Analyzing Solvent in Protein Cavities: Methods and Application to Fatty Acid Binding Protein

Dirk Bakowies

Laboratory of Physical Chemistry, Swiss Federal Institute of Technology,
ETH Hönggerberg, CH-8093 Zürich, Switzerland.

1. Introduction

Many globular proteins share internal cavities as a common structural feature [1, 2]. Often these cavities are large enough to accommodate one or several water molecules. Crystal structure determinations show positions of well-ordered "structural" water molecules, but they usually fail to detect disordered or mobile ones [3, 4]. Hence many protein cavities appear to be void of solvent. Nuclear magnetic resonance (NMR) experiments sometimes indicate additional, more mobile, water molecules in protein cavities [3], and certain NMR techniques even provide estimates of their number, orientational disorder, and residence times [5, 6]. On the other hand, NMR experiments lack the atomistic detail of crystal structure determinations. In view of the anticipated importance of internal water for protein stability [2], ligand binding [7], and enzyme catalysis [8, 9], a more detailed knowledge of both structural and mobile solvent in protein cavities would certainly be desirable. Molecular dynamics (MD) simulations may contribute some of this information, as they provide a detailed and dynamic atomistic picture of a protein in its aqueous environment. The analysis of the MD trajectories obviously requires a suitable procedure to distinguish between internal and external solvent. Once such a classification is made, one may concentrate on analyzing water molecules found to be internal.

A number of methods have been developed in the past to identify interior cavities by analyzing the protein geometry or topology rather than monitoring ligands or solvent in the protein interior. Many of them are based on the concept of solvent-accessible [10] or molecular contact surfaces [11] which are obtained by spherical probes rolling over the molecule [12]. Cai *et al.* suggest to approximate Connolly's analytical description [12] by a triangular mesh built around the embracing ellipsoid which is iteratively deflated until it touches the original surface [13]. Alpha-shape theory [14] and related methods [15]

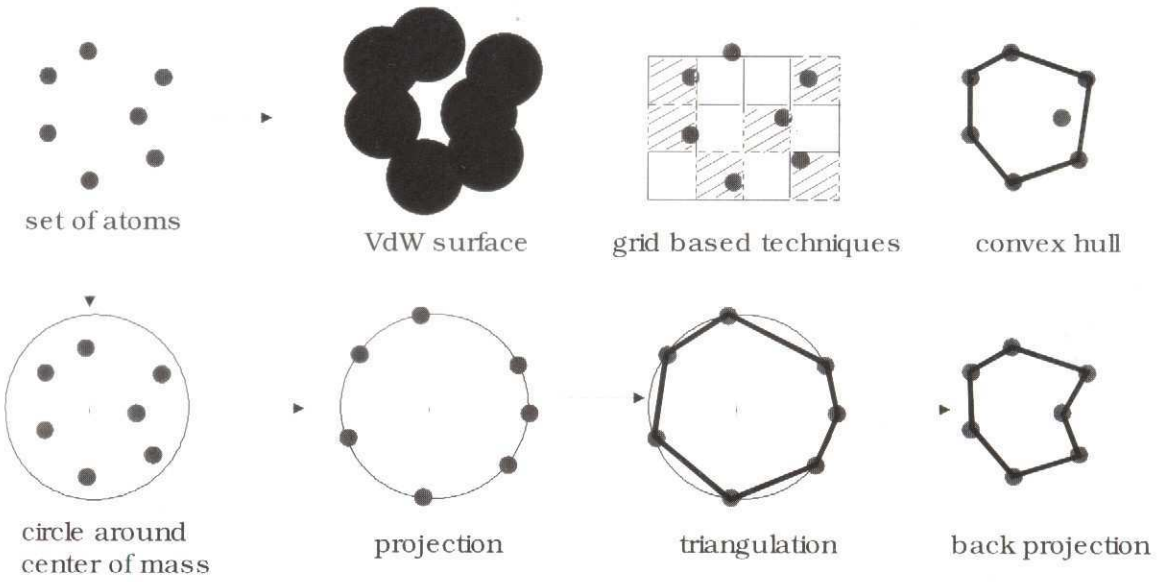


FIGURE 1. **Illustration (in 2D) of various methods used to define molecular surfaces and to identify cavities.** The lower row demonstrates the approach presented here.

also represent a molecule by van der Waals spheres, but they do not depend on auxiliary rolling probes. Instead they use Delaunay tessellations [16] of the molecule and distinguish between fully and partially occupied tetrahedra to identify both surface pockets [17] and interior cavities [18]. Alard and Wodak present a method to detect cavities by analyzing surfaces built from sets of interpenetrating spheres [19]. Procedures that first map the protein structure onto a grid and then use simple distance information between protein atoms and grid points to detect cavities have been proposed by Voorintholt *et. al.* [20] and by Levitt and Banaszak [21]. A modification using pattern recognition techniques and cellular logic operations has been proposed by Delaney [22]. While many of these approaches (see Fig. 1), and Connolly's algorithm in particular, have found wide-spread use in computer-graphics and related applications, they are of only limited use in the present context. If a protein contains an orifice or a channel connecting the interior cavity with the outside, most of the aforementioned approaches would - depending on input parameters like probe radii or grid sizes - capture these features as well. In the absence of a closed surface it appears impossible to unequivocally discriminate between 'interior' and 'exterior'.

In this paper we present a simple method to generate a closed surface which embraces a set of predefined atoms. The central idea is to construct a polyhedron whose vertices represent the set of chosen atoms and to find all solvent molecules enclosed by the polyhedron. The proposed procedure entails five steps (see Fig. 2):

