) \phi_{out}(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} . \tag{27}$$

Eq. (26) can be rewritten:

$$\begin{aligned} &\Phi_{in}(\vec{r}) = \frac{1}{2}\Phi_{in}(\vec{r}) + \oint_{\Omega_{in}} \left[G_0(\vec{r},\vec{r}_{in}) \frac{\partial \Phi_{in}}{\partial \vec{n}}(\vec{r}_{in}) - \frac{\partial G_0}{\partial \vec{n}}(\vec{r},\vec{r}_{in}) \Phi_{in}(\vec{r}_{in}) \right] \mathrm{d}\Omega_{in} + \sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} G_0(\vec{r},\vec{r}_{in,k}) \\ &\Leftrightarrow \frac{1}{2}\Phi_{in}(\vec{r}) - \oint_{\Omega_{in}} \left[G_0(\vec{r},\vec{r}_{in}) \frac{\partial \Phi_{in}}{\partial \vec{n}}(\vec{r}_{in}) \right] \mathrm{d}\Omega_{in} - \oint_{\Omega_{in}} \left[\frac{\partial G_0}{\partial \vec{n}}(\vec{r},\vec{r}_{in}) \Phi_{in}(\vec{r}_{in}) \right] \mathrm{d}\Omega_{in} = \sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} G_0(\vec{r},\vec{r}_{in,k}). \end{aligned} \tag{28}$$

Using the continuity conditions, eq. (24), eq. (27) is modified further:

$$\begin{aligned} &\phi_{out}(\vec{r}) = \frac{1}{2}\phi_{out}(\vec{r}) + \oint_{\Omega_{out}} \left[-G_k(\vec{r},\vec{r}_{out}) \frac{\partial \phi_{out}}{\partial \vec{n}}(\vec{r}_{out}) + \frac{\partial G_k}{\partial \vec{n}}(\vec{r},\vec{r}_{out}) \phi_{out}(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} \\ &\Leftrightarrow \Phi_{out}(\vec{r}) = \frac{1}{2}\Phi_{out}(\vec{r}) + \oint_{\Omega_{out}} \left[-G_k(\vec{r},\vec{r}_{out}) \cdot \frac{\varepsilon_{in}}{\varepsilon_{out}} \frac{\partial \Phi_{out}}{\partial \vec{n}}(\vec{r}_{out}) + \frac{\partial G_k}{\partial \vec{n}}(\vec{r},\vec{r}_{out}) \Phi(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} \\ &\Leftrightarrow \frac{1}{2}\Phi_{out}(\vec{r}) + \int_{\Omega_{out}} \left[G_k(\vec{r},\vec{r}_{out}) \cdot \frac{\varepsilon_{in}}{\varepsilon_{out}} \frac{\partial \Phi_{out}}{\partial \vec{n}}(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} - \oint_{\Omega_{out}} \left[\frac{\partial G_k}{\partial \vec{n}}(\vec{r},\vec{r}_{out}) \Phi(\vec{r}_{out}) \right] \mathrm{d}\Omega_{out} = 0. \end{aligned} \tag{29}$$

The previous two steps are a rearrangement to facilitate the building of a linear equation system. To simplify the notation of the general form of the linear equation system, $S_{i,j}$ and $D_{i,j}$ are used as follows:

$$S_{i,j} \frac{\partial \Phi_{in}(\vec{c}_i)}{\partial \vec{n}} = \oint_{\Omega_j} \left[G_0\left(\vec{c}_i,\vec{r}_{in,j}\right) \frac{\partial \Phi_{in}}{\partial \vec{n}}\left(\vec{r}_{in,j}\right) \right] \mathrm{d}\Omega_j \,,\; D_{i,j}\Phi\left(\vec{\mathrm{c}}_i\right) = \oint_{\Omega_j} \left[\frac{\partial G_0}{\partial \vec{n}}\left(\vec{\mathrm{c}}_i,\vec{r}_{in,j}\right) \Phi_{in}\left(\vec{r}_{in,j}\right) \right] \mathrm{d}\Omega_j \tag{30}$$

$\vec{r}$ is replaced by $\vec{c}_i$ which denotes the centroid of triangular surface element i on the molecular surface. j is the index of a corner point of a triangular surface element i with position $\vec{r}_i$. With this information, a linear equation system for the description of the so-called non-derivative BE formalism is built for every element i on the molecular surface.

The general form of this linear equation system is as follows:

$$\begin{bmatrix} \left[\frac{1}{2}I + D^0_{i,j}\right] & \left[-S^0_{i,j}\right] \\ \left[\frac{1}{2}I - D^\kappa_{i,j}\right] & \left[\frac{\varepsilon_{out}}{\varepsilon_{in}} S^\kappa_{i,j}\right] \end{bmatrix} \begin{bmatrix} \Phi_j \\ .. \\ \frac{\partial \Phi_j}{\partial \vec{n}} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{N_q} \frac{q_k}{\varepsilon_{in}} G^0 \\ .. \\ 0 \end{bmatrix},$$

$$\rightarrow \mathbf{A}\,\vec{x} = \vec{b}.$$

where $G^0 = G_0(\vec{r}, \vec{r}_{in})$ and I is an identity matrix. $S^0_{i,j}$ and $S^\kappa_{i,j}$ are defined in eq. (30). $S^0_{i,j}$ and $S^\kappa_{i,j}$ use the Greens functions $G_0(\vec{r}, \vec{r}_{in})$ and $G_\kappa(\vec{r}, \vec{r}_{out})$, respectively, as defined in eq. (25). The four elements of the coefficient matrix $\mathbf{A}$ are submatrices which are built by running over the indices i and j. $S_{i,j}$ and $D_{i,j}$ are computed for every vertex so that the number of equations to be solved is $2n$, where n is the number of vertices. As for the FE method, boundary conditions have to be set where the solution for the potential vector is known.

The derivative BE formalism can be derived similarly to the non-derivative BE form by one additional differentiation in the direction of $\vec{n}_0 = \vec{n}(\vec{r})$ and the use of the continuity conditions, eq. (24). It is known that the non-derivative BE formalism can lead to singularities, whereas the derivative BE formalism is well-behaved [20].

In general, the coefficient matrix of the linear equation system for the BE method is smaller than that for the FD or FE method. On the other hand, $\mathbf{A}$ is denser, and the evaluation of $S_{i,j}$ and $D_{i,j}$ for every vertex of the molecular surface mesh is more challenging. There are very fast numerical methods to solve these derivatives, e.g., by Gauss quadrature. In addition, one may try to split the equations into an analytical and a non-analytical part before solving. Also fast multipole methods are used in the BE method. At present, the BE method is still under development [20], [24]–[28].

The Finite Element Method

The main focus of this work is on the FE method. The whole FE procedure has been described by Sakalli et al. [3]. Here, a summary of the new methodology and a brief introduction to the methods used by mFES (*m*olecular *F*inite *E*lement *S*olver) are given. Using the Finite Element method, it is possible to solve the *l*PBE very accurately. This is achieved by employing higher-order polynomials in the three Cartesian directions for discretizing the molecular model. The overall molecular model is built by filling the entire space with finite elements. In this work, a triangular mesh for the molecular surface and a tetrahedron mesh for the volume of the molecular system are used. Similar to the latter, the molecular surface of the protein is separated into an interior and an exterior region. While the boundaries in the FD method enclose cubes using a constant lattice space, in the FE method, an arbitrarily shaped boundary may be used. In this work, a sphere with large radius and low density of triangle points is used as asymptotic boundary because the electrostatic potential approaches zero at infinity. With mFES, it is possible to use more than one surface (e.g. SES, IEL, cavity surface region).

mFES uses LSMS (*L*evel-*S*et method for *M*olecular *S*urface generation), which was developed by Can et al. [2], to generate a fine-grained molecular surface, and NETGEN [1], which is being developed by J. Schöberl and coworkers, to generate a regularized molecular surface mesh and an optimized volume mesh. One has to keep in mind that covering volume completely with tetrahedrons is a computationally hard problem. The aim of this problem is to cover a specified volume regime completely with densely packed regular tetrahedrons. Using perfectly regular tetrahedrons, volume can be covered only up to 85% [29]. Therefore, the packing of space with tetrahedrons is an art. NETGEN faces similar problems when performing the volume meshing procedure.

mFES covers the molecular surface by triangles. The shape of the triangles is regularized using a predefined resolution, h, given in Ångström. This resolution is equivalent to the grid constant used in the FD method but is twice as accurate if using a second-order solution. The protein volume is filled by tetrahedrons with a resolution of 0.5 Å. In the outer region the resolution decreases towards the asymptotic boundary. It is challenging for a meshing algorithm to fill the volume regularly because each molecular surface element with its own constraints has to be meshed with tetrahedrons simultaneously for the inner and for the outer region. Neighbouring tetrahedrons touching the molecular surface from the inside and to the outside have congruent faces, so that the previously meshed molecular surface stays invariant. An asymptotic boundary surface is defined by using a sphere with the radius of 100,000 Å with a low resolution (23 grid points which corresponds to an edge length of nearly 80,000 Å). The molecule is placed in the centre of the sphere. Subsequently, the external volume is filled. The space between the molecular surface and the asymptotic boundary surface is meshed using tetrahedrons. This procedure starts from the triangles on the protein surface, enlarges the edge lengths of the tetrahedrons regularly by a grading parameter $g = 0.5$ and proceeds towards the triangles on the asymptotic sphere surface. The resulting tetrahedral mesh of the molecular model is optimized and simplified while keeping the triangles on the protein surface invariant. This step is important to get rid of ill-conditioned tetrahedrons (possessing vanishing volume) which can lead to singular behaviour of the linear equation solving the *l*PBE. Finally, the grid points of the tetrahedral mesh are used to discretize the *l*PBE and solve the linear equation system by a direct or iterative method. During the optimization procedure, tetrahedrons are assigned to quality classes and from the curve progression of the quality class assignment plot the user gets an idea about the quality of the final mesh.

Discretizing the FE Method

This subsection explains the determination of the electrostatic potential generated by a charge distribution $\rho(\vec{r})$ in an inhomogeneous dielectric medium using the Finite Element (FE) method. For this purpose, the *l*PBE which has been presented in eq. (19) has to be solved:

$$\vec{\nabla}(\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})) - \kappa^2(\vec{r})\Phi(\vec{r}) = -4\pi\,\rho^f(\vec{r}),$$

where $\varepsilon(\vec{r})$ is the dielectric constant and $\kappa^2(\vec{r})$ describes the ionic strength. In the following, the Poisson's equation with $\kappa^2(\vec{r}) = 0$ is considered.

For proteins, the charge distribution $\rho(\vec{r})$ is defined by N_q point charges q_k localized at the atom positions $\vec{\rho}_k$ yielding the Poisson's equation, using eq. (23) and eq. (9):

$$\vec{\nabla}(\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})) = -4\pi\sum_{k=1}^{N_q}\frac{q_k}{\varepsilon_{in}}\delta(\vec{r}-\vec{\rho}_k), \tag{31}$$

where $\delta(\vec{r})$ is the Dirac delta function.

For large and flexible proteins, a dielectric constant larger than unity is used, typically $\varepsilon_{in} = 4$ [5], [30]. A dielectric constant of $\varepsilon_{out} = 80$ is used for the solvent. A discussion about the dielectric constants can be found on page 13, "*Classical Electrostatics in Molecular Systems*."

In the following, a derivation of the weak form of Poisson's equation is presented. The weak form of Poisson's equation is solved by minimizing the residual

$$r^e(\vec{r}) = \vec{\nabla}(\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})) + 4\pi\,\rho(\vec{r}). \tag{32}$$

The minimization of eq. (32) is performed by summing up test functions $\omega(\vec{r})$ for the inner and outer region in the entire volume Ω:

$$\int_\Omega \omega(\vec{r})\vec{\nabla}(\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r}))\,\mathrm{d}\vec{r} = -4\pi\int_\Omega \omega(\vec{r})\,\rho(\vec{r})\,\mathrm{d}\vec{r}. \tag{33}$$

The volume Ω is a sum of small local domains Ω_k, i.e. $\Omega = \Sigma_k \Omega_k$, with each Ω_k being a tetrahedron. The left side of eq. (33) is replaced using the identity

$$\vec{\nabla}[\omega(\vec{r})\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})] = \varepsilon(\vec{r})\vec{\nabla}[\omega(\vec{r})]\,\vec{\nabla}[\Phi(\vec{r})] + \omega(\vec{r})\vec{\nabla}[\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})]\,. \tag{34}$$

Furthermore, Gauss's integral equation is used

$$\int_\Omega \vec{\nabla}[\omega(\vec{r})\varepsilon(\vec{r})\vec{\nabla}\Phi(\vec{r})]\mathrm{d}\vec{r} = \int_\Gamma \varepsilon(\vec{r})\,\omega(\vec{r})\,\vec{n}(\vec{r})[\vec{\nabla}\Phi(\vec{r})]\mathrm{d}\sigma\,, \tag{35}$$

where $\vec{n}(\vec{r})$ is a surface normal vector pointing outwards and Γ is the spherical boundary surface of the molecular system with a dirichlet boundary condition of $\Phi(\Gamma) = 0$.

Finally, using eq. (34) and eq. (35) the weak form is rewritten as

$$\int_\Omega \varepsilon(\vec{r})\vec{\nabla}[\omega(\vec{r})]\,\vec{\nabla}[\Phi(\vec{r})]\mathrm{d}\vec{r} = 4\pi\int_\Omega \rho(\vec{r})\,\omega(\vec{r})\,\mathrm{d}\vec{r} + \int_\Gamma \varepsilon(\vec{r})\omega(\vec{r})\,\vec{n}(\vec{r})[\vec{\nabla}\Phi(\vec{r})]\mathrm{d}\sigma\,, \tag{36}$$

which avoids a derivative of $\varepsilon(\vec{r})$.

As test functions $\omega_j(\vec{r})$, Lagrange interpolation polynomials are used, which are localized at individual grid points $\vec{r}_j$ to expand the electrostatic potential

$$\Phi(\vec{r}) = \sum_j \Phi_j\,\omega_j(\vec{r})\,. \tag{37}$$

The test functions $\omega_j(\vec{r})$ have the following properties. They are non-vanishing at their reference grid point $\vec{r}_j$ and the tetrahedrons $\Omega_k^{(j)}$ that possess $\vec{r}_j$ as corner point, but vanish at the other corner points of these tetrahedrons and outside of these tetrahedrons. The test functions are defined the next subsection.

Specifying the test function $\omega_j(\vec{r})$ in eq. (36) to refer to grid point *i* and inserting the expansion of eq. (37) in eq. (36), one obtains

$$\begin{aligned}&\sum_j\sum_k \int_{\Omega_k^{(i,j)}} \varepsilon(\vec{r})[\vec{\nabla}\omega_i(\vec{r})\;\vec{\nabla}\omega_j(\vec{r})]\mathrm{d}\vec{r}\;\Phi_j \\ &\quad = 4\pi\sum_k \int_{\Omega_k^{(i)}} \rho(\vec{r})\,\omega_i(\vec{r})\,\mathrm{d}\vec{r} + \sum_j\sum_k \int_{\Gamma_k^{(i,j)}} \varepsilon(\vec{r})\,\omega_i(\vec{r})[\vec{n}(\vec{r})\vec{\nabla}\omega_j(\vec{r})]\mathrm{d}\sigma\;\Phi_j.\end{aligned} \tag{38}$$

The sums in eq. (38) run over *k* and refer to tetrahedrons $\Omega_k^{(i,j)}$ that share the corner points $\vec{r}_j$ and $\vec{r}_i$ in common (or only $\vec{r}_i$ in case of $\Omega_k^{(i)}$). Introducing the abbreviations

$$\begin{aligned} \mathbf{A}_{ij} &= \sum_k \int_{\Omega_k^{(i,j)}} \varepsilon(\vec{r})[\vec{\nabla}\omega_i(\vec{r})\,\vec{\nabla}\omega_j(\vec{r})]\mathrm{d}\vec{r}, \quad (\vec{b})_i = 4\pi \sum_k \int_{\Omega_k^{(i)}} \rho(\vec{r})\omega_i(\vec{r})\,\mathrm{d}\vec{r}, \\ \mathbf{C}_{ij} &= \sum_k \int_{\Gamma_k^{(i,j)}} \varepsilon(\vec{r})\omega_i(\vec{r})[\vec{n}(\vec{r})\vec{\nabla}\omega_j(\vec{r})]\mathrm{d}\sigma, \end{aligned} \tag{39}$$

one is able to rewrite eq. (38) in matrix form:

$$(\mathbf{A}+\mathbf{C})\vec{\Phi} = \vec{b}. \tag{40}$$

$\mathbf{C}_{ij}$ vanishes because each triangle which contributes to the value of $\mathbf{C}_{ij}$ belongs to two different tetrahedrons except for tetrahedrons at the asymptotic boundary. The contributions of the triangles of two neighboring tetrahedrons cancel each other precisely, since the surface vectors $\vec{n}(\vec{r})$ of these triangles have opposite directions. If the electrostatic potential Φ vanishes at the asymptotic boundary surface, the surface integrals for triangles at the boundary can be neglected. Hence,

$$\mathbf{A}\vec{\Phi} = \mathbf{A}\vec{x} = \vec{b}. \tag{41}$$

The coefficient matrix $\mathbf{A}$ describes the molecular model. Each tetrahedron is represented within one row of the matrix with four or ten entries for first-order or second-order solution, respectively. Vector $\vec{b}$ on the right side of the equation describes the charge distribution. $\vec{\Phi}$ is the electrostatic potential vector whose elements refer to the tetrahedron corner points. The coefficient matrix $\mathbf{A}$ is sparse, since the matrix elements are non-zero only if $\vec{r}_i$ and $\vec{r}_j$ are equal or if they are adjacent grid points, i.e. being corners of the same tetrahedron.

Introducing Test Functions

The local test functions $\omega_j(\vec{r})$ used for the expansion, eq. (37), of the electrostatic potential $\Phi(\vec{r})$ are sums of Lagrange polynomials

$$\omega_{\mathrm{j}}(\vec{r}) = \sum_{\mathrm{k(j)}} p_{\mathrm{k(j)}}(\vec{r}) , \tag{42}$$

one for each tetrahedron, which has one corner at the grid point $\vec{r}_{\mathrm{j}}$. The coefficients of the Lagrange polynomials $p_{\mathrm{k(j)}}(\vec{r})$ are determined by the geometry of the tetrahedrons $\Omega_{\mathrm{k}}^{(\mathrm{j})}$, which possess the grid point $\vec{r}_{\mathrm{j}}$ as a corner point. In first order ($o = 1$) linear polynomials with four known coefficients are used yielding

$$p_{\mathrm{k(j)}}^{(1)}(\vec{r}) = c_0 + \sum_{\mathrm{i}=1}^{3} c_{\mathrm{i}}\, x_{\mathrm{i}} . \tag{43}$$

Using this definition, the electrostatic potential can be computed, eq. (37). Using second-order quadratic polynomials ($o = 2$), the geometry of the tetrahedron $\Omega_{\mathrm{k}}^{(\mathrm{j})}$ considered is defined by ten coefficients (see Fig 10):

$$p_{\mathrm{k(j)}}^{(2)}(\vec{r}) = c_0 + \sum_{\mathrm{i}=1}^{3} c_{\mathrm{i}}\, x_{\mathrm{i}} + \sum_{\mathrm{i}\geq\mathrm{l}=1}^{3} c_{\mathrm{i,l}}\, x_{\mathrm{i}} x_{\mathrm{l}} . \tag{44}$$

Numerical techniques for computing the elements in the coefficient matrix **A** using polynomial contributions of different tetrahedrons are described in [31], [32].

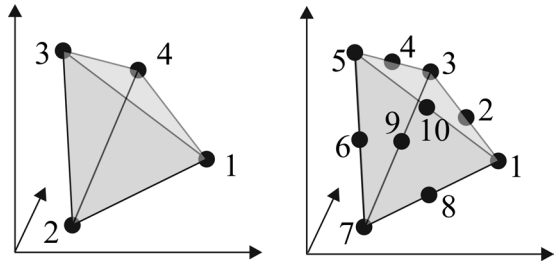


Fig 10. The left tetrahedron describes test functions with four control points (vertices). Using a higher-order polynomial, the description results in a formulation with ten control points, i.e. the six midpoints on the tetrahedron edges in addition to the corner points. Figure adopted from [3].

In applications described by Sakalli et al. [3] it is observed that the total number of grid points is increased by a factor of about 7, when changing from first- to second-order solutions.

Solving the Linear Equation System

In the previous section, a linear equation system (LES) was built ($\mathbf{A}\vec{x} = \vec{b}$.) which has to be solved. The coefficient matrix is obtained using different methods, like the FD, FE or BE method. To arrive at a unique solution, the boundary conditions have to be set. With the FE method, a vanishing electrostatic potential is used at the spherical boundary because it is possible to construct a boundary sphere which has a very large radius without much additional computational cost. Vanishing electrostatic potential at the asymptotic boundary is difficult to achieve using the FD method, because the space between the protein and the boundary is filled with a uniform grid point density in the same way as the protein interior. As a consequence the additional computational costs are high. To get rid of this pitfall, some efforts were made to approximate the boundary using a single (SDH) or multiple (MDH) Debye-Hückel approximation [33]. The Debye-Hückel approximation for the boundary potential $\Phi_i(\vec{r})$ writes as

$$\Phi_i(\vec{r}) = \sum_{j=1}^{N} \frac{q_j}{\varepsilon_{out} \mid \vec{r} - \vec{r}_j \mid} \cdot \exp\left(-\frac{\kappa}{\sqrt{\varepsilon_{out}}} \left| \vec{r}_j - \vec{r} \right| \right). \tag{45}$$

If the MDH is employed, N equals the total number of point charges in the molecular system with charge positions $\vec{r}_j$ and κ is the Debye screening parameter defined in eq. (18). If the SDH is employed, N equals one, and r_j is the pre-calculated centroid position of the protein, and Q is the total charge inside the protein. Normally, the SDH is used because of its simplicity and lower computational cost. One has to keep in mind that the SDH is exact for single charge in the center of a sphere (i.e. Born ion model; see appendix B). For a protein relation eq. (45) is only approximately valid. Techniques like focusing are used in the FD method to obtain accurate results (Fig 6 and Fig 7).

Methods to Solve the Linear Equation System

Solving a linear equation system is formally trivial if there are N equations with an equal number of unknowns. Nevertheless, there are different methods available to solve this problem for different cases. The choice of the algorithm depends on the structure of the coefficient matrix. In general, the solvers for a linear equation system make either use of (i) *direct methods* or of (ii) *iterative methods*. Direct methods are useful for matrices which are small or large but sparse. For large and dense matrices, iterative methods are used. They are able to solve a system approximately until the error is converged to a preset value. Depending on the implementation of these algorithms, they may run parallel or memory-efficient.

Direct Methods

Algorithms that solve a linear equation system with a direct method for a coefficient matrix that is quadratic and non-singular are the so-called *Gaussian elimination method* (GEM), the *LU-Factorization method* (LUM), and the *Cholesky decomposition method* (CDM). In the latter case the coefficient matrix must be also symmetric, which is fulfilled in all cases considered here. The *GEM* consists of a forward and a backward stage. By solving some unknowns, the algorithm builds an upper triangular coefficient matrix form. At the backward stage, using simple back substitutions to reduce the upper triangular matrix form, the algorithm produces the result of the linear equation system. The LU-decomposition method decomposes the coefficient matrix $\mathbf{A}$ by building a lower triangular part $\mathbf{L}$ and an upper triangular part $\mathbf{U}$. It holds that $\mathbf{LU} = \mathbf{A}$. Thus, the linear equation system can be rewritten as $\mathbf{A}\vec{x} = \vec{b} \Leftrightarrow \mathbf{LU}\vec{x} = \vec{b}$. This method is a variant of the Gaussian elimination method and can be solved similarly due to having a well-suited and simpler, decomposed coefficient matrix form. Other types of decomposition are for example the CDM

which is similar to the LUM but twice as efficient because of employing $\mathbf{LL}^* = \mathbf{A}$. In a first step, $\mathbf{L}\vec{y} = \vec{b}$ is computed using the forward-substitution method and in a second step, $\mathbf{L}^*\vec{x} = \vec{y}$ is computed using the back-substitution method.

