ESES: Software for Eulerian Solvent Excluded Surface

Beibei Liu,[a] Bao Wang,[b] Rundong Zhao,[a] Yiying Tong,*[a] and Guo-Wei Wei*[b,c,d]

Solvent excluded surface (SES) is one of the most popular surface definitions in biophysics and molecular biology. In addition to its usage in biomolecular visualization, it has been widely used in implicit solvent models, in which SES is usually immersed in a Cartesian mesh. Therefore, it is important to construct SESs in the Eulerian representation for biophysical modeling and computation. This work describes a software package called Eulerian solvent excluded surface (ESES) for the generation of accurate SESs in Cartesian grids. ESES offers the description of the solvent and solute domains by specifying all the intersection points between the SES and the Cartesian grid lines. Additionally, the interface normal at each intersection point is evaluated. Furthermore, for a given biomolecule, the ESES software not only provides the whole surface area, but also partitions the surface area according to atomic types. Homology theory is utilized to detect topological features, such as loops and cavities, on the complex formed by the SES. The sizes of loops and cavities are measured based on persistent homology with an evolutionary partial differential equation-based filtration. ESES is extensively validated by surface visualization, electrostatic solvation free energy computation, surface area and volume calculations, and loop and cavity detection and their size estimation. We used the Amber PBSA test set in our electrostatic solvation energy, area, and volume validations. Our results are either calibrated by analytical values or compared with those from the MSMS software. © 2017 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.24682

Introduction

Biophysical modeling, analysis and computation hold the key to the understanding of the self-organizing biomolecular world. However, the emergent complexity of self-organizing biomolecular systems poses a grand challenge to their quantitative description, modeling and analysis. Indeed, a detailed description in terms of electrons or atoms is computationally intractable even for a single macromolecule in most cases, not to mention an organelle or a cell. Therefore, multiscale approaches, such as discrete-continuum and quantum mechanics coupled molecular mechanics (QM/MM), are indispensable in quantitative biology. Technically, interfaces arise naturally to separate the descriptions at different scales. At the molecular scale, such an interface is a molecular boundary which describes the molecular shape. This is particularly true for the implicit solvent models,[1–6] including the Poisson–Boltzmann (PB) models,[2,3,7,8] generalized Born models,[9–13] and polarizable continuum models[14–16] for electrostatic potentials. These models admit a static atomistic representation of the biomolecules or solutes while adopting a continuum dielectric description of the solvent. In particular, the ionic contribution to the electrostatic potential is treated as a mean field approximation, namely, the Boltzmann distribution of the ionic concentration. Implicit solvent models are very popular because 65–90% of cell mass is water, whose explicit description is computationally very expensive and physically unnecessary in many situations, where the experimental measurement is performed on the solute. The solute charges can be assigned from existing force fields or obtained from quantum mechanical calculations coupled to the electrostatic potential. Similarly, solvent–solute interface is indispensable in the Poisson–Nernst–Planck (PNP) model,[17–22] in which the ionic dynamics is described by the continuum Nernst–Planck equation. Therefore, the PNP model is able to deal with the charge transport in a nonequilibrium setting. The next generation of multiscale models improves the static solute atom approximation in implicit solvent models and describes the dynamics of the solute molecule by molecular mechanics.[23–25]

An important issue is thus how to define solvent–solute interface, molecular surface, or molecular shape. The shape of a molecule is crucial to the conceptualization of invisible biomolecular systems. The visualization of the biomolecular world, which consists of proteins, DNAs, RNAs, viruses, molecular motors, and subcellular organelles, is of paramount importance in the understanding of biomolecular structure, function, dynamics, and transport.[26–28] Molecular visualization requires a definition of molecular surfaces.

Therefore, various definitions of molecular surfaces or solvent–solute interfaces have found widespread applications in

[a] B. Liu, R. Zhao, Y. Tong
Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan 48824
E-mail: ytong@msu.edu E-mail: wei@math.msu.edu
[b] B. Wang, G.-W. Wei
Department of Mathematics, Michigan State University, East Lansing, Michigan 48824
[c] G.-W. Wei
Department of Electrical and Computer Engineering, Michigan State University, East Lansing, Michigan 48824
[d] G.-W. Wei
Department of Biochemistry and, Molecular Biology, Michigan State University, East Lansing, Michigan 48824

This work is dedicated to Charles L. Brooks III on the occasion of his 60th birthday.

Contract grant sponsor: NSF; Contract grant number: IIS-1302285, IIS-0953096, and DMS-1160352; Contract grant sponsor: NIH; Contract grant number: R01GM-090208; Contract grant sponsor: MSU Center for Mathematical Molecular Biosciences Initiative

molecular modeling and visualization. In 1953, Corey and Pauling introduced the atom and bond model of molecules, which is still playing a crucial role in physical sciences. The van der Waals surface, which is the union of exposed atomic surfaces, is a simple definition that can be used for both molecular modeling and visualization. However, van der Waals surface gives rise to numerous geometric singularities on the surface and thus causes numerical instability in molecular modeling. As a remedy to this issue, solvent excluded surface (SES) was introduced by Lee and Richards.[29,30] Connolly formulates the mathematical representation of the SES for arbitrary biomolecules in terms of convex patches, saddle patches, and concave patches.[31,32] Practical construction and visualization of SESs are provided by many researchers.[33,36] MSMS is a popular SES software package,[34] in which the surface is eventually tessellated into a triangle mesh, also known as a Lagrangian representation, which is convenient for the purpose of visualization assisted by modern graphics hardware. To utilize their surfaces for the implicit solvent models where the electrostatic potential is defined on the whole computational domain, it is often necessary to carry out a conversion from the Lagrangian representation to the so-called Eulerian representation, see Fig. 1. Rocchia et al. construct approximated SESs in Cartesian grids, producing an Eulerian representation, which facilitates the subsequent implicit solvent calculations using finite difference (FD) approaches.[37] For instance, FD methods for Poisson–Boltzmann (PB) equation and Poisson–Nernst–Planck (PNP) equations rely on an Eulerian representation of the solvent–solute interface. Currently, SES is still the gold standard for implicit solvent models in computational biophysics for various applications and methodological development.

However, there are known problems associated with the use of solvent excluded surface. One of these problems is the fact that they admit geometric singularities where intersecting surfaces underlying the patches lead to tips and cusps and eventually to computational difficulty.[32,34,35] To overcome this difficulty, many alternative surface definitions have been proposed in the literature. Various Gaussian surfaces[38–41] have been employed to remove the geometric singularities, which leads to various software packages.[42] The parametrization of Gaussian surfaces, including surface extraction from the volumetric data, has been discussed.[42] Skinning surface is another computational approach to generate smooth surfaces.[43] Based on radial basis functions, the flexibility-rigidity index (FRI) method offers a family of rigidity surfaces or FRI surfaces, which include the Gaussian surfaces as a special case.[44,45] A comparison of a few surface definitions was discussed in Ref. [46].

Another problem associated with the SES and many other surface definitions is the presence of sharp features. Since biomolecules consist of atoms, and atoms are comprised of tiny nucleuses and orbiting electrons, technically, there should not have been such sharp features within a molecule or between solvent and solute molecules, regardless of their chemical or physical properties. Therefore, it is often desirable to use a fuzzy characteristic function, or hypersurface function to identify the solvent–solute interface.[47] Physically, such an approach allows the overlap of the solvent and solute domains. Mathematically, such a representation embeds the two-dimensional (2D) surface into a three-dimensional (3D) manifold and avoids geometric singularities. One of the first batch of such approaches for biomolecules was introduced in 2005, generated by curvature driven geometric partial differential equations (PDEs).[47] Later, the first variation-based molecular surface was proposed.[48–50] This approach is based on differential geometry of surfaces. The variation of the surface free energy leads to a minimal surface expression, which can be computed for biomolecules by the mean curvature flow or Laplace–Beltrami flow. The generalized Laplace–Beltrami flow can further incorporate a potential driving term in the surface formation.[51]

Yet another problem associated with the SES and similar surface definitions is that the solvent–solute interface, once prescribed, is static in all applications. However, in the solvation process, there are strong solvent–solute interactions, such as solvent polarization, induced dipolar, ionic rearrangement, solute response to solvent polarization and ion distribution, hydrophobicity, and so forth. These interactions lead to solvent–solute coupling and solvent–solute interface reconstruction. Such a surface reconstruction is not accounted for by classic SES or Gaussian surfaces. Dzubiella et al. proposed the first theory to couple nonpolar and polar solvation free energies in implicit solvent models.[52] Wei introduced a family of differential geometry-based multiscale models which facilitate solvent–solute coupling and interface reconstruction.[53] In Wei's models, nonpolar and polar energies are complemented by entropic effect of mixing, molecular mechanics, fluid mechanics and elastic energies. Some of these models have been validated in the past few years.[54–61]

Implicit solvent models and related multiscale models are mainly validated by physical measurements, such as solvation free energies for polar and nonpolar molecules, current and voltage (I–V) curves of electrophysiology. However, these models typically adopt certain parameters, such as dielectric constants, and atomic radii and charges. Therefore, researchers often look into physical pictures and properties in the original implicit solvent models, such as the SES, as a reference for new development.[42] Additionally, new numerical methods have been developed[62–65] and will be continuously proposed for solving the PB equation. Such a development in the community also requires robust and accurate SES software packages. Furthermore, there are still some deficiencies in the current public software. Although the MSMS software is available for applications, its triangulation quality is limited. Additionally, although it offers relatively good approximation to the SES with a sufficiently dense mesh, this output is still not guaranteed to resemble the analytical one. Furthermore, MSMS surfaces cannot be generated successfully at some given densities for some molecules in the benchmark test. Finally, as mentioned earlier, it is more desirable to generate the SES directly in the Cartesian grid for implicit solvent applications. As such, there is a pressing need to develop a robust software for analytical Eulerian representation of SESs.

The objective of the present work is to develop a software package, called Eulerian solvent excluded surface (ESES), for the construction of SESs in the Cartesian representation. The proposed ESES robustly generates meshes readily available for being used in solving the PB or PNP type of equations in the Cartesian grid. The

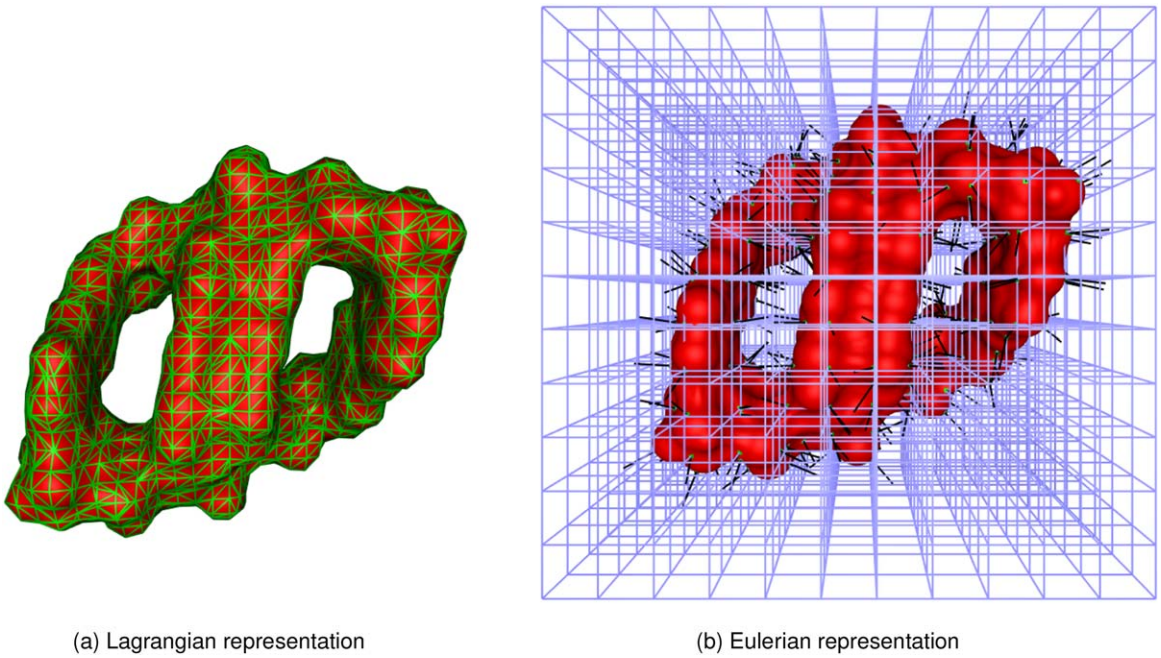

Figure 1. Lagrangian and Eulerian representations of molecular surface. Lagrangian representation is a triangular tessellation of the molecular surface which can rotate and translate with the molecule, while Eulerian representation is a description of the molecular surface on a spatially fixed Cartesian grids, typically denoted by the interface locations on the regular grid lines along with the associated surface normal directions. [Color figure can be viewed at wileyonlinelibrary.com]

surface information in ESES is provided in terms of intersection points between the grid lines and the interface, that is, the data for the coordinates of each intersection point and the surface normal at the point. Additionally, each grid point can be labeled as either inside or outside the interface. This information is useful for initializing the PB equation and for the enforcement of interface conditions. The analytical nature of intersection point evaluation of the present ESES enables high order PB and PNP solvers to

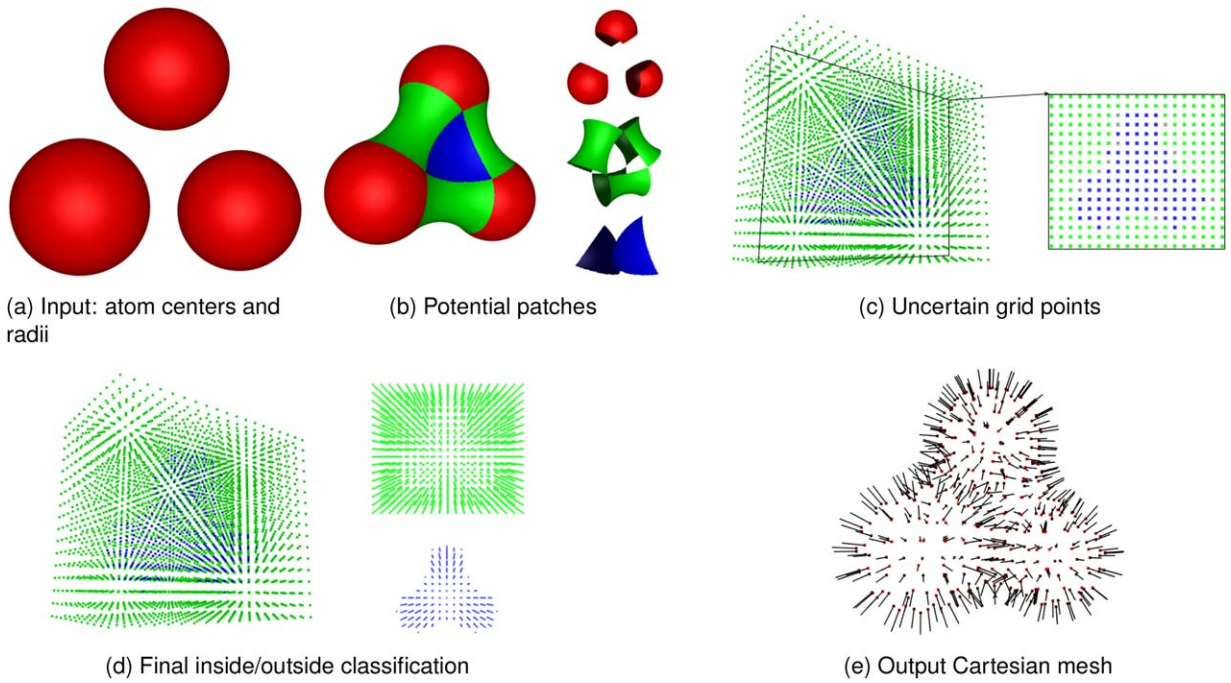

Figure 2. Algorithm overview. a) A list of atom positions and radii as input. b) Result of the step of collecting potential patches based on a probe radius, including convex (red), saddle (green), and concave (blue) patches. c) Initial classification: most grid points are already classified as inside (blue) and outside (green), with few undetermined grid points (gray) visible in the cross-section view. d) Completed classification: based on intersection count calculation, all gray grid points are classified as inside or outside. e) Final output of cartesian mesh: intersection points (red) and the corresponding normals (black arrows). a) Input: atom centers and radii b) Potential patches c) Uncertain grid points d) Final inside/outside classification e) Output Cartesian mesh. [Color figure can be viewed at wileyonlinelibrary.com]

achieve their designed order of convergence or accuracy in the L_∞ norm, which is essential in judging the performance of these elliptic interface methods for complex biomolecular surfaces.

