JOURNAL OF STRUCTURAL BIOLOGY **116,** 138–143 (1996)
ARTICLE NO. 0022

The Contour-Buildup Algorithm to Calculate the Analytical Molecular Surface

MAXIM TOTROV

Skirball Institute of Biomolecular Medicine, NYU Medical Center, 540 First Avenue, New York, New York 10016

AND

RUBEN ABAGYAN

Department of Biochemistry and Skirball Institute of Biomolecular Medicine, NYU Medical Center, Courant Institute of Mathematics, 540 First Avenue, New York, New York 10016

Received October 1, 1995, and in revised form October 19, 1995

A new algorithm is presented to calculate the analytical molecular surface defined as a smooth envelope traced out by the surface of a probe sphere rolled over the molecule. The core of the algorithm is the sequential build up of multi-arc contours on the van der Waals spheres. This algorithm yields substantial reduction in both memory and time requirements of surface calculations. Further, the contour-buildup principle is intrinsically "local", which makes calculations of the partial molecular surfaces even more efficient. Additionally, the algorithm is equally applicable not only to convex patches, but also to concave triangular patches which may have complex multiple intersections. The algorithm permits the rigorous calculation of the full analytical molecular surface for a 100-residue protein in about 2 seconds on an SGI indigo with R4400 processor at 150 Mhz, with the performance scaling almost linearly with the protein size. The contour-buildup algorithm is faster than the original Connolly algorithm an order of magnitude. **© 1996 Academic Press, Inc.**

INTRODUCTION

Generating a smooth surface of a molecule is important for a number of applications involving molecular recognition, electrostatics, energetics, and drug design [1–7], as well as molecular graphics [8, 9]. Almost 20 years ago Richards introduced an idea of molecular surface [1]. In contrast to the *solvent accessible surface* which is displaced outward from the van der Waals surface by the radius of the solvent probe [10–16], the *molecular surface* is a smooth envelope touching the van der Waals surface of atoms as the solvent probe rolls over the molecule. According to Richards, the molecular surface consists of two parts. The first part—the contact surface—is the van der Waals surface of the molecule capable of being in direct contact with the surface of the solvent probe. The second part—the reentrant surface—is composed of interior-facing parts of the probe sphere touching simultaneously two or more atoms (Fig. 1).

A number of numerical algorithms have been developed for calculating approximations of the molecular surface. These methods include a grid algorithm of Geer and Bush [18], dot surface algorithms [19], cube algorithms [20–23], and the Voronoi construction [1, 24].

Connolly [12] was the first to introduce a computer algorithm to generate the precise analytical molecular surface, subsequently implemented in the PMSQ program [8]. In his algorithm, which may be referred to as a "torus-centric" algorithm, the calculation is based on the descriptions of positions and parameters of all toruses, one of the three types of surface elements forming the molecular surface (Fig. 2). The disadvantage of this algorithm is the fact that to calculate the contour of the convex spherical patches forming the contact surface one needs to store and analyze all the torus segments accumulated for every two atoms at a distance closer than their van der Waals radii plus the diameter of a water molecule. This procedure results in large memory and time requirement.

This paper presents a new fast algorithm which calculates the molecular surface by the sequential buildup of the contours bordering the convex contact surface patches that are taken to be the primary