Iterative Methods

The Jacobi Method. The simplest method to solve a linear equation system iteratively is the Jacobi method (JM). The JM is used if the diagonal of the coefficient matrix is positive-definite and possesses the largest values, which is true if the diagonal values are larger than the sum of their respective row values. In this case the result will converge although it might also converge if the diagonal does not dominate. Starting from an initial guess (e.g. zero solution), the neighboring equations are solved by

$$x_i^{r+1} = \frac{1}{a_{ii}}\left\{ b_i - \sum_{j \neq i}^{n} a_{ij} x_j^r \right\}, \tag{46}$$

where the next value x_t^{r+1} at position i of iteration $(r+1)$ is defined by the diagonal element a_{ii} and the solution b_i at same the position i. Every summand consists of a multiplication of a row value a_{ij} multiplied by the previous iteration result x_j^r at position j. This iterative procedure is repeated for the whole matrix several times until a predefined convergence criterion (error residual) is attained.

Gauss-Seidel Method. The Gauss-Seidel method (GSM) is similar to the Jacobi method in that the solution converges if the diagonal is dominant, i.e. the diagonal is larger than the sum of its row elements. This can be proved using the convergence theorem [34]. The coefficient matrix has to be quadratic. In some cases (e.g. a rectangular matrix) this can be achieved by inverting the coefficient matrix if possible, but inverting the matrix needs more CPU time. An iteration step of the GSM writes as

$$x_i^{r+1} = \frac{1}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{r+1} - \sum_{j=i+1}^{n} a_{ij} x_j^r \right\}. \tag{47}$$

Here, beside its dependence on the values of iteration r, the value x_i^{r+1} of iteration $r+1$ depends on the sum of the previously calculated x_j^{r+1} of the iteration step $r+1$. The GSM may be very efficient in terms of storage and memory requirements. It is fast for sparse matrices and it is possible to adjust the method if a predefined error is attained. One drawback is that even if the diagonal is dominant, in some cases this method might not converge.

Successive Over-Relaxation Method. The Successive over-relaxation method (SOR) takes the general form of the Gauss-Seidel method with an additional constant ω for the iteration step k, leading to

$$x_i^{r+1} = (1-\omega) x_i^r + \frac{\omega}{a_{ii}} \left\{ b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{r+1} - \sum_{j=i+1}^{n} a_{ij} x_j^r \right\}. \tag{48}$$

ω acts like a scaling factor for the vector, pointing towards the exact result. If $\omega = 1$, the SOR is equal to the GSM. The SOR is used because it is known that for some applications the solution converges faster. For example, the software MEAD (*M*acroscopic *E*lectrostatics with *A*tomic *D*etail) [35] uses $\omega = 1$ for $k = 1$ and a so-called Jacobi radius, $r_S = 0.989$, in every iteration step $k+1$ taking the following form:

$$\omega^{k+1} = \frac{1}{1 - 0.25\left(r_S^2 \cdot \omega^k\right)} \ .$$

Using the Chebyshev acceleration [35]–[37], one may reduce the number of iteration steps for faster convergence.

Conjugate Gradient Method. The Conjugate gradient method (CGM) is a fast and reliable method for solving large linear equation systems. It can be applied for a quadratic, positive-definite, symmetric coefficient matrix and is best used for sparse matrices.

In this context, conjugate means that vectors $\vec{u}$ and $\vec{v}$ fulfill the requirement

$$\vec{u}^T A\vec{v} = \vec{0}.$$

To explain the CGM, first the quadratic form of $\mathbf{A}\vec{x} = \vec{b}$ is written as

$$f(\mathrm{x}) = \frac{1}{2}\vec{x}^T \mathbf{A}\vec{x} - \vec{x}^T\vec{b}\,. \tag{49}$$

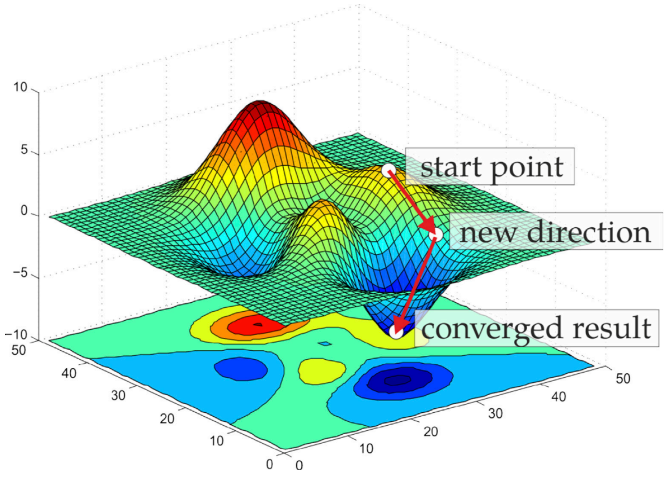


Fig 11. Example of the conjugate gradient method. Starting from an initial guess (start point), an exact solution is reached after a maximum of N steps, where N is the number of unknowns in the linear equation system.

This means that if there is an $\vec{\hat{x}}$ (initial guess) which is set as the approximated vector $\vec{x}$, $f(\vec{\mathrm{x}})$ will become smaller if $\vec{\hat{x}}$ approaches the exact solution. If $f(\vec{\mathrm{x}}) = \vec{0}$, the exact solution is attained.

After the starting point is defined (see Fig 11; normally it is $\vec{\hat{x}} = \vec{0}$), one has to minimize $f(\vec{\mathrm{x}})$ to get the exact solution to find the – hopefully global – sink. Another way is to define a residual $\vec{r}$ which will be minimized and where $\vec{r} = \vec{0}$ results in the same exact solution:

$$\mathbf{A}\vec{x} = \vec{b} \Leftrightarrow \vec{0} = \vec{b} - \mathbf{A}\vec{x}_k \doteq \vec{r}^{\,k}\,, \tag{50}$$

where k is the current iteration step and $\vec{r}^{\,k}$ the current result using $\vec{x}_k \doteq \vec{\hat{x}}_k$. With this it is possible to find a vector p_k which points towards a "new direction" (see Fig 11), using the expression:

$$\vec{p}_k = \vec{r}_k - \sum_{i<k} \frac{\vec{p}_i^{\,\mathrm{T}} \mathbf{A}\,\vec{r}_k}{\vec{p}_i^{\,\mathrm{T}} \mathbf{A}\,\vec{p}_i}\vec{p}_i\,, \tag{51}$$

where $\vec{p}_k$ is conjugate to the residual $\vec{r}_k$. A new solution vector for the next iteration step $k+1$ is calculated as

$$\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k ,$$

where

$$\alpha_k = \frac{\vec{p}_k^{\mathrm{T}} \vec{b}}{\vec{p}_k^{\mathrm{T}} \mathbf{A} \vec{p}_k} = \frac{\vec{p}_k^{\mathrm{T}} (\vec{r}_{k-1} + \mathbf{A} \vec{x}_{k-1})}{\vec{p}_k^{\mathrm{T}} \mathbf{A} \vec{p}_k} = \frac{\vec{p}_k^{\mathrm{T}} \vec{r}_{k-1}}{\vec{p}_k^{\mathrm{T}} \mathbf{A} \vec{p}_k} . \tag{52}$$

After $k = N$ steps, the CGM converges to the exact solution. At first, the CGM looks more complicated than other methods, but summarising the method in pseudo-code (Listing 1) shows that the CGM just needs one matrix-vector multiplication and a few inner products for each step. There are many ways to reduce the computational cost of the CGM. Generally, in the worst case it has a runtime of $O(\mathrm{n}^3)$, but if the matrix is sparse, the CPU time reduces significantly. Using a preconditioner before solving the linear equation system may help to further reduce CPU time [38].

$\vec{r}_0 \doteq \vec{b} - \mathbf{A}\vec{x}_0$
$\vec{p}_0 \doteq \vec{r}_0$
$k \doteq 0$
repeat
  $\alpha_k = \frac{\vec{r}_k^{\mathrm{T}} \vec{r}_k}{\vec{p}_k^{\mathrm{T}} \mathbf{A} \vec{p}_k}$
  $\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k$
  $\vec{r}_{k+1} = \vec{r}_k + \alpha_k \mathbf{A} \vec{p}_k$
  if r_{k+1} fulfils convergence criterion, break repeat
  $\beta_k = \frac{\vec{r}_{k+1}^{\mathrm{T}} \vec{r}_{k+1}}{\vec{r}_k^{\mathrm{T}} \vec{r}_k}$
  $\vec{p}_{k+1} = \vec{r}_k + \beta_k \vec{p}_k$
  $k = k + 1$
end repeat
return x_{k+1}

Listing 1. Pseudocode for the conjugate gradient method.

Multigrid Method

The multigrid method (MGM) is a technique to minimize the number of convergence steps by using intermediate steps which are called *(i)* the smoothing, *(ii)* the restriction, and *(iii)* the prolongation step (see Fig 12). A fine-grained problem is reduced into smaller coarse-grained problems. The results of the coarse-grained problems are interpolated to obtain boundary solutions to solve the finer-grained problems until the original fine-grained problem is solved. Using a good initial guess obtained in previous computations results in a minimization of convergence steps for the original fine-grained problem. This significantly saves computation time. Within a smoothing step *(i)*, some Gauss-Seidel iteration steps are performed (see page 43). For example, the software APBS (*A*daptive *P*oisson–*B*oltzmann *S*olver) [18], [39]–[41] uses an advanced Gauss-Seidel procedure called the Red-Black Gauss-Seidel method [39] which boosts the GSM by solving black-labeled indices first and red-labeled indices afterwards (this can be imagined as a generalized three-dimensional chess board). A restriction step *(ii)* transforms the fine-grained solution into a coarser one. A prolongation step *(iii)* interpolates the coarser grid solution onto a finer grid.

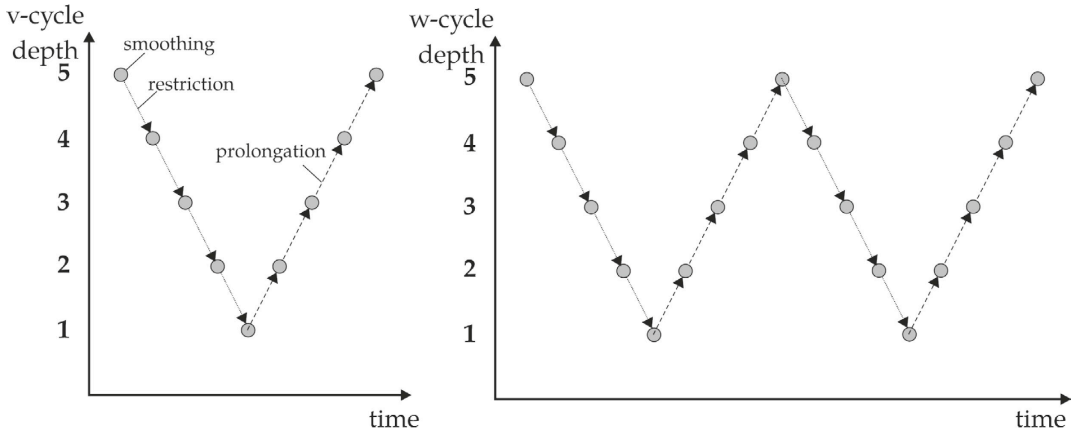


Fig 12. Schematic representation of the multigrid method with *(i)* smoothing, *(ii)* restriction and *(iii)* prolongation steps; left: v-cycle; right: w-cycle; here, both cycles have a depth of 5.

Multifrontal Method

Using mFES, it is possible to reduce the number of linear equations and build a sparse, positive-definite and symmetric coefficient matrix **A**. mFES uses MUMPS (*M*ultifrontal *M*assively *P*arallel sparse direct *S*olver) [42]–[44] to solve the *l*PBE numerically. MUMPS is adapted to sparse coefficient matrices and utilizes the symmetry of the coefficient matrix. It uses a multi-frontal solver method employing the Gaussian elimination techniques described on page 43 for every subproblem, a so-called front. The frontal solver is designed to skip the large number of zeros in the sparse coefficient matrix by building up smaller dense matrices. A front is generated by an LU matrix decomposition technique building independent submatrices which can be solved independently using Gaussian elimination techniques. On account of having independent submatrices to compute the electrostatic potential of a molecular system by solving the *l*PBE, the method is called the multi-frontal method. MUMPS generates many fronts which can be used to solve the linear equation system with a sparse matrix by parallelization. In standard configuration mFES does not use this option. Instead, it uses an iterative procedure of MUMPS for solving the *l*PBE with a second-order solution which leads to converged results after two iteration steps.

NETGEN [1] provides the coefficient matrix **A** and the right-side vector $\vec{b}$ for a higher-order solution of the *l*PBE. For a third- and fourth-order solution three iteration steps are necessary to get sufficiently converged results.

Discretization Pitfalls

Artificial Grid Energy

PDB files contain information about the crystal structure of a protein in terms of atom and residue types of Cartesian coordinates and of B-factors (*P*rotein *D*ata *B*ank file [45]). Information about atomic vdW radii and charges are added to these data [46]–[48] and the molecular model is discretized from free space onto a regular grid (FD method), irregular grid (FE method), or onto an irregular molecular surface grid (BE method). This preparation is crucial for all following steps.

An artefact arises from distributing a point charge q_i, over neighbouring grid points by linear or polynomial interpolation. Thus, fractional charges appear at the grid points, which start to interact. But, even if an atomic charge is by chance exactly on a grid point it is subject to self-interaction. Another point charge artefact occurs because the electrostatic potential at a point charge is infinite, eq. (3). The resulting self-energy depends on the geometrical arrangement of the neighbour grid points where the fractional charges are localized, generating artificial grid energies. It is not possible to compute the grid energy a priori because every grid may be different and the same is true for the self-energy contribution of a grid point to the total electrostatic energy of the system. The higher the density of grid points is, the shorter is the distance between newly created partial point charges and the higher is the total self-energy contribution. The problem is that one generally needs a high grid point density to compute very accurate electrostatic energies for a molecular system which results in large grid energy artefact.

To overcome the divergence of the self-energy resulting from the grid artefact, the artefact free electrostatic free energy ΔG is calculated by computing the energy difference $\Delta G_{gas\rightarrow \mathrm{solv}} = G_{solv} - G_{gas}$, where *gas* refers to the environment where a protein

is in the gas phase with homogeneous dielectric medium and *solv* refers to the model considering the protein in a solvent represented by a dielectric continuum using the same grid points as in the gas phase. There are alternative techniques for removing the singular contribution of point charges in the *l*PBE equation explicitly by using the reaction field energy approach [49], [50].

Surface Discretization Pitfalls

Discretizing a molecular surface (Fig 13) is performed by discriminating between the inner and outer region at protein surface.

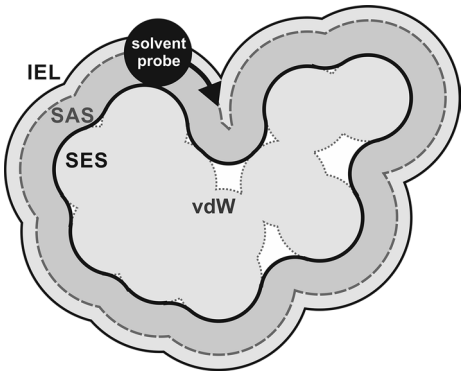


Fig 13. Surface types for a protein; IEL: ion exclusion layer; SAS: solvent accessible surface; SES: solvent excluded surface; vdW: van-der-Waals surface. A solvent probe with a given radius is rolling over the vdW surface creating the SES which is also known as the molecular surface.

When using the FD method, it has to be decided if the intersecting cube in the simple cubic grid belongs to the interior or the exterior of a protein moiety. In the following, an example of a general FD algorithm creating the SES is sketched (APBS [18], [39]–[41]):

- Mark all cubes on the grid as exterior;
- Iterate over all atom positions with extended sphere radius (vdW radius + probe radius) and mark all cubes which are partially inside of the extended spheres as belonging to the protein volume;

- Create evenly distributed points with a given point density located on the vdW spheres (vdW points);
- Delete all vdW points which are located in intersecting vdW spheres;
- Draw a sphere with probe radius around each remaining vdW point and mark all cubes within this radius as exterior again.

The determination of molecular surface points, performed by this procedure, bears an uncertainty of $\pm 0.5h$, where h is the grid constant of the finest focusing step used in the FD method algorithm. In [3] the effect on simple electrostatic solvation energy computations for different proteins is shown. The FD and FE methods differ by more than a few kJ/mol, depending on the surface area of the protein analysed which in general is proportional to the size of the protein.

The FE method is flexible in distributing grid points in space to discretize the molecular model. To model the protein surface faithfully, a high grid point density is selected on the molecular surface, while in the volume of the interior and exterior protein region, the density is lower. With mFES, the high molecular surface grid point density is achieved using a level set method implemented in LSMS [2]. To regularize the resulting molecular surface, NETGEN [1] is employed which uses the advancing front method [51], [52] to generate triangles with good quality on the surface. In the resulting molecular model, one can establish finer-grained molecular models with the same number of grid points or even less than needed for the FD method without the necessity of using a focusing technique. In the following, some common problems are treated, which have to be solved by establishing an initial fine-grained molecular surface and compare the performance of different programs computing the SES.

In general, one is interested in a triangulated protein surface which is accurate and which produces electrostatic energies and pK_A values with high quality after solving the *l*PBE. The algorithm has to be robust and flexible to construct triangles on the molecular surface which have nearly same edge lengths (regular triangles). Different meshing algorithms were explored, like EDTSurf [53], MSMS [54] and LSMS [2], as well as other algorithms like ball-pivoting on point clouds [55] to reach these goals.

EDTSurf turned out to be unusable for the task because EDTSurf eventually simplifies the marching cubes algorithm to 23 cases even though up to $2^8 = 256$ cases are possible to mesh a surface, depending on the inside/outside pattern of the cube-corner points. In an important but still very easy case, a sphere, EDTSurf was not able to mesh its surface with sufficient quality (see Fig 14).

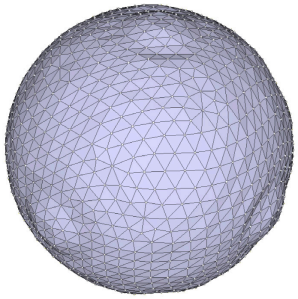


Fig 14. The surface of a sphere is generated with EDTSurf. EDTSurf uses the vertex-connected marching cubes algorithm which is not performing well in this example.

There is not much control over the meshing parameters in detail, and every mesh is finalized with a Laplacian smoothing by default. From experience this smoothing might lead to uncontrollable surface shrinkage which in turn might result in triangles with low quality. On the other side, volumes and the solvent accessible surface area are approximated well, as stated in the paper by Zhang [53], but EDTSurf is expected to have difficulties in meshing the molecular surface with high quality if there are titratable groups near the surface which is a challenging task.

MSMS is able to compute an analytical surface and nearly uniformly distributed points on the molecular surface, but (1) triangle quality is not measured and (2) sometimes the triangulated surface features self-intersections which is a difficult problem to solve (see Fig 15 and Fig 16).

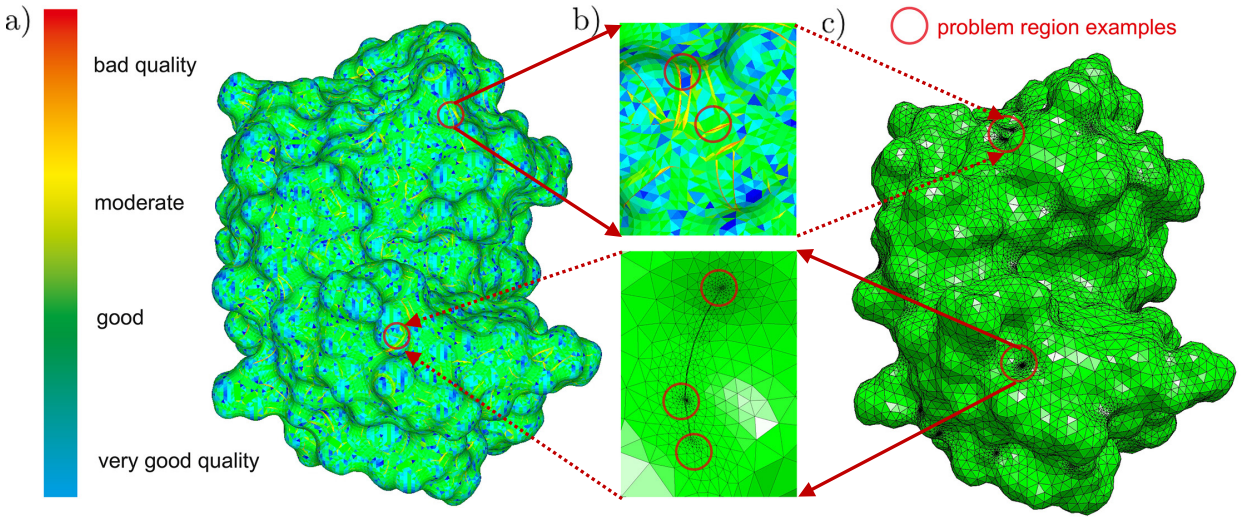


Fig 15. Lysozyme (PDB id 2lzt [56]) meshing example with trouble spots (circled in red); a) triangulated MSMS mesh; b) *upper part:* zoom into some trouble spots of the MSMS-computed surface which are hard to mesh; b) *lower part:* NETGEN mesh with bad triangles using MSMS mesh; c) an attempt of NETGEN to mesh the MSMS surface of lysozyme; some trouble spots cannot be repaired.

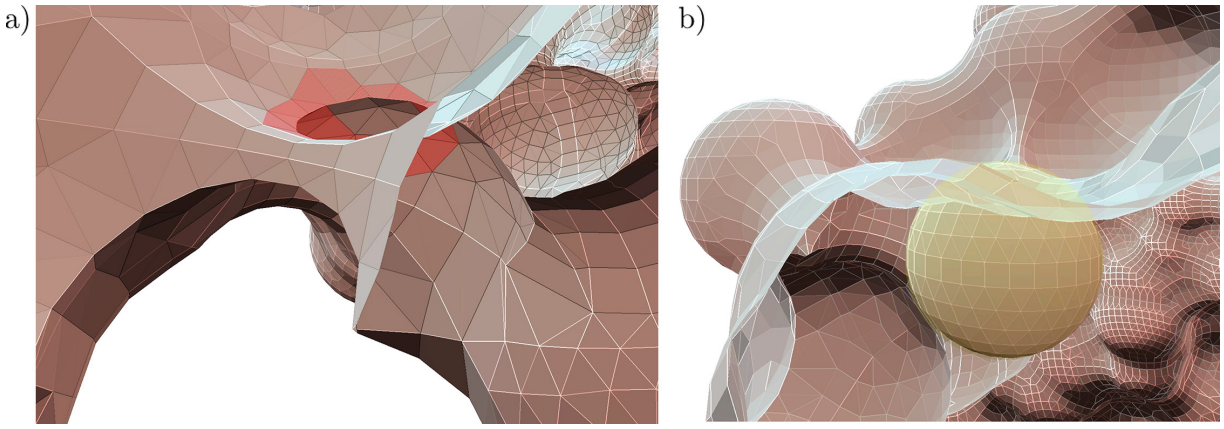


Fig 16. Ribonuclease H (PDB id 2RN2 [57]) meshing example with trouble spots obtained by the MSMS algorithm; a) *left:* exterior of the protein, intersecting (red) faces of the molecular surface; b) *right:* interior of the protein; irregularity in the mesh: a complete sphere inside the protein (yellow).

Another possibility is to use a ball-pivoting algorithm, which creates points on the molecular surface. This point cloud is generated by extracting information from different programs (e.g. APBS [18], [39]–[41], MSMS [54]). A sphere with radius r starts at a seed grid point (vertex) and triangulates every three vertices to a face, which is repeated until all vertices were visited at least once. With this algorithm described by Bernardini et al. [55], it is probable to obtain a surface with holes. It is not feasible to close all holes in the surface automatically without getting self-intersections (see Fig 17), therefore, this method was not used.