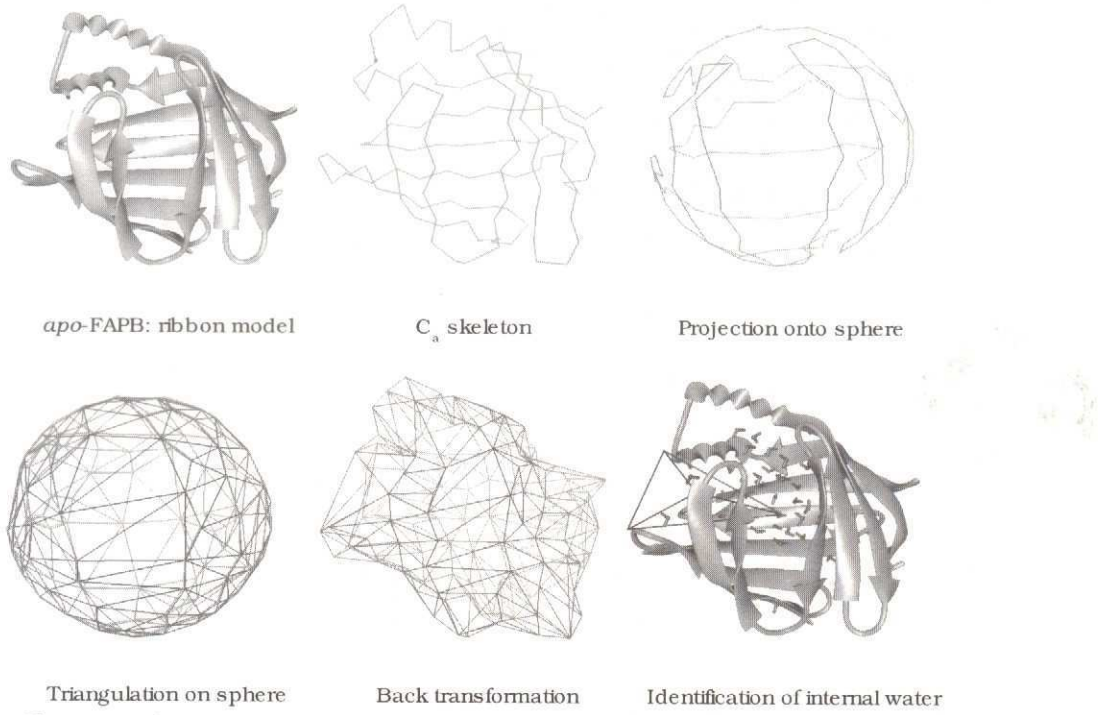


FIGURE 2. **Illustration of the procedure to triangulate the protein surface.** A color version can be found on the front cover of the 2001 C4 report book.

(1) Choice of a set of protein atoms which define the boundaries of the protein interior, the protein cavity, or more generally the protein region of interest.
(2) Projection of this set of atoms onto a sphere around their geometrical center.
(3) Triangulation of the spherical surface.
(4) Back transformation to the original coordinate system.
(5) Identification of solvent molecules inside the so-generated polyhedron.

This procedure is applied to 5 ns long MD trajectories of *apo-* and *holo-* fatty acid binding protein (FABP, see Fig. 2). The β-barrel type structure of FABP encompasses a large solvent-filled cavity, and only some of the internal water is expelled upon introduction of the palmitate ligand. In this report we concentrate on structural and dynamical aspects of internal water and on differences between the *apo* and *holo* forms of the protein. In particular, we focus on water densities, coordination, and interaction, and also discuss pathways of exchange with the bulk. A more comprehensive analysis may be found elsewhere. [23, 24]

Although the procedure proposed here has been developed to analyze solvent in the interior cavity of FABP, it should be useful for applications to other proteins as well. This includes all members of the lipid binding protein family [25] as well as some other transport proteins [26] which show a large solvent-filled cavity as a common feature.

Another example from the recent literature is the KcsA potassium channel which contains polarizable water believed to facilitate passage of ions through the low dielectric membrane [27, 28].

2. Methods

The objective is to find a simple description of a bounding surface for the protein interior. A well-defined volume is enclosed by this surface, and any solvent molecule contained in this volume is considered to be in the protein interior. Given a set of protein atoms as vertices, one obvious way to find a bounding surface is to form triangles from these vertices and to assemble them as faces of a polyhedron. Chemical knowledge may assist in a proper choice of vertices, and different choices may be used to identify various portions of the protein. Simple protein shapes such as the β-barrel of FABP can simply be represented by their backbone geometries, and the entire set of C_α atoms appears to be a reasonable choice.

Surface triangulation is a well-known subject in computational geometry, and Delaunay triangulation is the method of choice for two-dimensional problems [16]. Delaunay triangulation and its dual structure, the Voronoi diagram, are solutions to an important kind of nearest neighbor problem in geometry. Let $A = \{a_1, \cdots, a_n\}$ be a set of sites in the two-dimensional plane. Then the Voronoi diagram is composed of polygons around each site $a_i \in A$ enclosing the area of all points closer to a_i than to any other site a_j $(j \neq i)$. The set of points with two or more nearest sites a_i, a_j, $\cdots$ form the edges and vertices of Voronoi polygons. As the dual of Voronoi diagrams, Delaunay triangulation connects by lines any two sites a_i and a_j which border a common edge in the Voronoi representation.

Delaunay triangulation is well-suited for two-dimensional problems, but application to three-dimensional surfaces appears less straight-forward. Increasing dimensionality involves tessellation of three-dimensional space and affords tetrahedra rather than triangles. The bounding surface of the generated Delaunay complex is a convex hull enclosing all given vertices [16]. This might be used for our purposes, but vertices (protein atoms) hidden by the surface would then be discarded (see also Fig. 1). In applications to protein structure, concave crevices on the protein exterior would be counted inside, and the number of protein atoms defining the surface would be allowed to change as function of time. It is clear that there is no *a priori* definition of a surface containing *all given* vertices, and the difficulty apparently lies in finding a *suitable* definition *prior* to triangulation. We propose to use the analytical description of a simple surface which approximates the shape of the protein reasonably well and to project the set of chosen vertices onto this auxiliary surface. Spherical surfaces are reasonable approximations to all convex shapes

including the β-barrel of FABP. Hence for the purpose of this study, we choose to project main chain atoms onto a sphere, then perform the triangulation, back-transform the triangulated surface to the original coordinate frame, and finally identify water molecules inside the polyhedron bounded by this surface. The center of the sphere is always chosen as the center of the selected protein atoms. Figure 2 illustrates the outlined sequence of steps.

Reflecting the importance for computational geometry, there are a large number of algorithms available to perform Delaunay triangulation on planar surfaces [16,29]. Simple algorithms show $O(N_V^4)$ scaling with the number of sites or vertices (N_V), and much effort has been invested to improve this scaling behavior. Advanced techniques, based on the divide and conquer paradigm, for example, achieve a scaling as low as $O(N_V \log N_V)$. The computational efficiency comes at the expense of increased algorithmic complexity, however. On the other hand, the set of vertices used to represent the protein surface is quite limited, such that computational effort affords no unsurmountable burden. Hence we prefer to use a simple algorithm which can easily be applied to spherical surfaces, accepting for the moment its unfavorable scaling behavior. Later it will be seen that specific properties of spherical triangulations may be used for significant improvements which render the computer program being executed in trivial amounts of computer time.

2.1. **Triangulation procedure: implementation A.** There is one property of Delaunay triangles which lends itself to simple implementations both for planar and spherical surfaces: *Three sites (vertices) form a Delaunay triangle if and only if no other site is*

A

```
for each vertex (a_i) do
   for each vertex (a_j) do
      if not shared (a_i, a_j) then
         for each vertex (a_k) do
            flag := true
            if not (shared(a_i, a_k) or shared(a_j, a_k)) then
               l    := 0
               repeat
                  increment l
                  if (a_l) in circumcircle of (a_i, a_j, a_k) then
                     flag := false
                  endif
               until ( (not flag) or (l=N_v) )
               if (flag) then
                  add (a_i, a_j, a_k) to list of triangles
               endif
            endif
         end for
      endif
   end for
end for
```

FIGURE 3. **Core of the triangulation algorithm: implementation A.** The chart has been simplified for clarity and assumes that $i \neq j \neq k \neq l$. The value of `shared(`a_i`,`a_j`)` is `true` if the edge (a_i, a_j) is shared by two triangles already recorded and `false` otherwise.