1047-8477/96 $18.00

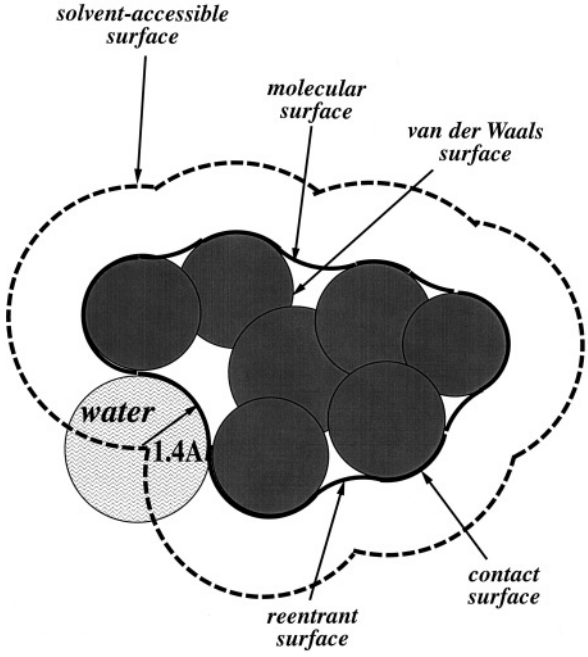


FIG. 1. Basic types of surfaces used in molecular modeling and graphics. Van der Waals surface consists of the parts of individual atomic van der Waals spheres unobstructed by other spheres. Solvent accessible surface is the locus of all possible positions of the center of the probe sphere when it touches but does not penetrate the van der Waals surface. Molecular surface can be best described as a combination of two kinds of surfaces: contact and reentrant. Contact surface is the part of van der Waals surface which can be in contact with the probe sphere without penetration. Reentrant surface is defined by the interior facing part of the probe when it is simultaneously in contact with more than one atom.

surface elements (Fig. 2). Tori and concave reentrant patches are then derived from these contours and the same contour-buildup algorithm is used to efficiently calculate the intersections of the concave spherical triangles. The algorithm scales almost linearly with the protein size and allows fast calculation of partial molecular surfaces.

METHOD

The key element of the procedure is the routine which determines the contour of the contact surface of an atom. This contour consists of one or more cycles of arcs on the van der Waals sphere. The contact surface has the following important property: if we consider the solvent accessible surface (see Fig. 1. for the definition) of the same atom, these two surfaces have exactly the same shape and only differ in radii by the solvent probe radius. Thus, we can calculate the patch of the solvent accessible surface of an atom and then scale it down to obtain the contact surface. The atomic patches of the solvent accessible surface are formed by the van der Waals spheres expanded by the probe radius and bounded by arcs of intersections with the neighboring expanded spheres. Instead of calculating all possible arcs and then trying to filter out those which do not belong to the patch boundary, we employ the local sequential contour-buildup algorithm.

The core of the algorithm applied to each accessible atom is a loop through all its neighbors in which the contour is gradually modified. Neighbors are the atoms within $R_i + R_j$ cutoff from the atom i, where R_i is its radius, R_j is the neighbour's radius, and both radii are expanded by the probe radius. The first neighbor atom creates a single circle, the next one creates a second circle or breaks the first circle into two arcs, and so on. More specifically, in each step we (i) find all the crossings between the new tentative circle and all the currently existing arcs, (ii) remove the arcs which are covered, (iii) modify those which were crossed, and (iv) add arcs of the new circle. After going through all neighbors, the contour consisting of n consecutive arcs A_j ($j = j_1, \ldots, j_n$) is built (Fig.3).

The above procedures deserve a more detailed description. The topology of the border is represented as one or more cycles. Each cycle points to one of its arcs, which we call the first arc of the cycle. Arcs are stored as a linked circular list. Each arc has the previous and the next arc and two vertices (ends). Vertex 1 is shared with the previous arc and vertex 2 is shared with the next arc. For each arc A_k, we store (i) the radius-vector r_k from the center of atom i to the arc's center and its vertices (if it's a full circle, we only store the center); (ii) the next and the previous arcs; and (iii) the neighboring atom which created the arc.

For each new neighboring atom j, the circle of intersection of its sphere and the sphere of atom i is determined first. The radius-vector r_j from the atom i to the center of this circle is determined by the equation

$$r_j = r^j_{\text{at}} \frac{R_i^2 + (r^j)^2_{\text{at}} - R_j^2}{2 \cdot r^{j^2}_{\text{at}}},$$

where R_i and R_j are the expanded radii of the two atoms and r^j_{at} is the radius - vector from the atom i to the atom j.

Then, each arc A_k of the current contour is checked for an *intersection with the new circle.* To identify whether the arc and the circle intersect, and to calculate new vertices if they do, let us introduce an auxiliary vector r_m. If we consider an imaginary line of intersection of two planes, one plane containing the new circle and the second plane containing an arc, r_m is the radius vector of the point where this line goes through the third plane, which is defined by the circle center, arc center, and atom i

$$r_m = r_j \cdot \frac{(r_j^2 - r_i \cdot r_k) \cdot r_k^2}{r_j^2 \cdot r_k^2 - (r_j \cdot r_k)^2} + r_k \cdot \frac{(r_k^2 - r_j \cdot r_k) \cdot r_j^2}{r_j^2 \cdot r_k^2 - (r_j \cdot r_k)^2}.$$

If this point is outside the sphere of atom i, i.e., $r_m^2 > R_i^2$, the new circle and the circle containing the arc do not intersect.