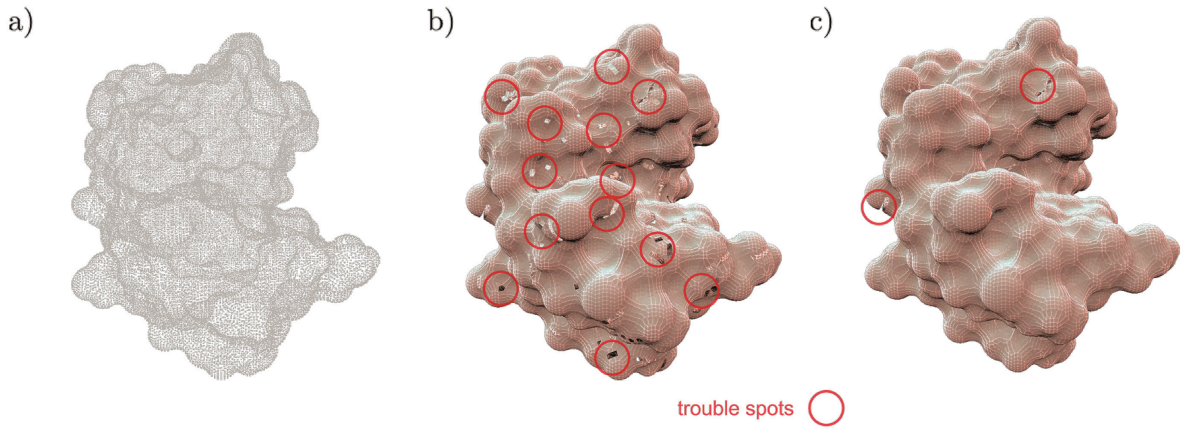


Fig 17. Application of the ball-pivoting method [55]. Trouble spots are circled in red. a) point cloud generated using e.g. MSMS [54] for modelling the surface of lysozyme (PDB id 2lzt [56]); b) result of the ball-pivoting algorithm with holes marked as trouble spots; c) result of the automatic hole closing algorithm after some filtering using Meshlab [58]. Not all holes could be closed. Hence, the surface remains inconsistent.

Applications

Computation of the Electrostatic Solvation Energy

The electrostatic solvation energy is part of the solvation free energy which is the energy to transfer a molecule from vacuum (gas phase) into water (aqueous phase). The presence of ions affect the solvation free energy. Regardless of the specific distribution of the ionic charges, the electrostatic solvation energy decreases with ionic strength [59].

The difference $\Delta G_{gas\to solv}$ of the electrostatic energy of a point charge distribution in two different dielectric environments (*gas* and *solv*) is given by

$$\Delta G_{gas\to \mathrm{solv}} = G_{solv} - G_{gas} = \frac{1}{2}\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i}}\left(\Phi_{\mathrm{i}}^{\,solv} - \Phi_{\mathrm{i}}^{\,gas}\right), \tag{53}$$

where *gas* refers to the environment where a protein is in the gas phase ($\varepsilon_{\mathrm{in}} = \varepsilon_{\mathrm{out}} = 1$ to 4) and *solv* refers to the model using same protein in a solvent represented by a dielectric continuum ($\varepsilon_{\mathrm{out}} = 80$ outside and $\varepsilon_{\mathrm{in}} = 1$ to 4 inside of the protein, see Fig 18).

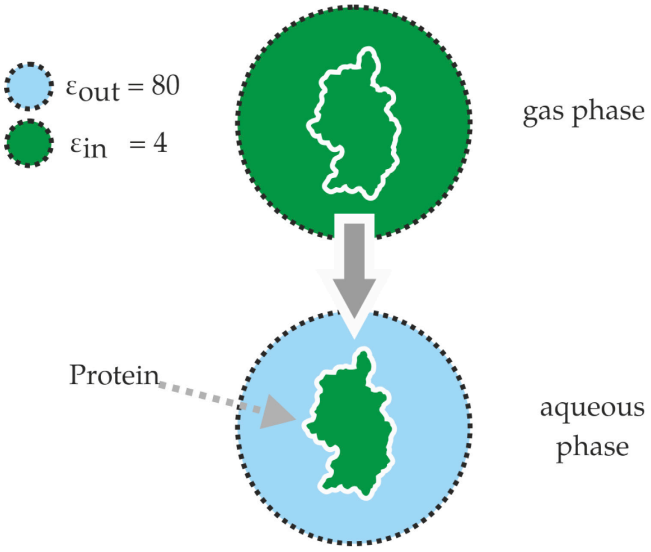


Fig 18. Schematic representation of a dielectric continuum model. The electrostatic solvation energy is computed using the same grid for the protein in gas and aqueous phase. The dashed circle denotes the boundary surface used in numerical computations to set up boundary conditions.

Determination of pK_A Values in Proteins

The Danish chemist Søren Sørensen established the pH scale in 1909 [60]. He determined the proton concentration $[H^+]$ in solution and defined the pH value as the negative decadic logarithm of this concentration. Lawrence Joseph Henderson described acid-base equilibration reactions in 1908 [61] and Karl Albert Hasselbalch extended Henderson's equations in 1916 [62] which lead to the Henderson-Hasselbalch equation, eq. (60), which connects the pH, the acid-base equilibration reactions, and the pK_A. The latter is defined as the negative decadic logarithm of the acid constant in acid-base equilibration reactions. The equilibrium of a pair of protonation states of a titratable group can described by the pK_A value which has to be determined experimentally (if possible) or by computation.

Table 1. Model pK_A values measured in different labs. Table adopted from [66], p. 1215.

group	**Cohn and Edsall** [67]	**Nozaki and Tanford** [68]	**Gurd Lab** [69]	**Wuthrich lab** [70]	**Creighton** [71]
α-carboxyl	3.0–3.2	3.8	3.3	-	3.5–4.3
Asp	3.0–4.7	4.0	3.9	3.9	3.9–4.0
Glu	4.4	4.4	4.2	4.2	4.3–4.5
His	5.6–7.0	6.3	6.8	6.9	6.0–7.0
α-amino	7.6–8.4	7.5	8.1	–	6.8–8.0
Cys	9.1–10.8	9.5	–	–	9.0–9.5
Tyr	9.8–10.4	9.6	10.0	10.2	10.0–10.3
Lys	9.4–10.6	10.4	10.5	11.0	10.4–11.1
Arg	11.6–12.6	12.0	–	–	12.0

The pK_A values were measured under different conditions using various approaches, see references for details. In general, these measurements have an error of ± 0.1–0.2 pH units.

Experimental pK_A values of titratable groups in solution are determined using potentiometric titration that varies the pH value [63] or indirect techniques such as NMR to monitor ionization states [64]. The resulting experimental values of the degree of protonation are fitted to the Henderson-Hasselbalch equation to obtain so-

called model pK_A values [65]. Table 1 lists measured pK_A values. These are sometimes controversial, and can be subject to small errors, which dependent on experimental conditions [66]. Experimental pK_A values of titratable groups in a protein environment are more difficult to measure. Hence, computing pK values of titratable groups in proteins is an important task.

pK_A Determination *in silico*

For the pK_A determination in silico, the ionization state of every titratable group has to be determined by using the crystal structure information of the protein. The ionization state depends on the pH value of the solvent. A high pH value means low proton concentration in solution and a low pH value means high proton concentration. Depending on the pH value of the solvent environment of a protein, the solvated protons interact with atoms of the protein and titratable groups might change their protonation state. The different types of titratable groups can either be in their protonated or in their deprotonated state, which depends on the difference between pK_A and pH values. The pK_A value lies at that point of the pH scale where the probability of a titratable group of being protonated or deprotonated is 0.5. Table 2 lists some values used for model pK_A computations.

MD (*m*olecular *d*ynamic) simulations are one example where an initial ionization state has to be assigned to a protein before starting a simulation. In general, the "standard" protonation corresponding to the pK_A values of titratable groups in solution is used for proteins in a physiological environment with a pH value of about 7. This assignment is often not correct for protein environment. The protonation state of a protein depends on the structure and the environment and has to be recomputed for each protein conformation generated by the MD simulation.

Table 2. Experimental model pK_A values used in thermodynamic cycle computations to calculate the energy differences between the charged and the uncharged state of a titratable group.

group	**pK_A^{model}** [68], [72]
Asp	4.0
Glu	4.4
Lys	10.4
Arg	12.0
Hisδ	7.0
Hisε	6.6
Tyr	9.6
Nterm	7.5
Cterm	3.8

Titration Curves for a Single Isolated Titratable Group

To determine pK_A values, one has to compute titration curves which in principle follow the Henderson-Hasselbalch equation, eq. (60). This results in the assumption that the titration of residues in proteins produces similar sigmoidal curves. In the following, the mathematical formulation of the Henderson-Hasselbalch equation is derived.

An acid-base equilibrium is described by the Henderson-Hasselbalch equation. The general forms of an acid, eq. (54), and a base reaction, eq. (55), writes as

$$\mathrm{HA} \underset{K_B}{\overset{K_A}{\rightleftharpoons}} \mathrm{A^-} + \mathrm{H^+}, \tag{54}$$

$$\mathrm{BH^+} \underset{K_B}{\overset{K_A}{\rightleftharpoons}} \mathrm{B} + \mathrm{H^+}. \tag{55}$$

The equilibrium constants are defined by the laws of mass action, resulting in

$$K_A = \frac{c\left(H^+\right)c(A^-)}{c(HA)} \quad \text{(acid equilibrium constant)}, \tag{56}$$

$$K_B = \frac{c\left(BH^+\right)}{c\left(\mathrm{B}\right) + c\left(H^+\right)} \quad \text{(base equilibrium constant)}. \tag{57}$$

K_A equals unity if the concentration of H^+ and A^- on the right side of eq. (54) equals the concentration of HA. If K_A is less than unity, the protonated (neutral) acid dominates, and if K_A is greater than unity the dissociated (negatively charged) acid dominates.

The pH value is the negative decadic logarithm of the H^+ concentration and the pK_A the negative decadic logarithm of the acid constant, one writes

$$\mathrm{pH} = -\log_{10}(\mathrm{c}(\mathrm{H}^+)), \tag{58}$$

$$\mathrm{pK_A} = -\log_{10}(K_A) = -\frac{\ln(K_A)}{\ln(10)}. \tag{59}$$

Combining eq. (56) and the previous definitions above leads to the Henderson-Hasselbalch equation which is the dimensionless negative decadic logarithm of the acid equilibrium constant:

$$\mathrm{pK_A} = -\log_{10}\left(\frac{c(\mathrm{H}^+)c(\mathrm{A}^-)}{c(\mathrm{HA})}\right) = pH - \log_{10}\left(\frac{c(\mathrm{A}^-)}{c(\mathrm{HA})}\right) = pH - \underbrace{\frac{1}{2.303}}_{=\ln(10)} \cdot \ln\left(\frac{c(\mathrm{A}^-)}{c(\mathrm{HA})}\right). \tag{60}$$

The pK_A value of an acid equals the pH value at which the concentrations of its protonated and its deprotonated forms are equal. Therefore, the pK_A is measured by determining the equilibrium point of a titration curve where $c(A^-) = c(HA)$. For titratable groups, these values can be obtained experimentally (page 56 and table 1).

Changes in pK_A have to be related to changes in free energy to be able to perform thermodynamic cycle computations. The Gibbs free energy links the free energy with the pK_A, eq. (59), at standard conditions (temperature: 298.15 °K; pressure: 1 bar). The Gibbs free energy ΔG^0_{prot} is defined as

$$\mathrm{pK_A} = -\frac{\ln(K_A)}{\ln(10)} \Leftrightarrow \ln(K_A) = -\ln(10)\,\mathrm{pK_A} \Rightarrow \Delta G^0_{prot} = RT \cdot \ln(\mathrm{K}_A) = -\mathrm{RT} \cdot \ln(10) \cdot \mathrm{pK_A}, \tag{61}$$

where RT is a scaling factor for the energy defined as $k_B T \cdot N_A$ (k_B is the Boltzmann factor, T is the absolute temperature in Kelvin, and N_A is the Avogadro constant; see appendix A).

Using eq. (61), the acid constant is rewritten as

$$\frac{\Delta G^0_{prot}}{RT} = \ln(\mathrm{K}_A) \Leftrightarrow K_A = e^{\frac{\Delta G^0_{prot}}{RT}}. \tag{62}$$

The protonation probability $\langle x \rangle$ is computed for every titratable group depending on pH. Rewriting the Henderson-Hasselbalch equation, eq. (60), using $1 - \langle x \rangle = c(\mathrm{A}^-) / \left(c(\mathrm{A}^-) + c(\mathrm{HA}) \right)$ results in

$$\mathrm{pK_A} = \mathrm{pH} - \log\left(\frac{1 - \langle x \rangle}{\langle x \rangle} \right). \tag{63}$$

The change in the free energy due to a change in the pH environment is defined as

$$\Delta G^0_{prot} = \mathrm{RT} \cdot \ln(10) \cdot \left(\mathrm{pH} - \mathrm{pK_A} \right). \tag{64}$$

Using eq. (63) and eq. (64) leads to

$$\langle x \rangle = \frac{\mathrm{e}^{-\frac{\Delta G^0_{prot}}{\mathrm{RT}}}}{\mathrm{e}^{-\frac{\Delta G^0_{prot}}{\mathrm{RT}}} + 1} = \frac{1}{10^{(\mathrm{pH} - p\mathrm{K_A})} + 1}. \tag{65}$$

Finally, a titration curve is drawn by plotting the probability $\langle x \rangle$ as a function of the pH which results in the well-known sigmoidal titration curve (an example is shown in Fig 19) defined by the Henderson-Hasselbalch equation, eq. (60).

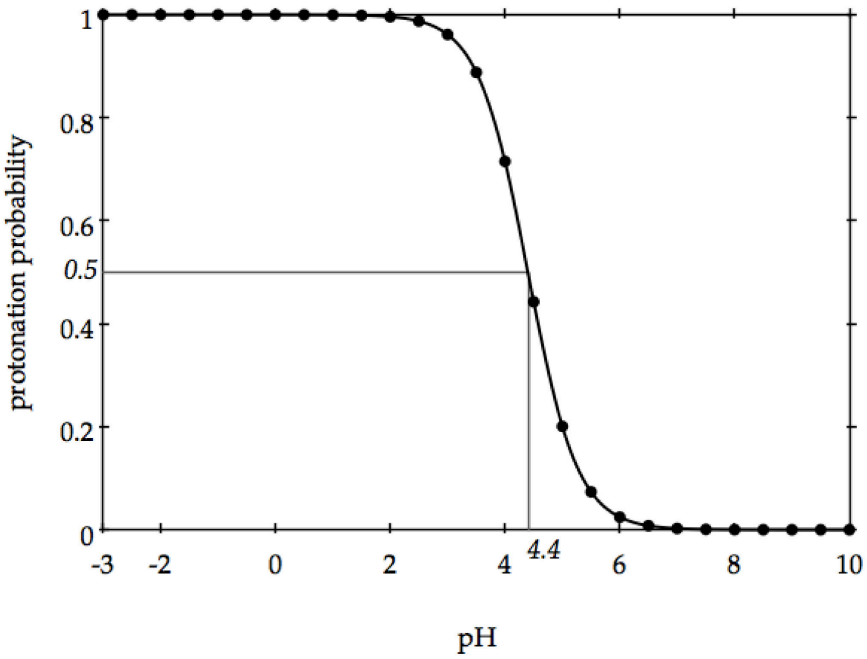


Fig 19. Exemplary titration curve of Glu with the model pK_A value 4.4. The sigmoidal shape of the curve is described by the Henderson-Hasselbalch equation.

Titration Curves of Multiple Titratable Groups

In the previous subchapters the determination of pK_A values has been explained which is sufficient to describe the titration behaviour of one titratable group in solution. In the following, the pK_A values of multiple interacting titratable groups of a single protein conformer will be determined. Observing pK_A shifts while e.g. changing the protein conformation or the internal charges can give clues to protein function or to possible proton transport channels [73].

In general, titratable groups obey the Henderson-Hasselbalch equation, i.e. they have a sigmoidal titration curve. Inside the protein, their behaviour might be different due to interactions between neighbouring titratable groups, other charges of polar groups, or interactions with ions near the molecular surface. These effects can all be accounted for by electrostatic interactions [68]. Therefore, it is possible to

determine the ionization states of titratable groups if all electrostatic effects are included in the mathematical description. This is achieved by the continuum electrostatic model using different dielectric media. The energetics of different protonation patterns are computed in different dielectric media using a thermodynamic cycle (Fig 20). A reasonable choice of dielectric constants ε_{in} and ε_{out} has to be made as well as an accurate description of the protein by discretization methods. There are error cancellation effects, and the different philosophies regarding the "best" thermodynamic cycle are subject to on-going debate in the scientific community [74], [75].

Using the thermodynamic cycle displayed in Fig 20, the free-energy difference for one titratable group in a dielectric environment is defined by

$$\Delta G_X(\mathrm{AH},\mathrm{A}) = \mathrm{G}_X(\mathrm{A}^-) - \mathrm{G}_X(\mathrm{AH}) = \mathrm{RT}\ln(10)\,(\mathrm{pH} - \mathrm{pK}_A^X), \tag{66}$$

where X denotes one of the three environments: gas phase (G), aqueous phase/solvent (S), or protein environment (P). Generally, one is interested in $\Delta G_P(\mathrm{AH} \to \mathrm{A}^-)$ which is the free-energy difference between the protonated and the deprotonated form of the titratable group within the protein environment. The difference ΔG_P is obtained by computing the other free-energy differences occurring in the thermodynamic cycle and combining them as follows:

$$\Delta G_P(\mathrm{AH} \to \mathrm{A}^-) = -\Delta G_{S\to P}(\mathrm{AH}) - \Delta G_{G\to S}(\mathrm{AH}) + \Delta G_G + \Delta G_{G\to S}(\mathrm{A}^-) + \Delta G_{S\to P}(\mathrm{A}^-), \tag{67}$$

where $\Delta G_G(\mathrm{AH} \to \mathrm{A}^-)$ is known as the proton affinity (in the gas phase). This procedure works for small heterogeneous groups of compounds as has been demonstrated to be feasible using the DFT (*d*ensity *f*unctional *t*heory) [74], [76].

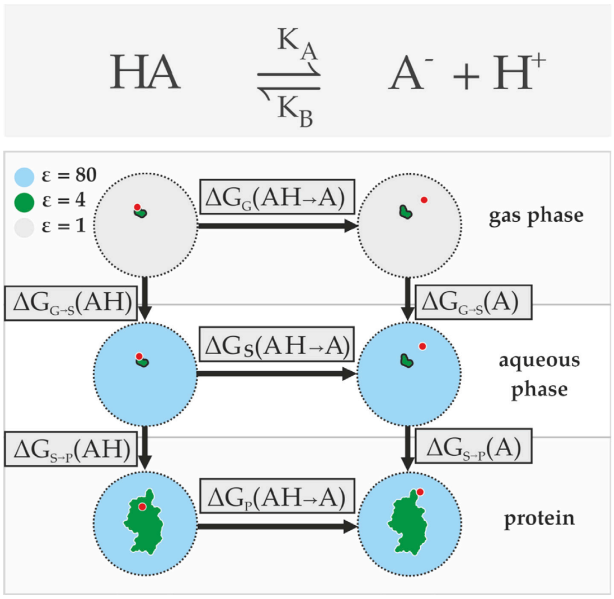


Fig 20. A general thermodynamic cycle is shown for computing the pK_A value of a titratable group using a single conformer of the protein. The gas phase, aqueous phase and protein environment have varying setups using different dielectric constants for the solvent and/or solute. The location of the H^+ is denoted by a red dot. For the gas and aqueous phase only the titratable group and for the protein environment the whole protein is marked in green; left: one titratable group is protonated; right**:** the same titratable group is deprotonated, i.e. H^+ is solvated.

Although similar approaches have systematically improved this method for metal compounds [77], a large and computationally expensive basis set is needed to achieve an appropriate accuracy for *ab initio* computations of pK_A values. This is why an alternative way is chosen for skipping the expensive quantum-chemical gas-phase calculation by using experimental pK_A^{model} values for $\Delta G_{G\to S}(\mathrm{AH},\mathrm{A}^-)$ which results in a simplified thermodynamic cycle computation:

$$\Delta G_P(\mathrm{AH}\to\mathrm{A}^-) = -\overbrace{\Delta G_{S\to P}(\mathrm{AH})}^{unknown} + \overbrace{\Delta G_{S\to P}(\mathrm{A}^-)}^{unknown} + \overbrace{\Delta G_{G\to S}(\mathrm{AH},\mathrm{A}^-)}^{known}. \tag{68}$$

The pK_A value of titratable group μ in protein environment correspondingly writes

$$pK_{A,\mu}^{P} = pK_{A,\mu}^{model} + \frac{\left(\Delta G_{S\to P}(\mathrm{A}^-) - \Delta G_{S\to P}(\mathrm{AH})\right)}{RT\cdot\ln(10)} = pK_{A,\mu}^{model} + \frac{\Delta\Delta G_{\mu}^{P}}{RT\cdot\ln(10)}. \tag{69}$$

The two unknown parts of $\Delta\Delta G_{\mu}^{P}$ are computed by calculating the electrostatic energy differences between *gas* phase and *aqueous* phase and between *aqueous* phase and *protein* phase, as shown in Fig 20. For every electrostatic energy term

$$\Delta G_{gas\to\mathrm{solv}}(\alpha)=\Delta G_{S\to P}(\alpha)=G_P(\alpha)-G_S(\alpha)=\tfrac{1}{2}\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i}}\left(\Phi_{\mathrm{i}}{}^{P}(\alpha)-\Phi_{\mathrm{i}}{}^{S}(\alpha)\right), \tag{70}$$

is used, where $\alpha\in\{AH,A^{-}\}$, N_q is the number of charges of the titratable group, q_i is the i$^{\mathrm{th}}$ charge, and Φ_i is the potential at the location of charge q_i for the dielectric medium corresponding to the protein (P) or in the solvent environment (S), i.e. the aqueous phase. $\Delta\Delta G_{\mu}^{P}$ is often computed using a continuum dielectric model with the two double differences of $\Delta\Delta G_{born}$ and $\Delta\Delta G_{back}$ energy terms:

$$\begin{aligned}&\Delta\Delta G_{solv,\mu}=\Delta\Delta G_{born,\mu}=-\Delta G_{S\to P,\mu}(HA)+\Delta G_{S\to P,\mu}\left(A^{-}\right)=\\&-\left(\tfrac{1}{2}\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i},\mu}\left(\Phi_{\mathrm{i}}{}^{P}(HA)-\Phi_{\mathrm{i}}{}^{S}(HA)\right)\right)+\left(\tfrac{1}{2}\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i},\mu}\left(\Phi_{\mathrm{i}}{}^{P}\left(A^{-}\right)-\Phi_{\mathrm{i}}{}^{S}\left(A^{-}\right)\right)\right).\end{aligned} \tag{71}$$

$\Delta\Delta G_{solv,\mu}$ is known as the Born energy contribution which is the solvation energy needed to transfer a titratable group from an aqueous phase into a protein environment. In more detail, the sum N_q runs over all atoms of titratable group μ where the atomic partial charges depend on the actual protonation state. The equation

$$\Delta\Delta G_{back,\mu}=-\left(\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i},\mu}\left(\Phi_{\mathrm{i}}{}^{S}(A)-\Phi_{\mathrm{i}}{}^{S}(HA)\right)\right)+\left(\sum_{\mathrm{i}=1}^{\mathrm{N_q}} q_{\mathrm{i},\mu}\left(\Phi_{\mathrm{i}}{}^{P}(A)-\Phi_{\mathrm{i}}{}^{P}(HA)\right)\right), \tag{72}$$

is used to calculate the influence of charges inside a protein which are due to other titratable groups and neutral charges. In this equation, N_q runs over all background charges in the protein, i.e. the charges of all other titratable groups in their reference state and all charges in the protein which are not located in titratable groups, as well as all charges of the titratable groups μ which are *not changing* if the protonation

state is changed (Fig 21). Hydrogen-bond donors and acceptors might be present and have to be accounted for in the pK_A computation of a titratable group μ in protein environment by inserting eq. (71) and eq. (72) into eq. (69) [78], [79]:

$$pK_{A,\mu}^{\text{intrinsic}} = pK_A^{\text{model}} + \frac{1}{RT \cdot \ln(10)}\left(\Delta\Delta G_{\text{solv},\mu} + \Delta\Delta G_{\text{back},\mu}\right). \tag{73}$$

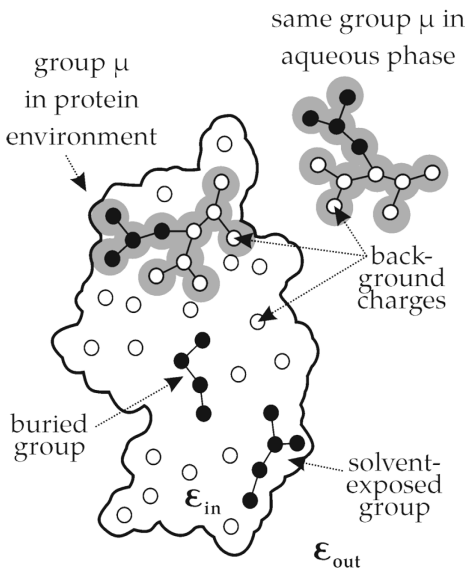


Fig 21. A titratable group in protein environment and in aqueous solution is shown; *black circles:* charges are changing if the protonation state of the titratable group is changing; *white circles:* background charges.