The rest of article is organized as follows. In Algorithm section, the algorithm for constructing the ESES is presented. The most important piece of the Eulerian representation of the SES is the intersection coordinates between the surface and the Cartesian grid lines as well as the corresponding normal directions. The analytical representation of potential SES patches is first computed based on Connolly's work,[31] where accurate convex patches, saddle patches and concave patches are generated and stored for fast access. Each Cartesian grid point is then classified either as inside the SES or outside the SES based on its relation with respect to these three types of patches in its neighborhood. For each Cartesian edge with one Cartesian end point inside the SES and the other outside, we compute its analytical intersection position and corresponding normal direction with respect to the outermost surface patch. These tasks are quite routine except for vicinities of geometric singularities, that is, cusps and tips formed by self-intersection of underlying spherical and toroidal surfaces of singular patches. In these situations, we develop appropriate algorithms to robustly handle the intersection evaluation. In Validation section, a brief review of the MSMS surface is given. The issues of MSMS for some molecules are discussed. Surfaces generated by the ESES and MSMS software packages are compared qualitatively through a surface morphology comparison. The electrostatic solvation free energies based on both ESES and MSMS are evaluated in electrostatic solvation free energy section. We demonstrate that with increasing surface density specified for the SES generation, the energies associated with the MSMS surface converge to those of the analytical ESES. MSMS and ESES are also compared quantitatively in this section. In surface area and enclosed volume section, the ESES areas and enclosed volumes are validated for the Amber PBSA test set. Additionally, the atomic surface area partitioning is presented. In topological features section, the loops and cavities on the manifold formed by the ESES are analyzed based on persistent homology theory and level-set method. A conclusion is given in concluding remarks section.

Algorithm

Our algorithm is designed to robustly evaluate the aforementioned Eulerian representation of SES from the molecular information Figure 1 input. Figure 2 provides an overview of the algorithm. More precisely, we take as input a molecule, represented by a collection of atoms along with their van der Waals radii. We denote the collection as a set A,

$$A=\{(\mathbf{c}_i, r_i)\}_{i=1\ldots N}, \tag{1}$$

where N is the number of atoms, $\mathbf{c}_i$ is the center of atom i, and r_i is its van der Waals radius.

From the input molecule data and a specified Cartesian grid that contains the entire molecule, a 3D boolean array Inside$[I, m, n]$ is built to indicate whether the grid point at (Ih, mh, nh) is inside the SES. Here, h is the grid spacing size. Without loss of generality, the starting corner of grid is assumed to be the origin of the coordinate system. In addition to Inside$[\cdot,\cdot,\cdot]$, we also output a list of edges intersecting the SES along with the intersection point locations $\{[(I, m, n), dir, t]_k\}_{k\in E}$, where E is the set of edges with one end inside and the other outside the SES, (I, m, n) is the starting point of the grid edge, *dir* is the direction of the edge (*X*, *Y*, or *Z*), and $t \in [0, 1)$ is the location of the intersection on the grid edge. For example, $[(3, 4, 5), Y, 0.3]$ indicate the intersection point located at $(3h, 4.3h, 5h)$. If the downstream application also requires a triangular mesh, it is straightforward to assemble the connectivity information using the marching cubes algorithm.[66]

We carry out the two tasks of classifying the grid points and locating the intersection points in three stages in our framework. First, all possible patches of the analytical SES are collected from the input atom location data. Second, we classify the majority of the Cartesian grid points as inside or outside points. Finally, we use the intersection with potential SES patches to determine the rest of the grid points as inside or outside, and the intersection points and associated normals are computed for each grid edge with one Cartesian point inside and the other outside. The overview pseudo-code is provided in Algorithm 1. We elaborate on these three stages in the following subsections.

Algorithm 1 Eulerian SES Construction

Require: atom locations and radii $A=\{(\mathbf{c}_i, r_i)\}_{i=1\ldots N}$ and grid spacing size h.
Ensure: grid point classification Inside$[I, m, n]$ and intersecting edges containing intersection points $\{[(I, m, n), dir, t]_k\}_{k\in E}$.

{Stage 1: collecting potential patches}

1. Singletons: store all atoms not entirely buried inside other atoms
2. Pairs: store all pairs of atoms potentially forming saddle patches
3. Triplets: store all triplets potentially forming concave patches

{Stage 2: initialize Inside$[\cdot,\cdot,\cdot]$}

4. Inside$[I, m, n] \leftarrow$ false for all grid points (Ih, mh, nh)
5. Inside$[I, m, n] \leftarrow$ uncertain for all (Ih, mh, nh) inside SAS
6. for all $\mathbf{p}=(Ih, mh, nh)$ inside any atom i: InAtom$_i$ $(\mathbf{p})=t$rue, Inside$[\mathbf{p}/h] \leftarrow$ true
7. for all $\mathbf{p}$ covered by saddle patch s: InVS$_s$ $(\mathbf{p})$&!InTorus$_s(\mathbf{p})$=true, Inside$[\mathbf{p}/h] \leftarrow$ true
8. for all $\mathbf{p}$ covered by concave patch c: InVT$_c(\mathbf{p})$&!InProbe$_c(\mathbf{p})$=true, Inside$[\mathbf{p}/h] \leftarrow$ true

{Stage 3: finalize Inside$[\cdot,\cdot,\cdot]$ and output intersections}

9. **for** each remaining grid edges $(\mathbf{p}_1, \mathbf{p}_2)$ with one uncertain end $\mathbf{p}_1$ **do**
10. Calculate number of intersection points I along the edge
11. **if** I is even **then**
12. Inside$[\mathbf{p}_1/h] \leftarrow$ Inside$[\mathbf{p}_2/h]$
13. **else**
14. Inside$[\mathbf{p}_1/h] \leftarrow$!Inside$[\mathbf{p}_2/h]$
15. **end if**
16. **end for**
17. Output all intersection points (for each edge with multiple intersections, pick the outermost one)

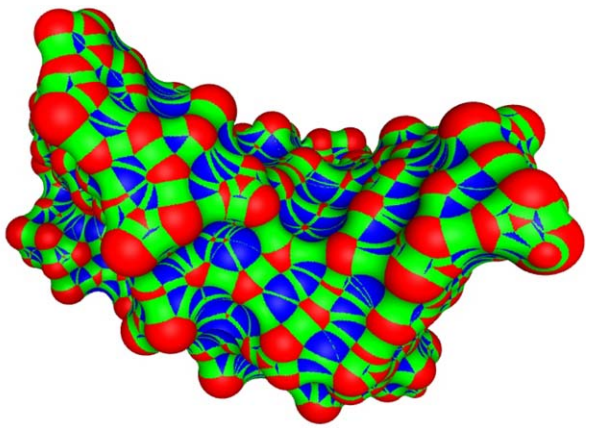

Figure 3. Three types of patches for SES: convex patches (red), saddle patches (green), and concave patches (blue). [Color figure can be viewed at wileyonlinelibrary.com]

Construction of potential patches

In the first stage, we collect all possible patches that may appear on the analytical SES, the geometric information of which was defined in the equations provided by Connolly's work.[31] With these potential patches, we can classify most of the grid points, and we can leave the often ambiguous patch intersection cases to our third stage. We suspect that the numerical issues in calculating such patch boundaries led to some inconsistent surfaces cases in MSMS surface generation, which was treated by increasing the radii of the atoms involved.[34] Note that we include all potential patches instead of just those on the reduced surface in MSMS, as we need to classify the grid points for our intended Eulerian-representation-based applications.

There are three types of patches generated in the SES, namely, *convex patches*, *saddle patches*, and *concave patches*, shown as red, green, and blue patches, respectively, in Figure 3. These convex, saddle, or concave patches are determined by singletons, pairs, or triplets of atoms, respectively.

Convex patches. A convex patch is the exposed part of van der Waals surface of one of the atoms, which can be touched by the probe revolving around atom *i*. It is part of the 0th level set of the quadratic function

$$A_{i,r_i}(\mathbf{p}) = |\mathbf{p}-\mathbf{c}_i|^2 - r_i^2. \tag{2}$$

It is a comparison of the squared distance to the atom center $\mathbf{c}_i$ with radius r_i. A spherical cap-shaped convex patch is typically bounded by a number of convex spherical arcs on the van der Waals sphere of the atom, unless it is a stand-alone atom.

Saddle patches. The saddle-shaped part of SES is formed by the envelopes of the rolling probe surface touching a pair of atoms *i* and *j* at the same time. This part can thus be decomposed into patches of tori of the following form:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} (r_{ij}+r_p\cos\theta)\cos\varphi \\ (r_{ij}+r_p\cos\theta)\sin\varphi \\ r_p\sin\theta \end{pmatrix}, \tag{3}$$

where the major radius of the torus, r_{ij}, is the distance from the probe center touching both atoms to its projection $\mathbf{c}_{ij}$ on the line segment connecting $\mathbf{c}_i\mathbf{c}_j$, and the minor radius is simply the probe radius r_p. The torus surface can also be represented as the 0th level set of the quartic function

$$S_{ij}(\mathbf{p}) = 4r_{ij}^2(\mathbf{p}_z^2 - r_p^2) + (r_p^2 + r_{ij}^2 - |\mathbf{p}|^2)^2. \tag{4}$$

The above equations assume a local coordinate system with origin at $\mathbf{c}_{ij}$ and *Z*-axis aligned to $\mathbf{c}_i\mathbf{c}_j$, which can always be implemented by a rotation $\mathbf{R}_{ij}$ and a translation $-\mathbf{c}_{ij}$ from the global coordinate system. A typical saddle patch is bounded by two convex arcs and two concave arcs, which can be encoded as the limits of the parameters: $\theta \in [\theta_{\min,ij}, \theta_{\max,ij}]$ and $\varphi \in [\varphi_{\min,ij}, \varphi_{\max,ij}]$. All the parameters can be straightforwardly computed (see, e.g., Connolly's work[31]).

Concave patches. When the probe touches three atoms simultaneously, a concave patch of SES is formed as the spherical triangle (the part bounded by three arcs on the sphere $\{\mathbf{p}|\, V_{ijk}(\mathbf{p})=0\}$, with

$$V_{ijk}(\mathbf{p}) = |\mathbf{p}-\mathbf{c}_{ijk}|^2 - r_p^2, \tag{5}$$

where $\mathbf{c}_{ijk}$ is the center of the probe touching atoms *i*, *j*, and *k*). The arcs bounding the spherical triangle are regarded as concave arcs since they bend towards the normal direction of the SES. If the probe touches more than three atoms, the concave patch can be obtained by concatenating the spherical triangles formed by some triplets of these atoms.

Boundary arcs. The boundary curves of the *convex patches* are marked as *convex edges* and the boundary curves of the *concave patches* are *concave edges*. All these edges can be evaluated based on the angle limits of the saddle patches. When a saddle/torus patch is *free*, that is, the probe touching both atoms cannot reach any other atom, there is no limit on φ and its boundary convex edges are the complete contact circles. If there are no associated concave edges generated for a non-free torus, this torus is marked as *blocked*. As aforementioned, the equations in Connolly's work[31] can be used to compute the centers, radii, and the boundary planes/edges/points of all the possible patches. The order of the evaluation of parameters in our implementation is to first construct tori, then place probes blocked by other atoms, and last construct the potential saddle rectangles.

Singularities. With a large number of atoms and their possible configurations in space, numerous singular patches can be present in practice. They are mainly formed by two cases: when the torus of the saddle patch is a spindle torus ($r_{ij} < r_p$); and when the spherical cap around $\mathbf{c}_{ijk}$ of the concave patch is eroded by probe location from the other side of the triangle $\mathbf{c}_i\mathbf{c}_j\mathbf{c}_k$. We defer the handling of singularities to the second and third stages of our framework.

Data structure. We use a simple spatial indexing structure (i.e., a look-up table that maps a location **p** to a list of nearby objects) to allow fast nearest neighbors search. We store a dynamical length array in each cell of a regular Cartesian grid (possibly with different resolution of the output grid) similar to Ref. 67. Assuming that the cell diameter is greater than the sum of the largest atom radius and the probe radius ($r_{\max}+r_p$), all possible pair and triplets potentially forming saddle or concave patches can be found within a 3×3×3 subgrid in $O(1)$ time (constant time), since the number of atoms within each grid cell has an upper bound. Thus, the construction can be done with $O(N)$ time complexity where N is the atom count. We also use the same spatial indexing to allow a constant time search for nearby patches in the next two stages. The storage required for this spatial indexing structure depends on the grid size instead of just the atom count. So there can be improvements to space complexity based on k–d tree, which is a special case of binary space partitioning (BSP) trees,[34] or spatial hash table. However, we choose the grid-based indexing structure for the locality of the memory access to facilitate parallelization in future.

Initial classification of Cartesian grid points

In the second stage, we classify Cartesian grid points in the computational domain as either inside or outside the SES. We start by labeling the ones that are easy to classify, and leave the rest to the third stage, during which we count the number of times a grid edge passes through the interface.

Any grid point outside of the solvent accessible surface (SAS) is clearly outside of the SES. So we first label all grid points that are outside of the union of the augmented atomic balls as outside points. The augmented atomic ball for atom i is the ball centered at $\mathbf{c}_i$, with the radius of the $r_{a,i}=r_i+r_p$ (where r_i is the radius of the atom and r_p is the radius of the probe). Every grid point outside the union can definitely be reached by the probe, and is thus outside of SES.

For efficiency, we simply initialize $\mathrm{Inside}[\cdot,\cdot,\cdot]$ to the boolean value *false*, and then set the Cartesian grid points within the *augmented atoms* to *uncertain*. This can be achieved by going through each atom i defined in eq. (2), and test whether $A_{i,r_{a,i}}(\mathbf{p})\le 0$ for each grid point $\mathbf{p}\in[\mathbf{c}_x-r_{a,i},\mathbf{c}_x+r_{a,i}]\times[\mathbf{c}_y-r_{a,i},\mathbf{c}_y+r_{a,i}]\times[\mathbf{c}_z-r_{a,i},\mathbf{c}_z+r_{a,i}]$. As the number of grid points checked for each atom has an upper bound, the overall time for this step is $O(N)$, that is, linear time complexity in terms of atom count N.

For all the *uncertain* grid points, we now label those easily determined as inside points. One sufficient condition that we use in our framework to determine $\mathrm{Inside}[\mathbf{p}/h]=\mathrm{true}$ can be summarized as

$$\begin{aligned}&\mathrm{InAtom}(\mathbf{p})+\sum_{s\in S}[(\mathrm{InVS}_s(\mathbf{p})\&!(\mathrm{InTorus}_s(\mathbf{p}))]\\&+\sum_{c\in C}[\mathrm{InVT}_c(\mathbf{p}))\&!(\mathrm{InProbe}_c(\mathbf{p}))]=\mathrm{true},\end{aligned}\tag{6}$$

where the summation is the boolean OR operator (e.g., true + true = true), S denotes the set of potential saddle patches, and C denotes the set of potential concave patches; the boolean predicate $\mathrm{InAtom}(\cdot)$ indicates whether the point is within the van der Waals surface; the predicate $\mathrm{InVS}_s(\cdot)$ indicates whether the point is inside the visibility sphere of a saddle patch s, which is the sphere containing points potentially blocked from any probe by the two atoms generating s; $\mathrm{InTorus}_s(\cdot)$ determines whether the torus formed by the probe rotating around the axis connecting the two atoms would cover that point, in which case the point is not to be labeled inside; $\mathrm{InVT}_c(\cdot)$ indicates whether the point is inside the visibility tetrahedron of the concave patch c, which is the tetrahedron containing points potentially blocked by the three atoms generating c; $\mathrm{InProbe}_c(\cdot)$ indicates whether the point is contained in some probe ball in any nearby locations. Each predicate is discussed below in detail. Note, however, that the above equation does not provide a *necessary* condition.

InAtom. In this step, we label all the grid points inside the van der Waals surface with $\mathrm{Inside}[\mathbf{p}/h]=\mathrm{true}$. Recall that each atom i in the input has its center at $\mathbf{c}_i$ and its van der Waals radius given as r_i. For a Cartesian grid point $\mathbf{p}=(lh,mh,nh)$, $\mathbf{InAtom}(\mathbf{p})$ is *true* if and only if

$$\exists i\ A_{i,r_i}(\mathbf{p})\le 0,\tag{7}$$

with $A_{i,r_i}(\cdot)$ defined in eq. (2). In implementation, this step can be performed during the step of checking whether the grid point inside SAS. Thus, the added time complexity is also $O(N)$.