There are four types of mutual position of the two nonintersecting circles j and k on a sphere each requiring different processing: (1) k-circle is outside j-circle, we can proceed to the next arc; (2) k-circle and j-circle together cover the whole sphere, current atom has no surface; (3) k-circle is within j-circle, so we may proceed to the next neighboring atom since atom j can not change the current border; (4) k-circle, and hence arc A_k, is covered by j-circle. In the latter case no action is needed since arc A_k will be later automatically discarded.

If the circles do intersect, we have to find the points of intersection. These are given by the formula

$$r_{jk}^{\text{cross}1,2} = r_m \pm [r_j \times r_k] \cdot \sqrt{\frac{(R_i^2 - r_m^2)}{[r_j \times r_k]^2}}.$$

If the arc A_k is a full circle—i.e., it has no vertices—at this point we modify it by giving it vertices, store the vertices in the list for creation of the arc(s) on the new circle, and proceed to the next arc. Otherwise, we have to determine where are the two crossing points located on the circle containing the old arc with respect to the ends of the arc.

We found that three parameters c_1, c_2, and d fully describe all the cases of mutual arrangements of the two new intersection points. These parameters are given by

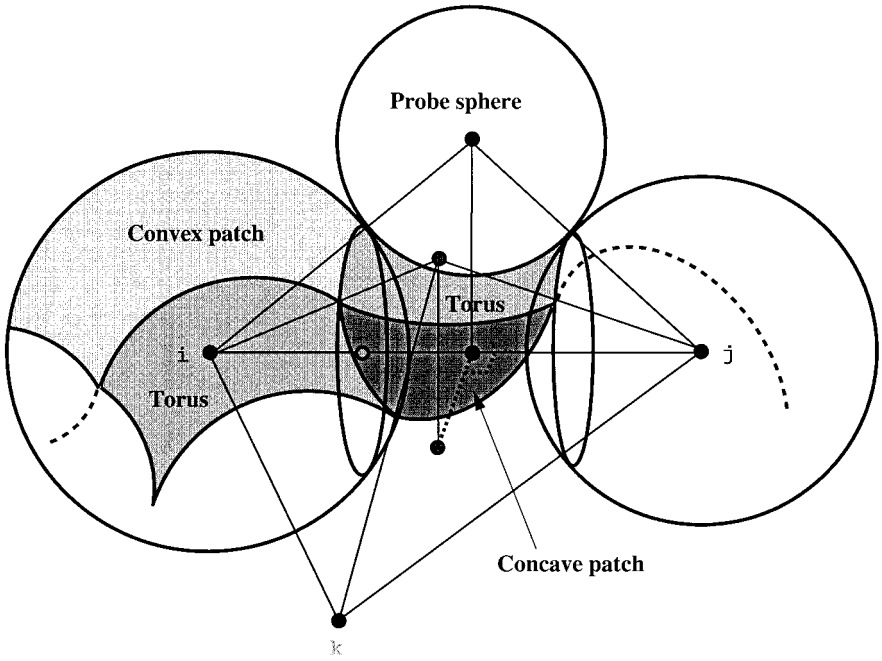


FIG. 2. Three main types of molecular surface elements: convex spherical patch of atomic contact surface, saddle toroid segments, and concave spherical triangle. Convex patch is the area where the probe can only contact one atom. Toroid segments emerge when the probe touches two atoms and rolls between them. Simultaneous contact of the probe with three atoms results in concave spherical triangle.

$$c_1 = r_{\text{at}}^j \cdot (r_k^{\text{vx1}} - r_j)$$

$$c_2 = r_{\text{at}}^j \cdot (r_k^{\text{vx2}} - r_j)$$

$$d = s_k \cdot ([r_k^{\text{vx1}} \times r_{jk}^{\text{cross1}}] \cdot r_k^{\text{vx2}}),$$

where $r_k^{\text{vx1,2}}$ are radius vectors of the two ends (vertices) of the arc and s_k is the arc sign defined by mutual position of the arc and the contributing atom.