Eq. (73) for computing the pK_A value is the so-called intrinsic pK_A value which applies if there are no other titratable groups in the protein or if all other titratable groups are in their uncharged reference state. This assumption is not necessarily fulfilled which is why one has to consider the influence of interactions between titratable groups in the protein environment in more detail. This is done using an interaction matrix which accounts for the interaction between titratable groups μ and v. By summation over all variable charges of the titratable group μ one obtains:

$$W_{\mu v} = \sum_{\mu=1}^{N}\left[\left(q_\mu - q_{\text{ref},\mu}\right)\left(\Phi_{\text{v}}^{P} - \Phi_{\text{ref,v}}^{P}\right)\right]. \tag{74}$$

This matrix is symmetric. It is multidimensional because titratable groups can be in different tautomeric states corresponding to multiple protonation states. The diagonal element $W_{\mu\mu}$ is zero because the self-energy term is included in the $\mathrm{pK}_{\mathrm{A},\mu}^{\mathrm{intrinsic}}$.

Summing up the above derivations yields the equation for the electrostatic energy of protonation state x related to the reference protonation state x_{ref} of a titratable group μ, which in analogy to eq. (64) writes as

$$G_{prot}^{\mu} = \left(x_{\mu} - x_{ref,\mu}\right) \cdot \ln(10) \cdot \mathrm{RT} \cdot \left(\mathrm{pH} - \mathrm{pK}_{\mathrm{A},\mu}^{\mathrm{intr}}\right) + \sum_{\nu \neq \mu} \delta_{\mu\nu} W_{\mu\nu}, \tag{75}$$

where $x_{\mu} \in \{0,1\}$ is the current protonation state of titratable group μ; $x_{\mu} = 0$ if the group is in its deprotonated state and $x_{\mu} = 1$ if the group is in its protonated state. $x_{ref} \in \{0,1\}$ is the reference protonation state which is the state of the uncharged group. $x_{ref} = 1$ if the group is an acid and $x_{ref} = 0$ if the group is a base.

The delta function $\delta_{\mu\nu}$ equals unity if both titratable residues (μ and ν) are in their non-reference, i.e. the charged state.

Determining the occupancy $\langle x_i \rangle$ by evaluating the Boltzmann sum for state i one gets:

$$\langle x_i \rangle = \frac{e^{-\frac{G_i^n}{RT}}}{\sum_j e^{-\frac{G_j^n}{RT}}}. \tag{76}$$

Here the previous notation is used to write the so-called thermodynamic average of x_{μ} over all its 2^N protonation states. This average is divided by the sum over all possible protonation states of every titratable group:

$$\left\langle x_\mu \right\rangle = \frac{\sum_{n=1}^{2^N} x_\mu \cdot \mathrm{e}^{-\frac{G^n}{RT}}}{\sum_{n=1}^{2^N} \mathrm{e}^{-\frac{G^n}{RT}}} . \qquad (77)$$

To evaluate the above Boltzmann average it is necessary to compute the energy of every protonation pattern, which involves 2^N different protonation states. Because a "brute-force" method is computationally expensive, a Monte Carlo method (MCM) is used [80], [81].

This probability is plotted for every titratable group μ depending on the pH. The pK_A value for group μ is determined as that pH value for which the group is half-protonated and, therefore, $\left\langle x_\mu \right\rangle = 0.5$ holds true. This method is very appropriate if the titration curve obeys the Henderson-Hasselbalch equation which is the case for majority of titratable groups. In cases, where several titratable groups are strongly coupled they may possess very irregular titration curves. In these cases a separate inspection of the titration curves is necessary in order to obtain meaningful results.

In most cases, better results are obtained if protein flexibility is considered, allowing relaxation of the protein structure, by using appropriate protocols [82], [83]. To account for protein flexibility, one needs to add the electrostatic energy difference G^l of conformation l with respect to the reference conformation n in its reference protonation state to eq. (75):

$$\begin{aligned} G^{n,l} \quad &= \sum_{\mu=1}^{N} \left(x_\mu^n - x_{ref,\mu} \right) \cdot \ln(10) \cdot \mathrm{RT} \cdot \left(\mathrm{pH} - \mathrm{pK}_{\mathrm{A},\mu}^{\mathrm{intr}} \right) \\ &+ \sum_{\mu=1}^{N} \sum_{\mu \neq v}^{N} \delta_{\mu v} \cdot W_{\mu v}^l \\ &+ G^l . \end{aligned} \qquad (78)$$

As a consequence, the probability of a protonation pattern changes from that presented in eq. (77) to

$$\left\langle x_\mu \right\rangle = \frac{\sum_{l=1}^{L}\sum_{n=1}^{2^N} x_\mu \cdot \mathrm{e}^{-\frac{G^{n,l}}{RT}}}{\sum_{l=1}^{L}\sum_{n=1}^{2^N} \mathrm{e}^{-\frac{G^{n,l}}{RT}}}, \tag{79}$$

where L is the number of conformations.

For most of the above-mentioned energy terms it is assumed that the electrostatic contributions of all titratable groups are additive. This is true if the *l*PBE that is solved is valid and the electrostatic potentials follow the principle of superposition [84]. With a non-linear PBE description, the electrostatic potentials are not additive. At low ion concentration the *l*PBE is a good approximation of the general non-linear PBE. High ion concentration might occur for highly charged molecules like RNA or DNA requiring high ionic strength to be efficiently solvated under physiological conditions. To compute the electrostatic potential for such systems with an appropriate accuracy, the electrostatic potential has to be calculated explicitly for each protonation state.

II. mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations

This chapter is based upon a peer-reviewed publication:

I. Sakalli, J. Schöberl, and E. W. Knapp, "mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations," *J. Chem. Theory Comput.*, vol. 10, no. 11, pp. 5095–5112, Oct. 2014.

http://dx.doi.org/10.1021/ct5005092

Contributions

- Development of mFES (research and implementation)
- Development of webpage
- Generating results and performing analyses with new tools
- Manuscript preparation

Summary and Discussion

This thesis presents the computation of electrostatic potentials by using the Finite Element (FE) method for solving the linear Poisson-Boltzmann equation (*l*PBE). It explains the underlying algorithm of the software mFES (*m*olecular *F*inite *E*lement *S*olver) and why it is able to outperform the electrostatic solvation energy computations compared of well-established programs employing the Finite Difference (FD) method.

> ***"To the best of our knowledge, mFES is the first FE method, which is competitive with well-established FD methods and sufficiently robust to calculate electrostatic properties for large proteins accurately."***
>
> Sakalli, **2014** in *Journal of Chemical Theory and Computation*.

After a short introduction to different software using the FE method, the FD method and the Boundary Element (BE) method to solve the *l*PBE, the mathematical expressions of the FE method for discretizing the *l*PBE to an irregular grid are derived. Grid artefacts occurring in the course of the discretization of point charges are discussed as well as how to obtain a high accuracy for electrostatic potentials. For these purposes, higher order polynomials are utilized and are evaluated via test functions to obtain second- or even higher-order polynomial solutions.

The definition and generation of a protein surface is described and different programs to compute molecular surfaces are discussed. Afterwards, a level-set method is introduced for generating a fine-grained molecular surface. The LSMS (*L*evel-*S*et method for *M*olecular *S*urface generation) fulfils the need for high-quality molecular surface generation.

Next, NETGEN is described and how it is used to generate a model of the molecular volume to solve the *l*PBE numerically. First, the molecular surface is covered with triangles which are then regularized via an advancing front method [51], [52]. Subsequently, the protein volume is filled with tetrahedrons with a given resolution. The asymptotic boundary is defined by means of a coarse sphere with the protein in the centre. The space between the protein and the asymptotic surface of the sphere is filled with tetrahedrons whose edge lengths increase regularly towards the asymptotic surface. The resulting tetrahedral mesh of the model is then optimized and the grid points of the resulting tetrahedral mesh are used to discretize the *l*PBE and to solve the resulting linear equation system with MUMPS (*Mu*ltifrontal *M*assively *P*arallel sparse direct *S*olver).

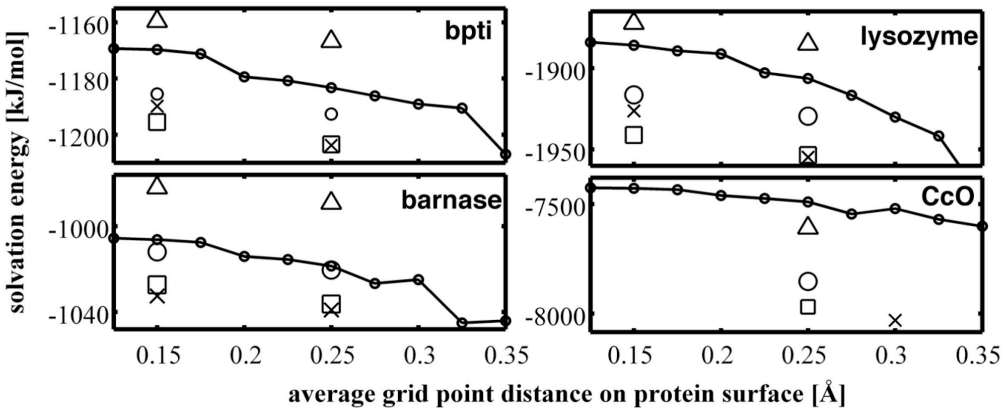


Fig 22. Computed electrostatic solvation energies in [kJ/mol] for bpti (bovine pancreatic trypsin inhibitor), barnase, lysozyme and C*c*O (cytochrome *c* oxidase) as a function of the average nearest-neighbor grid point distances. *mFES:* solid line with dots; *APBS:* for the construction of the molecular surface three different point densities on the atomic vdW surfaces are considered: □, 10 points/Å^2; ○, 3 points/Å^2; △, 1 point/Å^2; *MEAD:* results are marked with × and use same coarse and fine resolutions for the second focusing step as does APBS.

As a proof of principle the Born ion model is computed with different methods and with different ion concentration. The highest accuracy is obtained with mFES as compared to finely resolved computations performed with other methods.

Subsequently, solvation energies are computed for four proteins: bpti (bovine pancreatic trypsin inhibitor), barnase, lysozyme and C*c*O (cytochrome *c* oxidase) (Fig 22). For the FD method it is not clear which point density should be used on the vdW sphere surfaces to obtain converged results. The FE method yields results with errors smaller than 1%. It is obvious that the accuracy of the FE method is increased by lowering the average triangle edge length on the protein surface for all proteins.

With lysozyme as a test case, the Hausdorff distance [85], [86] is utilized to compare molecular surfaces computed with different FE methods (Fig 23). These surfaces are used to compute electrostatic solvation energies.

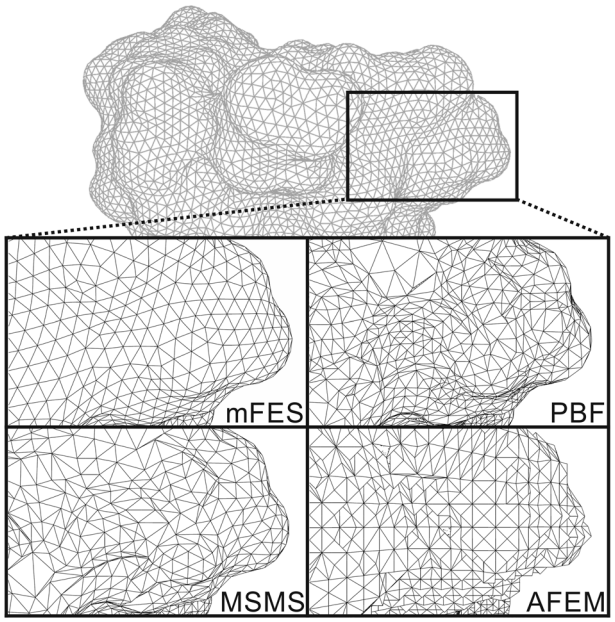


Fig 23. Comparison of molecular surfaces (SES) generated with different FE methods, focusing on a small region of bpti [87]. MSMS [54] is not able to perform solvation energy computations but generates molecular surfaces and is known to deliver a fine representation.

The CPU times needed for solving the *l*PBE are stated as well as the parameters used in mFES (h_S: average edge length of the molecular surface triangles; h_V: average edge length of the tetrahedrons in molecular volume; g: grading parameter reducing the edge lengths of adjacent tetrahedrons while approaching the protein surface) and how these parameters influence the results and reduce the number of unknowns. The parameters are compared with respect to the resulting accuracy in the computations.

> ***"Compared to FD methods the number of linear equations to solve the lPBE is reduced by one to two orders magnitude."*** [3]

Maps of the potential-energy surfaces obtained with the FD method and the FE method are shown. Both methods yield visually very similar results but the FE method outperforms the FD method for large proteins, e.g. the adenovirus with PDB id 4CWU [88] containing 193k atoms. This computation demonstrates the robustness of mFES.

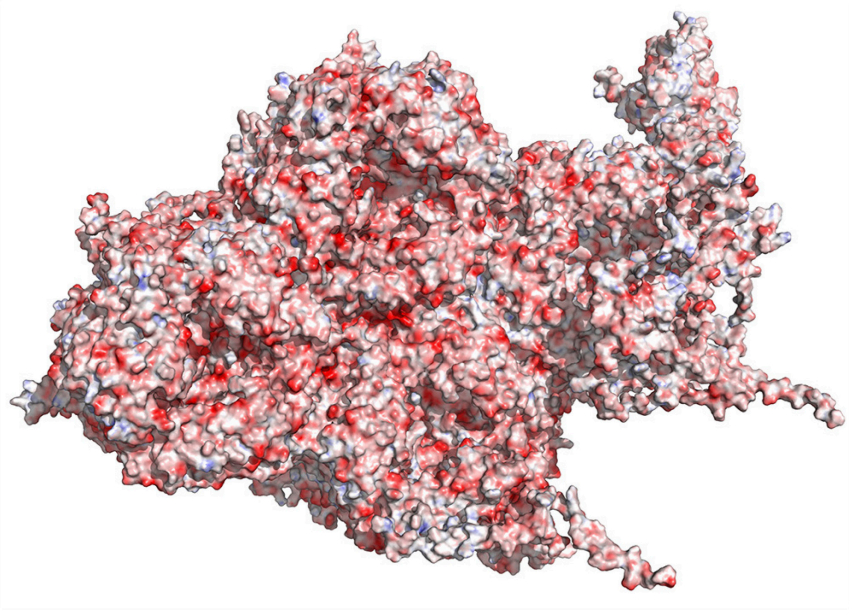


Fig 24. Electrostatic potential map for the adenovirus (PDB id 4CWU) [88] with 193k atoms in the range from -40 to 40 [kT/e] computed with mFES with moderate resolution (h_S = 0.5 Å) and visualized with PyMOL [89]

Peer-Reviewed Paper

http://dx.doi.org/10.1021/ct5005092

III. pK_A in Proteins Solving the Poisson-Boltzmann Equation with Finite Elements

Introduction

Knowledge on pK_A values is essential to understand protein structure, function, dynamics and stability under different environmental conditions. This section describes how to determine pK_A values from computed titration curves using the Finite Element (FE) and Finite Difference (FD) methods. pK_A-RMSD (*r*oot-*m*ean-*s*quare *d*eviation) values are obtained by comparing computed pK_A values with experimentally measured pK_A values. The respective theory is presented in "*Determination of pK_A Values In Proteins*", beginning at page 56. Although different computational methods have been developed in the last three decades to determine pK_A values [78], [82], [83], [90]–[101], so far there have been no pK_A computations performed with the FE method which are comparable in accuracy to the well-established FD method.

This chapter is based on a manuscript, which has been submitted for publication. In the following, results obtained with the FD and FE methods are summarized. Details of the algorithmic procedure are presented for Karlsberg+ which was developed by Kieseritzky et al. in 2008–2010 [102], [103]. Karlsberg+ provides a framework to compute pK_A values with the FD method. The RMSD to known pK_A values is 1.1 pH units if different minimization protocols are used that include protein flexibility.

In this proof of concept, lysozyme (PDB id 2lzt [56]) is analysed and a benchmark set of 342 experimentally measured pK_A values [102], [104] is used to test the FE method. A pH adapted conformer (PAC) [102] with variable hydrogen atom positions at pH7 is computed where the self-consistent cycle optimizing the

hydrogen atom positions for a specific protonation pattern is aborted after first iteration step. This simplification is applied to guarantee that the same atom coordinates are used for FE and FD method computations.

Lysozyme has coupled titratable groups [102] which may pose a challenge to the computation of correct electrostatics interactions. A cluster of coupled titratable groups is present, consisting of the three aspartates Asp52, Asp48 and Asp66 which are all influenced by Tyr53 (Fig 25). Salt bridges are present between Asp48 and Arg61, Asp66 and Arg68, and Asp119 and Arg125. A cavity is located near Asp87. The distances between this cavity and some titratable groups are stated in Fig 25. Computations done with the FD method including or excluding the cavity yields pK_A values which are deviating from each other by 0.1 pH units only.

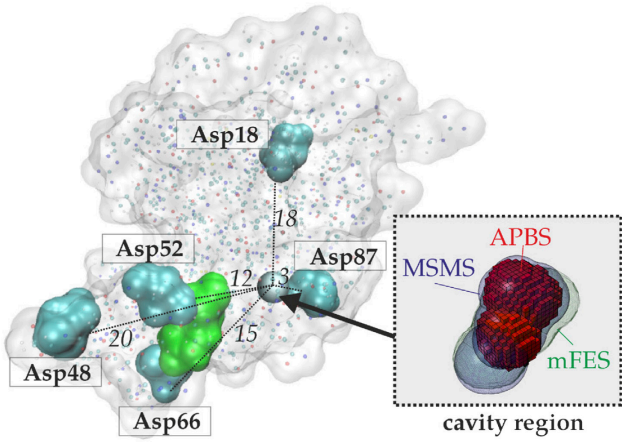


Fig 25. Aspartates near a cavity in lysozyme (PDB id 2lzt). Tyr53 is highlighted in green. Dashed lines indicate distances in Å to the small cavity near Asp87. Asp52, Asp48 and Asp66 are influenced by Tyr53. Salt bridges are present between Asp48 and Arg61, Asp66 and Arg68, and Asp119 and Arg125. The insert shows the enlarged cavity structure as calculated by different software (MSMS [54], APBS [18] and mFES [3]).

Methods

A pH adapted conformer (PAC) at pH7 is a protein conformation with fixed backbone coordinates [102], [105]. To each protein conformation hydrogen atoms are added with the HBUILD functionality of CHARMM (*C*hemistry at *HAR*vard *M*acromolecular *M*echanics) [46]–[48] and the hydrogen atom positions are energy minimized. The charges used for different protonation states are listed in appendix D, table S3. The experimental pK_A values used are listed in table 2 in the chapter "*Determination of pK_A Values in Proteins*."

The FE and FD methods use different thermodynamic cycles because of algorithmic necessities. The FD method performs a three-step focusing. First, an equidistant grid is generated which is nearly four times as large as the largest dimension of the protein. It has a varying grid constant of 2–3 Å. In the second step, the first focusing with a resolution of 1 Å is performed including the whole protein. Finally, a second focusing with a resolution of 0.25 Å is performed fitting the grid around the titratable group. Fig 20 and Fig 26 show the thermodynamic cycles, which are solved for every protonation state of every titratable group in a protein. Using the FD method, only *cycle1* (Fig 26) needs to be computed in which the upper gas phase part is replaced with the aqueous phase.

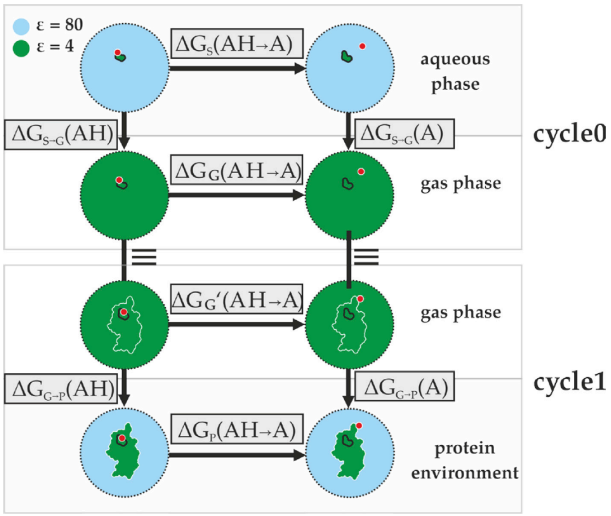


Fig 26. Two thermodynamic cycles for the computation of pK_A values with mFES are shown. Two different methods are implemented in mFES: The explicit method using both *cycle0* and *cycle1* (two-cycle approach) and the implicit method using *cycle1* and optimized values for *cycle0* (implicit two-cycle approach). The different dielectric constants used for the aqueous phase, the gas phase and the protein environment are stated. The gas phase of *cycle0* is set equal to the gas phase of *cycle1* although a different molecular volume mesh is used with different grid point locations.

Utilizing the FE method, one can choose between two different approaches: the two-cycle and the implicit two-cycle approach. The two-cycle approach performs both *cycle1* and *cycle0* computations for every titratable group, generating one model volume for the protein and one model volume for each titratable group. The implicit, two-cycle approach generates one model volume for the protein and uses this for every titratable group computation again. *cycle0* computations are implicitly included by using optimized values for them. Because the generation of model volumes is expensive when using the FE method, the implicit two-cycle approach requires less CPU time.

The Algorithmic Procedure for Computing Titration Curves. This subsection introduces the algorithmic steps for computing titration curves. It presents a pseudocode-like expression verbalising the steps performed by Karlsberg+ using

mFES (Fig 27). The three main steps are divided between *(i)* CHARMM, *(ii)* mFES, and *(iii)* Karlsberg2.

Input: A protein is chosen and the charge variable groups are selected which are going to be titrated. Charges have to be assigned to the different groups for each state. A molecular boundary surface has to be generated. Standard input files are used for charge assignment and boundary options.

CHARMM

- A standard ionization, the so-called standard protonation pattern, is assigned to the protein where every titratable group is in its reference state: all acidic groups are protonated and all basic groups including histidines are deprotonated.
- Hydrogen atoms are added to the crystal structure of the protein by HBUILD and are energy minimized while keeping all other atoms fixed because hydrogens are not included in protein crystal structures from the PDB [45].

mFES

- Either the implicit two-cycle or the two-cycle approach is employed.
- For every state of every titratable group, mFES computes the energy shift of protonation states to the reference state using model pK_A values and the energy terms $\Delta\Delta G_{born}$, eq. (71), and $\Delta\Delta G_{back}$, eq. (72). This step yields the intrinsic pK_A values for every titratable group.
- Finally, the interaction matrix *W* for the interactions between all titratable groups for all different states is computed (Fig 27).

Karlsberg2

- The previously computed intrinsic pK_A and *W* matrix values are used as well as the reference state information.

- The pH range and the step size are chosen for the computation of the titration curves.
- Karlsberg2 uses a Monte Carlo method to perform the Boltzmann averaging, eq. (77), of the different protonation patterns.
- The protonation probabilities of the titratable groups are computed at different pH values. In general, if the protonation pattern has changed compared to the previous protonation state, all steps are repeated by Karlsberg+, beginning again with "CHARMM" and using the latest protonation pattern, until the protonation pattern does not change any more. This produces a self-consistent so-called pH adapted conformer (PAC). Here, one iteration step is performed to be sure to obtain same protonation pattern and atom coordinates using FE and FD method.