InVS and InTorus. The predicate $\mathrm{InVS}_s(\cdot)$ is an abbreviation for *Inside the Visibility Sphere of the saddle patch s*. The concept of visibility sphere was proposed in the work by Krone et al.[67] to compute the saddle patch—only the points of the torus surface that is inside the visibility sphere can be on the saddle patch. We use the sphere to locate candidate grid points that can be inside the SES due to the associated saddle patch. The center and the radius of the *visibility sphere* can be computed from the following equations:

$$\mathbf{c}_{vs}=\frac{|\mathbf{c}_p-\mathbf{c}_i|}{|\mathbf{c}_p-\mathbf{c}_i|+|\mathbf{c}_p-\mathbf{c}_j|}\mathbf{c}_j+\frac{|\mathbf{c}_p-\mathbf{c}_j|}{|\mathbf{c}_p-\mathbf{c}_i|+|\mathbf{c}_p-\mathbf{c}_j|}\mathbf{c}_i,\tag{8}$$

$$r_{vs}=\left|\frac{r_i}{|\mathbf{c}_p-\mathbf{c}_i|}(\mathbf{c}_p-\mathbf{c}_i)+\mathbf{c}_i-\mathbf{c}_{vs}\right|,\tag{9}$$

where $\mathbf{c}_{vs}$ is the center of the visibility sphere, r_{vs} is its radius, $\mathbf{c}_i$ and $\mathbf{c}_j$ are the centers of the two atoms forming the saddle patch s, and $\mathbf{c}_p$ is the center of an arbitrary probe ball touching both atoms. See Figure 4 for a typical visibility sphere. Being within the visibility sphere is a necessary condition for a grid point to be in the inaccessible region formed by the saddle patch s, but apparently not a sufficient condition. In other words, the region where $\mathbf{InVS}_s(\mathbf{p})=\mathrm{true}$ includes some points that are accessible from certain probe location (colored blue in Fig. 4). Thus, we use $\mathbf{InTorus}_s$ to further determine whether a point is inside the torus, which can be scooped out by the probe rotating around the axis connecting atom i and atom j. The yellow dashed circles show a cross-section of the torus. Note that when the saddle patch s is not free, we only check the part of the visibility sphere that is located within the

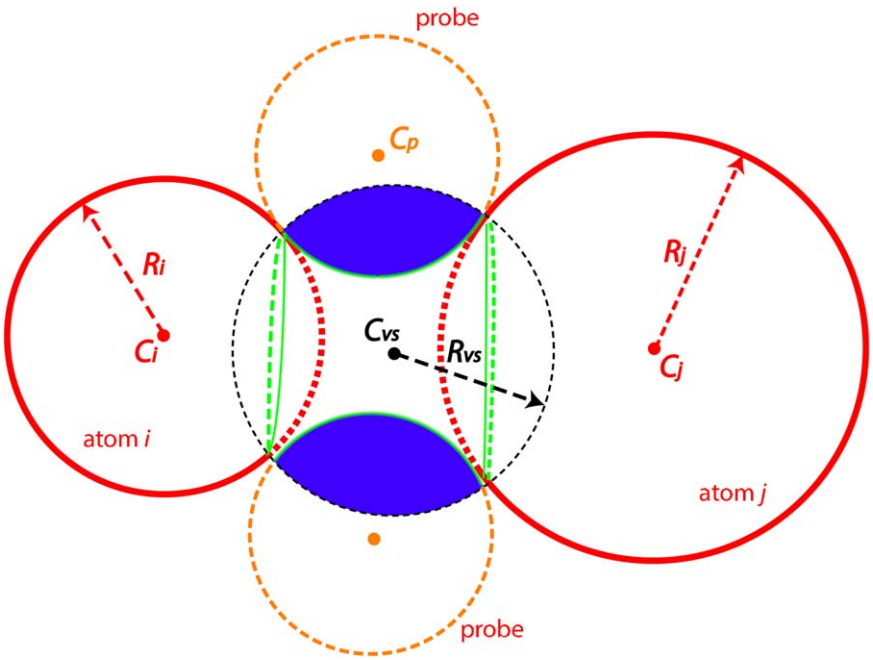

Figure 4. Figure for **InsideVS**. Solid lines show a cross-section of a simple SES composed of two atoms (red) and one saddle patch (green). Black dashed circle is the cross-section of the corresponding visibility sphere when the probe touches atom *i* and atom *j* simultaneously. **InsideVS** returns true when the point is determined to be inside the visibility sphere. [Color figure can be viewed at wileyonlinelibrary.com]

border of the saddle patch. Details for computing these two boolean predicates are provided below. As we leave some grid points to be uncertain, we only ensure that $\mathrm{InVS}_s(\mathbf{p})\backslash\&!\ \mathrm{InTorus}_s(\mathbf{p})$ is a *sufficient condition* for a point to be inside the SES.

When the torus associated to a saddle patch *s* is a ring torus (the major radius greater than or equal to the minor radius, that is, $r_{vs} \geq r_p$), the evaluation of both $\mathrm{InVS}_s(\cdot)$ and $\mathrm{InTorus}_s(\cdot)$ is straightforward:

$$\mathrm{InVS}_s(\mathbf{p}) \equiv |\mathbf{p}-\mathbf{c}_{vs,s}|^2 - r_{vs,s}^2 \leq 0,$$

and

$$\mathrm{InTorus}_s(\mathbf{p}) \equiv S_{ij}(\mathbf{p}) \leq 0,$$

where *i* and *j* are the indices of the atoms forming the torus, which is assume to be a torus free from intersection with any other atoms. When other atoms block the saddle patch, two-additional (2D) conditions are used in conjunction with both of the above conditions:

$$\theta_{\mathbf{p}} \in [\theta_{\min,s}, \theta_{\max,s}]$$

and

$$\varphi_{\mathbf{p}} \in [\varphi_{\min,s}, \varphi_{\max,s}],$$

where $[\theta_{\min,s}, \theta_{\max,s}]$ and $[\varphi_{\min,s}, \varphi_{\max,s}]$ are the ranges of longitudinal and latitudinal angles of the rectangular saddle patch *s*, and $\theta_{\mathbf{p}}$ and $\varphi_{\mathbf{p}}$ are the longitudinal and latitudinal angles of the projection of **p** onto the torus. The projection (the closest point) on the torus can be evaluated efficiently even though the level set function S_{ij} is quartic, as we can decompose the projection into two steps. First, the longitudinal angle $\theta_{\mathbf{p}}$ can be found by the projection of **p** onto the major circle, and then the latitudinal angle $\varphi_{\mathbf{p}}$ can be found by the projection of **p** onto the minor circle at the angle $\theta_{\mathbf{p}}$.

Note that all the evaluations in this subsection are based on the coordinates of **p** in a local coordinate system aligned to the torus associated to the saddle patch *s*, that is, we use $\tilde{\mathbf{p}} = \mathbf{Rp}+\mathbf{t}$ instead of **p**, where the rotation **R** and the translation **t** can transform the torus into one centered around the origin lying flat on the *XY*-plane, as described in construction of potential patches section.

When the torus is a spindle torus (i.e., the minor radius is greater than the major radius $r_p > r_{vs,s}$, causing the toric surface to intersect with itself), the evaluation is much more complicated than the regular ring torus case. So we stick to the sufficient (but not necessary) condition for the point to be inside, by disjunction (boolean operator or) of the above $\mathrm{InTorus}_s(\cdot)$ with $\mathrm{InLemon}_s(\cdot)$, which removes all the points in the self-intersection part of the spindle torus. The spindle (the lemon shaped intersection) is removed as the points in it are potentially touched by the probe rotating along the other side of the major circles. Note that this part is not automatically removed by $S_{ij}(\mathbf{p}) \leq 0$ as the points actually produce positive S_{ij} values due to the self-intersection. The surface of the self-intersection can be described as a lemon,[68] defined by

$$z^2 = r_p^2 - (\sqrt{x^2+y^2} + r_{vs,s})^2.$$

Thus, for point **p** to be inside the lemon, the following predicate must be true:

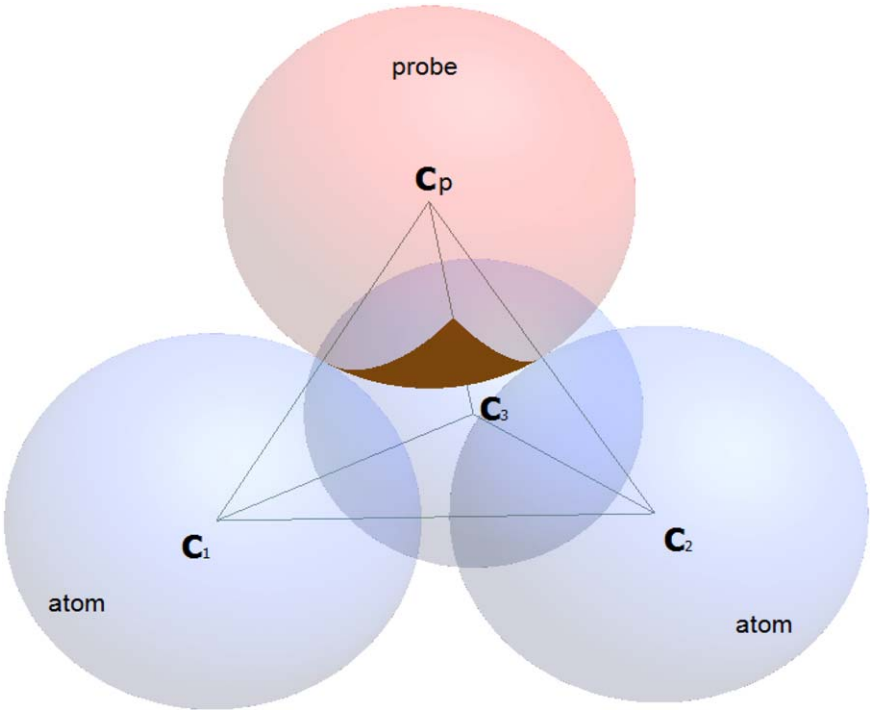

Figure 5. An illustration of **InsideVT**. A probe (red) centered at $\mathbf{C}_p$ is in contact with three atoms (blue) centered at $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$, respectively. **InsideVT** is tagged as true when the point is detected as inside the tetrahedron $\mathbf{C}_p\mathbf{C}_1\mathbf{C}_2\mathbf{C}_3$. [Color figure can be viewed at wileyonlinelibrary.com]

$$\mathrm{InLemon}_s(\mathbf{p}) \equiv \left[\mathbf{p}_z^2 < r_p^2 - (\sqrt{\mathbf{p}_x^2+\mathbf{p}_y^2}+r_{vs,s})\right] \;\&\; \left[\sqrt{\mathbf{p}_x^2+\mathbf{p}_y^2}+r_{vs,s}-r_p < 0\right].$$

In this test, we do not check the range of $\theta_{\mathbf{p}}$ and $\varphi_{\mathbf{p}}$ since the lemon region can be eroded from the other side of torus.

InVT and InProbe. The predicate $\mathbf{InVT_c}$ is an abbreviation for *Inside the Visibility Tetrahedron of the concave patch c*, which is defined as the tetrahedron formed by the centers of the three atoms associated to the concave patch *c* and the probe touching the three atoms, as shown in Figure 5. The evaluation of $\mathrm{InVT}_c(\mathbf{p})$ is carried out by the conjunction of four linear tests to see if $\mathbf{p}$ is on the correct side of the four planes containing the triangular faces of the tetrahedron.

To form a sufficient condition for a grid point to be inside, we need to exclude the Cartesian grid points inside the probe spheres with fixed positions *in the neighborhood* of the concave patch *c*, since the concave patch may develop singularities and be eroded by other concave patches nearby. So the associated $\mathbf{InProbe}_c$ is true as long as it is inside one of those probes, as a safety measure against mishandling the singularity case:

$$InProbe_c(\mathbf{p}) \equiv \sum_{\mathbf{c}_{ijk}\in N(\mathbf{p})} [|\mathbf{c}_{ijk}-\mathbf{p}|^2 < r_p^2],$$

where $N(\mathbf{p})$ is the ball around $\mathbf{p}$ with the radius of $r_{\max}+r_p$, $\mathbf{c}_{ijk}$ are the centers of fixed location probes, which can be found with constant time complexity using the spatial indexing data structure, and $\sum$ indicates the boolean operator OR. Note that in taking this precautionary step, we again allow false negatives of the test of being inside, but not false positives.

Intersection calculation

Since eq. (6) only provides a *sufficient* condition for tagging *inside* Cartesian grid points, there may exist unlabeled inside Cartesian grid points. They exist even if we include the pairs and triplets that are not on the reduced surface, possibly due to the interior "tunnel" structure of the SES (see Fig. 6). Any of these points can be correctly tagged by counting the number of intersection points with SES along any ray originated from it. For locality and efficiency of computation, instead of counting intersections along a ray to infinity, we can use the Cartesian grid edge with one unlabeled end grid point $\mathbf{p}_1$ and the other end point $\mathbf{p}_2$ already labeled. As long as such grid edges exist, there are unlabeled grid points; and conversely, when all those edges are turned into edges with both ends labeled, all grid points can be properly labeled.

Assuming that on such a grid edge, there are I intersection points with the SES, and point $\mathbf{p}_1$ shares the same Boolean label $\mathrm{Inside}[\mathbf{p}_1/h]$ with that of the point $\mathbf{p}_2$ if I is even. Otherwise, $\mathbf{p}_1$ is tagged with the opposite label of $\mathbf{p}_2$. In practice, we typically find $I = 0$ or $I = 1$, unless the grid spacing h is particularly large.

For robustness, we first retrieve all potentially intersecting patches using the spatial indexing data structure, and accumulate the intersection calculation for each of them. However, in this case, we need the intersections to be exactly on SES, instead of including ones that are either inside or outside the

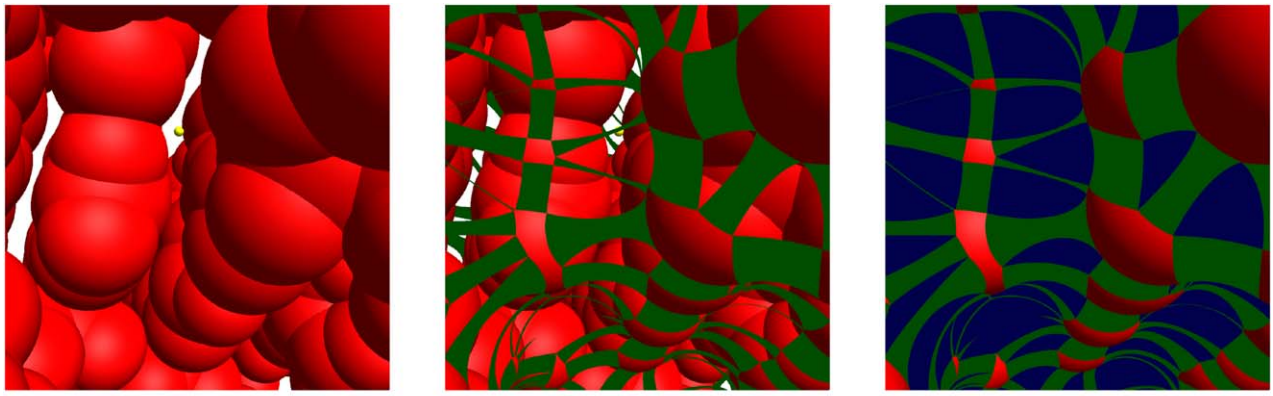

Figure 6. An illustration of an unlabeled Cartesian grid point. The subfigures from left to right highlight part of the 1A2E model with different patches respectively, where we can see the Cartesian point (yellow) should be tagged as *inside* the SES. However, it does not satisfy the *sufficient* conditions proposed in eq. (6). a) Visualization with atoms only b) Visualization with atoms and saddle patches only c) Visualization for overall SES. [Color figure can be viewed at wileyonlinelibrary.com]

true SES. Thus, we validate the intersection points against all possible false positive cases:

- Valid intersections with the *convex patch* (Fig. 7) should not be
  1. inside of the nearby atoms, checked through the distance to the neighboring atom centers, or
  2. covered by its neighboring *saddle patches*, checked through the boundary normal of the bordering convex edge.
- Valid intersections with the *saddle patch* (Fig. 8) should have
- *InLemon* false (i.e., not in the self-intersection of spindle torus), and
- *InVS* true (i.e., inside of its associated visibility sphere).
- Valid intersections with the *concave patch* should be located
- in spherical triangle within the concave arcs defined by the three triangle faces ($\mathbf{c}_1\mathbf{c}_2\mathbf{c}_p, \mathbf{c}_2\mathbf{c}_3\mathbf{c}_p, \mathbf{c}_1\mathbf{c}_p\mathbf{c}_3$) of the visibility tetrahedron $\mathbf{c}_p\mathbf{c}_1\mathbf{c}_2\mathbf{c}_3$, where $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3$ are the centers of the three touching atoms (Fig. 5), and
- outside of any nearby probe spheres with fixed locations.

Computing the analytical intersection with convex patches or concave patches amounts to solving a simple quadratic equation

$$|(1-t)\mathbf{p}_1+t\mathbf{p}_2-\mathbf{c}|^2-r^2=0,$$

where $\mathbf{c}$ and r are the center and radius of the associated sphere respectively, and t is checked to be real and within range $[0, 1]$.