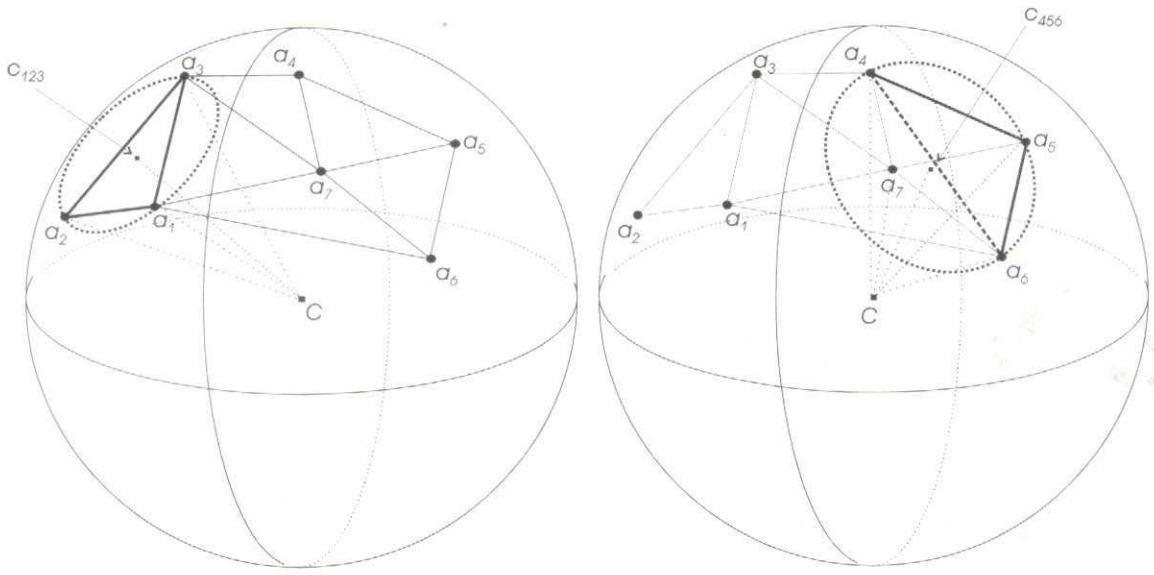


FIGURE 4. **Delaunay triangulation on spherical surfaces.** Based on the circumcircle criterion, the left panel shows a valid triangle (a_1,a_2,a_3) and the right panel shows an invalid triangle (a_4,a_5,a_6). See text for details.

contained in their circumcircle [16]. The algorithmic realization is immediately evident (see Fig. 3): For every triple of sites (a_i, a_j, a_k) compute the center c_{ijk} and radius r_{ijk} of the circumcircle and check if the distance between any other site a_l and c_{ijk} is smaller than r_{ijk}. If there is no such site a_l, then (a_i, a_j, a_k) is a Delaunay triangle. On spherical surfaces, three sites form a triangle if no other site is contained in the sphere segment lined by their circumcircle. The distance check is replaced by a comparison of the angle $\alpha_{ijk} = \angle(c_{ijk}, C, a_i) = \angle(c_{ijk}, C, a_j) = \angle(c_{ijk}, C, a_k)$ with all angles $\alpha_l = \angle(c_{ijk}, C, a_l)$ where C denotes the center of the sphere and $l \neq i \neq j \neq k$ (see Fig. 4).

The algorithm outlined above obviously scales as $O(N_V^4)$ with the number of vertices: It contains three nested loops considering all triples of vertices (a_i, a_j, a_k) which may potentially form triangles. A fourth loop runs over all remaining vertices, but finishes as soon as a vertex is found inside the circumcircle of (a_i, a_j, a_k). The computational effort is somewhat reduced by skipping edges which have already been identified in two triangles.

2.2. **Improvements: implementations B and C.** We shall refer to the above procedure as implementation A (Fig. 3) and introduce two major improvements which enhance computational efficiency significantly. First, knowledge about the total number of triangles permits early termination of the triangulation scheme (implementation B). Second, Delaunay triangulation is a local property by construction, allowing to restrict circumcircle tests to near neighbors (implementation C).

```
generate initial edge (a1, a2)                                    B,C
initialize stack with edge (a1, a2)
initialize NF
while (NF < 2NV-4)
   retrieve edge (ai, aj) from stack
   if not shared (ai, aj) then
      k    := 0
      repeat
         increment k
         flag := false
         if not (shared (ai, ak) or shared (aj, ak)) then
            flag := true
            l    := 0
            repeat
               increment l
               if (al) in circumcircle of (ai, aj, ak) then
                 flag := false
               endif
            until ( (not flag) or (l=NV) )
            if (flag) then
               add (ai, aj, ak) to list of triangles
               increment NF
               if not shared (ai, ak)  put (ai, ak) on stack
               if not shared (aj, ak)  put (aj, ak) on stack
               if (NF = 1)             put (ai, aj) on stack
            endif
         endif
      until (flag)
   endif
end while
```

FIGURE 5. **Core of the triangulation algorithm: implementations B and C.** See also caption of Fig. 3. The stack not only records Delaunay edges (a_i, a_j) but also the corresponding vertices they form triangles with to prevent double-counting of triangles. The initial edge is determined either from a given vertex and its nearest neighbor (implementation B) or within the large loop (implementation C). Finally, implementation C limits the inner two loops to near neighbors of a_i. See text for further details.

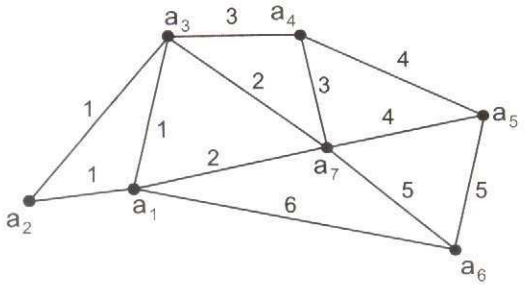


FIGURE 6. **Delaunay triangulation, implementations B and C.** Using the same example as in Figure 4, the graph illustrates how the algorithm walks through the triangulation. After identifying a new triangle, it proceeds with one of the newly found edges knowing that it must be shared by another triangle.