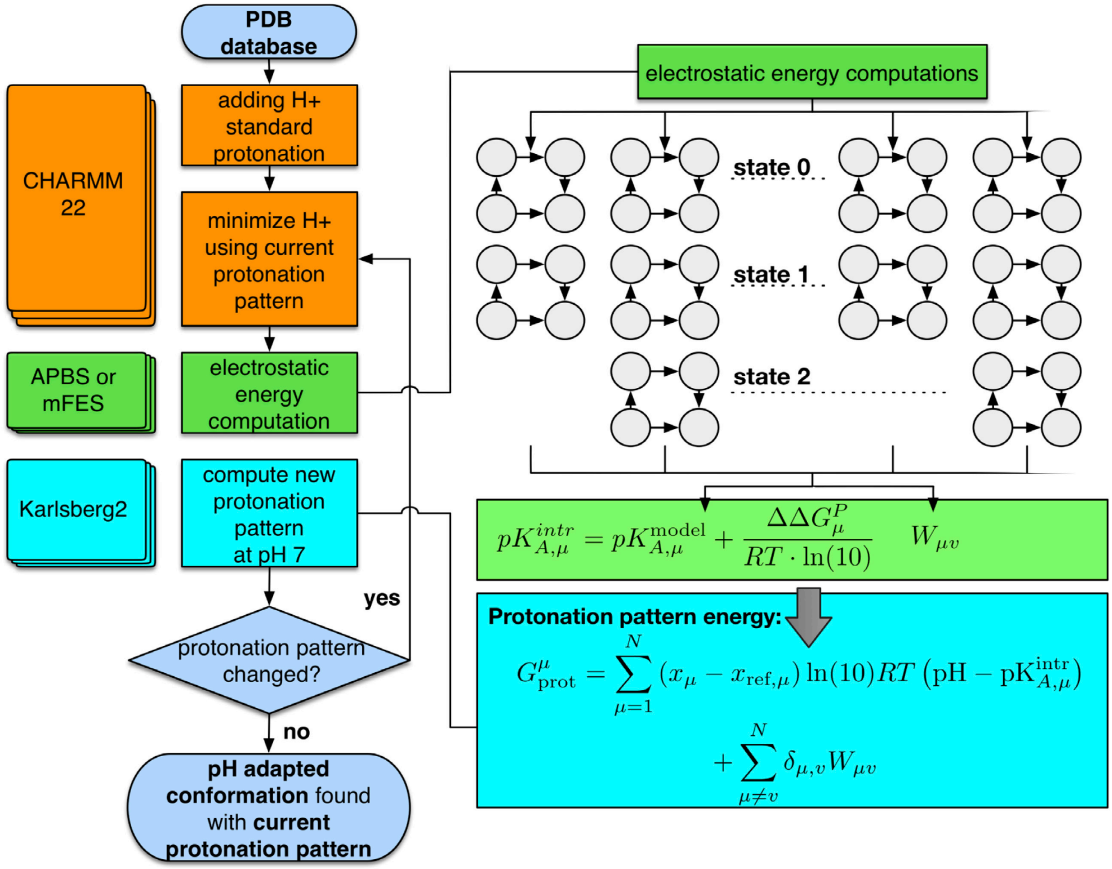


Fig 27. Overview of the algorithmic workflow for a single pH-adapted conformer (PAC) using either the FD or the FE method within the Karlsberg+ framework [79], [81], [102], [106].

Table 3. Comparison of pK_A values obtained by the FD and the FE method for lysozyme (PDB id 2lzt [56]). For the sake of simplicity the small cavity in lysozyme close to residue Asp87 (see Fig 25) is filled with an uncharged dummy atom. All titratable groups for which experimental pK_A values are available are considered here. Groups marked in bold letters are coupled. The FE method computations are done using second-order solution, average edge length $h_S = 0.3$ Å (fine) or 0.5 Å (coarse), grading parameter of $g = 0.5$, ion concentration of 0.1 M, radius of 2 Å for ions and probe sphere rolling over the IEL with radius of 2 Å.

group	FD[a] fine	FE[b] coarse	FE[c] fine
Cter129	3.0	3.1	3.1
Asp18	1.9	1.8	1.9
Asp48	-0.7	-0.8	-0.8
Asp52	1.7	1.2	1.3
Asp66	-2.5	-2.8	-2.8
Asp87	-1.2	-1.3	-1.4
Asp101	4.4	4.4	4.5
Asp119	2.7	2.7	2.6
Glu7	3.1	2.9	3.1
Glu35	4.4	4.3	4.2
His15	5.7	5.3	5.6
Lys1	9.3	9.1	9.2
Lys13	10.6	10.5	10.4
Lys33	10.7	10.8	10.4
Lys96	10.7	10.4	10.4
Lys97	10.9	11.0	10.8
Lys116	8.7	8.7	8.8
Tyr20	15.8	15.5	15.7
Tyr23	10.8	10.9	10.9
Tyr53	23.7	23.2	23.4
	RMSD	**0.24**	**0.18**

[a] The first two focusing steps use $n^3 = 381^3 = 5.5\ 10^7$ grid points with 1 Å and 0.25 Å resolution, respectively, covering the whole protein. The third and last focusing step with the same protein center uses $n^3 = 449^3 = 9.1\ 10^7$ grid points with 0.125 Å lattice constant.

[b] The average grid point distance on the triangular surface is $h_S = 0.5$ Å for the protein and the group volume model. The resulting protein volume model consists of $1.3\ 10^5$ grid points ($1.1\ 10^6$ grid points for second-order solution).

[c] The average grid point distance on the triangular surface is $h_S = 0.3$ Å for the protein and the group volume model. The resulting protein volume model consists of $4.7\ 10^5$ grid points ($3.8\ 10^6$ grid points for second-order solution).

Results and Discussion

The pK_A values are computed with the FE method [3] and the well-established FD method [18], [39]–[41] using the crystal structure of 15 proteins. Lysozyme (PDB id 2lzt [56], table 3) is analysed in more detail. A single pH-adapted conformer at pH7 is generated for every protein by the application Karlsberg+ [102], [103] using one iteration step to guarantee that the same coordinates are used for the FD and FE method.

Tyr53 is forming a salt-bridge with Asp52. Maximum deviations of computed pK_A values between the high resolution FD and the low (high) resolution FE method are observed for Tyr53 [0.56 pH (0.35 pH) units] and for Asp52 [0.44 pH (0.31 pH) units] (see table 3). The FE method guarantees that the molecular surface is defined by grid points, which are precisely on the surface. This is not the case for the FD method and may be the reason for some of these moderate deviations.

Analysing the electrostatic energy terms obtained with the FD and FE method it was found that the pK_A values at the reference protonation state (i.e. the intrinsic pK_A) are practically the same (appendix D, table S1), as well as the interactions of the titratable residues. Hence, a reason for the deviations of the computed pK_A values between the FD and FE methods could be the different definition of the ion exclusion layer. The FD method APBS uses a vdW surface with atom radii extended by 2 Å (radius of ions) whereas the FE method rolls with a probe sphere of ion radius 2 Å over the same vdW surface. The resulting smoothened ion exclusion layer makes the algorithm to mesh the surface by triangles more robust. To explore the effect of the difference in the ion exclusion layer, a small rolling sphere radius of 0.3 Å was used for the FE method yielding deviations in the pK_A values, which are smaller than 0.2 pH units.

Table 4. pK_A-RMSD values of computed pK_A values compared with measured experimental pK_A values for lysozyme (PDB id 2lzt [56]). One pH adapted conformer at pH 7 is used to obtain pK_A values from computations with the Karlsberg+ program [102], [103] utilizing one iteration step. mFES uses an average edge length of h_S = 0.5 Å, a grading parameter of g = 0.5, an ion concentration of 0.1 M, a radius of 2 Å for ions and a probe radius for the rolling-sphere over the ion-exclusion layer of 2 Å. More detailed values are given in appendix D, table S2. Tyr53 is excluded because it does not titrate under physiological conditions using the crystal structure information and Lys1 is excluded because it is located at the N-terminus and has an unusual structure.

titratable group	pK_A-RMSD FD	pK_A-RMSD FE (two-cycle, 2nd order)	pK_A-RMSD FE (implicit 2-cycle, 2nd order)	pK_A-RMSD FE (implicit 2-cycle, 1st order)
Asp	2.0	2.2	1.9	1.4
Glu	1.3	1.3	1.8	1.8
Tyr	3.9	3.8	3.5	3.5
His	0.3	0.1	0.3	0.2
Lys	0.7	0.8	1.0	1.3

Comparing the results from the two methods with the experimental pK_A values of lysozyme, qualitatively similar pK_A-RMSD values are obtained for both methods (Table 4). While the pK_A-RMSD for the FD method is 1.95, the pK_A-RMSD for the FE method using the two-cycle approach is 2.00 which, therefore, is practically of the same quality. Interestingly, for the implicit two-cycle FE method with optimized electrostatic pK_A^{solv} values[2] a pK_A-RMSD of 1.91 is obtained. The coarse FE method using the implicit method with a first-order instead of a second-order solution yields a better agreement with the experimental values, resulting in a pK_A–RMSD of 1.77, because the pK_A^{solv} values were optimized for the coarse FE method. The coarse-grained first-order geometry molecular surface yields the same

[2] The following optimized solution pK_A values are used for the implicit two-cycle approach: Arg: 10.51; Lys: 12.68; Glu: 11.63; His-δ: 8.78; His-ε: 0.29; Tyr: 9.01; Asp: 12.55.

accuracy for pK_A computations as does the more fine-grained second order geometry for solvation energies.

Additionally, a benchmark set of 15 proteins involving 185 residues with measured pK_A values [102] is utilized to compare the FE and FD method. Results are shown in Fig 28. The qualitative pK_A value difference between both methods is comparable to pK_A values obtained for lysozyme. The pK_A-RMSD between both methods is 0.3. The pK_A-RMSD of both methods computed based on the crystal structures of the considered 15 proteins involving 342 measured pK_A values for 185 residues is 2.5.

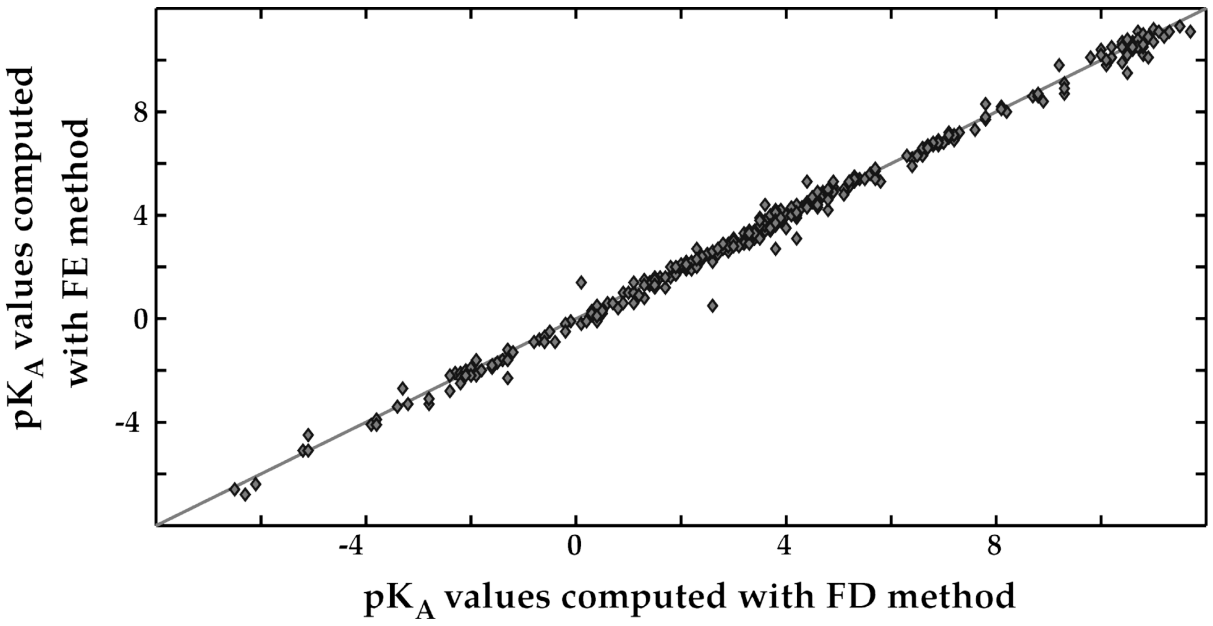


Fig 28. pK_A values are computed with the FD and FE method. A benchmark set [102] with 15 proteins (involving 342 measured pK_A values) is used for this plot. The FE method uses a two-cycle approach, a second-order solution and a resolution of $h_S = 0.5$ Å. The FD method performs a three step focusing procedure with a resolution of 0.25 Å. The RMSD value between both methods is 0.3.

Conclusions

For the first time it has been demonstrated that both the FD and the FE method yield results of similar quality. The FE fine and coarse computations are as precise as the FD fine computations using the two-cycle thermodynamic cycle approach of mFES for lysozyme. The values obtained by the FE method are compared with 342 experimentally measured pK_A values. Both computational methods, FE and FD, yield pK_A values of the same quality as shown by the respective pK_A-RMSD to the experimental values.

Solvent pK_A values are optimized to be subsequently used in the implicit two-cycle FE approach. Employing these values, the pK_A-RMSD obtained with a two-cycle approach using the FE or the FD method have the same quality as with the FE method with the implicit two-cycle approach, which means that the values used in this work are close to optimum values.

Although electrostatic solvation energy computations need a second-order solution and a meshing of the molecular surface with a resolution of $h_S = 0.5$ Å, computing the pK_A values needs less accuracy and requires only a first-order solution. Combining a first-order solution with the implicit two-cycle approach using the FE method delivers nearly the same pK_A-RMSD values (table 4). For the pK_A values, a quality similar to that of the solvation energies is achieved, since in the case of the pK_A values double differences $\Delta\Delta G$ are computed which are less error-prone than single differences ΔG. Hence, single differences like solvation energies need higher accuracy of the molecular surface because there is less error cancellation.

IV. mFES Implementation and mFES+ Web Services

Implementation

mFES is a stand-alone program written in C/C++ and is available for free on the respective web page (agknapp.chemie.fu-berlin.de/mfes) of Prof. Knapp's workgroup or at github (github.com/imago). Installation scripts and ebuilds are also available on the web page under the GNU lesser general public license v2.0 (LGPLv2) and is free for the community to extend and modify provided that the contributors to this project are cited.

mFES is based on the Boost C++ library and includes vcglib [107] to provide a variety of algorithms to process meshes. LSMS [2] is included to perform the level-set method [108] as well as the marching cubes algorithm [109]. NETGEN [1] is incorporated because of its high quality advancing front method implementation [51], [52] as well as its volume meshing, optimization and analyzing capabilities [110]–[114]. NGSolve [115], which is on top of NETGEN, provides an interface to different solvers like MUMPS [42]–[44], Hypre [116], and Pardiso [117], [118] as well as other multigrid, algebraic multigrid [119] or direct methods.

Different parallelization techniques are ready to be used. In principle, a large molecule can be split into several subproblems, which are then solved in parallel by different CPUs. The solutions of the subproblems are subsequently merged in the same process (*divide-and-conquer* method) using ParMETIS [120], [121] which implements parallel graph-partitioning algorithms. ParMETIS is compiled together with NETGEN and mFES. The parallelization features may be provided in future mFES releases.

An alternative molecular surface mesher is being developed with the support of Joachim Schöberl. The so-called "voxelizer" uses a level-set method. It works well

for small molecules without cavities or titratable groups. It runs in parallel, is fast, and delivers even more precise molecular surfaces.

Utilizing mFES in other frameworks (developed by other groups) may yield results which lead to better pK_A-RMSD values or deliver new insights into protein structure, function and stability. An ongoing project is to insert mFES into CHARMM with the help of Milan Hodošček and to make this software useful for a wider community.

mFES$^+$ Web Services

The mFES$^+$ web application is a service for solving different computational problems:

i. solvation energies of proteins (ΔG)

ii. pK_A values of titratable groups in proteins.

These web services are written in PHP and are based on Drupal [122], jQuery [123], jsc3d [124] and Google APIs like Google Charts [125]. Several scripts have been written during the implementation of mFES and are provided in a tools directory.

Computation of Solvation Energies. At the current stage a web service is provided for computing electrostatic solvation energies. Molecules can be selected by their PDB id or by uploading the coordinates of a protein structure in PDB format. The user is free to specify different parameters, like ion concentration, solvent probe radius, ion radius, resolution of the molecular surface, dielectric constants, and the number of Taubin smoothing steps [126] to be performed. Jobs are submitted to a compute cluster queue. The user is informed via e-mail when a job is successfully processed, and a link to the results is provided. As an example, the output for 1div is presented in Fig 29.

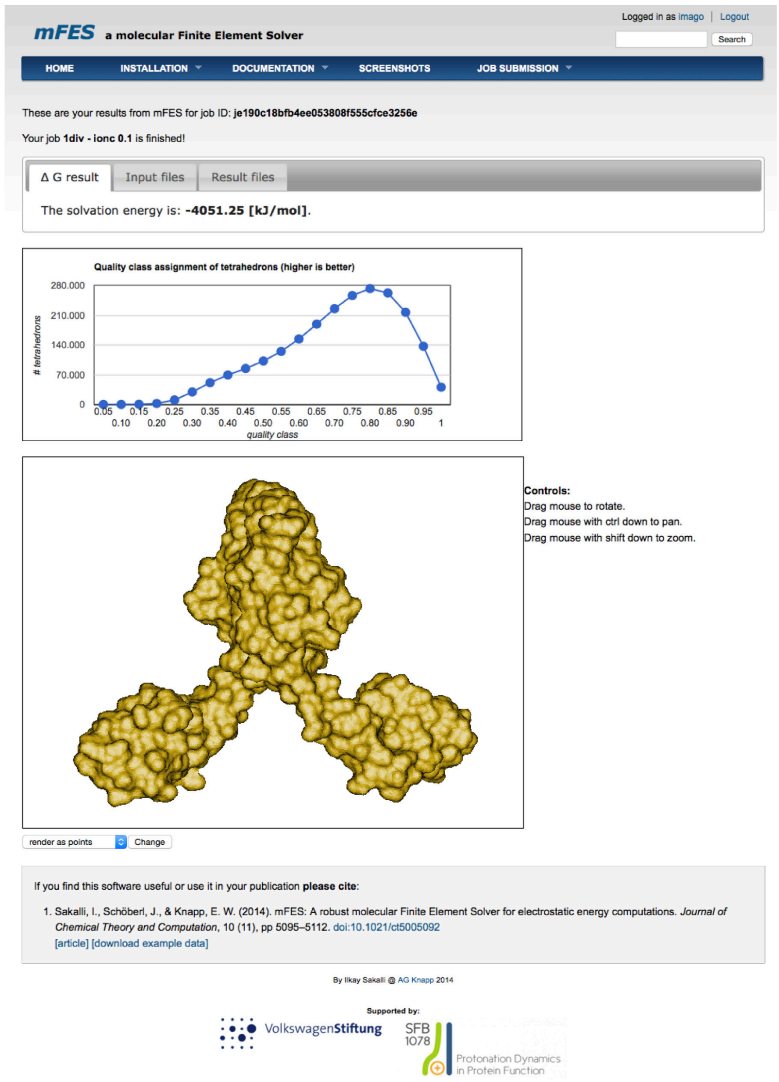


Fig 29. Screenshot of the mFES^{+} web results for the computation of the electrostatic solvation energy of the ribosomal protein L9 (PDB id 1div [127]). The interactive visualization of the molecular surface is shown as well as the quality class assignment of the tetrahedrons for the protein volume.

The user is able to download the molecular volume model as well as the molecular surface and other files, like the input files which may be used to compute jobs on a local machine after installing mFES.

Computation of pK_A Values. While this thesis is being written, the module for pK_A computations is in beta phase. These computations are based on the web service Karlsberg+ utilizing mFES instead of APBS. The user can choose a protein structure

by providing its PDB id or upload a protein structure in PDB format and will receive an e-mail with a link to the results. An example output for 2lzt is presented in Fig 30 and Fig 31.

The pK_A values are plotted ordered by titratable group and residue ids and titration curves are plotted for every group, showing both the probability for protonated and the deprotonated group. It is possible to download the protonation pattern at different pH values and temperatures as a PDB or PQR file. The FE lattice of the protein, both for the volume and the surface, is provided with 3D navigation or as a download.

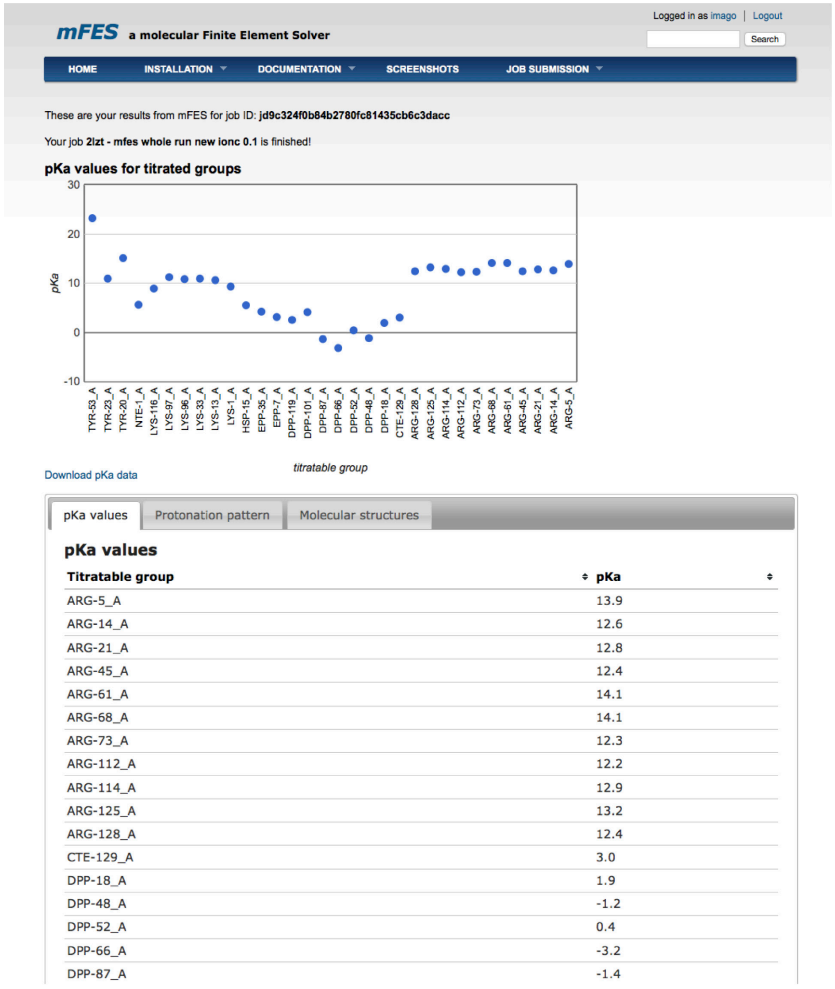


Fig 30. Screenshot of the pK_A values computed with mFES⁺ for pH7 adapted conformer of lysozyme (PDB id 2lzt [56]). This screen is continued in the next figure.

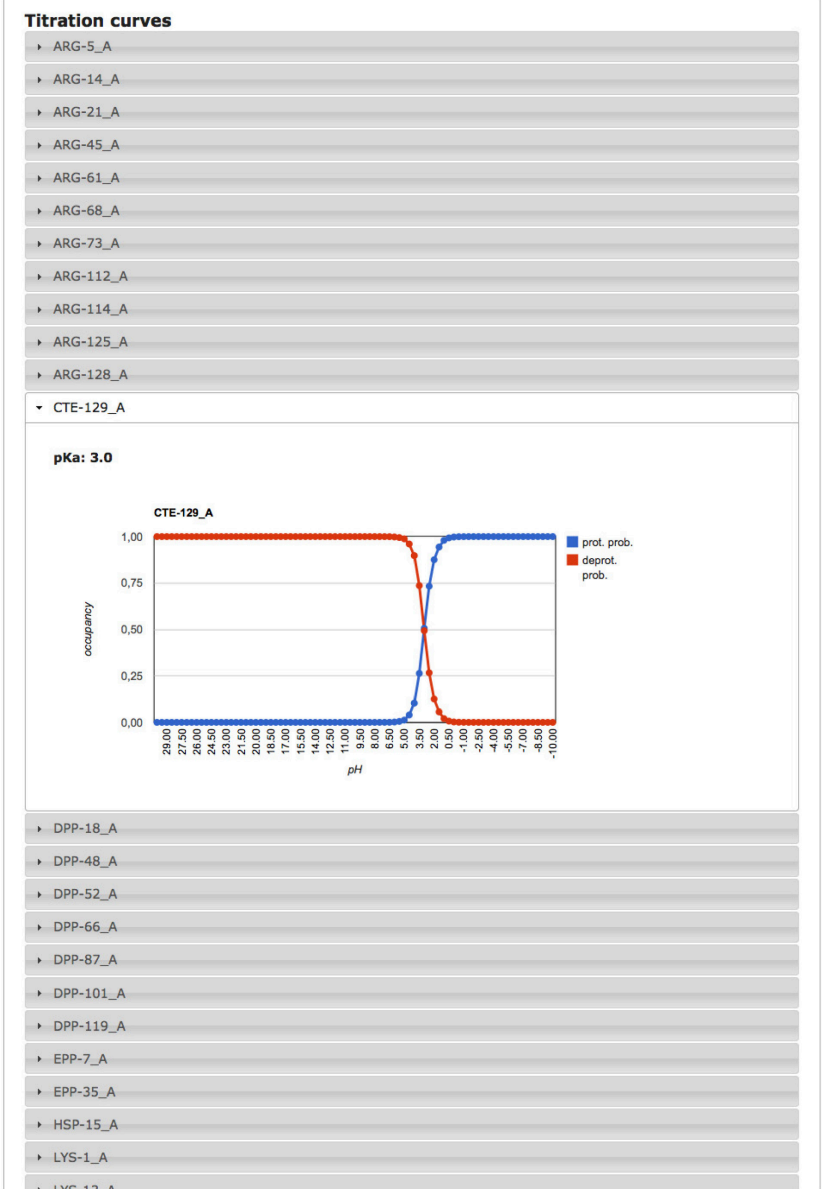


Fig 31. Screenshot of titration curves computed with mFES+ for the pH7 adapted conformer of lysozyme (PDB id 2lzt [56]). All titratable groups are listed.