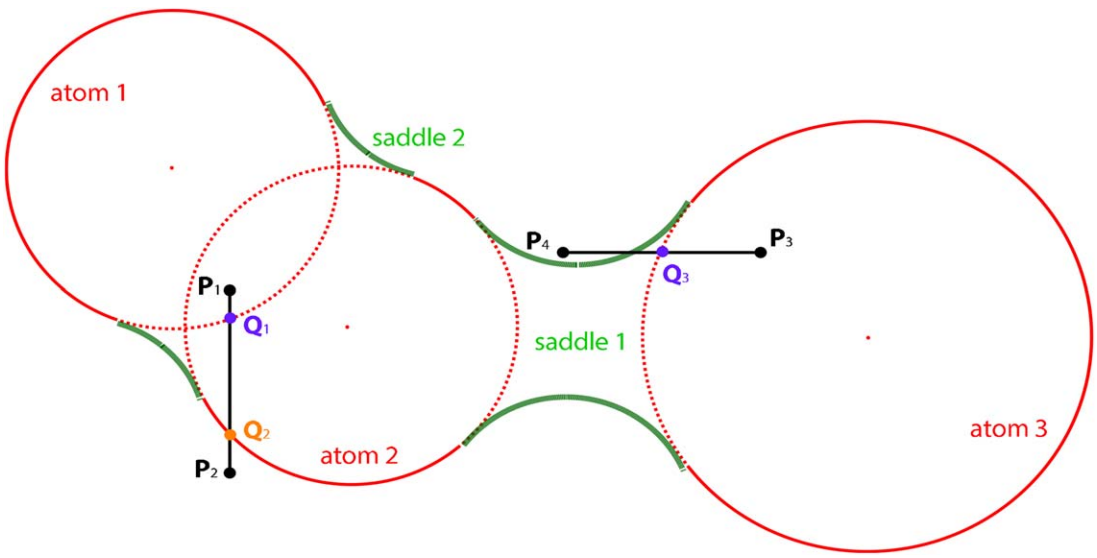

Figure 7. Validation for intersection with convex patches. Solid lines illustrate the outline of a simple SES composed of three atoms (red) and two saddle patches (green). $\mathbf{P}_1\mathbf{P}_2$ and $\mathbf{P}_3\mathbf{P}_4$ are Cartesian edges. Note here we only mark the possible intersections with convex patches. Edge $\mathbf{P}_1\mathbf{P}_2$ seems to have two intersections $\mathbf{Q}_1$ and $\mathbf{Q}_2$ with convex patches, where $\mathbf{Q}_1$ is actually an invalid intersection since it is inside of atom 2. Similarly, intersection point $\mathbf{Q}_3$ for edge $\mathbf{P}_3\mathbf{P}_4$ is not valid, since it is included by saddle patch 2. [Color figure can be viewed at wileyonlinelibrary.com]

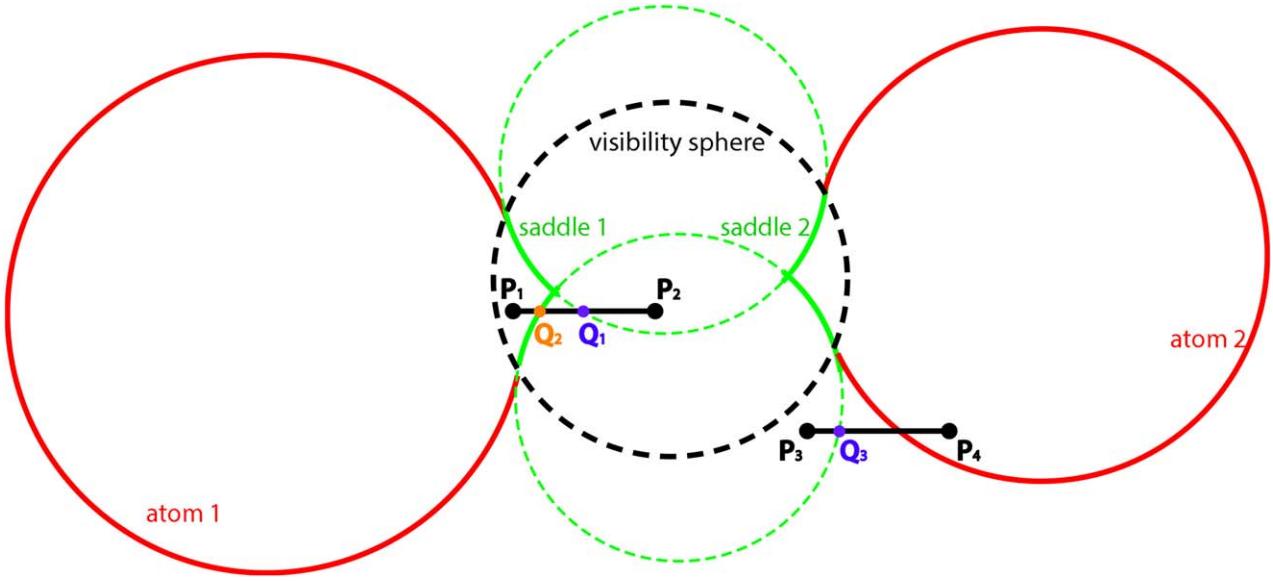

Figure 8. Validation for intersection with saddle patches. Solid lines illustrate the outline of a simple SES composed of two atoms (red) and two saddle patches (green), where two separate saddle patches belong to the same torus. The black dashed circle is the outline of the corresponding visibility sphere. Singularities are generated in this case and they are appropriately handled implicitly with our designed conditions. $\mathbf{P}_1\mathbf{P}_2$ and $\mathbf{P}_3\mathbf{P}_4$ are the Cartesian edges. Note here we only mark the possible intersections with saddle patches. There are two intersections $\mathbf{Q}_1$ and $\mathbf{Q}_2$ for edge $\mathbf{P}_1\mathbf{P}_2$ with saddle patches, where $\mathbf{Q}_1$ is invalid since its associated Boolean tag IsInLemon is true. The intersection point $\mathbf{Q}_3$ for edge $\mathbf{P}_3\mathbf{P}_4$ with saddle patches is not valid either, since it is outside of the corresponding visibility sphere. [Color figure can be viewed at wileyonlinelibrary.com]

For saddle patches it is a quartic equation

$$S_{ij}((1-t)\mathbf{p}_1+t\mathbf{p}_2)=0,$$

where S_{ij} is the associated quartic level-set function of the associated torus. In our tests, Newton's method may lead to missing or double-counting of intersection points due to instability and dependence on initial guesses. So we adopt Jenkins–Traub algorithm[69] with a few modifications for robustness to solve the quartic equation for better numerical stability.

Output of intersection coordinates. After labeling of the Cartesian grid points is complete, we only need to compute the intersection coordinates and associated normals for Cartesian edges with one Cartesian point inside SES and the other outside. The same set of equations and same validation rules discussed above are used for determining the intersection coordinates with nearby patches. In fact, we store the locations of some of the intersection points during the labeling procedure in the third stage for efficiency. So only the grid edges with one end inside and the other end outside labeled in the second stage need to be examined. The evaluation of the normal direction for spheres and tori with analytical expressions is straightforward.

For typical applications using the Eulerian representation, the Hermite data (intersection point locations with associated surface normals) contain only one intersection per intersecting grid edge. So when the intersection count is greater than 1, we simply choose the outermost intersection point: smallest t if the outside end point is at $t=0$ and largest t otherwise. Note that when multiple intersections with grid edges occur frequently, it indicates that the resolution of the Cartesian grid is insufficient to resolve the details of the SES.

Validation

Surface morphology

One of the most popular SES software packages is the Michel Sanner molecular surface (MSMS) software package.[34] The MSMS surface is based on the structure of reduced surface, which is one of the first programs that can handle the geometric singularities, such as self-intersecting surfaces. The MSMS method consists of four critical steps for computing the triangulated solvent excluded surface:

1. Compute the reduced surface of a molecule,
2. Build an analytical representation of the SES from the reduced surface,
3. Remove all the self-intersecting parts from the analytical SES built above, and
4. Produce a triangulation of the reduced surface based on the SES.

More detailed description and numerical implementation of the MSMS surface is provided in Ref. [34].

Currently, owing to the efficiency and robustness of the MSMS software, the surface has already been incorporated into several comprehensive molecular mechanics software packages, including the visual molecular dynamics software (VMD) and the UCSF Chimera software.

The MSMS surface generation depends on an additional parameter other than the intrinsic parameters for controlling

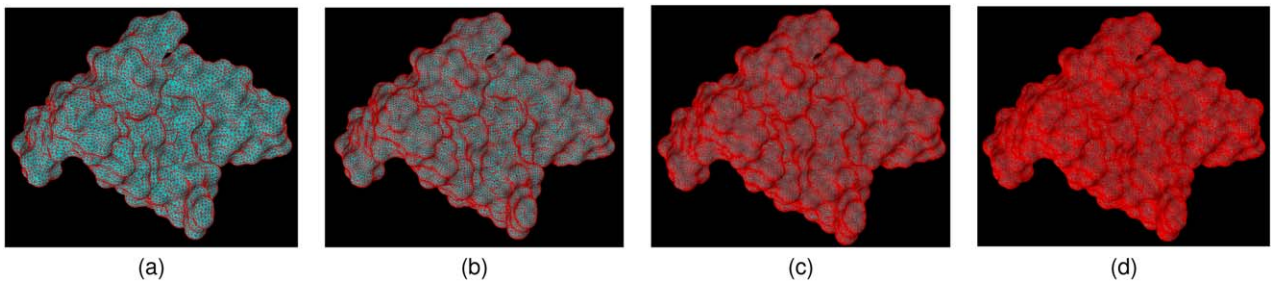

Figure 9. MSMS surface for biomolecule 1A2E. a)–d) are the MSMS surface with densities 5, 10, 25, and 50, respectively. The MSMS surfaces are all generated successfully. [Color figure can be viewed at wileyonlinelibrary.com]

the triangulation of the SES, namely, the surface density, which is the approximated number of triangles on per unit Å^2 area. Usually, the larger the density is, the closer the triangulated MSMS surface is to the analytical solvent excluded surface, provided the surface can be generated successfully.

In comparison, ESES method provides accurate SESs at all grid density. Thus, ESES is density independent.

Figure 9 shows the MSMS surface for a nucleic acid molecule (PDB ID: 1A2E) generated at densities 5, 10, 25, and 50, respectively. The surface can be generated successfully at all the tested densities. Obviously, with the increasing vertex density, the MSMS surface becomes smoother.

Figure 10 depicts the analytical SES for molecule 1A2E generated by the ESES software. It can be seen that the molecular surfaces generated by both ESES and MSMS software packages are qualitatively consistent with each other.

Despite the great success of the MSMS surface software, it has some problems. In particular, MSMS software might fail to generate an SES at some densities for certain molecules. Occasionally, the MSMS software provides incorrect surfaces at certain high density. Figure 11 illustrates one example. The molecular surface can be generated successfully at densities 5, 10, and 25. However, the MSMS output at density 50 is incorrect for 1dqz-A.p22.

For the comparison, Figure 12 depicts the ESES surface for protein 1dqz-A.p22, which is qualitatively consistent with the correct surfaces generated by MSMS.

In all the tests, the protein or nucleic acid molecular structure comes from the Amber PBSA test set,[70] which has already been parameterized. The data is available from web page http://rayl0.bio.uci.edu/rayl/. The version of the MSMS surface software used is MSMS 2.6.1 Unix/Linux i86_64.

Electrostatic solvation free energy

To further validate the present ESES software, we consider electrostatic solvation free energy calculations using both ESES and MSMS surfaces. Here, the electrostatic solvation free energies are computed based on the Poisson–Boltzmann model, which is an implicit solvent model that couples the atomistic modeling of the solute molecule with the continuum treatment of the solvent. Mathematically, the PB model can be formulated as an elliptic interface problem defined in an open domain $\Omega=\Omega_s\cup\Gamma\cup\Omega_m\subset\mathbb{R}^3$, where Ω_m is the solute domain enclosed by the molecular surface, Ω_s is the solvent domain, and Γ is the interface of the solvent and solute domains, that is, the solvent excluded surface.[3] The governing equation of the PB model is given by the Poisson–Boltzmann equation (PBE):

$$-\nabla\cdot(\epsilon(\mathbf{r})\nabla\phi(\mathbf{r}))=4\pi(\rho_m(\mathbf{r})+\rho_s(\mathbf{r})), \tag{10}$$

where $\phi(\mathbf{r})$ is the electrostatic potential, and $\rho_m(\mathbf{r})=\sum_{i=1}^{N_m}Q_i\delta(\mathbf{r}-\mathbf{r}_i)$ is the charge density of the solute with Q_i being the partial charge of the ith atom at position $\mathbf{r}_i$, and N_m the number of charged atoms. Here, $\rho_s(\mathbf{r})=\sum_{\alpha=1}^{N}q_\alpha c_\alpha e^{-q_\alpha\phi(\mathbf{r})/kT}$ is the continuum charge distribution of ions in the solvent, with N being the number of ionic species, c_α the bulk concentration of αth ion species, q_α the charge of αth ionic species, k the Boltzmann constant, and T the absolute temperature. In eq. (10), $\epsilon(\mathbf{r})$ is the permittivity which is assumed to be constants in both the solute and solvent domains:

$$\epsilon(\mathbf{r})=\begin{cases}\epsilon_s=80, & \text{if } \mathbf{r}\in\Omega_s,\\ \epsilon_m=1, & \text{if } \mathbf{r}\in\Omega_m.\end{cases}$$

The PBE is solved with the Debye–Hückel boundary condition, which reads for the pure water solvent:

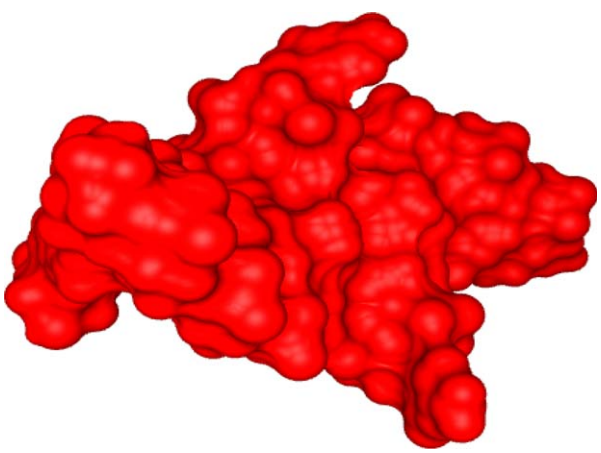

Figure 10. The ESES for biomolecule surface (PDB ID: 1A2E). [Color figure can be viewed at wileyonlinelibrary.com]

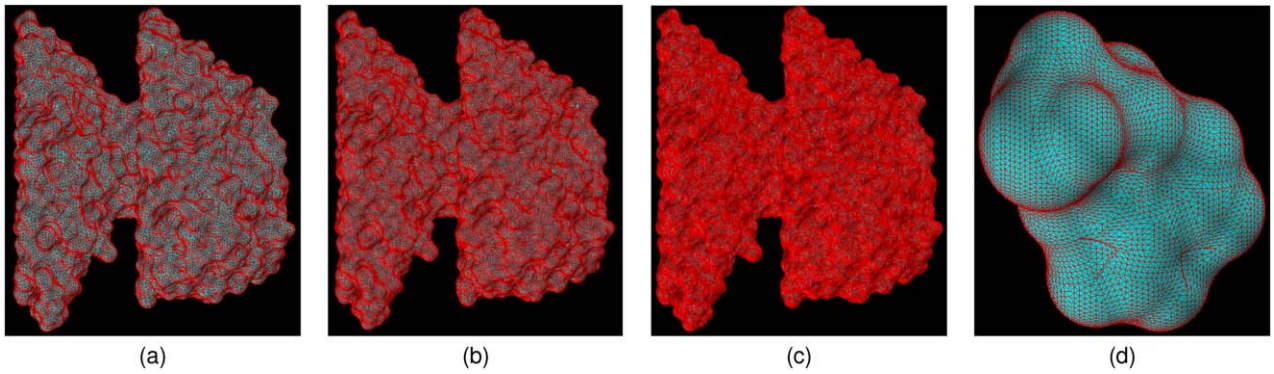

Figure 11. The MSMS surfaces for molecule 1dqz-A.p22. a)–d) are the MSMS surfaces with densities 5, 10, 25, and 50, respectively. The MSMS fails to generate the correct surface at density 50, while it works well at other densities. [Color figure can be viewed at wileyonlinelibrary.com]

$$\phi(\mathbf{r})=\sum_{i=1}^{N_m}\frac{Q_i}{\epsilon_s|\mathbf{r}-\mathbf{r}_i|}. \tag{11}$$

Across the solvent–solute interface Γ, the continuities for electrostatic potential and flux are enforced:

- Continuity of electrostatic potential:

$$[\phi]|_\Gamma=\phi_s(\mathbf{r})-\phi_m(\mathbf{r})=0. \tag{12}$$

- Continuity of electrostatic potential flux:

$$[\epsilon\phi_\mathbf{r}]=\epsilon_s(\mathbf{r})\nabla\phi_s\cdot\mathbf{n}-\epsilon_m\nabla\phi_m\cdot\mathbf{n}=0, \tag{13}$$

where $\mathbf{n}=(n_x,n_y,n_z)$ is the outer normal direction of the interface Γ, which points from the solute domain to the solvent domain, and ϕ_s and ϕ_m are the limiting values at the interface Γ from Ω_s and Ω_m, respectively.