Euler's formula [30] relates the number of vertices (N_V), edges (N_E), and faces (N_F). It is valid for *any* polyhedron of sphere topology:

$$N_V + N_F = N_E + 2 \tag{1}$$

The number of vertices is known and simply equals the number of selected protein atoms. By construction, all polygonal faces of the polyhedron are triangles, hence every face is surrounded by exactly three edges, and every edge is shared by exactly two faces:

$$3N_F = 2N_E \tag{2}$$

Combining equations (1) and (2) yields the total number of triangles:

$$N_F = 2N_V - 4 \tag{3}$$

Triangulation can thus be terminated as soon as the pre-calculated number of triangles has been generated. By itself this measure would only reduce the number of circumcircle tests but not affect the time complexity of the algorithm. However, the additional knowledge that every edge is shared by exactly two triangles can not only be used as auxiliary check, it may also be implemented as a guide to proceed in the triangulation. Implementation B thus replaces the outer two loops by a single loop over all edges known to belong to one more triangle (Figs. 5, 6). A stack implemented for this purpose records edges of triangles generated so far. Then, one edge (a_i, a_j) is retrieved from stack, and the loops over k and l are executed until a *new* triangle (a_i, a_j, a_k) is identified. At this point, edges (a_i, a_k) and (a_j, a_k) are pushed onto the stack. The procedure terminates as soon as the known number of triangles has been generated. An alternative (and equivalent) termination condition would be an empty stack. Note that by construction a second triangle for a given edge may have been found *before* it is retrieved from stack (e.g. edge (a_1, a_7) in Fig. 6). The algorithm checks for this case (line 6 of Fig. 5) and determines whether it should continue or retrieve the next edge. Additional checks are optional but help to avoid unnecessary circumcircle tests (line 11) and to limit the size of the stack (lines 23, 24). Implementation B requires N_F full cycles through the outer loop and up to N_V cycles through each of the inner loops, resulting in $O(N_V^3)$ scaling for the worst case scenario. An initial edge is required as seed for the triangulation. This can easily be obtained from an arbitrary vertex a_i and its nearest neighbor a_j, since nearest neighbors are known to form edges of Delaunay triangles [16]. Obviously this initial step is carried out in $O(N_V)$ time.

Realizing that vertices close in space are the most likely to form triangles, we may additionally reduce the number of required circumcircle tests by using presorted lists of vertices. The innermost loop only needs to consider vertices a_l whose distance to a_i (or a_j, a_k) is smaller than the diameter ($2r_{ijk}$) of the circumcircle of (a_i, a_j, a_k). No other vertex can possibly be inside the circumcircle. Valid Delaunay triangles and their circumcircles are typically small, hence identification will only require very few cycles provided that the loop index l runs through a list of vertices sorted for distance with a_i. On the other hand,

the majority of vertex triples contain other vertices inside their circumcircles, and most of these circles are quite large. However, only one vertex inside the circumcircle needs to be identified, and vertices close to a_i are the most likely candidates, again limiting the number of necessary cycles through the innermost loop. Similar considerations apply to the loop with index k: Vertices a_k close to a_i (or a_j) are the most likely to form triangles with (a_i,a_j). Exactly one triangle needs to be identified, and only very few cycles through this loop are required, if the list of vertices a_k is sorted for distance with a_i. Delaunay triangulation is thus effectively treated as a local property. In the ideal case, the number of cycles (per edge (a_i,a_j)) through loops k and l will not depend on the number of vertices, but only on how homogeneously these vertices are distributed. Taking into account a total of N_F cycles through the outer loop, we thus expect (near) linear scaling, $O(N_V)$, for implementation C. One remaining problem is to generate a sorted list of vertices. The calculation of the distance matrix can only be carried out in $O(N_V^2)$ time, the subsequent sorting requires at least $O(N_V^2 \cdot logN_V)$ steps. Initial tests have shown that even with very efficient non-recursive implementations of Quicksort [31] the sorting becomes the dominant part of the calculation as N_V exceeds ~500. Fortunately, however, it is not really necessary to compute and sort the *entire* distance matrix. It will be sufficient for every vertex to know all its nearest neighbors within a certain distance r_{max}. This information may be obtained with grid-cell based techniques such as those suggested by Quentrec and Brot [32] and by Hockney *et al.* [33] in the context of molecular dynamics simulations. Using these techniques we achieve near linear scaling for the full triangulation procedure (see Table 1).

		t_{cpu}/s for implementation		
Sample	N_V	A	B	C
apo-FABP	131	0.7	0.1	0.01
Random distribution	1600	1062.0	6.1	0.19
Random distribution	6400	74096.2	109.8	0.91
Particle insertion	1600	1054.3	6.0	0.19
Particle insertion	6400	70581.3	105.9	0.87

TABLE 1. **Computational Aspects of Delaunay Triangulation.** The chosen samples for triangulation include the C_α skeleton of apo-FABP as well as sets of N_V particles placed on a sphere either randomly or by particle insertion. Random distributions are intentionally biased towards polar regions by choosing each pair of sphere angles (θ, ϕ) with equal probability. The particle insertion technique yields a much more homogeneous distribution because it chooses every position on the sphere with equal likelihood. CPU times have been measured on one processor of a Pentium III/500 Xeon server.

In summary, implementation C establishes a very efficient variant of a simple Delaunay triangulation algorithm which can easily be applied to long MD trajectories. Differences in efficiency between particle distributions of varying homogeneity are noticeable but not excessive.

2.3. **Identification of interior water.** Once the triangulation is known, simple geometry is used to decide whether a solvent molecule or, more generally, a particular point P is located in the interior of the generated polyhedron. The choice of a sphere as auxiliary surface for triangulation entails that every face of the polyhedron is visible from its center and not hidden by other faces. Consequently, the ray originating in center C and passing through P will pass through one (and only one) triangular face $F_{ijk}=(a_i, a_j, a_k)$ of the polyhedron (see Fig. 2). Solving the system of linear equations,

$$\overrightarrow{CP} = \gamma_i \overrightarrow{Ca_i} + \gamma_j \overrightarrow{Ca_j} + \gamma_k \overrightarrow{Ca_k} \tag{4}$$

for coefficients γ_i, γ_j, and γ_k will tell if the ray from C through P passes through triangle F_{ijk} ($\gamma_i > 0$, $\gamma_j > 0$, $\gamma_k > 0$), and, if so, if P is located inside the polyhedron ($\gamma_i+\gamma_j+\gamma_k<1$), on the surface ($\gamma_i+\gamma_j+\gamma_k=1$), or outside ($\gamma_i+\gamma_j+\gamma_k>1$). In the numerically unlikely case that any of γ_i, γ_j, γ_k equal zero, the ray from C to P passes through an edge or vertex of the polyhedron.