Depending on c_1, c_2, and d we perform different operations. If c_1 is positive and c_2 is negative, it means that vertex 1 of the old arc is inside the sphere of the atom j while vertex 2 is outside and the arc has to be shortened by replacement of vertex 1 with the crossing point 1 if d is positive and the crossing point 2 if it is negative. Similarly, if c_2 is positive and c_1 is negative, vertex 2 has to be replaced in the same fashion. If both c_1 and c_2 are positive, both vertices are inside the j sphere. If d is positive, the whole arc is inside, and it will be automatically discarded later. If d is negative, part of the arc between crossing points is still outside and both ends of the arc are replaced by the crossing points. If c_1 and c_2 are negative, both vertices are outside of j sphere. If d is positive, part of the arc is inside the sphere and the arc has to be cut in two, otherwise it is completely outside and has to be left intact.

In each of the above cases, the crossing points which are involved in modification of the old arcs are stored in a list. For each of them the flag which defines if the crossing point became vertex 1 or vertex 2 of the modified old arc is stored. After all old arcs were processed, the list of the accumulated crossing points is sorted in such a manner that they sit on the new circle in a clockwise order if we look from atom j. Then, starting from any point which was marked as being vertex 2 of an old arc, we connect pairs of consecutive points on the circle into new arcs and link the new arcs with the old arcs. As a result, some cycles may become split into several new ones. At this point we look for the new cycles and add them to the list of cycles.

Some cycles may completely disappear. To account for them we check whether during the processing of the arcs of a particular cycle any of the arcs are modified or lay completely outside of the j sphere. If not, we conclude that the whole cycle is inside the j sphere and should be discarded. Also, if the new circle does not intersect with and is not covered by any of the old arcs and their containing circles, it might create a new cycle. However, we still have to prove that it is not covered or intersected by other circles eliminated during the procedure, which is done by checking it against all previously considered neighbors. If it is not, we add a new cycle and give it the new circle as the first arc.

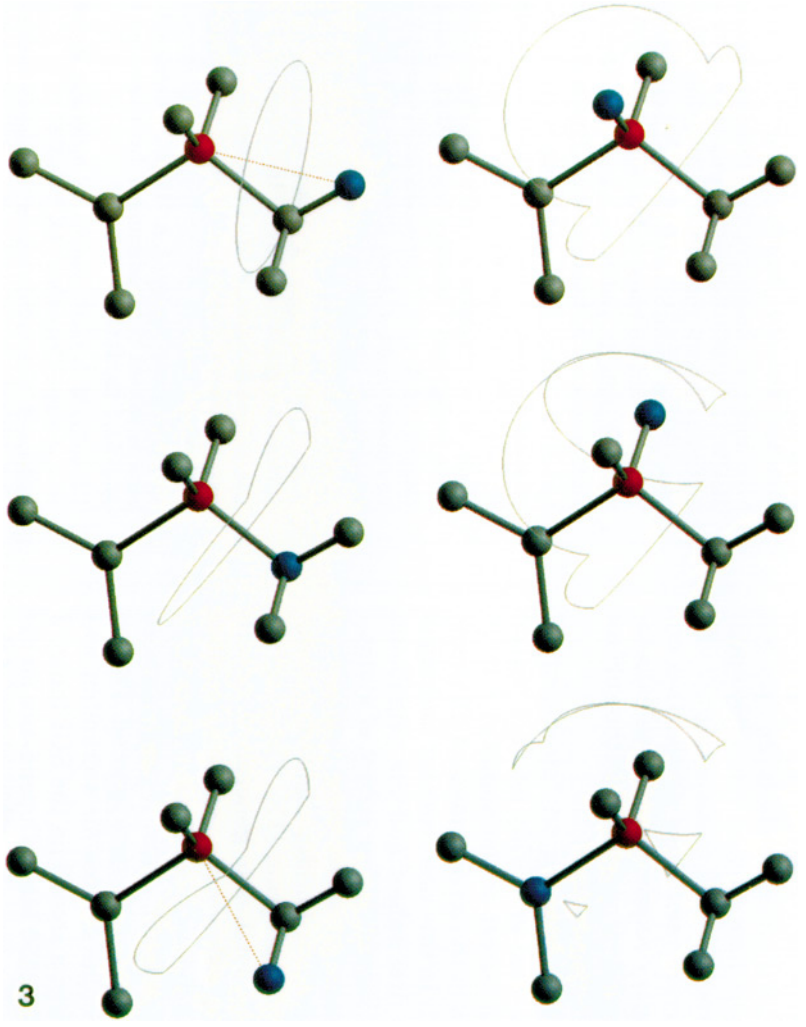
The procedure described deals with *all* possible cases except the singularities, when one or more of the parameters c_1, c_2, and d is close to zero. When using double-precisions arithmetics, such occurrences are quite rare because the positions of at least four atoms have to satisfy a stringent condition to cause the singularity. Currently, in such a case, we skip the calculations for the particular atom. In many instances this results in only rather minor defects of the surface; however, if we need the complete surface, small random translations of atom positions may be applied to bypass the singularity.