The electrostatic solvation free energy ΔG_{sol} is calculated as:

$$\Delta G_{\text{sol}}=\frac{1}{2}\sum_{i=1}^{N_m}Q_i(\phi(\mathbf{r_i})-\phi_{\text{homo}}(\mathbf{r}_i)),\quad \mathbf{r}_i\in\Omega_m \tag{14}$$

where Q_i is the partial charge at ith atom, ϕ is solved from the above PB equation, and ϕ_{homo} is solved from the following Poisson equation:

$$-\epsilon_m\nabla^2\phi_{\text{homo}}(\mathbf{r})=4\pi(\rho_m(\mathbf{r})+\rho_s(\mathbf{r})), \tag{15}$$

by the fast Fourier transform (FFT) when there is no salt.

Obviously, for a given molecule with the same force field assignment and the same PB solver, the calculated electrostatic solvation free energies should be the same when the SESs generated by different software packages are consistent with each other. In the following, we demonstrate that with the increasing vertex density of the MSMS surface, the MSMS surface-based electrostatic solvation free energies converge to those based on the ESES surface.

Figure 13 shows the convergence of the electrostatic solvation free energies calculated using MSMS surfaces to those obtained by ESES surfaces. All the calculations are carried out by a highly accurate PB solver, MIBPB software,[62–65,71] in which the PB equation is solved on a Cartesian mesh. The MSMS surface is generated with densities varying from 10 to 100. Since the MSMS surface is given by the Lagrangian representation, that is, the triangulation of the molecular surface, a Lagrangian to Cartesian transformation is employed to embed the MSMS surface to the Cartesian mesh.[72]

Apparently, with the increasing vertex density, the MSMS surface will converge to the analytical molecular surface, SES, provided that the MSMS software does generate the correct surface at the tested series of surface densities without modifying the atom radii. Results demonstrated in Figure 13 indicate the following two observations:

- The SES generated by the ESES software leads to results predicted by analytical or exact SES.
- The MSMS surface converges to the analytical SES with increasing mesh density.

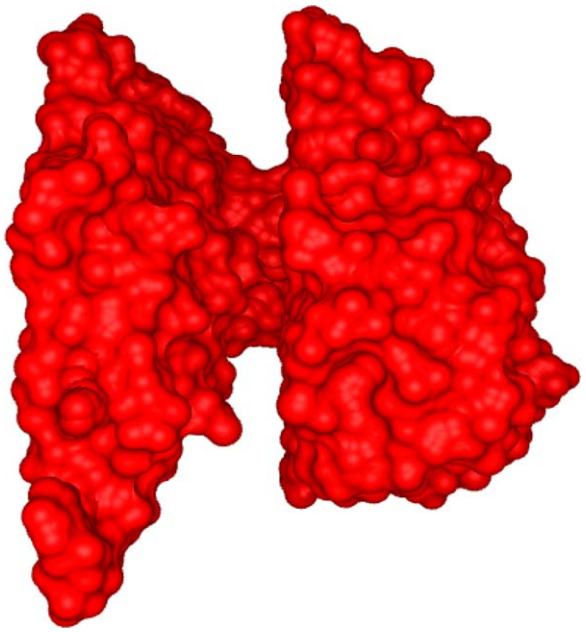

Figure 12. The ESES surface for protein 1dqz-A.p22. [Color figure can be viewed at wileyonlinelibrary.com]

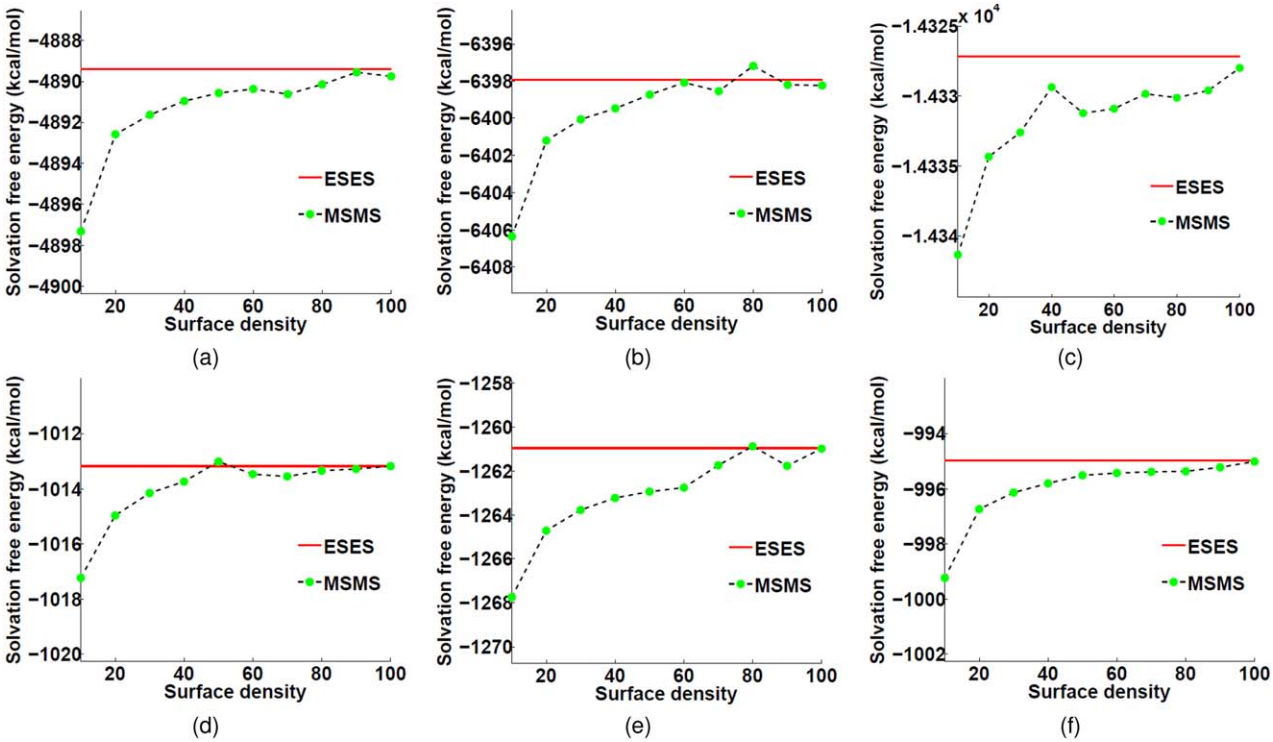

Figure 13. The convergence of the electrostatic solvation free energies calculated using MSMS surfaces to that obtained using the ESES surface. a)–c) are for nucleic peptide molecules with PDB IDs 1A2E, 1BNA, and 1L4J, respectively. d)–f) are for protein molecules with PDB IDs 1a93-B.p22, 1aca-0.p22, and 1b8w-A.p22, respectively. All the energies are obtained by the MIBPB software at grid size 0.5 Å. Note that the energies calculated using the ESES are density independent. [Color figure can be viewed at wileyonlinelibrary.com]

Surface area and enclosed volume

The surface area and enclosed volume are widely used in the modeling of the nonpolar solvation free energy in biophysics. The accurate calculation of the surface area and enclosed volume is of essential importance in the solvation analysis. The surface area and enclosed volume calculations on the triangular mesh was discussed in our earlier work.[73] In this work, similar calculations on the Cartesian mesh are tested. The ESES embeds an SES to the Cartesian mesh, which is labeled with the set of grid points J_1 inside the surface Γ and another set of grid points J_2 outside the surface. ESES also offers information on the set of coordinates where the interface Γ intersects with the Cartesian mesh lines and the corresponding normal directions. Additionally, we identify the set of irregular grid points, J_{Irr}, where a grid point is said to be irregular when at least one of its neighbor grid point is located on the other side of the interface. For a second-order scheme, the area of the SES Γ can be evaluated by

$$\text{Area}=\int_{\Gamma} dS \approx \sum_{(i,j,k)\in I} \left(|n_x|+|n_y|+|n_z|\right)h^2, \qquad (16)$$

where h is the grid size, and I is the set of irregular grid points that is either inside Γ or on the Γ. Here, $|n_x|$ is the magnitude of the first component of the normal direction at intersecting point (x_0, y_j, z_k), which is the intersection of Γ with the x-mesh line that passes through (x_i, y_j, z_k). In eq. (16), $|n_y|$ and $|n_z|$ are defined analogously. Note that for higher order schemes, the whole J_{Irr} set should be used, while the last (or the first) irregular point on each mesh line should be omitted.

Similarly, for a second-order scheme, the volume enclosed by the molecular surface Γ can be evaluated by

$$\text{Volume}=\int_{\Omega_m} d\mathbf{r} \approx \frac{1}{2}\left(\sum_{(i,j,k)\in J_1} + \sum_{(i,j,k)\in J_1 \cup J_{\text{Irrin}}}\right)h^3, \qquad (17)$$

where J_{Irrin} is the set of irregular grid points that are inside the SES.

Smereka originally proposed a similar scheme[74] for evaluating the surface integration and analyzed its convergence. We have utilized a similar scheme for calculating the surface area of SESs.[25]

Table 1 shows the grid refinement analysis of the numerical scheme given by eqs. (16) and (17) on the sphere of radius 2. Obviously the numerical scheme for area calculations is of second-order convergence, while that for the calculation of surface enclosed volume is roughly of third-order convergence.

Table 2 displays the area and volume calculation of the sphere of radius 2 with MSMS at different densities, it is easy to see that the higher the density is, the more accurate the area and volume calculations are.

In the following, the above numerical schemes for area and volume calculations are utilized for the calculations of molecular surfaces generated by the ESES software for the PBSA

Table 1. The grid refinement analysis of the area and volume calculation for the sphere of radius 2.

Grid size	Area	Error	Order	Volume	Error	Order
1.0	43.755	6.51		47.00	13.490	
0.5	48.824	1.44	2.18	35.28	1.865	2.85
0.25	49.839	0.43	1.76	33.80	0.287	2.70
0.125	50.193	0.07	2.56	33.57	0.058	2.30
0.0625	50.243	0.02	1.71	33.52	0.007	3.05
Exact Value	50.28			33.52		

test set, which contains 937 molecules with the numbers of atoms ranging from several hundreds to around nine thousand.

Figure 14 illustrates the convergence of the numerical schemes eqs. (16) and (17) for computing the surface area and enclosed volume. The convergence is measured in a relative sense. We compute the relative errors of the numerical surface area and volume compared to those obtained at the finest grid utilized in the calculation, 0.2 Å in our tests. Charts (a) and (b) in Figure 14 effectively verify the convergence of both schemes for a large set of 937 biomolecules.

To confirm the accuracy of the above area and volume calculation schemes, the consistency of the surface area and enclosed volume calculated by ESES and the MSMS software is presented in Figure 15. The grid spacing of 0.2 Å is employed in eqs. (16) and (17) for the area and volume calculations on the Cartesian grid. Meanwhile, surface density 100 is used for the MSMS surface volume computations, and analytical surface area calculated by the ESES software are employed for the comparison. Due to the lack of robustness in the MSMS surface generation, we perform a surface area and volume consistency comparison on a subset of the PBSA test set, containing 740 molecules for which MSMS can successfully generate surfaces at density 100. This subset is listed in the Appendix.

The results depicted in Figure 15 indicate an excellent consistency between results calculated by MSMS and ESES. Based on the test results shown in Table 2, it is seen that at surface density 100 the volume calculation is extremely accurate.

The whole surface area is usually employed for a crude model of the area contribution in nonpolar solvation free energy:

$$G_{\mathrm{np}}^{\mathrm{area}}=\gamma \mathrm{Area},$$

with γ being surface tension. A more accurate area model for modeling the nonpolar solvation effects is based on the atomic surface areas, in which the different types of atoms admit different surface tensions. As such, one can model the area contribution to the nonpolar solvation free energy by

$$G_{\mathrm{np}}^{\mathrm{area}}=\sum_{\alpha}\gamma_{\alpha}\sum_{i\in\alpha}\mathrm{Area}_i, \tag{18}$$

where γ_α is the atomic surface tension of the αth type of atoms and Area_i is the atomic surface area of the ith atom. Therefore, in addition to the surface area calculation, the ESES software also provides the atomic surface area calculation.

In the SES, each convex patch of the contact surface is contributed by a single atom. Each toric patch is generated due to the contact of the probe with two atoms simultaneously. Each concave patch is associated with three atoms. This fact is inherited to the surface area partitioning. Note that in our area formula eq. (16), the whole area is the cumulative contribution from each intersection point. In the atomic area partitioning, we distribute the area associated with each surface intersection region to the associated atoms. For a small surface patch of area $A_0(\mathbf{r})$ at position $\mathbf{r}$, and the partition of the area is based on the following criteria:

- If the patch is associated with only one atom, the whole area $A_0(\mathbf{r})$ is assigned to the corresponding atom.
- Assume that the patch of area $A_0(\mathbf{r})$ is associated with N atoms of positions $\mathbf{r}_i$ and radii r_i, $i=1,2,\cdots,N$. We define a set of squared distances as $d_i=\sqrt{||\mathbf{r}-\mathbf{r}_i||^2-r_i^2}$. We partition the area into fractions $A_{0i}=w_iA_0(\mathbf{r})$ and the nonnegative weights satisfy $\sum_{i=1}^{N}w_i=1$. There are two ways to assign weights. The winner-take-all weights are given by

$$w_i=\begin{cases}1 & d_i\le d_j,\ \forall j\neq i\\ 0 & \text{otherwise.}\end{cases} \tag{19}$$

Alternatively, we can choose

$$w_i=\frac{1/d_i^2}{\sum_{j=1}^{N}1/d_j^2} \tag{20}$$

or

$$w_i=\frac{-d_i+\sum_{j}^{N}d_j}{(N-1)\sum_{j}^{N}d_j} \tag{21}$$

The idea for this area partitioning is similar to that of the power diagram.[75] The winner-take-all strategy is used in the present work for atomic surface area distribution.

Table 2. The area and enclosed volume calculations for the sphere of radius 2 at different MSMS surface densities.

Surface density	Area	Error	Order	Volume	Error	Order
10	49.928	0.35		33.09	0.43	
20	50.108	0.17	1.03	33.32	0.21	1.07
40	50.185	0.09	0.86	33.42	0.11	0.90
80	50.227	0.05	0.84	33.46	0.06	0.90
160	50.245	0.04	0.60	33.49	0.03	0.80
Exact Value	50.28			33.52		

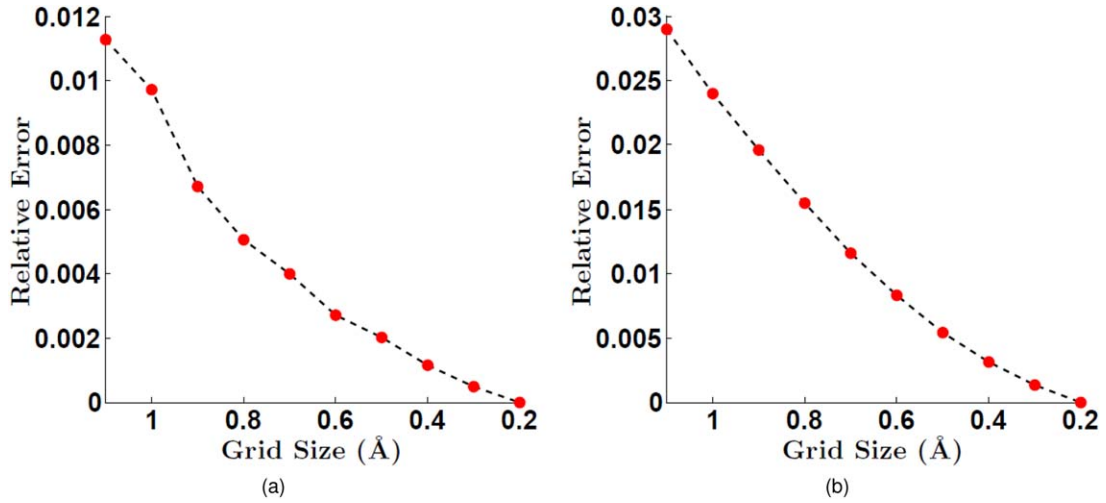

Figure 14. Convergence test of the surface area and enclosed volume for the PBSA test set ($N = 937$) compared to the results obtained at grid size 0.2 Å. a) Area; b) Volume. [Color figure can be viewed at wileyonlinelibrary.com]

Topological features

Loops and cavities are omnipresent in SESs. Usually these loops or cavities are related to the binding pockets or binding sites. Accurate and efficient algorithms for detecting the loops and cavities together with measuring the sizes of these topological features play an important role in the practical applications, including drug design. In this section, we provide an accurate and efficient numerical algorithm based on the homology theory[76–78] for the loop and cavity detection. Furthermore, we propose a level set method-based filtration to generate persistent bar codes which characterize the size of loops and cavities.