Preferred pathways of exchange may be identified by recording and clustering surface triangles involved in water exchange. This procedure will be referred to as portal cluster analysis and it may be outlined as follows: Whenever a water molecule changes its state (from internal to external or vice versa), the surface triangle closest in space is recorded. If this triangle shares a vertex with one already recorded triangle, both are registered in the same cluster. If it shares vertices with triangles in two or three different clusters, all these clusters are combined, and the triangle is registered with this new cluster. If none of its vertices are shared by any of the previously recorded triangles, the triangle is registered with a new cluster. After processing the entire trajectory, there are $N_{cluster}$ different clusters which represent $N_{cluster}$ distinct regions of the protein (orifices) involved in water exchange.

3. Application to MD trajectories of *apo*- and *holo*-FABP

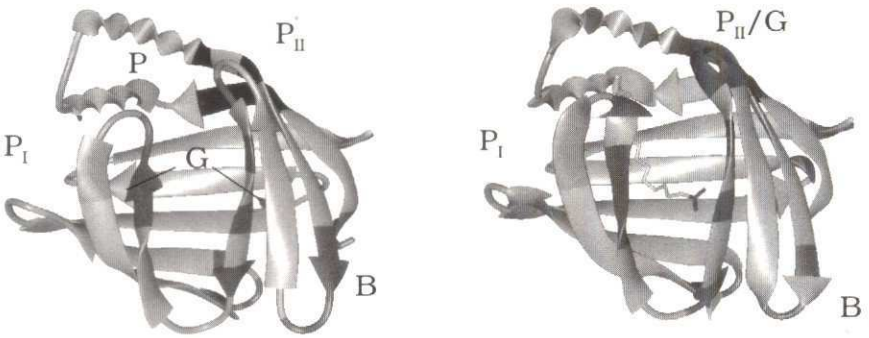
We have applied the procedure outlined above to identify water molecules in the interior of *apo*- and *holo*-FABP and to capture pathways of exchange with the bulk. Two molecular dynamics simulations of 5 ns length were carried out for FABP in *apo*-form and with bound palmitate. The simulations were started from crystal structures of the

protein which were inserted in large boxes containing equilibrium configurations of water molecules. Truncated octahedron periodic boundary conditions were applied. Since protein atoms and water molecules leaving the central box on one side were replaced by their periodic images on the opposite side, coordinates needed to be prepared prior to analysis. For every time frame, the protein chain was restored, and the minimum image convention was applied for water molecules with respect to the protein center of mass. Triangulation was then performed for all 131 C_α atoms, capturing water molecules inside the β-barrel of the protein.

3.1. **Water inside the protein.** The simulations show a total of 25 and 7 water molecules, respectively, which never leave the β-barrel of the protein. These numbers compare reasonably well with the number of well-ordered internal water molecules identified in the crystal structures of the two forms of the protein. Depending on the exact distinction between interior and exterior, experimental numbers vary between 23-24 and 7-9, respectively [6, 34–36]. Many water molecules exchange between the protein interior and the bulk and are thus unlikely to be detected in crystal structure experiments. On average, there are 8 and 14 additional water molecules, respectively, which are detected in the protein interior for some period of the MD simulation.

3.2. **Water exchange pathways.** Portal cluster analysis (see above) identifies several pathways of exchange with the bulk (see Table 2, Fig. 7). The most prominent orifices discussed in the experimental literature [36] are found. These include the gap G in the β-barrel and the portal near the helices. The portal appears to be composed of three separate areas which are denoted P_I (left), P (center), and P_{II} (right) and are shown in Figure 7. Another site of exchange has been identified on the opposite side of the protein barrel and will be referred to as the backside portal (B). This orifice has not found as much attention in the experimental literature, but clearly shows up in the MD simulations as an additional site of exchange. There are a number of interesting differences observed between *apo*- and *holo*-FABP:

- *holo*-FABP shows a higher rate of exchange than *apo*-FABP. This is counter-intuitive given the reduced gap width (see above) and the smaller number of water molecules inside the cavity.
- *holo*-FABP shows a neutral balance of entry and exit events. There is a net gain of 2 water molecules by exchange through the portal region P_{II} and the gap and a net loss of 2 water molecules by exchange through the portal region P_I and the backside portal. *apo*-FABP, however, loses 9 water molecules by expulsion through the portal region P_I (-4), the gap (-3), and the backside portal (-2). This



apo-FABP *holo*-FABP

FIGURE 7. **Water exchange pathways.** Different shades of gray indicate different orifices (P, P_I, P_{II} ,B , G) identified by portal cluster analysis.

Orifice		Contributing Residues	N_{enter}	N_{leave}
apo-FABP	P_I	22, 23, 76, 77, 97	15	19
	P_{II}	8, 9, 33, 35	12	12
	P	14, 15, 29, 32	1	1
	G	56, 58, 60, 70-72	5	8
	B	1, 2, 42, 44, 84, 86, 87, 108	67	69
holo-FABP	P_I	21, 22, 75, 76	23	24
	P_{II}/G	8, 9, 33-35, 55, 56, 58, 60, 70, 71, 73	73	71
	B	1, 44, 86, 87	37	38

TABLE 2. **Portal cluster analysis: orifices observed in water exchange.** For counting all entry (N_{enter}) and exit (N_{leave}) events, a time window of 5 ps has been used to define the minimum period required between two consecutive changes of state (internal to external or external to internal). Residues contributing at least 15 % to all events counted for a given opening are listed.

may indicate that much longer simulations are needed for adequate equilibration of water exchange in *apo*-FABP.

- Most of the water exchange in *holo*-FABP proceeds via the gap and portal pathways. In contrast, *apo*-FABP shows more exchange through the backside portal.

In summary, portal cluster analysis confirms common assumptions about major solvent exchange pathways and identifies an additional orifice at the backside of the protein.

3.3. **Water densities.** Water positions have been mapped onto a three-dimensional grid which discretizes the space occupied by the protein molecule. Histograms of water positions are then recorded from the entire trajectory (see Figures 8 and 9). The bulk of the so-determined water density is concentrated in the center of the protein, populating

an area typically referred to as the actual interior protein cavity. Note, however, that many other locations inside the protein exhibit considerable water density as well. These additional sites of water density were only identified because the cavity was deliberately chosen to include the entire β-barrel lined by the protein backbone. The isolated area of water density near the loop at the bottom of the gap G, for example, corresponds to a well-ordered water molecule (*W135*) known from crystal structure determinations [34].