To calculate the two remaining types of surface elements, namely, tori and the concave patches, we just have to realize that all the information about them is already in the contour. Each arc A_j corresponds to one torus segment between atoms i and j. The line connecting the centers of the two atoms defines the axis of the torus and the coordinates of the arc's ends define the extent of the segment. Similarly, each two consecutive arcs A_j and A_{j+1} ($j = j_1, \ldots, j_n$) define the concave patch, completing the construction of the molecular surface.

Unfortunately, in many cases the reentrant surface contains more complex elements than torus segments and spherical triangles. The radius of the inner circle of the torus may become negative.

$$\sqrt{R_i^2 - \left(\frac{R_i^2 + D_{ij}^2 - R_j^2}{2 \cdot D_{ij}}\right)^2} - r_{\text{probe}} < 0.$$

In this case the central part of the saddle collapses and the



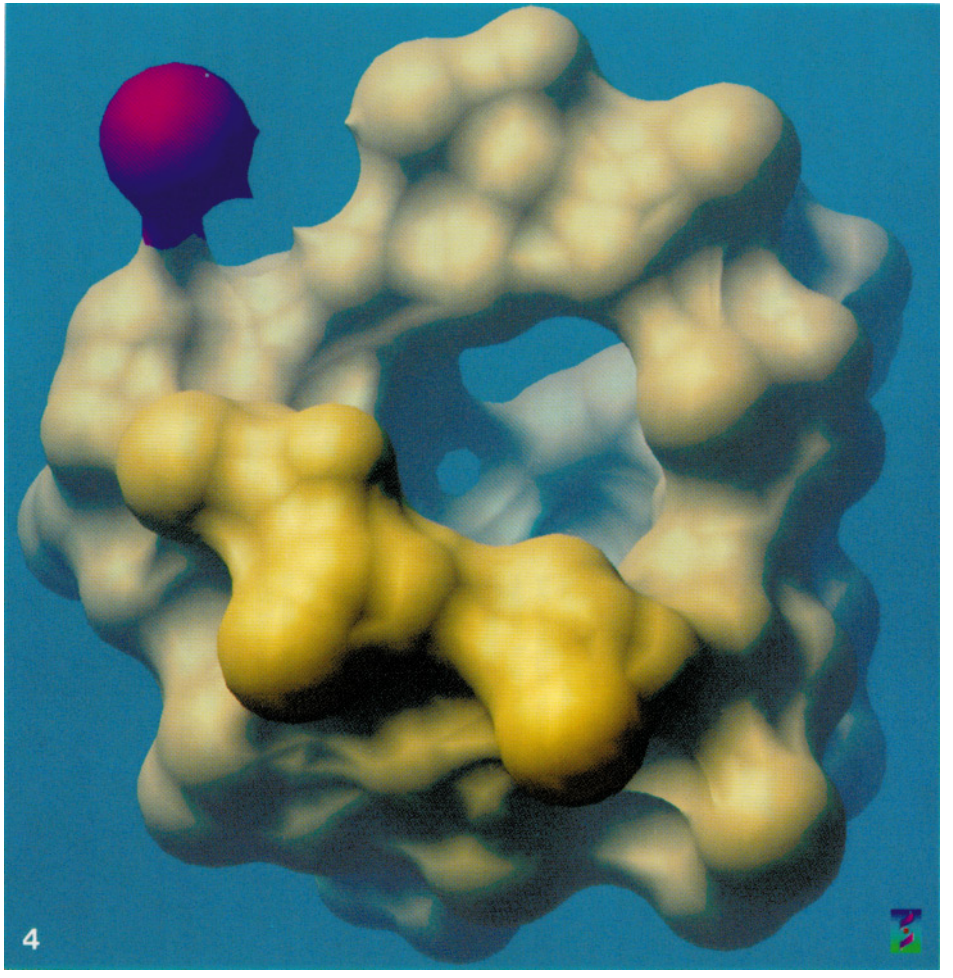


FIG. 3. Example of the convex patch boundary buildup for the C_α atom of the amino acid glycine. Various types of intersection of the new circle with the old cycle can be observed.