Loop and cavity detection. Homology group, which is an algebraic group associated to the topology of a manifold, provides an effective theoretical framework for computing loops and cavities on the manifold. The homology group constructed based on cubical complexes provides a practical methodology for computing the loops and cavities for the ESES. For details on homology theory in a cubical complex setting, readers are referred to Ref. [76].

In the following, all the homology computations are carried out by the Perseus persistent homology software,[79] which is an efficient persistent homology program that can handle both simplicial and cubical complexes.[80] ESES output provides a suitable input data for the Perseus software.

We consider a benchmark test example, a C_{60} molecule, to illustrate the ESES performance in loop and cavity generation.[78,81] Figure 16 depicts C_{60} generated at the atomic radius of 0.8 Å for all the Carbon atoms with the probe radius of 0.1 Å. The structure has 32 rings. Figure 17 provides the

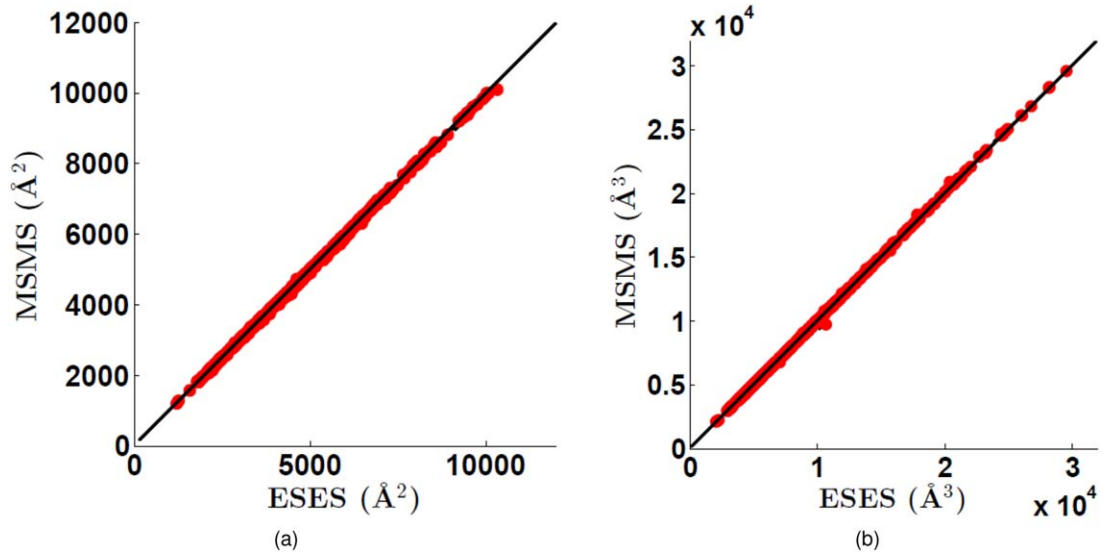

Figure 15. The comparison of areas and volumes calculated by MSMS and ESES software packages for a subset ($N = 740$) of the PBSA test set. The ESES results are generated with grid spacing 0.2 Å. a) For the area comparison, the correlation is 0.9999, and the best fitting line is $y = 0.9934x + 14.3307$; b) For the volume comparison, the correlation is 0.9999, and the best fitting line is $y = 1.0040x - 23.9577$. [Color figure can be viewed at wileyonlinelibrary.com]

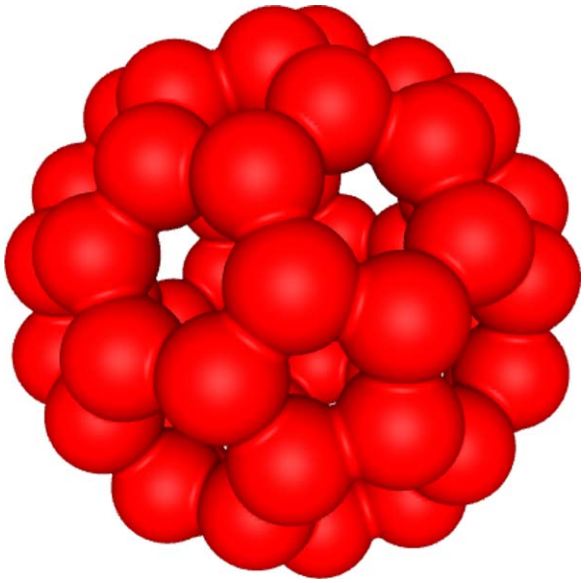

Figure 16. The SES for a C_{60} molecule. To illustrate the geometry, the probe radius is chosen as 0.1 Å, and the van der Waals radius of Carbon atoms is set to 0.8 Å. [Color figure can be viewed at wileyonlinelibrary.com]

number of loops generated by ESES at different grid sizes. When the grid spacing used for the surface generation is coarser than 0.6 Å, the numerical method cannot capture all the small loops, which are about 0.5 Å in diameter, in C_{60} molecule. However, when the grid spacing is finer than 0.6 Å, all the loops can be resolved. Note that persistent homology reports only 31 loops because one of the 32 loops can be expressed as a linear combination of all other loops.

Figure 18 shows the SES generated with probe radius 1.4 Å for some molecules from the PBSA test set. Their corresponding numbers of loops and cavities are also presented and calculated at the grid resolution of 0.3 Å.

The size of topological features: Persistence. The method for detecting the loops and cavities of the manifold formed by the molecular surface has been investigated in the previous part. Nevertheless, we also need an effective way to characterize the size of a loop and/or a cavity.

Persistent homology theory provides a way to measure the sizes of the loops and cavities. The measure is called persistence in the filtration. We have introduced a time propagation approach for generating the persistent homology filtration in our recent work.[81] In the present work, we replace the geometric flow propagation in our earlier work by a constant speed propagation so as to uniformly measure the sizes of different loops or cavities.

To define the persistent homology, we need a filtration, that is, a complex K together with nested sequence of subcomplexes $\{K^i\}_{0\le i\le n}$, such that:

$$\varnothing = K^0 \subset K^1 \subset \cdots \subset K^n = K.$$

Each subcomplex K^i in the filtration has an associated chain group C_k^i, cycle group Z_k^i and boundary group B_k^i, for $\forall i \geq 1$, and thus one has the following definition.

Definition 1. *The p-persistent of kth homology group K^i is*:

$$H_k^{i,p} = Z_k^i / \left(B_k^{i+p} \cap Z_k^i\right),$$

where $H_k^{i,p}$ captures the topological features of the filtrated complex that persists for at least p steps in the filtration.

To measure the size of loops and cavities of the manifold formed by the SES, a minor modification is needed in the previous definition of the filtration. The modified filtrated sequence of complexes is defined as:

$$\Omega_m = K_0 \subset K_1 \subset \cdots \subset K_n = K,$$

where K_0 is the manifold formed by the SES. K is the evolved manifold when all loops and cavities of K_0 are filled.

To measure the persistence of the loops and cavities in the SES, the level set method is employed for the surface propagation with constant speed.

For the molecular surface Γ, we can give it a level set representation $\psi(\mathbf{r})$, such that:

$$\psi(\mathbf{r}) \begin{cases} >0, & \mathbf{r} \in \Omega_s \text{ outside the molecular surface,} \\ =0, & \mathbf{r} \in \Gamma \text{ on molcular surface,} \\ <0, & \mathbf{r} \in \Omega_m \text{ inside the molecular surface.} \end{cases} \quad (22)$$

$\psi(\mathbf{r})$ can be chosen as the signed distant function to the molecular surface Γ.

To propagate the surface with a given velocity in the Eulerian representation, we introduce the time variable t to the level set function, that is, $\psi(\mathbf{r}, t)$. By taking the derivative with respect to t of the SES level set function $\psi(\mathbf{r}, t) = 0$, one has

$$\frac{\partial \psi}{\partial t} + \nabla \psi \cdot \frac{\partial \mathbf{r}}{\partial t} = 0.$$

Note that $\frac{\partial \mathbf{r}}{\partial t}$ is exactly the surface propagation velocity, denoted as $\mathbf{v}$. Projecting the velocity on to the outer normal

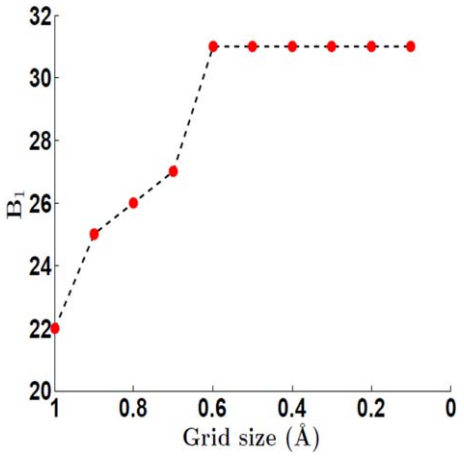

Figure 17. The number of loops calculated at different grid sizes for the C_{60} solvent excluded surface. [Color figure can be viewed at wileyonlinelibrary.com]

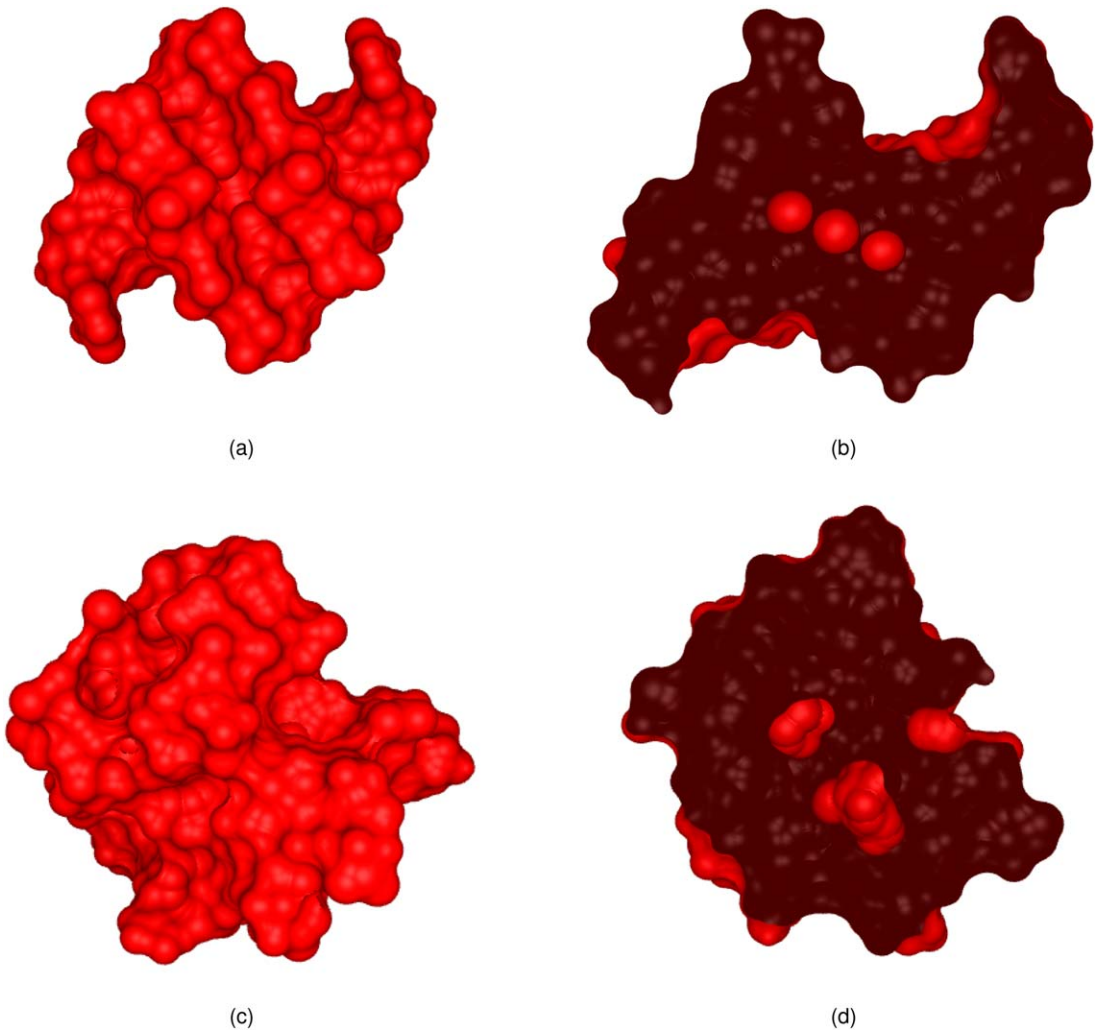

Figure 18. The solvent excluded surfaces and their topologies of two biomolecules. a) The ESES result for protein 2AVH with three loops and one cavity. b) A cross-section of protein 2AVH showing the loops. c) The ESES result for protein 1afo-A.p22 with two loops and two cavities. d) A cross-section of protein 1af8 showing the loops and cavities. [Color figure can be viewed at wileyonlinelibrary.com]

direction of the surface $\frac{\nabla\psi}{|\nabla\psi|}$, one has the projected velocity along the normal direction $v_N \doteq \mathbf{v}\cdot\frac{\nabla\psi}{|\nabla\psi|}$.

Finally, the level set equation for describing the surface propagation along the outer normal direction can be written as:

$$\frac{\partial\psi}{\partial t}+v_N|\nabla\psi|=0, \tag{23}$$

which is a Hamilton–Jacobi equation. The level set eq. (23) is solved by the simple upwind scheme with periodic boundary condition. For detailed theoretical description and numerical

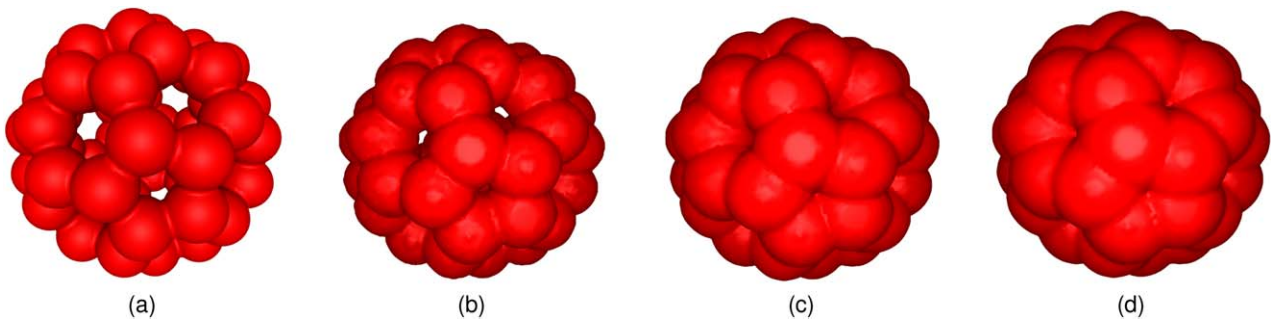

Figure 19. The level set representation of the SES for C_{60} molecule. a) the level set representation of the SES; b)–d) represent the frames of the evolution at time 20, 40, and 60, respectively. [Color figure can be viewed at wileyonlinelibrary.com]

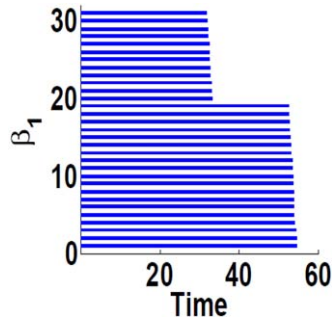

Figure 20. The persistence diagram of the loops in the SES of the C_{60}. [Color figure can be viewed at wileyonlinelibrary.com]

implementation of the level set method, readers are referred to Refs. [82,83].

First, we consider the benchmark test, the SES of C_{60} generated by the same parameters as aforementioned. The loops on the SES can be classified into two categories, that is, the pentagon and hexagon loops. These two types of loops are of different sizes. In the following, we solve the level set equation at grid resolution 0.1 Å in the spatial domain discretization with the velocity 0.1 Å per unit time along the outer normal direction. To ensure the Courant–Friedrichs–Lewy (CFL) condition, the resolution of 0.002 is employed in the time discretization. Figure 19 depicts some frames of the evolution procedure of the C_{60} SES driven by the level set equation. Here, different frames show that the SES is propagated along the outer normal direction in a constant speed. After some time, both types of loops are closed. The persistent time of these types of loops reflects the size of loops, which is proportional to the size of loops since the surface grows with a constant speed.