Conclusion and Outlook

"We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity,
and especially because it produces objects of beauty."
— Donald E. Knuth (*1955)

This thesis provides insight into the performance of the new software mFES (*m*olecular *F*inite *E*lement *S*olver) which employs the Finite Element method. mFES is capable of computing solvation energies and pK_A values of macromolecules. It generates an irregular object adapted protein surface and volume grid and builds a linear equation system to solve the *l*PBE yielding electrostatic potentials of proteins. mFES is extendable and available free of charge under LGPLv2 licence conditions. It uses a number of open-source software to fulfil its tasks. Before this thesis was written, there was no Finite Element solver which had the ability to compete with well-established FD methods in terms of reliability, precision and CPU time. mFES outperforms the FD method in computing electrostatic solvation energies and yet delivers pK_A values of the same quality compared to FD computations.

Singular behaviour of the linear equation system of the *l*PBE derived from the FE method might occur if the molecular surfaces are not generated with sufficient care. Thus, results of this thesis might also help the scientific community when using the Boundary Element (BE) method which requires a faithful representation of molecular surfaces.

Installation scripts are available for the compilation of mFES on general-purpose UNIX systems. Web services have been set up for public users to perform computations without the need to install mFES locally. mFES may help other software frameworks to achieve accurate results for other problems which require a faithful representation of molecular surfaces and volumes.

Several features are still under development. One of those is the option of using a second-order surface definition to reduce the number of molecular grid points while maintaining the same precision of results. It might be interesting to try out new algorithms like quadric-based mesh decimation [128] techniques that can further reduce the number of surface points without loss of molecular surface curvature information. The number of grid points is equal to the number of coupled linear equations. Hence, reducing the number of grid points results in a faster algorithm. Parts of mFES can be parallelized. The most promising approach here lies in splitting a molecular model into portions using ParMETIS [120], [121], solving the subproblems on different CPUs, and combining the solutions to find the solution of the original problem (*divide-and-conquer* method).

Finite Element methods are used in industry to perform stress tests, for product optimization, or to compute electromagnetic or fluid dynamics. The application of the FE method to such problems is relatively new, so it is believed that there will be progress in solving problems with the FE method also in other fields. In return, by utilizing the experiences made in the course of this progress, mFES might perform better in the future by using e.g. more parallelism techniques to help the community to find answers to difficult questions in computational chemistry and to deal with large systems.

Summary in English

In this interdisciplinary work, the computer program *mFES* (*m*olecular *F*inite *E*lement *S*olver) has been developed for solving electrostatic problems associated with proteins. It is fast and yields results with a higher accuracy than that obtained with the more traditional software based on finite difference methods.

Electrostatics is an important topic in computational chemistry. This work focuses on the computation of electrostatic properties of small molecules like peptides as well as large proteins like viruses. mFES is based on the well-defined Finite Element (FE) method and solves the linear Poisson-Boltzmann equation, a second-order partial differential equation, where a solution algorithm was for instance described by Warwicker and Watson in 1982 [129].

Besides electrostatic potentials $\Phi(r)$, single and double energy differences $\triangle G$ and $\triangle\triangle G$ are computed to determine different properties, like the electrostatic solvation energy and pK_A values, using a robust generation of the molecular model.

Fundamental progress has been made in this thesis which continues preceding approaches of Friesner et al. [130]–[132] and Holst et al. [41], [133] mostly more than a decade ago. mFES is based on LSMS [2] and NETGEN [1] in building molecular models and on MUMPS [42]–[44] in solving the linear Poisson-Boltzmann equation.

The key improvement lies in the way molecular surfaces (solvent excluded surfaces) are generated. While the molecular surface itself is well-defined [54], [134], [135], it remained difficult up to now to compute very precise molecular surfaces without singularities. Using the advancing front method [51], [52] implemented by Schöberl et al. [1], this problem is solved by producing a triangulation of the molecular surface which is controllable by parameters like the average edge length of the triangles on the molecular surface. Another key improvement of the Finite

Element solver lies in the way the tetrahedral volume elements are optimized, thus producing an overall molecular model of high quality.

Comparing the FE method with Boundary Element and Finite Difference methods, the focus lies on the latter because it is well-established and most commonly used in the electrostatics community. The Finite Element methods that have been available up to now and which are directly compared are not able to yield an accuracy which could compete with that of the Finite Difference method. This is now solved by mFES.

Electrostatic solvation energies are computed for average- and large-sized proteins (bovine pancreatic trypsin inhibitor, barnase, lysozyme, cyctochrome c oxidase) and the adenovirus serves as an example for a very large protein with nearly 193,000 atoms, including hydrogen atoms. Improvements in accuracy resulted in electrostatic solvation energy differences as high as 30 [kJ/mol] for average-sized proteins. All molecular systems based on the FE method needed much fewer equations to solve the Poisson-Boltzmann equation compared with using the FD method and still higher accuracy is achieved.

A proof of concept is the computation of pK_A values with mFES. pK_A values for lysozyme and other proteins are computed and compared with 342 experimentally measured pK_A values as well as with results obtained with the Finite Difference method. These computations are done using the classical Karlsberg+ program developed by Kieseritzky et al. [102], [103]. pK_A-RMSD values with respect to measured values computed by mFES are of the same quality compared with the values obtained by the FD method. $mFES^+$ web application services for using mFES without the need for a local installation are available at

http://agknapp.chemie.fu-berlin.de/mfes

Installation routines for the stand-alone mFES program are provided as well as example runs and a documentation to facilitate the use of mFES in other frameworks.

Zusammenfassung auf Deutsch

In dieser interdisziplinären Arbeit ist das Softwareprogramm *mFES* (*m*olecular *F*inite *E*lement *S*olver) entstanden, um elektrostatische Fragestellungen für Proteine zu lösen. mFES ist schnell und berechnet Ergebnisse mit höherer Präzision als traditionelle Software, die auf der Finite Differenzen Methode basiert.

Die Elektrostatik ist ein wichtiges Thema der computergestützten Chemie. Diese Arbeit legt ihren Fokus auf die Berechnung elektrostatischer Eigenschaften kleiner Moleküle wie Peptide bis hin zu großen Proteine wie z. B. Viren. mFES basiert auf einer wohldefinierten Finite Elemente (FE) Methode und löst die lineare Poisson-Boltzmann-Gleichung, eine partielle Differentialgleichung zweiter Ordnung, für die zum Beispiel Warwicker und Watson 1982 einen Lösungsalgorithmus beschrieben haben [129].

Neben elektrostatischen Potentialen $\Phi(r)$ werden einfache und doppelte Energiedifferenzen, $\triangle G$ und $\triangle\triangle G$, berechnet, um verschiedene Eigenschaften, wie elektrostatische Solvatisierungsenergien und pK_A Werte, zu berechnen, wofür robuste molekulare Modelle generiert werden.

Fundamentale Fortschritte gelangen mit dieser Arbeit, welche an die Arbeiten von u. a. Friesner et al. [130]–[132] und Holst et al. [41], [133] anknüpft, die meist mehr als ein Jahrzehnt zurückliegen. mFES benutzt zur Generierung molekularer Modelle LSMS [2] und NETGEN [1] und zur Lösung der linearen Poisson-Boltzmann Gleichung MUMPS [42]–[44].

Eine entscheidende Verbesserung der elektrostatischen Berechnungen liegt in der Art der Erzeugung molekularer Oberflächen (solvent excluded surface). Obwohl diese Oberfläche wohldefiniert ist [54], [134], [135], ist es bis heute schwierig, eine nahezu exakte Oberfläche ohne Singularitäten zu generieren. Durch die Nutzung der Advancing-Front-Methode [51], [52], welche u. a. von Joachim Schöberl [1] entwickelt wurde, wird dieses Problem gelöst, indem eine Triangulation der molekularen Oberfläche generiert wird, die durch verschiedene Parameter, beispielsweise die mittlere Kantenlänge der Dreiecke, kontrollierbar ist. Eine weitere entscheidende Verbesserung des Finite Elemente Lösers liegt in der Art der Optimierung der

Tetraeder-Volumenelemente, welche ein molekulares Gesamtmodell mit hoher Qualität erzeugt.

Beim Vergleich der Finiten Elemente Methode mit der Rand Elemente Methode und der Finite Differenzen Methode liegt der Fokus auf letzterer, weil diese etabliert ist und überwiegend von den Experten benutzt wird. Die bisher verfügbaren Finite Elemente Methoden, die direkt miteinander verglichen werden, liefern im Vergleich zur Finite Differenzen Methode keine hohe Genauigkeit. Dieses Problem wird mit mFES gelöst.

Elektrostatische Solvatisierungsenergien wurden für kleine und große Proteine berechnet (Rinderpankreas-Trypsininhibitor, Barnase, Lysozym und Cyctochrom-*c*-Oxidase), aber auch für Adenovirus-Proteine, welche ein Beispiel für sehr große Proteine mit fast 193.000 Atomen inklusive der Wasserstoffatome darstellen. Verbesserungen der Präzision ergaben elektrostatische Solvatisierungsenergie Unterschiede von bis zu 30 [kJ/mol] für durchschnittlich große Proteine. Alle molekularen Systeme, die auf der Finiten Elemente Methode basieren, benötigen hierbei weniger lineare Gleichungen zur Lösung der Poisson-Boltzmann-Gleichung als die Finite Differenzen Methode.

Ein Machbarkeitsbeweis ist die Berechnung von pK_A-Werten mit mFES. pK_A-Werte wurden für Lysozym und andere Proteine berechnet und mit 342 experimentell ermittelten pK_A-Werten und mit den Ergebnissen der Finite-Differenzen-Methode verglichen. Für die Berechnungen wurde das klassische Programm Karlsberg+ benutzt, welches von Kieseritzky et al. [102], [103] entwickelt wurde. pK_A–RMSD-Werte in Bezug auf gemessene Werte beweisen, dass die mit mFES berechneten pK_A-Werte die gleiche Güte haben wie pK_A-Werte, die mit der FD-Methode berechnet wurden. Webservices für die Nutzung von $mFES^+$ web ohne eine lokale Installation sind verfügbar unter

http://agknapp.chemie.fu-berlin.de/mfes

Routinen zur Installation von mFES stehen zur Verfügung, ebenso einige Beispielrechnungen und eine Dokumentation, um die Möglichkeit bereitzustellen, mFES in andere Frameworks einzubinden.

STATUARY DECLARATION

Hereby, I testify that this thesis is the result of my own work and research, except of references given in the bibliography. This work contains material that is the copyright property of others, which cannot be reproduced without the permission of the copyright owner. Such material is clearly identified in the text.

Ilkay Sakalli,

May 2015, Berlin, Germany

Bibliography

[1] J. Schöberl, "NETGEN An advancing front 2D/3D-mesh generator based on abstract rules," *Comput. Vis. Sci.*, vol. 1, no. 1, pp. 41–52, Jul. 1997.

[2] T. Can, C.-I. Chen, and Y.-F. Wang, "Efficient molecular surface generation using level-set methods.," *J. Mol. Graph. Model.*, vol. 25, no. 4, pp. 442–54, Dec. 2006.

[3] I. Sakalli, J. Schöberl, and E. W. Knapp, "mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations," *J. Chem. Theory Comput.*, vol. 10, no. 11, pp. 5095–5112, Oct. 2014.

[4] F. Fogolari, A. Brigo, and H. Molinari, "The Poisson-Boltzmann equation for biomolecular electrostatics: a tool for structural biology.," *J. Mol. Recognit.*, vol. 15, no. 6, pp. 377–92, Jan. .

[5] C. N. Schutz and A. Warshel, "What are the dielectric 'constants' of proteins and how to validate electrostatic models?," *Proteins*, vol. 44, no. 4, pp. 400–17, Sep. 2001.

[6] M. K. Gilson and B. Honig, "Calculation of the total electrostatic energy of a macromolecular system: solvation energies, binding energies, and conformational analysis.," *Proteins*, vol. 4, no. 1, pp. 7–18, Jan. 1988.

[7] H. Tjong and H.-X. Zhou, "On the Dielectric Boundary in Poisson-Boltzmann Calculations.," *J. Chem. Theory Comput.*, vol. 4, no. 3, pp. 507–514, Mar. 2008.

[8] M. K. Gilson and B. H. Honig, "The dielectric constant of a folded protein.," *Biopolymers*, vol. 25, no. 11, pp. 2097–119, Nov. 1986.

[9] L. Li, C. Li, Z. Zhang, and E. Alexov, "On the Dielectric 'Constant' of Proteins: Smooth Dielectric Function for Macromolecular Modeling and Its Implementation in DelPhi.," *J. Chem. Theory Comput.*, vol. 9, no. 4, pp. 2126–2136, Apr. 2013.

[10] C.-A. Coulomb, "Construction et usage d'une balance électrique, fondée sur la propriété qu'ont les fils de métal, d'avoir une force de réaction de torsion proportionnelle à l'angle de torsion," *Hist. l'Académie [royale] des Sci. avec les mémoires mathématiques Phys. partie "Mémoires" [1785]*, pp. 569–577, 1788.

[11] P. Signell, "Gauss's Law For Spherical Symmetry," *Physnet, MISN-0-132, Michigan State Univ.*

[12] M. E. Davis and J. A. McCammon, "Electrostatics in biomolecular structure and dynamics," *Chem. Rev.*, vol. 90, no. 3, pp. 509–521, May 1990.

[13] K. A. Sharp and B. Honig, "Calculating total electrostatic energies with the nonlinear Poisson-Boltzmann equation," *J. Phys. Chem.*, vol. 94, no. 19, pp. 7684–7692, Sep. 1990.

[14] G. N. Lewis and M. Randall, "The activity coefficient of strong electrolytes. 1," *J. Am. Chem. Soc.*, vol. 43, no. 5, pp. 1112–1154, May 1921.

[15] A. J. Schwab, *Begriffswelt der Feldtheorie*, vol. 3. Springer-Verlag Berlin, Heidelberg, New York, 1993.

[16] A. H. Boschitsch and M. O. Fenley, "A Fast and Robust Poisson-Boltzmann Solver Based on Adaptive Cartesian Grids.," *J. Chem. Theory Comput.*, vol. 7, no. 5, pp. 1524–1540, May 2011.

[17] K. B. Lipkowitz, R. Larter, and T. R. Cundari, Eds., *Biomolecular Applications of Poisson–Boltzmann Methods*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2005.

[18] N. A. Baker, D. Sept, S. Joseph, M. J. Holst, and J. A. McCammon, "Electrostatics of nanosystems: application to microtubules and the ribosome.," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 98, no. 18, pp. 10037–10041, Aug. 2001.

[19] J. Liang and S. Subramaniam, "Computation of molecular electrostatics with boundary element methods.," *Biophys. J.*, vol. 73, no. 4, pp. 1830–41, Oct. 1997.

[20] A. H. Boschitsch, M. O. Fenley, and H.-X. Zhou, "Fast Boundary Element Method for the Linear Poisson–Boltzmann Equation," *J. Phys. Chem. B*, vol. 106, no. 10, pp. 2741–2754, Mar. 2002.

[21] S. A. C. Chandrajit Bajaj, "Efficient and accurate higher-order fast multipole boundary element method for poisson boltzmann electrostatics.," *Univ. Texas.*, 2009.

[22] M. D. Altman, J. P. Bardhan, J. K. White, and B. Tidor, "Accurate solution of multi-region continuum biomolecule electrostatic problems using the linearized Poisson-Boltzmann equation with curved boundary elements.," *J. Comput. Chem.*, vol. 30, no. 1, pp. 132–53, Jan. 2009.

[23] A. Juffer, E. F. . Botta, B. A. . van Keulen, A. van der Ploeg, and H. J. . Berendsen, "The electric potential of a macromolecule in a solvent: A fundamental approach," *J. Comput. Phys.*, vol. 97, no. 1, pp. 144–171, Nov. 1991.

[24] B. Lu, D. Zhang, and J. A. McCammon, "Computation of electrostatic forces between solvated molecules determined by the Poisson-Boltzmann equation using a boundary element method.," *J. Chem. Phys.*, vol. 122, no. 21, p. 214102, Jun. 2005.

[25] R. J. Zauhar and R. S. Morgan, "A new method for computing the macromolecular electric potential," *J. Mol. Biol.*, vol. 186, no. 4, pp. 815–820, Dec. 1985.

[26] R. J. Zauhar and R. S. Morgan, "The rigorous computation of the molecular electric potential," *J. Comput. Chem.*, vol. 9, no. 2, pp. 171–187, Mar. 1988.

[27] M. D. Altman, J. P. Bardhan, B. Tidor, and J. K. White, "FFTSVD: A Fast Multiscale Boundary-Element Method Solver Suitable for Bio-MEMS and Biomolecule Simulation," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 25, no. 2, pp. 274–284, Feb. 2006.

[28] B. Lu and J. A. McCammon, "Improved Boundary Element Methods for Poisson–Boltzmann Electrostatic Potential and Force Calculations," *J. Chem. Theory Comput.*, vol. 3, no. 3, pp. 1134–1142, May 2007.

[29] A. Haji-Akbari, M. Engel, A. S. Keys, X. Zheng, R. G. Petschek, P. Palffy-Muhoray, and S. C. Glotzer, "Disordered, quasicrystalline and crystalline phases of densely packed tetrahedra.," *Nature*, vol. 462, no. 7274, pp. 773–7, Dec. 2009.

[30] S. Takashima and H. P. Schwan, "Dielectric Dispersion of Crystalline Powders of Amino Acids, Peptides, and Proteins 1," *J. Phys. Chem.*, vol. 69, no. 12, pp. 4176–4182, Dec. 1965.

[31] I. Milton Abramowitz; Stegun A., *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables*. New York, N. Y., USA: Dover Publications, 1965.

[32] A. C. Polycarpou, *Introduction to the Finite Element Method in Electromagnetics*, Synthesis. San Rafael, Cal., USA: Morgan & Claypool Publishers, 2006.

[33] M. J. Holst, "The Poisson-Boltzmann Equation: Analysis and Multilevel Numerical Solution," University of Illinois at Urbana-Champaign, 1994.

[34] G. H. Golub and C. F. Van Loan, *Matrix Computations*. JHU Press, 1996.

[35] D. Bashford, "An Object-Oriented Programming Suite for Electrostatic Effects in Biological Molecules," in *Proceedings of the first International Conference, ISCOPE*, 1343rd ed., vol. 1343, R. R. Ishikawa, Y., Oldehoeft, Ed. Marina del Rey, Cal., USA: Springer, 1997, pp. 233–240.

[36] A. Nicholls and B. Honig, "A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation," *J. Comput. Chem.*, vol. 12, no. 4, pp. 435–445, May 1991.

[37] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[38] M. R. Hestenes, "Methods of conjugate gradients for solving linear systems," *J. Res. Natl. Bur. Stand. (1934).*, vol. 49, no. 6, 1952.

[39] M. Holst and F. Saied, "Multigrid solution of the Poisson- Boltzmann equation," *J. Comput. Chem.*, vol. 14, no. 1, pp. 105–113, Jan. 1993.

[40] M. J. Holst and F. Saied, "Numerical solution of the nonlinear Poisson-Boltzmann equation: Developing more robust and efficient methods," *J. Comput. Chem.*, vol. 16, no. 3, pp. 337–364, Mar. 1995.

[41] M. Holst, "Adaptive Numerical Treatment of Elliptic Systems on Manifolds," *Adv. Comput. Math.*, vol. 15, no. 1–4, pp. 139–191, Jan. 2001.

[42] P. R. Amestoy, T. A. Davis, and I. S. Duff, "An Approximate Minimum Degree Ordering Algorithm," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 4, pp. 886–905, Oct. 1996.

[43] P. R. Amestoy, I. Duff, and J. Y. L'Excellent, "MUMPS MUltifrontal Massively Parallel Solver," *Tech. Report, TR/PA/98/02*, 1998.

[44] P. R. Amestoy, I. S. Duff, and J.-Y. L'Excellent, "Multifrontal parallel distributed symmetric and unsymmetric solvers," *Comput. Methods Appl. Mech. Eng.*, vol. 184, no. 2–4, pp. 501–520, Apr. 2000.

[45] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, W. Helge, I. N. Shindyalow, and P. E. Bourne, "The Protein Data Bank," *Nucleic Acids Res.*, vol. 28, no. 1, pp. 235–242, Jan. 2000.

[46] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus, "CHARMM: A program for macromolecular energy, minimization, and dynamics calculations," *J. Comput. Chem.*, vol. 4, no. 2, pp. 187–217, 1983.

[47] B. B. Alexander D. MacKerell, "CHARMM: The Energy Function and Its Parameterization with an Overview of the Program," in *Encyclopedia of Computational Chemistry*, P. von Ragué Schleyer, N. L. Allinger, T. Clark, J. Gasteiger, P. A. Kollman, H. F. Schaefer, and P. R. Schreiner, Eds. Chichester, UK: John Wiley & Sons, Ltd, 2002, pp. 271–277.

[48] B. R. Brooks, C. L. Brooks, A. D. Mackerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V.

Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, and M. Karplus, "CHARMM: the biomolecular simulation program.," *J. Comput. Chem.*, vol. 30, no. 10, pp. 1545–614, Jul. 2009.

[49] Z. Zhou, P. Payne, M. Vasquez, N. Kuhn, and M. Levitt, "Finite-difference solution of the Poisson-Boltzmann equation: Complete elimination of self-energy," *J. Comput. Chem.*, vol. 17, no. 11, pp. 1344–1351, Aug. 1996.

[50] W. Rocchia, S. Sridharan, A. Nicholls, E. Alexov, A. Chiabrera, and B. Honig, "Rapid grid-based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects," *J. Comput. Chem.*, vol. 23, no. 1, pp. 128–137, Jan. 2002.

[51] H. Jin and R. I. Tanner, "Generation of unstructured tetrahedral meshes by advancing front technique," *Int. J. Numer. Methods Eng.*, vol. 36, no. 11, pp. 1805–1823, Jun. 1993.

[52] P. L. George and E. Seveno, "The advancing-front mesh generation method revisited," *Int. J. Numer. Methods Eng.*, vol. 37, no. 21, pp. 3605–3619, Nov. 1994.

[53] D. Xu and Y. Zhang, "Generating triangulated macromolecular surfaces by Euclidean Distance Transform.," *PLoS One*, vol. 4, no. 12, p. e8140, Jan. 2009.

[54] M. F. Sanner, A. J. Olson, and J. C. Spehner, "Reduced surface: an efficient way to compute molecular surfaces.," *Biopolymers*, vol. 38, no. 3, pp. 305–20, Mar. 1996.

[55] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Trans. Vis. Comput. Graph.*, vol. 5, no. 4, pp. 349–359, 1999.

[56] M. Ramanadham, L. C. Sieker, and L. H. Jensen, "Refinement of triclinic lysozyme: II. The method of stereochemically restrained least squares.," *Acta Crystallogr. B.*, vol. 46 (Pt 1), pp. 63–69, 1990.

[57] K. Katayanagi, M. Miyagawa, M. Matsushima, M. Ishikawa, S. Kanaya, H. Nakamura, M. Ikehara, T. Matsuzaki, and K. Morikawa, "Structural details of ribonuclease H from Escherichia coli as refined to an atomic resolution," *J. Mol. Biol.*, vol. 223, no. 4, pp. 1029–1052, Feb. 1992.

[58] P. Cignoni, M. Corsini, and G. Ranzuglia, "MeshLab: an Open-Source 3D Mesh Processing System," *Ercim news*, 2008. .

[59] H.-X. Zhou and F. Dong, "Electrostatic contributions to the stability of a thermophilic cold shock protein.," *Biophys. J.*, vol. 84, no. 4, pp. 2216–22, Apr. 2003.

[60] S. P. L. Sørensen, "Enzymstudien. II: Mitteilung. Über die Messung und die Bedeutung der Wasserstoffionenkoncentration bei enzymatischen Prozessen," *Biochem. Zeitschrift (in Ger.*, vol. 21, pp. 131–304, 1909.

[61] L. J. Henderson, "Concerning the relationship between the strength of acids and their capacity to preserve neutrality," *Am J Physiol*, vol. 21, no. 2, pp. 173–179, Mar. 1908.