FIG. 4. Molecular surface of a molecule with the point cusps and the circular cusps resulting from multiple intersections of the concave triangles.

torus segment splits into two separate patches. The situation can be rather easily treated since the resulting surface elements are still fully described by the same parameters and equations as the torus and can be treated similarly except for minor adjustments in subsequent triangulation procedure.

Much more difficult is the case of intersecting triangular concave patches. This situation is difficult to detect since there might be no direct indication in the parameters of an individual concave triangle and can only be found on the pairwise basis, which potentially means checking all pairs of triangles. Fortunately, we found a criterion ruling out the majority of the concave triangles before the pairwise check: if the whole of the concave triangle is contained inside the pyramid formed by the centers of the three atoms and the probe sphere, the intersection is impossible. Thus, during the atom-by-atom creation of the molecular surface, we apply this criterion to each concave triangle and draw it immediately only if the criterion is satisfied, otherwise this triangle is stored for further processing.

Intersections may potentially produce concave patches of very complex shape. A single intersection leads to the removal of the patch fragments which happened to be within the other sphere. Luckily, the contour-buildup algorithm described above for the convex patches can be used in a similar manner to calculate the results of potentially multiple intersections of the concave patches. Basically, the only modification of the core algorithm is that we have to start from the triangular contour instead of the whole sphere. Again, we form the list of other spheres close to the current one and then go through it one by one, modifying the contour if necessary. The resulting concave patch is a rigorous solution of the collision problem. This is the last step of the algorithm.

RESULTS AND DISCUSSION

At no stage except for the concave triangle intersection analysis does this algorithm require any large (proportional to the surface) storage. All the elements can be drawn immediately after they are produced. Intersection lists are also relatively small since only a small fraction of all concave triangles do not satisfy the criterion mentioned above. This ensures minimal memory requirements. The calculation time is also roughly proportional to the surface in a wide range of molecule sizes.

Since the algorithm is based on "atom-centric" paradigm, it can be used to quickly produce partial surfaces for a small subset of the atoms of a large molecule or complex in the environment of all atoms or of another subset. In this case, contour buildup is applied only to the first subset and the neighbours are taken from the second, larger subset of atoms.

We calculated molecular surfaces for a number of proteins using the contour-buildup algorithm as implemented in the ICM program [17]. The program also triangulates the patches to generate a smooth solid surface. Figure 4 shows the molecular surface of a molecule. Table I shows calculation times for a series of proteins of different sizes. Water molecules as well as additional subunits were ignored. The exact analytical molecular surface for a crambin was calculated in less than a second on the SGI Indigo 2 workstation (R4400 at 150 Mhz). Dependence on the protein size is almost linear. Visual analysis of the generated surfaces shows no defects. The performance of the contour-buildup algorithm was compared with the Connolly algorithm implemented in his program PMSQ [8]. Table I shows that the CB algorithm is faster by more than an order of magnitude.

TABLE 1

Computation Times Required for Generation of All the Elements (i.e., Convex Spherical Patches, Torus Segments, and Concave Spherical Patches) of the Analytical Molecular Surface by the Contour-Buildup Method

Proteins	PDB code	Number of residues	Number of atoms	Calculation time (seconds)	
				CB	PQMS
Crambin	1crn	46	327	0.79	8.3
Plastocyanin	2plt	98	727	1.97	22.9
Phosphocarrier prot. III	1f3g	150	1115	3.23	41.2
Alpha-lytic protease	1p09	198	1392	4.01	55.1
Triose-phosphate isomerase	1tim	247	1870	5.98	177.7

Note. The times do not include triangulation of the surface patches. Computation times for Conolly's PQMS program are given for comparison.

We thank Dr. Rik Wierenga (EMBL, Heidelberg) for running the PMSQ program for five proteins. We also thank Jonathan Weider for careful reading of the manuscript.