Figure 20 illustrates the persistence of pentagonal and hexagonal loops on the SES under the filtration constructed by the level set equation. In persistence bar codes, there are obviously two types of loops, where the pentagonal loops are of shorter persistence while the hexagonal loops are of longer persistence. The numerical result is consistent with the ground truth.

After the above validation on the C_{60} molecular surface, we apply the above level set method-based persistent homology theory to biomolecules to investigate the sizes of loops and cavities of the corresponding manifold enclosed by the SESs. All the implementation is carried out at the grid resolution of 0.3 Å in spatial discretization and 0.01 at the temporal discretization which guarantees the stability of the numerical integrator. The surface is propagated with a constant speed 0.2 Å per unit time along the outer normal direction.

We study the persistence of the loops and cavities of protein 1clh-0.p22. The previous homology theory predicts that the SES generated with probe radius 1.4 Å with the Amber force field has five loops and five cavities. Figure 21 shows the frames of the growing surfaces at time 0, 25, 50, and 100, respectively. Intuitively, during the surface propagation, the surface points should offset uniformly. Our numerical results are consistent with this intuition.

The left and right charts of Figure 22 show the persistence of the loops and cavities, respectively on the molecular surface of protein molecule 1clh-0.p22. The persistence barcode of the loops shows that there are two quite short-lived loops, that is, two tiny loops on the surface, and one middle-sized loop, together with two long-persistent loops. The largest loop has a persistent length of 73, which corresponds to a diameter of 14.6 Å. The cavity persistent barcode demonstrates that there is no short-lived cavity in the SES manifold. Five cavities all have relatively long persistence. The largest cavity has persistence length around 93, which corresponds to a cavity length of 18.6 Å.

Limitations and future improvements

Our current implementation is optimized for robustness but not efficiency. There are several developments that we are investigating. For the second stage, marking all grid points inside SAS as uncertain is overkilling. We postulate that marking grid points as "uncertain" inside the union of van der Waals balls, visibility spheres, and visibility tetrahedra is sufficient. We are working on a mathematical proof and numerical verifications. For the third stage, the intersection should be performed on the potential patches that are on the reduced surface, and we are developing a robust construction of the reduced surface structure. Finally, the locality of all of our calculation indicates that we should be able to parallelize the entire procedure.

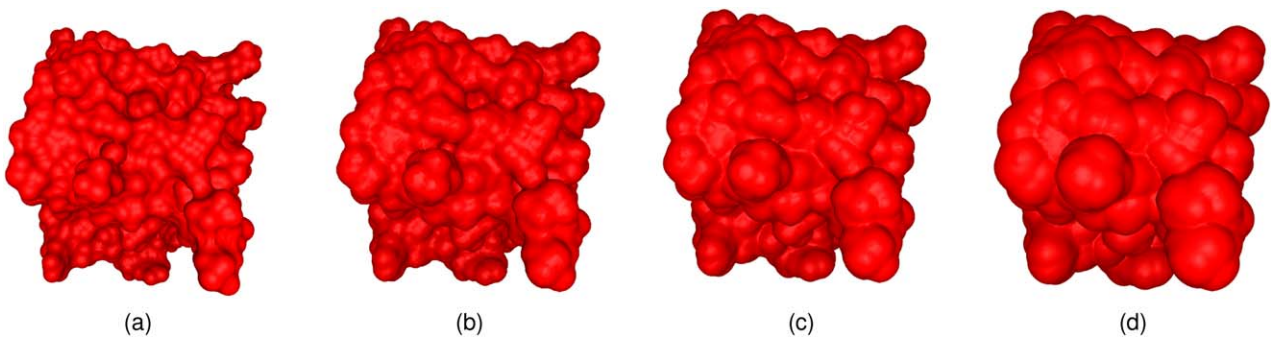

Figure 21. The level set representation of the SES for protein 1clh-0.p22. a) The level set representation of the original SES; b)–d) Frames of propagated surfaces at time 25, 50, and 100, respectively. [Color figure can be viewed at wileyonlinelibrary.com]

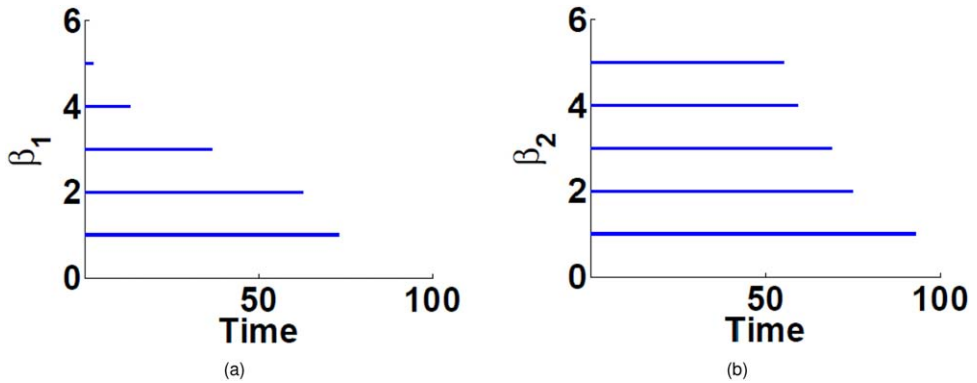

Figure 22. The persistence diagrams of the loops (Left chart) and cavities (Right chart) in the SES of protein 1clh-0.p22, respectively. [Color figure can be viewed at wileyonlinelibrary.com]

Concluding Remarks

Solvent excluded surface (SES) is the most popular surface definition in computational biophysics and molecular biology for biomolecular modeling and simulation. Existing SES software packages, such as MSMS,[34] typically provide SESs in the Lagrangian representation. For applications in implicit solvent models, one needs to convert the triangular SES into the Eulerian representation. Additionally, quality of MSMS depends on the density selected and the method might not work well at all required densities for certain molecules. Therefore, it is desirable to generate analytical SESs directly on the Cartesian mesh. This work offers a software package, called Eulerian solvent excluded surface (ESES), for the construction of SESs on the Cartesian mesh.

We generate analytical SESs based on Connolly's algorithm,[31] which divides an SES into three types of patches: convex patches, saddle patches, and concave patches. We immerse the analytical SES into the Euclidean space $\mathbb{R}^3$ and describe the surface or interface by its intersecting coordinates with the Cartesian mesh lines and associated normal directions at all the intersecting points.

The proposed ESES software is validated by a large number of benchmark tests, including morphological visualization, electrostatic solvation analysis, surface area and enclosed volume calculations, and topological feature analysis and characterization. We utilize the Amber PBSA test set in our validation. The MSMS software is employed for comparison. In the morphological visualization, it is shown that ESES successfully generate correct morphology while MSMS does not always work for the Amber test set at all densities. ESES also provides second-order accurate estimates for biomolecular surface area and enclosed volume. It is found that electrostatic solvation energies computed using the ESES are in close consistence with those calculated based on MSMS. A unique feature of the present software is that it provides atomic surface areas, which can be used for atomic modeling of nonpolar solvation free energies. Finally, we introduce homology theory to accurately detect topological features, namely, loops and cavities on/in SESs. A novel level set-based filtration is proposed to measure the sizes of loops and cavities.

The present ESES software will be improved on a few aspects. First, a better method for rendering the surface needs to be implemented for the surface visualization. Second, compared to the MSMS software, ESES is more robust but less efficient. Therefore, speeding up the ESES from multiple aspects is under our investigation.

Acknowledgment

BW thanks Dr. Zhan Chen and Dr. Zhixiong Zhao for discussions.

APPENDIX : THE PDB IDS OF THE AMBER TEST SET THAT ARE USED FOR THE SURFACE AREA AND VOLUME COMPARISON WITH MSMS SOFTWARE

100D, 109D, 110D, 118D, 126D, 127D, 131D, 137D, 138D, 151D, 152D, 157D, 158D, 160D, 161D, 165D, 181D, 182D, 184D, 190D, 192D, 196D, 198D, 1A2E, 1BD1, 1BNA, 1CSL, 1D10, 1D11, 1D12, 1D13, 1D15, 1D23, 1D32, 1D36, 1D37, 1D38, 1D39, 1D43, 1D44, 1D45, 1D46, 1D48, 1D49, 1D54, 1D56, 1D57, 1D58, 1D63, 1D67, 1D78, 1D79, 1D88, 1D8G, 1D8X, 1D96, 1DA0, 1DA9, 1DC0, 1DCG, 1DJ6, 1DL8, 1DN8, 1DNO, 1DNS, 1DNT, 1DNX, 1DNZ, 1DOU, 1DQH, 1EHV, 1EN3, 1EN8, 1EN9, 1ENE, 1ENN, 1EVP, 1F27, 1FD5, 1FDG, 1FMQ, 1FMS, 1FN2, 1FQ2, 1FTD, 1G4Q, 1I0T, 1I1P, 1I7J, 1ICG, 1ICK, 1ID9, 1IDW, 1IH1, 1IHA, 1IKK, 1IMR, 1IMS, 1JGR, 1JO2, 1JRN, 1JTL, 1K9G, 1KCI, 1KD3, 1KD4, 1KD5, 1L1H, 1L2X, 1L4J, 1LJX, 1M69, 1M6F, 1M6G, 1M6R, 1M77, 1MF5, 1MSY, 1NLC, 1NQS, 1NT8, 1NVN, 1NVY, 1O0K, 1OFX, 1OSU, 1P20, 1P4Y, 1P4Z, 1P79, 1PFE, 1PJG, 1PJO, 1Q9A, 1QCU, 1QYK, 1QYL, 1R68, 1RQY, 1RXB, 1S23, 1S2R, 1SGS, 1SK5, 1T0E, 1U8D, 1UB8, 1UE4, 1V9G, 1VJ4, 1VS2, 1VZK, 1WOE, 1WQY, 1XA2, 1XCS, 1XCU, 1XJX, 1XJY, 1XPE, 1XVK, 1XVN, 1XVR, 1Z3F, 1Z8V, 1ZEV, 1ZEX, 1ZEY, 1ZEZ, 1ZF0, 1ZF1, 1ZF2, 1ZF3, 1ZF4, 1ZF5, 1ZF6, 1ZF7, 1ZF8, 1ZF9, 1ZFA, 1ZFB, 1ZFC, 1ZFF, 1ZFG, 1ZNA, 1ZPH, 1ZPI, 200D, 212D, 215D, 220D, 221D, 222D, 224D, 232D, 234D, 235D, 236D, 240D, 241D, 243D, 245D, 248D, 251D, 255D, 258D, 259D, 260D, 272D, 276D, 279D, 284D, 288D, 292D, 293D, 295D, 2A43, 2A7E, 2ADW, 2AVH, 2B0K, 2B1B, 2B2B, 2B3E, 2D47, 2D94, 2D95, 2DCG, 2DES, 2DYW, 2DZ7, 2EES, 2EET, 2EEV, 2F8W, 2G32, 2G9C, 2GB9, 2GPM, 2GQ4, 2GQ5, 2GQ6,

2GQ7, 2GVR, 2GWA, 2GYX, 2HBN, 2HTO, 2I2I, 2I5A, 2IE1, 2O1I, 2OE5, 2OE8, 2OIY, 2OKS, 2PKV, 2PL4, 2PL8, 2PLB, 2PLO, 2PWT, 2Q1R, 2QEK, 2R22, 2V6W, 2V7R, 2VAL, 308D, 310D, 312D, 314D, 315D, 317D, 331D, 332D, 334D, 336D, 348D, 349D, 351D, 354D, 355D, 360D, 362D, 368D, 369D, 370D, 371D, 385D, 386D, 393D, 394D, 395D, 396D, 397D, 398D, 399D, 3C2J, 3C44, 3CGP, 3CGS, 3CJZ, 3CZW, 3D0M, 3DNB, 3ERU, 3EUM, 413D, 414D, 420D, 423D, 428D, 431D, 432D, 434D, 435D, 437D, 439D, 440D, 441D, 442D, 443D, 452D, 453D, 455D, 463D, 465D, 466D, 472D, 473D, 476D, 477D, 479D. 480D, 482D, 483D, 485D, 5DNB, 7BNA, 9BNA, 9DNA, 1a93-B.p22, 1ac0-0.p22, 1aca-0.p22, 1aci-0.p22, 1adn-0.p22, 1adr-0.p22, 1afo-A.p22, 1agg-0.p22, 1ah9-0.p22, 1ahl-0.p22, 1aiw-0.p22, 1aj3-0.p22, 1ajw-0.p22, 1ajy-A.p22, 1aml-0.p22, 1ao8-0.p22, 1aoy-0.p22, 1ap0-0.p22, 1ap7-0.p22, 1apc-0.p22, 1aps-0.p22, 1aq5-A.p22, 1auu-A.p22, 1aw0-0.p22, 1aw6-0.p22, 1awj-0.p22, 1axh-0.p22, 1axj-0.p22, 1ayj-0.p22, 1az6-0.p22, 1b1a-0.p22, 1b22-A.p22, 1b4r-A.p22, 1b8w-A.p22, 1b91-A.p22, 1b9p-A.p22, 1b9u-A.p22, 1bak-0.p22, 1bal-0.p22, 1baq-0.p22, 1bb8-0.p22, 1bbg-0.p22, 1bby-0.p22, 1bct-0.p22, 1bdc-0.p22, 1bds-0.p22, 1bfm-A.p22, 1bgf-0.p22, 1bgk-0.p22, 1bh4-0.p22, 1bhu-0.p22, 1bi6-H.p22, 1bip-0.p22, 1bj8-0.p22, 1bjx-0.p22, 1bku-0.p22, 1bl1-0.p22, 1blj-0.p22, 1blr-0.p22, 1bm4-A.p22, 1bmr-0.p22, 1bmx-0.p22, 1bno-0.p22, 1bnr-0.p22, 1bo0-0.p22, 1bo9-A.p22, 1boe-A.p22, 1bpv-0.p22, 1brv-0.p22, 1brz-0.p22, 1bve-A.p22, 1bw6-A.p22, 1bwx-0.p22, 1bxd-A.p22, 1byy-A.p22, 1bzg-0.p22, 1bzk-A.p22, 1c01-A.p22, 1c05-A.p22, 1c2n-0.p22, 1c3y-A.p22, 1c4e-A.p22, 1c55-A.p22, 1c5e-A.p22, 1c75-A.p22, 1c7k-A.p22, 1c7u-A.p22, 1c9q-A.p22, 1cch-0.p22, 1ccm-0.p22, 1cdb-0.p22, 1cdq-0.p22, 1ce4-A.p22, 1cf4-B.p22, 1cfe-0.p22, 1chc-0.p22, 1chl-0.p22, 1ck2-A.p22, 1cl4-A.p22, 1cmr-0.p22, 1cn2-0.p22, 1co4-A.p22, 1cou-A.p22, 1cur-0.p22, 1cw5-A.p22, 1cww-A.p22, 1cwx-A.p22, 1cye-0.p22, 1cyu-0.p22, 1d1h-A.p22, 1d6g-A.p22, 1d8b-A.p22, 1d8j-A.p22, 1daq-A.p22, 1dbf-A.p22, 1ddf-0.p22, 1de1-A.p22, 1dec-0.p22, 1def-0.p22, 1dfe-A.p22, 1dfs-A.p22, 1dgn-A.p22, 1dgq-A.p22, 1dip-A.p22, 1dl0-A.p22, 1dl6-A.p22, 1dlx-A.p22, 1dmc-0.p22, 1dny-A.p22, 1dp3-A.p22, 1dp7-P.p22, 1dpu-A.p22, 1dqb-A.p22, 1dqc-A.p22, 1dro-0.p22, 1dtv-A.p22, 1du2-A.p22, 1du6-A.p22, 1dv0-A.p22, 1dvh-0.p22, 1dwm-A.p22, 1dx7-A.p22, 1dx8-A.p22, 1dxz-A.p22, 1dz7-A.p22, 1e01-A.p22, 1e0a-B.p22, 1e0e-A.p22, 1e0h-A.p22, 1e0l-A.p22, 1e17-A.p22, 1e29-A.p22, 1e2b-0.p22, 1e4u-A.p22, 1e53-A.p22, 1e5g-A.p22, 1e5u-I.p22, 1e68-A.p22, 1e7l-A.p22, 1e88-A.p22, 1e8l-A.p22, 1e8r-A.p22, 1eci-A.p22, 1eds-A.p22, 1edv-A.p22, 1ef4-A.p22, 1egx-A.p22, 1eh2-0.p22, 1ehj-A.p22, 1ehs-0.p22, 1ehx-A.p22, 1eik-A.p22, 1eit-0.p22, 1ej5-A.p22, 1ekt-A.p22, 1elk-A.p22, 1emw-A.p22, 1enw-A.p22, 1eqo-A.p22, 1erc-0.p22, 1erd-0.p22, 1erx-A.p22, 1esx-A.p22, 1euw-A.p22, 1ev0-A.p22, 1ews-A.p22, 1exe-A.p22, 1ezg-A.p22, 1f0z-A.p22, 1f3c-A.p22, 1f41-A.p22, 1f53-A.p22, 1f81-A.p22, 1f8p-A.p22, 1fa3-A.p22, 1faf-A.p22, 1fbr-0.p22, 1fct-0.p22, 1fje-B.p22, 1fjk-A.p22, 1fjn-A.p22, 1fm0-D.p22, 1fmh-A.p22, 1fp0-A.p22, 1fqq-A.p22, 1fr3-A.p22, 1fre-0.p22, 1fu9-A.p22, 1fvl-0.p22, 1fwo-A.p22, 1fwp-0.p22, 1fwq-A.p22, 1fyb-A.p22, 1fyc-0.p22, 1fyj-A.p22, 1g1e-B.p22, 1g25-A.p22, 1g26-A.p22, 1g2h-A.p22, 1g4f-A.p22, 1g5v-A.p22, 1g66-A.p22, 1g6e-A.p22, 1g7d-A.p22, 1g7e-A.p22, 1g84-A.p22, 1g9l-A.p22, 1gab-0.p22, 1gd0-A.p22, 1ggw-A.p22, 1gh9-A.p22, 1ghc-0.p22, 1gp8-A.p22, 1gw3-0.p22, 1gyf-A.p22, 1h8c-A.p22, 1ha9-A.p22, 1hbw-A.p22, 1hdo-A.p22, 1hev-0.p22, 1hhn-A.p22, 1hks-0.p22, 1hnr-0.p22, 1hp8-0.p22, 1hre-0.p22, 1hs7-A.p22, 1hsq-0.p22, 1hx2-A.p22, 1hyi-A.p22, 1hyw-A.p22, 1hzn-A.p22, 1i1s-A.p22, 1i25-A.p22, 1i27-A.p22, 1i5h-W.p22, 1i5j-A.p22, 1ica-0.p22, 1ihv-A.p22, 1iie-A.p22, 1il6-0.p22, 1imt-0.p22, 1inz-A.p22, 1ioj-0.p22, 1irf-0.p22, 1itf-0.p22, 1jhb-0.p22, 1joy-A.p22, 1jun-A.p22, 1jwe-A.p22, 1kdx-A.p22, 1khm-A.p22, 1kjs-0.p22, 1kla-A.p22, 1koe-0.p22, 1lea-0.p22, 1lre-0.p22, 1lyp-0.p22, 1mgs-A.p22, 1mkc-A.p22, 1mkn-A.p22, 1mnt-A.p22, 1myf-0.p22, 1ncs-0.p22, 1noe-0.p22, 1ns1-A.p22, 1ntc-A.p22, 1olg-A.p22, 1om2-A.p22, 1paa-0.p22, 1pce-0.p22, 1pcn-0.p22, 1peh-0.p22, 1pfl-0.p22, 1pfs-A.p22, 1pls-0.p22, 1pmc-0.p22, 1pnb-A.p22, 1pnb-B.p22, 1pon-B.p22, 1psm-0.p22, 1qa5-A.p22, 1qck-A.p22, 1qdp-0.p22, 1qfd-A.p22, 1qfq-B.p22, 1qgp-A.p22, 1qhk-A.p22, 1qjo-A.p22, 1qk6-A.p22, 1qk7-A.p22, 1qk9-A.p22, 1qkf-A.p22, 1qkl-A.p22, 1qlo-A.p22, 1qn0-A.p22, 1qqi-A.p22, 1qqv-A.p22, 1qry-A.p22, 1qsv-A.p22, 1qtn-A.p22, 1qtn-B.p22, 1qto-A.p22, 1r2a-A.p22, 1rax-A.p22, 1rch-0.p22, 1rcs-A.p22, 1res-0.p22, 1rge-A.p22, 1rie-0.p22, 1rip-0.p22, 1rot-0.p22, 1rpr-A.p22, 1rxr-0.p22, 1sap-0.p22, 1scy-0.p22, 1sgg-0.p22, 1svf-B.p22, 1tba-A.p22, 1tba-B.p22, 1tbd-0.p22, 1tbn-0.p22, 1tfb-0.p22, 1tle-0.p22, 1tns-0.p22, 1tof-0.p22, 1tpm-0.p22, 1trl-A.p22, 1tsg-0.p22, 1u2f-A.p22, 1urk-0.p22, 1utr-A.p22, 1uwo-A.p22, 1uxc-0.p22, 1vgh-0.p22, 1vpu-0.p22, 1vre-A.p22, 1wdb-0.p22, 1whi-0.p22, 1xbl-0.p22, 1xnb-0.p22, 1xpa-0.p22, 1yui-A.p22, 1zta-0.p22, 1zto-0.p22, 2a3d-A.p22, 2alc-A.p22, 2end-0.p22, 2ezm-0.p22, 2fmr-0.p22, 2gat-A.p22, 2gcc-0.p22, 2gva-A.p22, 2hgf-0.p22, 2hir-0.p22, 2if1-0.p22, 2ife-A.p22, 2ifo-0.p22, 2jhb-A.p22, 2lfb-0.p22, 2mrb-0.p22, 2ncm-0.p22, 2nlr-A.p22, 2orc-0.p22, 2pth-0.p22, 2ptl-0.p22, 2rel-0.p22, 2sob-0.p22, 2tmp-0.p22, 2tps-A.p22, 2vik-0.p22, 3chb-D.p22, 3lri-A.p22, 3phy-0.p22, 3rpb-A.p22, 3vub-0.p22, 3znf-0.p22, 1svq-0.p22, 1swu-A.p22