[62] K. A. Hasselbalch, *Die Berechnung der Wasserstoffzahl des Blutes aus der freien und gebundenen Kohlensäure desselben, und die Sauerstoffbindung des Blutes als Funktion der Wasserstoffzahl*. Julius Springer, 1916.

[63] G. Gran, "Determination of the equivalence point in potentiometric titrations. Part II," *Analyst*, vol. 77, no. 920, pp. 661–671, 1952.

[64] W. R. Baker and A. Kintanar, "Characterization of the pH titration shifts of ribonuclease A by one- and two-dimensional nuclear magnetic resonance spectroscopy.," *Arch. Biochem. Biophys.*, vol. 327, no. 1, pp. 189–99, Mar. 1996.

[65] K. Bartik, C. Redfield, and C. M. Dobson, "Measurement of the individual pKa values of acidic residues of hen and turkey lysozymes by two-dimensional 1H NMR.," *Biophys. J.*, vol. 66, no. 4, pp. 1180–4, Apr. 1994.

[66] R. L. Thurlkill, G. R. Grimsley, J. M. Scholtz, and C. N. Pace, "pK values of the ionizable groups of proteins.," *Protein Sci.*, vol. 15, no. 5, pp. 1214–8, May 2006.

[67] E. Cohn and J. Edsall, "Proteins, amino acids and peptides," *Hafner Publishing Co., Inc.*, New York, pp. 370–381, 1943.

[68] Y. Nozaki and C. Tanford, "Examination of titration behavior," *Methods Enzymol.*, vol. 11, pp. 715–734, 1967.

[69] F. R. N. Gurd, P. Keim, V. G. Glushko, P. J. Lawson, R. C. Marshall, A. M. Nigen, R. A. Vigna, and J. Meinhofer, "Carbon-13 nuclear magnetic resonance of some pentapeptides," *Ann Arbor Sci. Publ. Ann Arbor, MI*, pp. 45–49, 1972.

[70] R. Richarz and K. Wüthrich, "Carbon-13 NMR chemical shifts of the common amino acid residues measured in aqueous solutions of the linear tetrapeptides H-Gly-Gly- X-L- Ala-OH," *Biopolymers*, vol. 17, no. 9, pp. 2133–2141, Sep. 1978.

[71] T. E. Creighton, *Proteins: Structures and Molecular Properties.*, Second Edi. New York: Freeman, 1993.

[72] G. M. Ullmann and E. W. Knapp, "Electrostatic models for computing protonation and redox equilibria in proteins.," *Eur. Biophys. J.*, vol. 28, no. 7, pp. 533–51, Jan. 1999.

[73] G. Kieseritzky and E. W. Knapp, "Charge transport in the ClC-type chloride-proton anti-porter from Escherichia coli.," *J. Biol. Chem.*, vol. 286, no. 4, pp. 2976–86, Jan. 2011.

[74] M. Schmidt am Busch and E. W. Knapp, "Accurate pKa determination for a heterogeneous group of organic molecules.," *Chemphyschem*, vol. 5, no. 10, pp. 1513–22, Oct. 2004.

[75] C. C. R. Sutton, G. V Franks, and G. da Silva, "First principles pKa calculations on carboxylic acids using the SMD solvation model: effect of thermodynamic cycle, model chemistry, and explicit solvent molecules.," *J. Phys. Chem. B*, vol. 116, no. 39, pp. 11999–2006, Oct. 2012.

[76] M. Schmidt Am Busch and E. W. Knapp, "One-electron reduction potential for oxygen- and sulfur-centered organic radicals in protic and aprotic solvents.," *J. Am. Chem. Soc.*, vol. 127, no. 45, pp. 15730–7, Nov. 2005.

[77] G. Galstyan and E. W. Knapp, "Computing pKA values of hexa-aqua transition metal complexes.," *J. Comput. Chem.*, vol. 36, no. 2, pp. 69–78, Jan. 2015.

[78] D. Bashford and M. Karplus, "pKa's of ionizable groups in proteins: atomic detail from a continuum electrostatic model," *Biochemistry*, vol. 29, no. 44, pp. 10219–10225, Nov. 1990.

[79] B. Rabenstein, G. M. Ullmann, and E. W. Knapp, "Calculation of protonation patterns in proteins with structural relaxation and molecular ensembles - application to the photosynthetic reaction center," *Eur. Biophys. J.*, vol. 27, no. 6, pp. 626–637, Sep. 1998.

[80] P. Beroza, D. R. Fredkin, M. Y. Okamura, and G. Feher, "Protonation of interacting residues in a protein by a Monte Carlo method: application to lysozyme and the photosynthetic reaction center of Rhodobacter sphaeroides.," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 88, no. 13, pp. 5804–8, Jul. 1991.

[81] B. Rabenstein and E. W. Knapp, "Calculated pH-dependent population and protonation of carbon-monoxy-myoglobin conformers.," *Biophys. J.*, vol. 80, no. 3, pp. 1141–50, Mar. 2001.

[82] H. W. van Vlijmen, M. Schaefer, and M. Karplus, “Improving the accuracy of protein pKa calculations: conformational averaging versus the average structure.,” *Proteins*, vol. 33, no. 2, pp. 145–58, Nov. 1998.

[83] J. E. Nielsen and G. Vriend, “Optimizing the hydrogen-bond network in Poisson-Boltzmann equation-based pKa calculations.,” *Proteins*, vol. 43, no. 4, pp. 403–12, Jun. 2001.

[84] J. D. Jackson, *Klassische Elektrodynamik*. Walter de Gruyter, 2006.

[85] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, “Comparing images using the Hausdorff distance,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, 1993.

[86] P. Cignoni, C. Rocchini, and R. Scopigno, “Metro: Measuring Error on Simplified Surfaces,” *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, Jun. 1998.

[87] K. D. Berndt, P. Güntert, L. P. Orbons, and K. Wüthrich, “Determination of a high-quality nuclear magnetic resonance solution structure of the bovine pancreatic trypsin inhibitor and comparison with three crystal structures.,” *J. Mol. Biol.*, vol. 227, no. 3, pp. 757–775, Oct. 1992.

[88] V. S. Reddy and G. R. Nemerow, “Structures and organization of adenovirus cement proteins provide insights into the role of capsid maturation in virus entry and infection.,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 111, no. 32, pp. 11715–20, Aug. 2014.

[89] “The PyMOL Molecular Graphics System.” Schrödinger LLC.

[90] M. K. Gilson, “Multiple-site titration and molecular modeling: two rapid methods for computing energies and forces for ionizable groups in proteins.,” *Proteins*, vol. 15, no. 3, pp. 266–82, Mar. 1993.

[91] A. S. Yang, M. R. Gunner, R. Sampogna, K. Sharp, and B. Honig, “On the calculation of pKas in proteins.,” *Proteins*, vol. 15, no. 3, pp. 252–65, Mar. 1993.

[92] J. Antosiewicz, J. A. McCammon, and M. K. Gilson, “Prediction of pH-dependent properties of proteins.,” *J. Mol. Biol.*, vol. 238, no. 3, pp. 415–36, May 1994.

[93] A. Karshikoff, “A simple algorithm for the calculation of multiple site titration curves,” *Protein Eng. Des. Sel.*, vol. 8, no. 3, pp. 243–248, Mar. 1995.

[94] E. Demchuk and R. C. Wade, "Improving the Continuum Dielectric Approach to Calculating pKas of Ionizable Groups in Proteins," *J. Phys. Chem.*, vol. 100, no. 43, pp. 17373–17387, Jan. 1996.

[95] E. G. Alexov and M. R. Gunner, "Incorporating protein conformational flexibility into the calculation of pH-dependent protein properties.," *Biophys. J.*, vol. 72, no. 5, pp. 2075–93, May 1997.

[96] X. Raquet, V. Lounnas, J. Lamotte-Brasseur, J. M. Frère, and R. C. Wade, "pKa calculations for class A beta-lactamases: methodological and mechanistic implications.," *Biophys. J.*, vol. 73, no. 5, pp. 2416–26, Nov. 1997.

[97] Y. Y. Sham, Z. T. Chu, and A. Warshel, "Consistent Calculations of pKa's of Ionizable Residues in Proteins: Semi-microscopic and Microscopic Approaches," *J. Phys. Chem. B*, vol. 101, no. 22, pp. 4458–4472, May 1997.

[98] S. T. Wlodek, J. Antosiewicz, and J. A. McCammon, "Prediction of titration properties of structures of a protein derived from molecular dynamics trajectories.," *Protein Sci.*, vol. 6, no. 2, pp. 373–82, Feb. 1997.

[99] A. Burykin and A. Warshel, "What really prevents proton transport through aquaporin? Charge self-energy versus proton wire proposals.," *Biophys. J.*, vol. 85, no. 6, pp. 3696–706, Dec. 2003.

[100] E. L. Mehler and F. Guarnieri, "A self-consistent, microenvironment modulated screened coulomb potential approximation to calculate pH-dependent electrostatic effects in proteins.," *Biophys. J.*, vol. 77, no. 1, pp. 3–22, Jul. 1999.

[101] L. Sandberg and O. Edholm, "A fast and simple method to calculate protonation states in proteins.," *Proteins*, vol. 36, no. 4, pp. 474–83, Sep. 1999.

[102] G. Kieseritzky and E. W. Knapp, "Optimizing pKa computation in proteins with pH adapted conformations.," *Proteins*, vol. 71, no. 3, pp. 1335–48, May 2008.

[103] G. Kieseritzky, "Shaping electrostatic energy computations in proteins: The ClC-type proton-chloride antiporter function," *Dissertation*, pp. 1–64, May 2011.

[104] C. L. Stanton and K. N. Houk, "Benchmarking p K a Prediction Methods for Residues in Proteins," *J. Chem. Theory Comput.*, vol. 4, no. 6, pp. 951–966, Jun. 2008.

[105] A. P. Gamiz-Hernandez, G. Kieseritzky, H. Ishikita, and E. W. Knapp, "Rubredoxin Function: Redox Behavior from Electrostatics," *J. Chem. Theory Comput.*, vol. 7, no. 3, pp. 742–752, Mar. 2011.

[106] B. Rabenstein, "Monte Carlo methods for simulation of protein folding and titration," Freie Universität Berlin, 2000.

[107] P. Cignoni and F. Ganovelli, "VCG Library." [Online]. Available: http://vcg.isti.cnr.it/vcglib/.

[108] S. Osher, R. Fedkiw, and K. Piechor, "Level Set Methods and Dynamic Implicit Surfaces," *Appl. Mech. Rev.*, vol. 57, no. 3, p. B15, 2002.

[109] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *ACM SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 163–169, Aug. 1987.

[110] T. J. Baker, "Developments and trends in three-dimensional mesh generation," *Appl. Numer. Math.*, vol. 5, no. 4, pp. 275–304, Jul. 1989.

[111] J. F. Thompson and N. P. Weatherill, "Aspects of numerical grid generation: current science and art," 1993.

[112] E. B. de L'isle and P. L. George, "Optimization of tetrahedral meshes," in *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, Springer, 1995, pp. 97–127.

[113] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[114] W. H. Frey and D. A. Field, "Mesh relaxation: A new technique for improving triangulations," *Int. J. Numer. Methods Eng.*, vol. 31, no. 6, pp. 1121–1133, May 1991.

[115] J. Schöberl, "C++11 Implementation of Finite Elements in NGSolve," ASC Report 30/2014, Institute for Analysis and Scientific Computing, Vienna University of Technology, 2014.

[116] U. M. Y. Robert D. Falgout, "hypre: a Library of High Performance Preconditioners."

[117] A. Kuzmin, M. Luisier, and O. Schenk, "Fast methods for computing selected elements of the green's function in massively parallel nanoelectronic device simulations," in *Euro-Par 2013 Parallel Processing*, Springer, 2013, pp. 533–544.

[118] C. G. Petra, O. Schenk, M. Lubin, and K. Gäertner, "An Augmented Incomplete Factorization Approach for Computing the Schur Complement in Stochastic Optimization," *SIAM J. Sci. Comput.*, vol. 36, no. 2, pp. C139–C162, Jan. 2014.

[119] Y. Shapira, "An Algebraic Multilevel Method with Applications," in *Matrix-Based Multigrid*, Springer, 2003, pp. 179–200.

[120] G. Karypis and V. Kumar, "A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering," *J. Parallel Distrib. Comput.*, vol. 48, no. 1, pp. 71–95, Jan. 1998.

[121] D. Lasalle and G. Karypis, "Multi-threaded Graph Partitioning," in *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*, 2013, pp. 225–236.

[122] D. Buytaert, G. Killesreiter, N. Drumm, G. Hojtsy, A. Byron, and N. Catchpole, "Drupal CMS." [Online]. Available: https://www.drupal.org/. [Accessed: 22-Jan-2015].

[123] J. Resig, "jQuery." [Online]. Available: http://jquery.com/. [Accessed: 22-Jan-2015].

[124] "jsc3d." [Online]. Available: https://code.google.com/p/jsc3d/. [Accessed: 22-Jan-2015].

[125] "Google Charts — Google Developers." [Online]. Available: https://developers.google.com/chart/. [Accessed: 22-Jan-2015].

[126] G. Taubin, "Curve and surface smoothing without shrinkage," in *Proceedings of 5. IEEE International Conference on Computer Vision*, 1995, pp. 852–857.

[127] D. W. Hoffman, C. Davies, S. E. Gerchman, J. H. Kycia, S. J. Porter, S. W. White, and V. Ramakrishnan, "Crystal structure of prokaryotic ribosomal protein L9: a bi-lobed RNA-binding protein.," *EMBO J.*, vol. 13, no. 1, pp. 205–12, Jan. 1994.

[128] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97*, 1997, pp. 209–216.

[129] J. Warwicker and H. C. Watson, "Calculation of the electric potential in the active site cleft due to α-helix dipoles," *J. Mol. Biol.*, vol. 157, no. 4, pp. 671–679, Jun. 1982.

[130] A. D. Bochevarov, E. Harder, T. F. Hughes, J. R. Greenwood, D. A. Braden, D. M. Philipp, D. Rinaldo, M. D. Halls, J. Zhang, and R. A. Friesner, "Jaguar: A high-performance quantum chemistry software program with strengths in life and materials sciences," *Int. J. Quantum Chem.*, vol. 113, no. 18, pp. 2110–2142, Sep. 2013.

[131] C. M. Cortis and R. A. Friesner, "An automatic three-dimensional finite element mesh generation system for the Poisson-Boltzmann equation," *J. Comput. Chem.*, vol. 18, no. 13, pp. 1570–1590, Oct. 1997.

[132] C. M. Cortis and R. A. Friesner, "Numerical solution of the Poisson-Boltzmann equation using tetrahedral finite-element meshes," *J. Comput. Chem.*, vol. 18, no. 13, pp. 1591–1608, Oct. 1997.

[133] M. Holst, J. A. McCammon, Z. Yu, Y. Zhou, and Y. Zhu, "Adaptive Finite Element Modeling Techniques for the Poisson-Boltzmann Equation," *Commun. Comput. Phys.*, vol. 11, no. 1, pp. 179–214, Jan. 2012.

[134] M. L. Connolly, "Analytical molecular surface calculation," *J. Appl. Crystallogr.*, vol. 16, no. 5, pp. 548–558, Oct. 1983.

[135] M. L. Connolly, "Molecular surface Triangulation," *J. Appl. Crystallogr.*, vol. 18, no. 6, pp. 499–505, Dec. 1985.

[136] P. J. Mohr, B. N. Taylor, and D. B. Newell, "CODATA Recommended Values of the Fundamental Physical Constants: 2010," *J. Phys. Chem. Ref. Data*, vol. 41, no. 4, p. 043109, Dec. 2012.

[137] C. Martin, V. Richard, M. Salem, R. Hartley, and Y. Mauguen, "Refinement and structural analysis of barnase at 1.5 A resolution.," *Acta Crystallogr. D. Biol. Crystallogr.*, vol. 55, no. Pt 2, pp. 386–398, Feb. 1999.

[138] M. Svensson-Ek, J. Abramson, G. Larsson, S. Törnroth, P. Brzezinski, and S. Iwata, "The X-ray crystal structures of wild-type and EQ(I-286) mutant cytochrome c oxidases from Rhodobacter sphaeroides.," *J. Mol. Biol.*, vol. 321, no. 2, pp. 329–339, Aug. 2002.

[139] S. Kuramitsu and K. Hamaguchi, "Analysis of the acid-base titration curve of hen lysozyme.," *J. Biochem.*, vol. 87, no. 4, pp. 1215–9, Apr. 1980.

List of Figures

List of Tables

List of Publications

Publications

- I. Sakalli, J. Schöberl, and E. W. Knapp, "mFES: A Robust Molecular Finite Element Solver for Electrostatic Energy Computations," *J. Chem. Theory Comput.*, vol. 10, no. 11, pp. 5095–5112, Oct. 2014. doi: 10.1021/ct5005092
- I. Sakalli, and E. W. Knapp, "pK_A in Proteins Solving the Poisson-Boltzmann Equation with Finite Elements," *submitted*

Thesis

- I. Sakalli, "Solving the Poisson-Boltzmann Equation on Massively Parallel Machines," Master thesis, FU Berlin, Oct. 2010.
- I. Sakalli, "Parallelization Strategies for a Brownian Dynamics Algorithm," Bachelor thesis, FU Berlin, Aug. 2008.

Conferences and Talks

2014 - Electrostatics Meeting,
Faculdade de Ciências da Universidade de Lisboa, *Portugal*

2013 - Collaboration meeting, TU Vienna, *Austria*

2012 - 3rd VW Status Symposium, Potsdam, *Germany*

2011 - Multiscale Simulation for Ion Channels, Berlin, *Germany*

2010 - Computer Simulation and Theory of Macromolecules 2010, Hünfeld, *Germany*

2009 - Methods in Molecular Simulation Summer School 2009, CCP5, *Sheffield*

2009 - Computer Simulation and Theory of Macromolecules 2009, Hünfeld, *Germany*

Supervised lectures and tutoria

2012-2013 - Mathematik für Chemiker I + II

2013 - Einführung in Unix

2012 - Computer simulation of biomolecules

Research funding, scholarships and prices

2014 - *SFB 1078*, Protonation Dynamics in Protein Function

2011 - Volkswagen Stiftung, New trends in simulating biological systems and soft matter

2010 - *SFB 765*, Multivalenz als chemisches Organisations- und Wirkungsprinzip

2009 - *SFB 740*, Collaborative Research Centre, Particle Dynamics of Cellular Processes

2006 - Studienstiftung des deutschen Volkes (nomination)

2005 - J. und E. Frauendorfer Förderung

2001 - German Scholastic Aptitude Test (GSAT-M)

2001 - William-Stern-Gesellschaft für besonders begabte Jugendliche

2000 - Internationaler Städtewettbewerb Mathematik Preis

1998 - Mathematik Olympiade Preis

COLOPHON

The title page shows a lysozyme. The molecular surface is triangulated with a resolution of $h_S = 0.5$ Å. The inner tetrahedral volume is drawn to show the regular tetrahedrons building up the molecular model. This image is generated using NETGEN v5.1.

This Word document was typeset using the typographical look-and-feel classicthesis of LaTeX.

Information for this document:
Font: TeXGyrePagella. *Final Version* from 17.05.2015 16:50 (version 344) has 26818 words, 3664 lines and 157 pages. Work done at geolocation: *(52.453694, 13.294565)*.

Appendix

A Units and Conversions

A.1 Units

Table 5. Expressions and units used in mFES. These values are mainly from NIST[136] or derived from NIST constants.

expression	Value	unit
ϵ_0	$8.85418782 \cdot 10^{-12}$	$\frac{As}{Vm}$
e	$1.60217656 \cdot 10^{-19}$	C
N_A	$6.0221415 \cdot 10^{23}$	$\frac{1}{mol}$
ΔG_{Born}	$694.6773088 \cdot \frac{z^2}{r} \cdot \left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}} \right)$	$\frac{kJ}{mol}$
κ^2	8.43249149	$\frac{I}{\epsilon} \cdot \frac{1}{\mathring{A}^2}$
k_b	$1.3806488 \cdot 10^{-23}$	$\frac{J}{K}$
T	300	K
F	96485.3365	$\frac{C}{mol}$

A.2 Conversions

Table 6. Useful Conversions

Conversion
$1J = \frac{1V}{1C}$
$1C = 1As$
$1\frac{mol}{l} = 10^3 \frac{mol}{m^3} = \frac{mol}{dm^3}$

B Born Model Derivation

Born model with charge q, in centre of a sphere with radius a. The dielectric constant in the sphere is ϵ_{in}, outside of the sphere the dielectric constant is ϵ_{out}. Outside of the larger sphere with radius b the ionic strength is κ^2 (Fig 32).

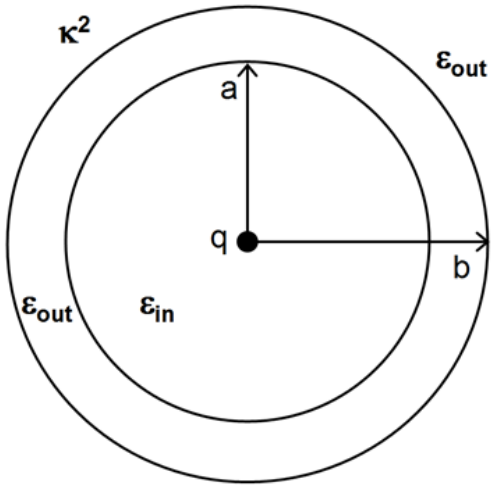


Fig 32. Born ion model. Sketch is used with permission from EW Knapp.

Close to the charge q the electrostatic potential ϕ is

$$\phi_1(r) = \frac{q}{\epsilon_{in} \cdot r} + c_1 . \tag{80}$$

Very far from the charge q, outside of both spheres, the electrostatic potential fulfils the Poisson-Boltzmann equation

$$\epsilon_{out} \vec{\nabla}^2 \phi(r) = \kappa^2 \phi(r) . \tag{81}$$

The radial part of the Laplace operator Δ is

$$\Delta_{radial} = \frac{1}{r^2} \frac{\partial}{\partial r} r^2 \frac{\partial}{\partial r} = \frac{2}{r} \frac{\partial}{\partial r} + \frac{\partial^2}{\partial r^2} .$$

Hence, a solution has to be found to

$$\left(\frac{2}{r}\frac{\partial}{\partial r}+\frac{\partial^2}{\partial r^2}\right)\phi(r)=\alpha^2\phi(r) \text{ with } \alpha^2=\kappa^2/\epsilon_{out}.$$

Two special solutions are

$$\phi_{\pm}(r)=c_3\frac{1}{r}e^{\pm\alpha r},$$

where the solution with the minus sign possesses the proper asymptotic.

To demonstrate that this is the proper solution, it is noted

$$\frac{\partial^2}{\partial r^2}\left(\frac{1}{r}e^{-\alpha r}\right)=-\frac{\partial}{\partial r}\left[\left(\frac{1}{r^2}+\frac{\alpha}{r}\cdot e^{-\alpha r}\right)\right]=\left(\frac{2}{r^3}+\frac{2\alpha}{r^2}+\frac{\alpha^2}{r}\right)\cdot e^{-\alpha\cdot r}$$

such that

$$\frac{\partial^2}{\partial r^2}\phi_{-}(r)=-\frac{\partial}{\partial r}\left[\left(\frac{1}{r}+\alpha\right)\phi_{-}(r)\right]=\left(\frac{2}{r^2}+\frac{2\alpha}{r}+\alpha^2\right)\phi_{-}(r)$$

and

$$\frac{2}{r}\frac{\partial}{\partial r}\phi_{-}(r)=-\left(\frac{2}{r^2}+\frac{2\alpha}{r}\cdot\phi_{-}(r)\right).$$

The general asymptotic solution reads

$$\phi_3(r)=c_3+\frac{1}{\epsilon_{out}\cdot r}e^{-r\frac{\kappa}{\sqrt{\epsilon_{out}}}}.$$

In the intermediate regime between the two spheres the electrostatic potential is

$$\phi_2(r)=\frac{q}{\epsilon_{out}\cdot r}+c_2.$$

At the two sphere boundaries the electrostatic potential and the radial component of the electrical displacement $\vec{D}=\epsilon\vec{E}$ must be continuous.