Dynamic information has been added to the plots shown in Figures 8 and 9 by separate analysis of successive time periods. The total density has further been divided into contributions arising from water molecules found inside the protein for the entire MD simulation (*permanent*), and those exchanging with the bulk. The latter molecules are labeled *core* if they are in contact with permanent waters (OW-OW distance of 0.3 nm or less) at any time during the simulation, and *peripheral* otherwise. This simple distinction enables us to focus on water molecules exchanging between the bulk and the center of the protein. A number of interesting conclusions can be drawn by comparing the time-resolved evolution of water densities in *apo-* and *holo*-FABP:

apo-FABP shows an apparent contraction of *permanent* water density. Particularly molecules in the backside portal region (B) migrate to the center of the cavity. At the same time, five *core* water molecules are lost by expulsion primarily through the backside

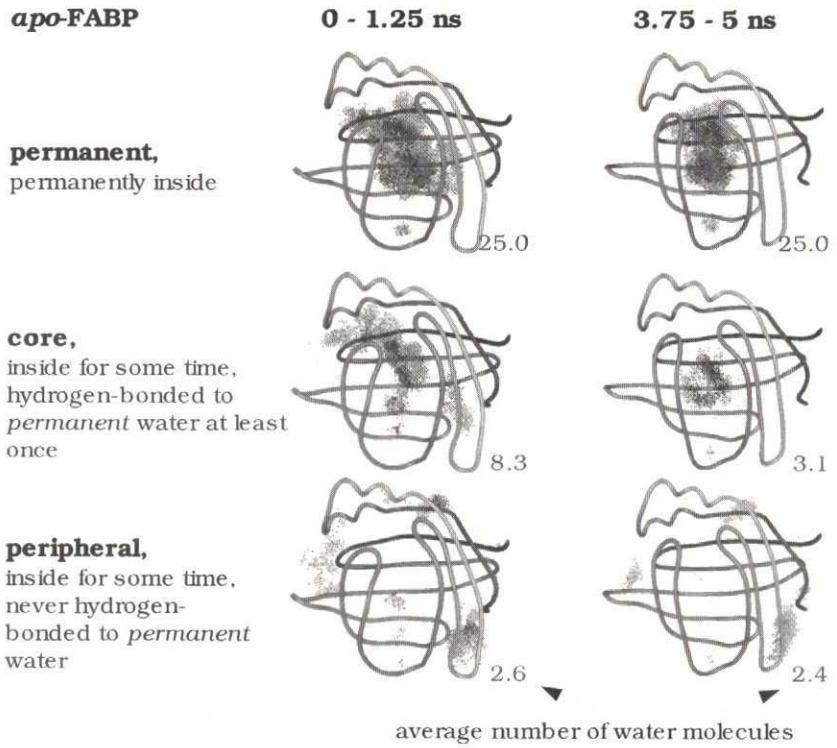


FIGURE 8. **Water density inside *apo*-FABP plotted on a 3D-grid.**

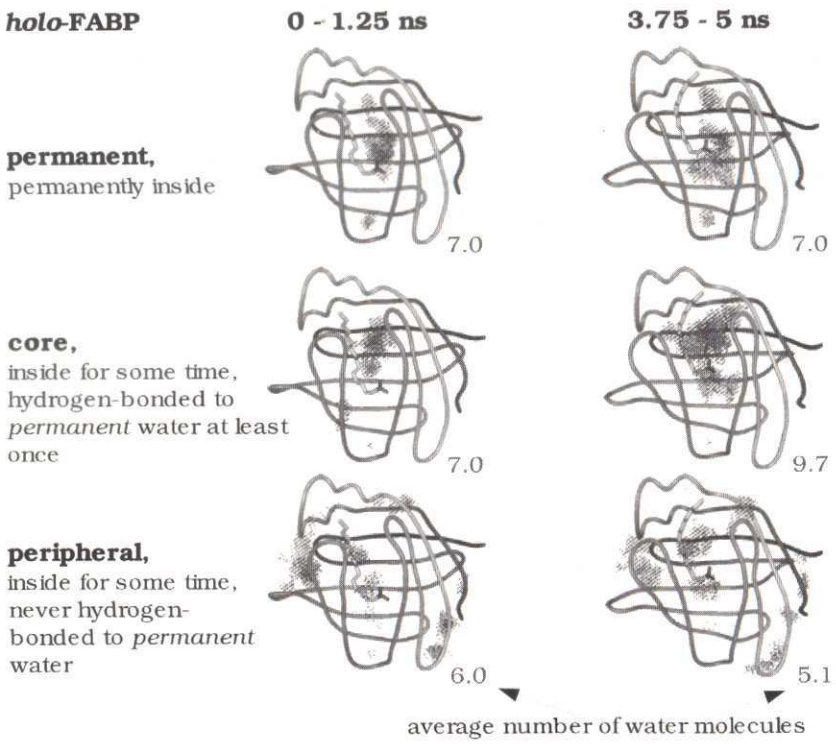


FIGURE 9. **Water density inside *holo*-FABP plotted on a 3D-grid.**

portal and through portal area P_I. The number of peripheral water molecules remains almost constant throughout the simulation.

The much smaller number of permanent water molecules in *holo*-FABP appears to be distributed over a region similar in size to that of *apo*-FABP. Compared to *apo*-FABP, a higher percentage of the total internal water density relates to water molecules exchanging with the bulk. No contraction or overall loss of internal water density is discernible at any time. Quite to the contrary, a significant number of water molecules flows in from portal area P_{II} and contributes to an increasing region of *core* water density in the center of the protein.

Core rather than *peripheral* water molecules contribute the most to the loss of water density in *apo*-FABP and to the gain of water density in *holo*-FABP. This nicely demonstrates that a significant portion of the observed water exchange relates to exchange between the protein center and bulk water.

It is also clear from Figures 8 and 9 that exchange through the backside portal B contributes little to the central water density. For *holo*-FABP, all water density observed in the vicinity of the backside portal corresponds to *peripheral* water which by definition has not had any contacts with *permanent* water in the center of the cavity. *apo*-FABP also

shows some *core* water density close to the backside portal which, however, appears disconnected from the central region. It seems unlikely that *core* water molecules migrating between the two regions would not have been detected at the chosen time resolution of 0.5 ps. Contacts between core and permanent waters are thus likely from the early phase of the simulation which still shows some *permanent* water density outside the central cavity. Hence we conclude that little or no exchange between central cavity and bulk water has occurred through the backside portal. This is probably due to a number of hydrophobic residues (*viz.* Trp6, Phe2, and Phe93) effectively closing off the backside of the β-barrel [35].