REFERENCES

1. Richards, F. M. (1977) Areas, volumes, packing and protein structure, *Annu. Rev. Biophys. Bioeng.* **6,** 151–176.
2. Voorintholt, R., Kosters, M. T., Vegter, G., Vriend, G., and Hol, W. G. J. (1989) Very fast program for visualizing protein surfaces, channels and cavities, *J. Mol. Graphics* **7,** 243–245.
3. Jiang, F., and Kim, S.-H. (1991) "Soft docking": Matching of molecular surface cubes, *J. Mol. Biol.* **219,** 79–102.
4. Tuñón, I., Silla, E., and Pascual-Ahuir, J. L. (1992) Molecular surface area and hydrophobic effect, *Protein Eng.* **5,** 715–716.
5. Sitkoff, D., Sharp, K. A., and Honig, B. (1994) Accurate calculation of hydration free energies using macroscopic solvent models, *J. Phys. Chem.* **98**(7), 1978–1988.
6. Honig, B., and Nicholls, A. (1995) 1995 Classical Electrostatics in Biology and Chemistry Science, **268,** 1144–1149.
7. Connolly, M. L. (1983) Solvent-accessible surfaces of proteins and nucleic acids, *J. Appl. Crystallogr.* **221,** 709–713.
8. Connolly, M. (1993) The molecular surface package. *J. Mol. Graph.* **11,** 139–141.
9. Pearl, L. H., Honegger, A. (1983) Generation of molecular surfaces for graphic display, *J. Mol. Graphics* **1,** 9–12.
10. Lee, B., and Richards, F. M. (1971) The interpretation of protein structures: Estimation of static accessibility, *J. Mol. Biol.* **55,** 379–400.
11. Shrake, A., and Rupley, J. A. (1973) Environment and exposure to solvent of protein atoms: Lysozyme and insulin, *J. Mol. Biol.* **79,** 351–371.
12. Connolly, M. L. (1983) Analytical molecular surface calculation, *J. Appl. Crystallogr.* **16,** 548–558.
13. Richmond, T. J. (1984) Solvent accessible surface area and excluded volume in proteins: Analytical equations for over-

lapping spheres and implications for the hydrophobic effect, *J. Mol. Biol.* **178,** 63–89.

14. Perrot, G., Cheng, B., Gibson, K. D., Vila, J., Palmer, K. A., Nayeem, A., Maigret, B., and Scheraga, H. A. (1992) MSEED: A program for the rapid analytical determination of accessible surface areas and their derivatives, *J. Comp. Chem.* **13,** 1–11.
15. Abagyan, R. A., Totrov, M. M., and Kuznetsov, D. A. (1994) ICM: A new method for structure modeling and design: Applications to docking and structure prediction from the distorted native conformation, *J. Comp. Chem.* **15,** 488–506.
16. Eisenhaber, F., Argos, P., Linzaad, P., Argos, P., Sander, C., Scharf, C. (1995) The double cubic lattice method: efficient approaches to numerical integration of surface area and volume and to dot surface contouring of molecular assemblies, *J. Comp. Chem.* **16,** 273–284.
17. Biosoft (1995) ICM 2.0 program manual.
18. Greer, J., and Bush, B. (1978) Macromolecular shape and surface maps by solvent exclusion, *Proc. Natl. Acad. Sci. USA* **75,** 303–307.
19. Pearl, L. H., and Honegger, A. (1983) Generation of molecular surfaces for graphic display, *J. Mol. Graphics* **1,** 9–12.
20. Mueller, J. J. (1983) Calculation of scattering curves for macromolecules in solution and comparison with results of methods using effective atomic scattering factors, *J. Appl. Crystallogr.* **16,** 74–82.
21. Pavlov, M. Yu., and Fedorov, B. A. (1983) Improved technique for calculating X-ray scattering intensities in solution: Evaluation of the form, volume, and surface of a particle, *Biopolymers* **22,** 1507–1522.
22. Lorensen, W., and Cline, H. (1987). Marching cubes: A high resolution 3D surface construction algorithm, *Comput. Graphics* **21,** 163–169.
23. Meyer, A. Y. (1988) Molecular mechanics and molecular shape. V. On the computation of the bare surface area of molecules, *J. Comp. Chem.* **9,** 18.
24. Finney, J. L. (1978) Volume occupation, environment, and accessibility in proteins. Environment and molecular area of RNase-S, *J. Mol. Biol.* **119,** 415–441.