Keywords: solvent excluded surface · Eulerian representation · analytic surface · robustness

How to cite this article: B. Liu, B. Wang, R. Zhao, Y. Tong, G.-W. Wei. *J. Comput. Chem.* **2017**, *38*, 446–466. DOI: 10.1002/jcc.24682

[1] N. A. Baker, In Reviews in Computational Chemistry, Vol. *21*; K. B. Lipkowitz, R. Larter, T. R. Cundari, Eds.; Wiley: Hoboken, NJ, **2005**.
[2] M. E. Davis, J. A. McCammon, *Chem. Rev.* **1990**, *94*, 509.
[3] K. A. Sharp, B. Honig, *Annu. Rev. Biophys. Biophys. Chem.* **1990**, *19*, 301.
[4] B. Honig, A. Nicholls, *Science* **1995**, *268*, 1144.
[5] B. Roux, T. Simonson, *Biophys. Chem.* **1999**, *78*, 1.
[6] R. Jinnouchi, A. B. Anderson, *Phys. Rev. B* **2008**, *77*, 245417.
[7] G. Lamm. In Reviews in Computational Chemistry; K. B. Lipkowitz, R. Larter, T. R. Cundari, Eds.; Wiley: Hoboken, NJ, **2003**; pp. 147–366.
[8] F. Fogolari, A. Brigo, H. Molinari, *J. Mol. Recognit.* **2002**, *15*, 377.
[9] B. N. Dominy, C. L. Brooks, III, *J. Phys. Chem. B* **1999**, *103*, 3765.
[10] D. Bashford, D. A. Case, *Annu. Rev. Phys. Chem.* **2000**, *51*, 129.
[11] V. Tsui, D. A. Case, *J. Am. Chem. Soc.* **2000**, *122*, 2489.
[12] A. Onufriev, D. A. Case, D. Bashford, *J. Comput. Chem.* **2002**, *23*, 1297.
[13] E. Gallicchio, L. Y. Zhang, R. M. Levy, *J. Comput. Chem.* **2002**, *23*, 517.

[14] E. Cances, B. Mennucci, J. Tomasi, *J. Chem. Phys.* **1997**, *107*, 3032.
[15] V. Barone, M. Cossi, J. Tomasi, *J. Chem. Phys.* **1997**, *107*, 3210.
[16] M. Cossi, V. Barone, R. Cammi, J. Tomasi, *Chem. Phys. Lett.* **1996**, *255*, 327.
[17] B. S. Eisenberg, D. Chen, *Biophys. J.* **1993**, *64*, A22.
[18] M. G. Kurnikova, R. D. Coalson, P. Graf, A. Nitzan, *Biophys. J.* **1999**, *76*, 642.
[19] A. E. Cardenas, R. D. Coalson, M. G. Kurnikova, *Biophys. J.* **2000**, *79*, 80.
[20] B. Roux, T. Allen, S. Berneche, W. Im, *Q. Rev. Biophys.* **2004**, *7*, 1.
[21] Q. Zheng, G. W. Wei, *J. Chem. Phys.* **2011**, *134*, 194101.
[22] Q. Zheng, D. Chen, G. W. Wei, *J. Comput. Phys.* **2011**, *230*, 5239.
[23] M. K. Gilson, M. E. Davis, B. A. Luty, J. A. McCammon, *J. Phys. Chem.* **1993**, *97*, 3591.
[24] N. V. Prabhu, M. Panda, Q. Y. Yang, K. A. Sharp, *J. Comput. Chem.* **2008**, *29*, 1113.
[25] W. Geng, G. W. Wei, *J. Comput. Phys.* **2011**, *230*, 435.
[26] J. Warwicker, H. C. Watson, *J. Mol. Biol.* **1982**, *157*, 671.
[27] D. Petrey, B. Honig, *Methods Enzymol.* **2003**, *374*, 492.
[28] N. A. Baker, J. A. McCammon, In Structural Bioinformatics; P. Bourne, H. Weissig, Eds.; Wiley: New York, **2003**; pp. 427–440.
[29] F. M. Richards, *Annu. Rev. Biophys. Bioeng.* **1977**, *6*, 151.
[30] B. Lee, F. M. Richards, *J. Mol. Biol.* **1971**, *55*, 379.
[31] M. L. Connolly, *J. Appl. Crystallogr.* **1983**, *16*, 548.
[32] M. L. Connolly, *J. Mol. Graph.* **1985**, *3*, 19.
[33] S. Sridharan, A. R. Nicholls, B. Honig, *Biophys. J.* **1995**, *68*, 220A.
[34] M. F. Sanner, A. J. Olson, J. C. Spehner, *Biopolymers* **1996**, *38*, 305.
[35] S. Sridharan, A. R. Nicholls, B. Honig, *Biophys. J.* **1997**, *73*, 722.
[36] J. Ryu, R. Park, D. S. Kim, *Comput. Aided Des.* **2007**, *73*, 1042.
[37] W. Rocchia, S. Sridharan, A. Nicholls, E. Alexov, A. Chiabrera, B. Honig, *J. Comput. Chem.* **2002**, *23*, 128.
[38] J. Blinn, *ACM Trans. Graph.* **1982**, *1*, 235.
[39] B. S. Duncan, A. J. Olson, *Biopolymers* **1993**, *33*, 231.
[40] Q. Zheng, S. Y. Yang, G. W. Wei, *Int. J. Numer. Methods Biomed. Eng.* **2012**, *28*, 291.
[41] Z. Y. Yu, M. Holst, Y. Cheng, J. A. McCammon, *J. Mol. Graph. Model.* **2008**, *26*, 1370.
[42] M. Chen, B. Lu, *J. Chem. Theory Comput.* **2011**, *7*, 203.
[43] H. L. Cheng, X. Shi, *Comput. Geom.* **2009**, *42*, 196.
[44] K. L. Xia, K. Opron, G. W. Wei, *J. Chem. Phys.* **2013**, *139*, 194109.
[45] K. Opron, K. L. Xia, G. W. Wei, *J. Chem. Phys.* **2014**, *140*, 234105.
[46] S. Decherchi, W. Rocchia, *PLoS One* **2013**, *8*, e59744.
[47] G. W. Wei, Y. H. Sun, Y. C. Zhou, M. Feig, Molecular Multiresolution Surfaces, arXiv:math-ph/0511001v1, 2005; pp. 1–11.
[48] P. W. Bates, G. W. Wei, S. Zhao, The minimal molecular surface. *arXiv:q-bio/0610038v1*, [q-bio.BM], **2006**.
[49] P. W. Bates, G. W. Wei, S. Zhao, In The Minimal Molecular Surface; *Midwest Quantitative Biology Conference*, Mission Point Resort, Mackinac Island, MI, September 29–October 1, 2006.
[50] P. W. Bates, G. W. Wei, S. Zhao, *J. Comput. Chem.* **2008**, *29*, 380.
[51] P. W. Bates, Z. Chen, Y. H. Sun, G. W. Wei, S. Zhao, *J. Math. Biol.* **2009**, *59*, 193.
[52] J. Dzubiella, J. M. J. Swanson, J. A. McCammon, *Phys. Rev. Lett.* **2006**, *96*, 087802.
[53] G. W. Wei, *Bull. Math. Biol.* **2010**, *72*, 1562.
[54] Z. Chen, N. A. Baker, G. W. Wei, *J. Comput. Phys.* **2010**, *229*, 8231.
[55] Z. Chen, N. A. Baker, G. W. Wei, *J. Math. Biol.* **2011**, *63*, 1139.
[56] Z. Chen, S. Zhao, J. Chun, D. G. Thomas, N. A. Baker, P. B. Bates, G. W. Wei, *J. Chem. Phys.* **2012**, *137*, 084101.
[57] S. Zhao, *Int. J. Numer. Methods Biomed. Eng.* **2011**, *27*, 1964.
[58] S. Zhao, *J. Comput. Phys.* **2014**, *257*, 1000.
[59] D. Chen, Z. Chen, G. W. Wei, *Int. J. Numer. Method Biomed. Eng.* **2012**, *28*, 25.
[60] D. Chen, G. W. Wei, *J. Chem. Phys.* **2012**, *136*, 134109.
[61] B. Wang, G. W. Wei, *J. Chem. Phys.* **2015**, *143*, 134119.
[62] W. Geng, S. Yu, G. W. Wei, *J. Chem. Phys.* **2007**, *127*, 114106.
[63] S. N. Yu, W. H. Geng, G. W. Wei, *J. Chem. Phys.* **2007**, *126*, 244108.
[64] S. N. Yu, Y. C. Zhou, G. W. Wei, *J. Comput. Phys.* **2007**, *224*, 729.
[65] Y. C. Zhou, M. Feig, G. W. Wei, *J. Comput. Chem.* **2008**, *29*, 87.
[66] W. E. Lorensen, H. E. Cline, *Comput. Graph.* **1987**, *21*, 163.
[67] M. Krone, K. Bidmon, T. Ertl, *IEEE Trans. Comput. Graph.* **2009**, *15*, 1391.
[68] E. W. Weisstein, "Lemon" From MathWorld-A wolffram Web Resource. http://mathworld.wolfram.com/Lemon.html.
[69] K. Haugland, Polynomial Equation Solver. Available at: http://www.codeproject.com/Articles/552678/Polynomial-Equation-Solver, **2013**.
[70] J. Wang, Q. Cai, Y. Xiang, R. Luo, *J. Chem. Theory Comput.* **2012**, *8*, 2741.
[71] B. Wang, G. W. Wei, *J. Chem. Theory Comput.* (in press).
[72] Y. C. Zhou, Matched Interface and Boundary (mib) Method and its Applications to Implicit solvent modeling of biomolecules; Ph.D. Thesis., Michigan State University, **2006**.
[73] X. Feng, K. L. Xia, Y. Y. Tong, G. W. Wei, *J. Comput. Chem.* **2013**, *34*, 2100.
[74] P. Smereka, *J. Comput. Phys.* **2006**, *211*, 77.
[75] F. Aurenhammer, *ACM Comput. Surv.* **1991**, *23*, 345.
[76] T. Kaczynski, K. Mischaikow, M. Mrozek, Computational Homology; Springer-Verlag, New York, Inc., **2004**.
[77] H. Edelsbrunner, J. Harer, Computational Topology: An Introduction; American Mathematical Soc, **2010**.
[78] K. L. Xia, X. Feng, Y. Y. Tong, G. W. Wei, *J. Comput. Chem.* **2015**, *36*, 408.
[79] V. Nanda, Perseus: The Persistent Homology Software. Software Available at: http://www.sas.upenn.edu/vnanda/perseus
[80] K. Mischaikow, V. Nanda, *Discrete Comput. Geom.* **2013**, *50*, 330.
[81] B. Wang, G. W. Wei, *J. Comput. Phys.* **2016**, *305*, 276.
[82] S. Osher, R. P. Fedkiw, *J. Comput. Phys.* **2001**, *169*, 463.
[83] J. A. Sethian, Level Set Methods and Fast Marching Methods, 2nd ed., Vol. *3* of Monographs on Appl. *Comput*. Math.; Cambridge University Press: Cambridge, **1999**.

Received: 6 July 2016
Revised: 2 November 2016
Accepted: 9 November 2016
Published online on 4 January 2017