3.4. **Water coordination numbers.** The analysis of water coordination provides additional information on water structure and discriminates between isolated molecules, clusters, and the bulk (see Fig. 10). The distribution of coordination numbers in bulk water can easily and accurately be simulated by considering only 'remote' exterior water molecules which are not in contact with any protein atoms. It compares very well with the distribution for bulk water calculated by García and Hummer [37]. The distribution shows a maximum at N_{coord}=5, corresponding to tetrahedrally coordinated water which is, on average, augmented by one additional water molecule located on one of the tetrahedron's faces. Coordination numbers of 4-6 are very common, but the probability vanishes quickly for lower *and* higher coordination numbers. In contrast, water molecules

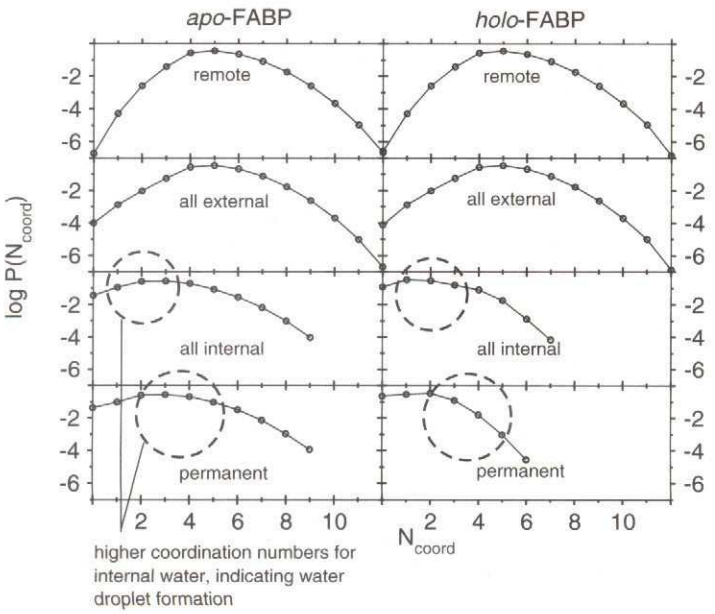


FIGURE 10. **Coordination numbers for water inside the protein and outside.** See text for details.

on the protein surface prefer lower coordination numbers. Consequently, the distribution calculated for the entire set of *all* exterior water shows a bias towards low N_{coord} (second row of Fig. 10).

Coordination numbers of *interior* water molecules are of particular interest to the present study, and unlike those of *exterior* water molecules, they differ considerably between *apo-* and *holo-* FABP. While *apo*-FABP shows a rather smooth distribution of coordination numbers for water molecules permanently in the protein interior, the corresponding distribution for *holo*-FABP vanishes rapidly for $N_{coord} > 3$. Around 10% of the water permanently inside *apo*-FABP is coordinated by 4 other water molecules, two orders of magnitude more than in *holo*-FABP. Low coordination numbers (0 or 1) of water permanently in the protein interior are less common for *apo-* than for *holo-* FABP. The distributions of coordination numbers for all interior water molecules peak at 2-3 for *apo*-FABP and at 1 for *holo*-FABP. The most favorable coordination numbers of bulk water, $N_{coord} = 4$ and N_{coord}=5, are found for 19% and 9%, respectively, of all water inside *apo*-FABP, but for only 8% and 2% of all water inside *holo*-FABP. In summary, all these observations indicate the formation of a water droplet in the cavity of the *apo*-protein.

3.5. **Water interaction potentials.** The geometric information provided by water densities and coordination numbers may be supplemented by energetic analyzes to further rationalize the differences in water structure and dynamics found between *apo-* and *holo-*FABP. The water interaction potential, mapped onto the same grid as the water density, appears to be a convenient measure of local differences in energy and will thus be analyzed in some detail.

We define the water interaction potential as the average total interaction energy of a water molecule located at a particular grid point. It is composed of Coulomb and van der Waals interactions with all other molecules and includes the usual reaction field

	apo-FABP	*holo*-FABP
Protein	-50.9	-56.8
Ligand	-	-11.8
Interior Water	-41.4	-16.5
Exterior Water	-1.4	-3.8
Total	-93.7	-89.0

TABLE 3. **Water interaction potential (kJ/mol).** Averages of the interaction potential experienced by all interior water molecules interacting with other water, protein, and ligand.

corrections. The potentials are averaged in time, and different water molecules may contribute to the same grid point. Table 3 reports averages for water molecules of a specific kind. Note that the tabulated values are averaged both in time and in space.

The notion of a water droplet forming in the cavity of *apo*-FABP is clearly supported by the data reported in Table 3. Water molecules in the protein interior show average interaction energies of more than -40 kJ/mol with other interior water. This amounts to about half the total interaction energy of a single water molecule in bulk water. As expected, internal water-water interactions are less significant for *holo*-FABP, but this is in part compensated for by favorable interactions with the ligand. In addition *holo*-FABP also shows slightly more attractive water-protein interactions which compensate for the lack of water droplet formation in *holo*-FABP. The total water interaction potential averages to -89 kJ/mol for *holo*-FABP which is only marginally (5 kJ/mol) less than for *apo*-FABP.

4. Discussion and Conclusions

A new computational procedure based on Delaunay triangulation has been presented to identify internal water in proteins and to capture exchange pathways. Unlike many other approaches previously reported in the literature [14, 15, 18–22], it does not attempt to identify solvent accessible volumes in the protein interior. Instead it provides a means to define a bounding surface of the protein by triangulating a set of selected protein atoms on an auxiliary sphere. A polyhedron is obtained, and any solvent molecule inside the polyhedron is considered to be in the protein interior. Exchange pathways and protein orifices are identified by clustering surface triangles next to a solvent molecule that leaves or enters the protein interior.

The triangulation algorithm is conceptually simple and uses the property of Delaunay triangulation that the circumcircle of every triangle is empty of other sites. While straightforward implementation of this principle affords computationally unfavorable scaling behavior ($O(N_V^4)$), *a priori* knowledge of the number of surface triangles combined with presorted lists of vertex pairs may be used to enhance computational efficiency significantly and eventually achieve near-linear scaling. In practice, the procedure requires trivial amounts of computer time to define the bounding surface of the protein, to identify internal water molecules and to determine portals of exchange with bulk water.

The approach presented in this paper can be applied to any convex shaped protein or region of a protein. It will fail for concave shapes, however, which cannot be reasonably represented by a sphere. Application to a torus, for example, would yield a dented ellipsoid rather than recover the torus after back transformation to the original coordinate frame. The choice of atoms used to define the bounding surface should be tailored to

the problem at hand. It should normally not include atoms in very flexible parts of the molecule (e.g. protein sidechains) which are not representative for the overall shape of the molecule. In the case of FABP the β-barrel spanned by main chain atoms provides a clear route to distinguish between interior and exterior, and fluctuations of the barrel shape observed along the MD trajectory should be reflected in fluctuations of the embracing polyhedron as well. In other cases, certain (buried) main chain atoms may be regarded as part of the interior rather than defining its bounding surface, and they should not be selected for triangulation. In summary, the manual selection of atoms certainly requires some judgement and it prevents full automation. But it also offers the flexibility needed for custom-tailored solutions.

Application to 5 ns long molecular dynamics trajectories of *apo-* and *holo-* FABP demonstrate the benefits of the approach. Protein orifices known from the crystal structures and discussed as putative pathways of exchange have been confirmed to be involved in water exchange between the interior and exterior of the protein. An additional opening on the backside of the protein has been identified. The internal water has been further analyzed in terms of water densities, coordination, and interaction potentials. A number of differences have been observed comparing *apo-* and *holo*-FABP:

First, *apo*-FABP experiences a continuous loss of internal water, indicating that equilibration has not finished within the 5 ns of simulation, and that the water content may fall below 30 eventually. *holo*-FABP shows a relatively constant count of 21-22 internal water molecules.