Hence, at the inner sphere radius $r=a$ there is

$$\epsilon_{in}=\frac{\partial}{\partial r}\phi_1(a)=\epsilon_{out}\frac{\partial}{\partial r}\phi_2(a).$$

This is fulfilled by $\phi_1(r)$ and $\phi_2(b)$, eq. (80) and eq. (81), respectively. At the outer sphere radius $r = b$ and there is

$$\epsilon_{out} \frac{\partial}{\partial r} \phi_2(b) = \epsilon_{out} \frac{\partial}{\partial r} \phi_3(\mathrm{b}),$$

yielding

$$-\frac{1}{b^2} c_3 \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right) e^{-b\frac{\kappa}{\sqrt{\epsilon_{out}}}} = -\frac{q}{b^2}$$

$$\phi_3(r) = \frac{q}{\epsilon_{out} r} e^{-r\frac{\kappa}{\sqrt{\epsilon_{out}}}} \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} e^{b\frac{\kappa}{\sqrt{\epsilon_{out}}}}.$$

Requesting that the electrostatic potential is continuous at the outer sphere $\phi_3(b) = \phi_2(b)$ yields

$$\frac{q}{\epsilon_{out} b} \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} = \frac{q}{\epsilon_{out} b} + c_2$$

and finally

$$c_2 = \frac{q}{\epsilon_{out} b} \left[\left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} - 1 \right] = -\frac{q}{\epsilon_{out} b} \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} \frac{b\kappa}{\sqrt{\epsilon_{out}}} = -\frac{q}{\epsilon_{out}} \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} \frac{\kappa}{\sqrt{\epsilon_{out}}},$$

such that

$$\phi_2(r) = \frac{q}{\epsilon_{out} r} \left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} \frac{r\kappa}{\sqrt{\epsilon_{out}}} \right].$$

To determine $\phi_1(r)$ the continuity condition at the inner sphere radius $r = a$, i.e. $\phi_1(a) = \phi_2(a)$ are used yielding

$$\frac{q}{\epsilon_{in} a} + c_1 = \frac{q}{\epsilon_{out} a} \left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}} \right)^{-1} \frac{\alpha\kappa}{\sqrt{\epsilon_{out}}} \right]$$

and

$$c_1 = \frac{q}{a}\left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}}\right) - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_o ut}}\right)^{-1} \frac{q}{\epsilon_{out}} \frac{\kappa}{\sqrt{\epsilon_{out}}}$$

such that

$$\phi_1(r) = \frac{q}{\epsilon_{in} r}\left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}}\right)^{-1} \frac{\epsilon_{in}}{\epsilon_{out}} \frac{r\kappa}{\sqrt{\epsilon_{out}}}\right] + \frac{q}{a}\left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}}\right).$$

Born Energy Formula

The self-energy of a point charge in the centre of a sphere as defined above is computed by loading the surface of the self-energy:

$$W_{self} = \int_0^q \phi_2(a)\mathrm{d}\hat{q} = \int_0^q \frac{\hat{q}}{\epsilon_{out} a}\left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}}\right)^{-1} \frac{a\kappa}{\sqrt{\epsilon_{out}}}\right]\mathrm{d}\hat{q} = \frac{q^2}{2 \cdot \epsilon_{out} a}\left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}}\right)^{-1} \frac{a\kappa}{\sqrt{\epsilon_{out}}}\right].$$

For the solvation energy the result is

$$W_{solv} = \frac{q^2}{2 \cdot \epsilon_{out} a}\left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{out}}}\right)^{-1} \frac{a\kappa}{\sqrt{\epsilon_{out}}}\right] - \frac{q^2}{2 \cdot \epsilon_{in} a}\left[1 - \left(1 + \frac{b\kappa}{\sqrt{\epsilon_{in}}}\right)^{-1} \frac{a\kappa}{\sqrt{\epsilon_{in}}}\right].$$

Example Unit Conversion

This is an example on how to obtain the analytical ΔG_{Born} term to calculate the energy for one atom in a dielectric and inhomogeneous medium with two different dielectric constants.

$$\Delta G_{Born} = \frac{1}{8\pi \cdot \epsilon_0} \cdot \frac{Q^2}{r} \cdot \left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}}\right) = \frac{10^{12}}{8\pi \cdot 8.85418782} \cdot \frac{(z \cdot 1.60217656 \cdot 10^{-19})^2}{r \cdot 10^{-10}} \cdot \left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}}\right)\left[\frac{Vm \cdot C^2}{As \cdot m}\right]$$

$$= \frac{z^2}{r} \cdot \frac{10^{12}}{8\pi \cdot 8.85418782} \cdot \frac{(1.60217656 \cdot 10^{-19})^2}{10^{-10}} \cdot \left(\frac{1}{\epsilon_{out}} - \frac{1}{\epsilon_{in}}\right)[J]$$

$$\Rightarrow \frac{z^2}{r}\frac{6.0221415\cdot 10^{23}}{1000}\cdot\frac{10^{12}}{8\pi\cdot 8.85418782}\cdot\frac{(1.60217656\cdot 10^{-19})^2}{10^{-10}}\cdot\left(\frac{1}{\epsilon_{out}}-\frac{1}{\epsilon_{in}}\right)\left[\frac{kJ}{mol}\right]$$

$$= \frac{z^2}{r}\cdot 694.6773088\cdot\left(\frac{1}{\epsilon_{out}}-\frac{1}{\epsilon_{in}}\right)\left[\frac{kJ}{mol}\right].$$

Example with Vanishing Ion Concentration

$r = 3\text{Å}, Q = 1C, \epsilon_{in} = 4, \epsilon_{out} = 80$.

$$\Delta G_{Born} = \frac{1^2}{3}\cdot 694.6773088\cdot\left(\frac{1}{80}-\frac{1}{4}\right)\left[\frac{kJ}{mol}\right]$$

$$= -54.995287\left[\frac{kJ}{mol}\right].$$

Calculation of κ^2

κ^2 is defined in eq. (18):

$$\kappa^2 = \frac{2\cdot N_A\cdot e^2\cdot I}{\epsilon_0\epsilon_r\cdot k_B\cdot T}.$$

Including known constants, it is obtained

$$\kappa^2 = \frac{2\cdot 6.0221415\cdot 10^{23}\cdot(1.60217656\cdot 10^{-19})^2\cdot I}{\epsilon_0\epsilon_r\cdot 1.3806488\cdot 10^{-23}\cdot 300}\left[\frac{\frac{1}{mol}C^2\frac{mol}{dm^3}}{\frac{As}{Vm}\cdot\frac{m^2\cdot kg}{s^2\cdot K}K}\right].$$

$$= 8.432\cdot 10^{17}\frac{I}{\epsilon_r}\left[\frac{C^2\frac{1}{(0.1m)^3}}{\frac{As}{Vm}\cdot\frac{m^2\cdot kg}{s^2}}\right]$$

$$= 8.432\cdot 10^{17}\frac{I}{\epsilon_r}\cdot 1000\left[\frac{C^2\frac{1}{m^3}}{\frac{As}{Vm}\cdot\frac{m^2\cdot kg}{s^2}}\right] = 8.432\cdot 10^{20}\frac{I}{\epsilon_r}\left[\frac{C^2\cdot Vm\cdot s^2}{As\cdot kg\cdot m^5}\right].$$

$$= 8.432\cdot 10^{20}\frac{I}{\epsilon_r}\left[\frac{C^2\cdot\frac{J}{C}\cdot m\cdot s^2}{C\cdot kg\cdot m^5}\right] = 8.432\cdot 10^{20}\frac{I}{\epsilon_r}\left[\frac{J\cdot s^2}{kg\cdot m^4}\right]$$

$$= 8.432\cdot 10^{20}\frac{I}{\epsilon_r}\left[\frac{1}{m^2}\right] = 8.432\cdot 10^{20}\frac{I}{\epsilon_r}\left[\frac{1}{(10^{10}\text{Å})^2}\right] = 8.432\cdot\frac{I}{\epsilon_r}\left[\frac{1}{\text{Å}^2}\right].$$

C Supporting Information to Chapter II

Table of contents

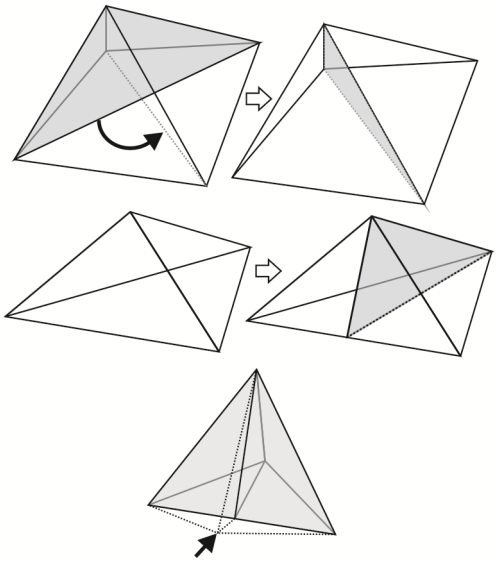


Figure S1. Optimization of tetrahedrons with NETGEN [1]. **top**: tetrahedron face swap: The separating wall between two adjacent tetrahedrons is swapped, which requires that one triangle from each tetrahedron must be in the same plane. If two triangles are only nearly in the same plane, the corresponding nodes are shifted slightly to establish planarity before applying the face swap. **middle**: tetrahedron split: A tetrahedron with a long edge is split in two by a plane which cuts the long edge and contains the two nodes opposite to this edge. **bottom**: tetrahedron collapse: If two triangles have a short common edge, the tetrahedrons built on top of such slim triangles can collapse to triangles by merging the two corner points of the short common edge. As a result one grid point and two tetrahedrons are eliminated.

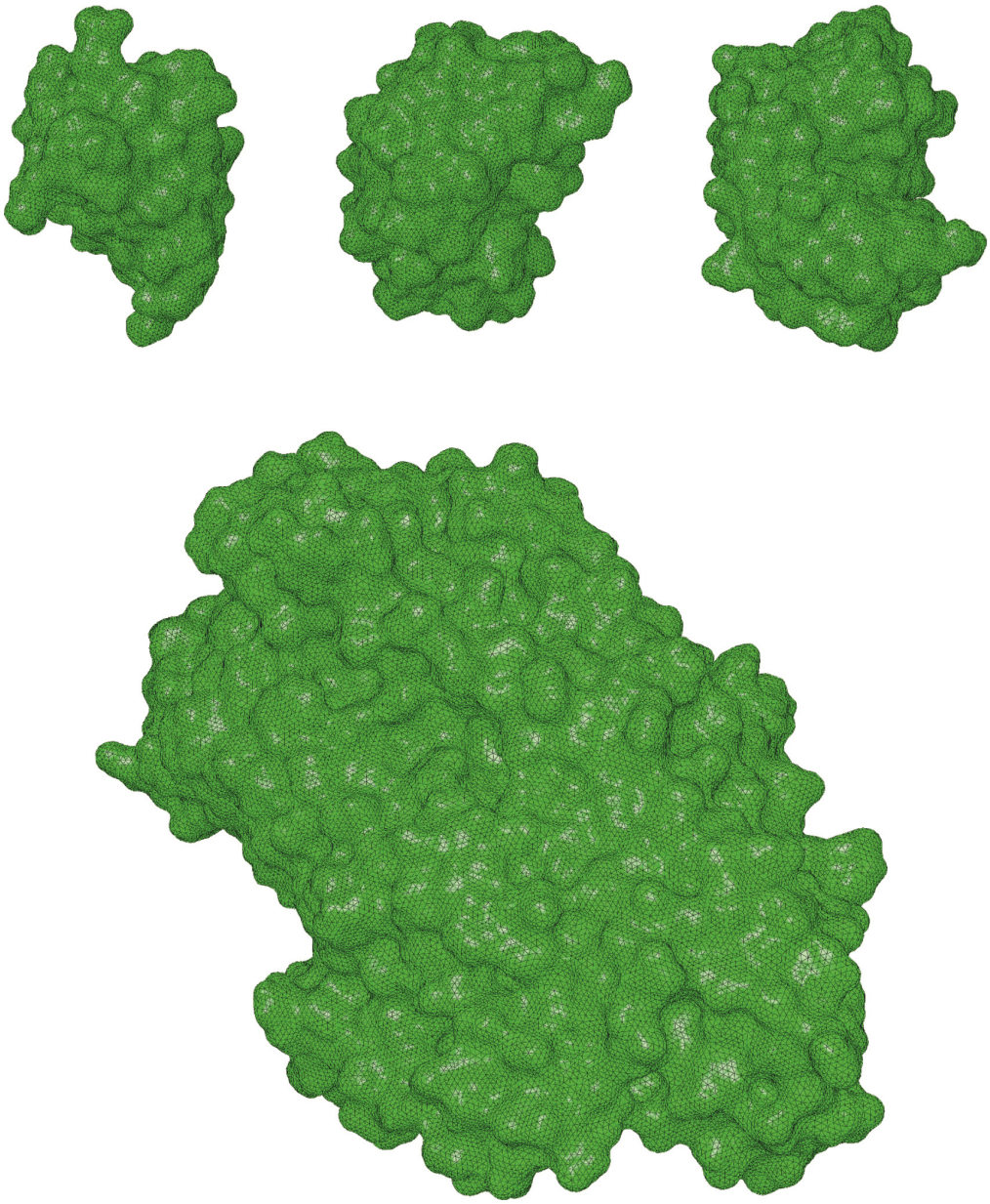


Figure S2. Four proteins used for the computation of solvation energies. **top, left to right**: bovine pancreatic trypsin inhibitor [87] (bpti), barnase [137], lysozyme [56]. **bottom**: cytochrome c oxidase [138].

Table S1. NIST constants and expressions used in calculations

expression / constant	values	units
$\boldsymbol{\varepsilon_0}$	8.85418782×10^{-12}	$\frac{s^4 \cdot A^2}{m^3 \cdot kg}$
$\mathbf{e_0}$	$1.60217656\times 10^{-19}$	C
$\mathbf{N_A}$	6.0221415×10^{23}	mol^{-1}
ΔG_{Born}	-164.98586	$\frac{z^2}{r} \cdot \frac{kJ}{mol}$
$\boldsymbol{\kappa^2}$	8.43249149	$\frac{I}{\varepsilon_r} \cdot \frac{1}{\mathring{A}^2}$

Table S2. ΔG_{Born} electrostatic solvation energy of a unit charge in center of sphere of radius $r_{Born} = 3$ Å, $\varepsilon_{in} = 4$, $\varepsilon_{out} = 80$. Comparison of APBS and mFES solver with varying ionic strength.

I [mol/l]	APBS fine[a,b]	APBS coarse[a,c]	mFES[d]	analytical result
0.01	-40.6542	-40.9927	-40.7276	-40.6188
0.02	-37.7509	-38.0969	-37.8258	-37.7153
0.05	-33.9976	-34.3215	-34.0675	-33.9565
0.1	-31.4309	-31.7182	-31.4927	-31.3823
0.15	-30.0910	-30.3521	-30.1470	-30.0374
0.2	-29.2216	-29.4631	-29.2734	-29.1643

[a] The point density at the atomic vdW spheres is set to 10 points/Å^2, which is the recommended value in APBS.

[b] $n^3 = 193^3 = 7.2\ 10^6$ grid points with 0.05 Å lattice constant

[c] $n^3 = 65^3 = 2.7\ 10^5$ grid points with 0.25 Å lattice constant

[d] Second-order approximation is used corresponding to an average distance between neighbor grid points of 0.175 Å inside the Born ion sphere resulting in a total of 34,335 grid points, which is 1/8 of the number grid points used for the coarse resolution with FD. The spherical asymptotic boundary surface is at a distance of 10^5 Å from the center.

CPU time ratio of solving linear equation systems

Solving the linear equation system is the computationally most expensive part in FD methods. Hence, CPU times for solving the linear equations for four different proteins are shown as a ratio between APBS and mFES (Fig. S3). Here, CPU times for preparing the linear equation system like generating the tetrahedral grid of the molecular model are not included. mFES reduces the CPU time to solve the linear equation system by at least one order of magnitude because the number of equations is significantly smaller with the FE method. mFES uses the linear equation solver MUMPS [42]–[44] (*M*ultifrontal *M*assively *P*arallel sparse direct *S*olver).

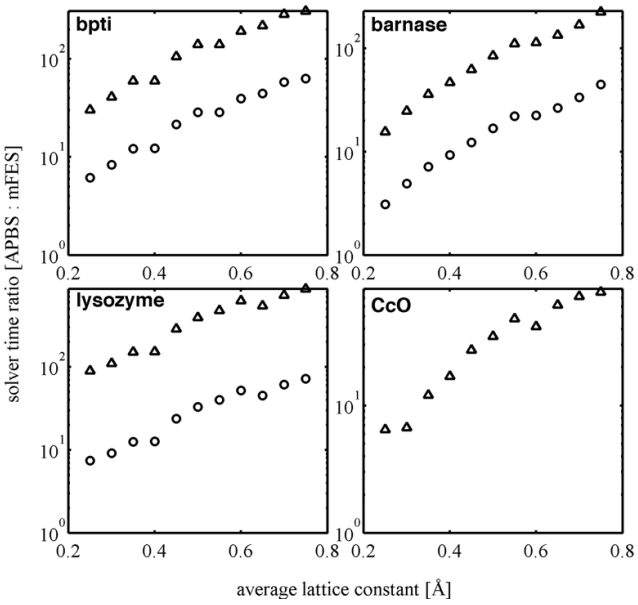


Figure S3. CPU time ratios solving linear equation systems for four proteins. Solver time ratio of APBS fine to mFES (△) and APBS coarse to mFES (○) are plotted versus the average edge length h_S on the molecular surface using mFES. Calculations are done with two APBS models (fine and coarse) for every molecule and one model for each average surface edge length generated with mFES. The ratio between APBS to mFES is increasing from lower to higher lattice constant because the molecular models computed by mFES are getting coarser without losing much accuracy in electrostatic calculations compared to FD method.

D Supporting Information to Chapter III

Table S1. Intrinsic pK_A values are compared using FD fine (resolution is 0.125 Å) and FE coarse (h_S = 0.5 Å) and fine (h_S = 0.3 Å) method using a second-order solution. Titratable groups are listed where experimental values are available. FE method uses an ion exclusion layer with a probe sphere which is rolling over inflated vdW surface. This probe sphere has a radius of 2 Å. Details about FD fine method is given in table 3.

group	SAS [Å²]	FD fine [pK]	FE coarse [pK]	FE fine [pK]
Cter129	341	3.9	4.0	4.0
Asp18	292	-1.1	-1.1	-1.0
Asp48	294	0.3	0.1	0.2
Asp52	299	1.0	0.7	0.9
Asp66	291	0.0	0.7	-0.6
Asp87	291	0.7	0.7	0.6
Asp101	294	5.7	5.7	5.7
Asp119	289	4.4	4.5	4.4
Glu7	321	2.5	2.4	2.3
Glu35	327	4.5	4.3	4.3
His15	313	-6.8	-6.5	-6.6
Lys1	367	-9.1	-8.9	-9.1
Lys13	353	-9.8	-9.7	-9.7
Lys33	348	-10.8	-10.9	-10.6
Lys96	357	-10.3	-10.2	-10.2
Lys97	343	-10.2	-10.3	-10.1
Lys116	360	-8.9	-8.9	-9.0
Tyr20	387	3.9	4.0	4.0
Tyr23	383	-1.1	-1.1	-1.0
Tyr53	388	0.3	0.1	0.2

Table S2. Comparison between two-cycle and implicit two-cycle approach of mFES is performed using a different order of solution for lysozyme (PDB id 2lzt [56]). Tyr53 is excluded from RMSD because it does not titrate under physiological conditions using the crystal structure. Lys1 has an unusual model pK_A due to its position at the N-terminus. FE method uses an ion exclusion layer with a probe sphere which is rolling over inflated vdW surface. This probe sphere has a radius of 2 Å. An average edge length of h_S = 0.5 Å is utilized to mesh the titratable groups.

group	**experimental**[65], [139]	**FD fine (no cavity)**	**FE coarse (order 2, two-cycle)**	**FE coarse (order 2, implicit two-cycle)**	**FE coarse (order 1, implicit two-cycle)**
Cter129	2.75	3.0	3.1	3.1	2.9
Asp18	2.66	1.9	1.8	1.6	1.7
Asp48	1.2	-0.7	-0.8	-1.3	-0.5
Asp52	3.68	1.7	1.2	2.9	3.4
Asp66	0.4	-2.5	-2.8	-3.2	-2.2
Asp87	2.07	-1.2	-1.3	0.1	0.5
Asp101	4.08	4.4	4.4	4.5	4.5
Asp119	3.2	2.7	2.7	2.8	2.8
Glu7	3.15	3.1	2.9	4.4	4.8
Glu35	6.2	4.4	4.3	3.9	4.2
His15	5.36	5.7	5.3	5.7	5.1
Lys1	10.8	9.3	9.1	18.5	17.86
Lys13	10.5	10.6	10.5	10.1	9.6
Lys33	10.36	10.7	10.8	10.1	8.9
Lys96	10.8	10.7	10.4	12.1	11.8
Lys97	10.3	10.9	11.0	11.2	10.5
Lys116	10.2	8.7	8.7	8.9	8.0
Tyr20	10.3	15.8	15.5	15.2	15.2
Tyr23	9.8	10.8	10.9	10.6	10.3

Table S3. Atomic partial charges used for pK_A computations utilizing CHARMM naming scheme. Charges are given in Coulomb [*C*].

histidine			**δ-histidine**	**ε-histidine**
atom	***charmm***	***protonated***	***deprotonated***	***deprotonated (2)***
C-β	CB	-0.05	-0.09	-0.08
C-δ_2	CD2	0.19	0.22	-0.05
H-δ_2	HD2	0.13	0.10	0.09
C-γ	CG	0.19	-0.05	0.22
N-ε_2	NE2	-0.51	-0.70	-0.36
H-ε_2	HE2	0.44	0.00	0.32
N-δ_1	ND1	-0.51	-0.36	-0.70
H-δ_1	HD1	0.44	0.32	0.00
C-ε_1	CE1	0.32	0.25	0.25
H-ε_1	HE1	0.18	0.13	0.13

aspartic acid				
atom	***charmm***	***Protonated***	***deprotonated***	***protonated (2)***
C-β	CB	-0.21	-0.28	-0.21
C-γ	CG	0.75	0.62	0.75
O-δ_1	OD1	-0.61	-0.76	-0.55
O-δ_2	OD2	-0.55	-0.76	-0.61
H-δ_1	HD1	0.44	0.00	0.00
H-δ_2	HD2	0.00	0.00	0.44

glutamic acid				
atom	***charmm***	***Protonated***	***deprotonated***	***protonated (2)***
C-γ	CG	-0.21	-0.28	-0.21
C-δ	CD	0.75	0.62	0.75
O-ε_1	OE1	-0.61	-0.76	-0.55
O-ε_2	OE2	-0.55	-0.76	-0.61
H-ε_1	HE1	0.44	0.00	0.00
H-ε_2	HE2	0.00	0.00	0.44

Cysteine			
atom	***charmm***	***protonated***	***deprotonated***
C-β	CB	-0.11	-0.25
S-γ	SG	-0.23	-0.93
H-γ_1	HG1	0.16	0.00

Tyrosine			
atom	***charmm***	***protonated***	***deprotonated***
C-γ	CG	0.000	-0.341
C-δ_1	CD1	-0.115	0.028
H-δ_1	HD1	0.115	0.072
C-ε_1	CE1	-0.115	-0.525
H-ε_1	HE1	0.115	0.124
C-ζ	CZ	0.110	0.769
O-η	OH	-0.540	-0.826
H-η	HH	0.43	0.000
C-δ_2	CD2	-0.115	0.028
H-δ_2	HD2	0.115	0.072
C-ε_2	CE2	-0.115	-0.525
H-ε_2	HE2	0.115	0.124

Arginine			
atom	***charmm***	***protonated***	***deprotonated***
N-ε	NE	-0.70	-0.81
H-ε	HE	0.44	0.44
C-ζ	CZ	0.64	0.71
N-η_1	NH1	-0.80	-0.90
H-η_{11}	HH11	0.46	0.27
H-η_{12}	HH12	0.46	0.27
N-η_2	NH2	-0.80	-0.90
N-η_{21}	HH21	0.46	0.27
N-η_{22}	HH22	0.46	0.27

Lysine			
atom	***charmm***	***Protonated***	***deprotonated***
N-ζ	NZ	-0.30	-0.97
H-ζ_1	HZ1	0.33	0.22
H-ζ_2	HZ2	0.33	0.22
H-ζ_3	HZ3	0.33	0.22

c-terminus			
atom	***charmm***	***Protonated***	***deprotonated***
C	C	0.34	0.34
OT1	OT1	-0.17	-0.67
OT2	OT2	-0.17	-0.67

n-terminus			
atom	***charmm***	***Protonated***	***deprotonated***
N	N	-0.300	-0.970
HT1	HT1	0.330	0.220
HT2	HT2	0.330	0.220
HT3	HT3	0.330	0.220