Second, *holo*-FABP experiences much more water exchange between the interior cavity and the bulk, primarily through portal region P_{II} . This observation is counter-intuitive given the smaller number of water molecules inside the cavity. A more detailed analysis is clearly needed to explain this unexpected behavior. [23]

Third, *apo*-FABP displays a significant concentration of water density in the center of the cavity. It appears to form a small water droplet with water-water interactions accounting for about half the total interaction energy of internal water. The notion of a water droplet is also supported by calculated coordination numbers. Water inside *holo*-FABP, on the other hand, appears to benefit from interactions with the carboxylate group of the ligand as well as slightly stronger water-protein interactions.

In summary, the approach discussed in this paper to identify water in protein cavities has proven to be a useful tool to study structure and dynamics of internal water and to complement experimental information from X-ray crystallography and NMR techniques with theoretical analyses of MD trajectories.

Acknowledgments

The molecular graphics images were generated using the MidasPlus package [38] from the Computer Graphics Laboratory, University of California, San Francisco (supported by NIH P41 RR-01081).

References

[1] S. J. Hubbard, K.-H. Gross, P. Argos, *Protein Eng.* **1994**, *7*, 613–626.

[2] M. A. Williams, J. M. Goodfellow, J. M. Thornton, *Protein Sci.* **1994**, *3*, 1224–1235.

[3] J. A. Ernst, R. T. Clubb, H.-X. Zhou, A. M. Gronenborn, G. M. Clore, *Science* **1995**, *267*, 1813–1817.

[4] J. A. Ernst, R. T. Clubb, H.-X. Zhou, A. M. Gronenborn, G. M. Clore, *Science* **1995**, *270*, 1848–1849.

[5] V. P. Denisov, K. Venu, J. Peters, H. D. Hörlein, B. Halle, *J. Phys. Chem. B* **1997**, *101*, 9380–9389.

[6] S. Wiesner, E. Kurian, F. G. Prendergast, B. Halle, *J. Mol. Biol.* **1999**, *286*, 233–246.

[7] M. L. Raymer, P. C. Sanschagrin, W. F. Punch, S. Venkataraman, E. D. Goodman, L. A. Kuhn, *J. Mol. Biol.* **1997**, *265*, 445–464.

[8] E. Meyer, *Protein Sci.* **1992**, *1*, 1543–1562.

[9] E. N. Baker, in *Protein-Solvent Interactions* (Ed. R. B. Gregory), Marcel Dekker, New York, 1995 pp. 143–189, pp. 143–189.

[10] B. Lee, F. M. Richards, *J. Mol. Biol.* **1971**, *55*, 379–400.

[11] F. M. Richards, *Ann. Rev. Biophys. Bioeng.* **1977**, *6*, 151–176.

[12] M. L. Connolly, *J. Appl. Cryst.* **1983**, *16*, 548–558.

[13] W. Cai, M. Zhang, B. Maigret, *J. Comp. Chem.* **1998**, *19*, 1805–1815.

[14] J. Liang, H. Edelsbrunner, P. Fu, P. V. Sudhakar, S. Subramaniam, *Proteins* **1998**, *33*, 1–17.

[15] S. Sastry, D. S. Corti, P. G. Debenedetti, F. H. Stillinger, *Phys. Rev. E* **1997**, *56*, 5524–5532.

[16] J. O'Rourke, *Computational Geometry in C*, 1st ed., Cambridge University Press, Cambridge, **1995**.

[17] J. Liang, H. Edelsbrunner, C. Woodward, *Protein Sci.* **1998**, *7*, 1884–1897.

[18] J. Liang, H. Edelsbrunner, P. Fu, P. V. Sudhakar, S. Subramaniam, *Proteins* **1998**, *33*, 18–29.

[19] P. Alard, S. J. Wodak, *J. Comp. Chem.* **1991**, *12*, 918–922.

[20] R. Voorintholt, M. T. Kosters, G. Vegter, G. Vriend, W. G. J. Hol, *J. Mol. Graphics* **1989**, *7*, 243–245.

[21] D. G. Levitt, L. J. Banaszak, *J. Mol. Graphics* **1992**, *10*, 229–234.

[22] J. S. Delaney, *J. Mol. Graphics* **1992**, *10*, 174–177.

[23] D. Bakowies, W. F. van Gunsteren, *J. Mol. Biol.* **2002**, *315*, 713–736.

[24] D. Bakowies, W. F. van Gunsteren, *Proteins* **2002**, *47*, 534–545.

[25] L. Banaszak, N. Winter, Z. H. Xu, D. A. Bernlohr, S. Cowan, T. A. Jones, *Advances in Protein Chemistry* **1994**, *45*, 89–151.

[26] M. D. Yoder, L. M. Thomas, J. M. Tremblay, R. L. Oliver, L. R. Yarbrough, G. M. Helmkamp, Jr., *J. Biol. Chem.* **2001**, *276*, 9246–9252.

[27] D. A. Doyle, J. M. Cabral, R. A. Pfuetzner, A. Kuo, J. M. Gulbis, S. L. Cohen, B. T. Chait, R. MacKinnon, *Science* **1998**, *280*, 69–77.

[28] B. Roux, R. MacKinnon, *Science* **1999**, *285*, 100–102.
[29] F. P. Preparata, M. I. Shamos, *Computational Geometry*, Texts and Monographs in Computer Science,, 1st ed., Springer Verlag, New York, **1985**.
[30] H. S. M. Coxeter, *Introduction to Geometry*, Wiley, New York, **1961**.
[31] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in Fortran 77 - The Art of Scientific Computing*, vol. 1, 2nd ed., Cambridge University Press, Cambridge, **1992**.
[32] B. Quentrec, C. Brot, *J. Comp. Phys.* **1973**, *13*, 430–432.
[33] R. W. Hockney, S. P. Goel, J. W. Eastwood, *J. Comp. Phys.* **1974**, *14*, 148–158.
[34] G. Scapin, J. I. Gordon, J. C. Sacchettini, *J. Biol. Chem.* **1992**, *267*, 4253–4269.
[35] J. C. Sacchettini, J. I. Gordon, L. J. Banaszak, *J. Mol. Biol.* **1989**, *208*, 327–339.
[36] J. C. Sacchettini, J. I. Gordon, *J. Biol. Chem.* **1993**, *268*, 18399–18402.
[37] A. E. García, G. Hummer, *Proteins* **2000**, *38*, 261–272.
[38] T. E. Ferrin, C. C. Huang, L. E. Jarvis, R. Langridge, *J. Mol. Graphics* **1988**, *6*, 13–27.