UJMM ONE + TWO

Undergraduate Journal of Mathematical Modeling: One + Two

Volume 8 | 2017 Fall 2017 Issue 1 | Article 5

Approximating Surface Area of Fluctuating Lipid Leaflets Using Weighted Grid Tessellation

Ahnaf Siddiqui
University of South Florida

Advisors:
David Milligan, Mathematics and Statistics
Sameer Varma, Cell Biology, Microbiology and Molecular Biology

Problem Suggested By: Sameer Varma

Abstract. The surface area per lipid is an important quantity useful in characterizing lipid bilayer structure and dynamics. However, simply dividing the flat surface area covered by the number of lipids does not give a sufficient characterization of surface dynamics in cases where parts of the lipid leaflets have curvaceous trajectories in three dimensions. An algorithm is proposed for modeling surface area per lipid using a three-dimensional tessellated grid. In this way both the vertical, as well as the more commonly considered the horizontal, fluctuations of lipids are used to accurately determine the area covered. This algorithm can reveal differences in dynamics between leaflets under different conditions that are otherwise invisible with a conventional flat surface study.

Keywords. lipid bilayer, molecular dynamics simulation, tessellation, surface area

Approximating Surface Area of Fluctuating Lipid Leaflets Using Weighted Grid Tessellation

PROBLEM STATEMENT

This paper presents a computational algorithm that solves the problem of approximating a single representative surface area for a fluctuating lipid leaflet by aggregating the surface over time using a weighted grid and then tessellating the grid.

MOTIVATION

The phospholipid bilayer that composes cell membranes is not static. Lipid molecules diffuse freely within a lipid leaflet, causing the cell membrane's structure to behave more like a fluid than a solid wall (Alberts, 1989). One consequence of this behavior is that a lipid layer undulates, causing its surface area to fluctuate over time (Waheed and Edholm, 2009). This complicates the quantification of lipid bilayer surface area, which is a characteristic aspect of lipid leaflet dynamics. Quantifying surface area is important for a number of studies involving lipid bilayers and membranes, such as studies involving viral diseases (Barz et al., 2008).
One method of quantification is to simply calculate the flat area covered by a number of lipids on a plane and divide that by the number of lipids in the set to obtain an average surface area per lipid (Feller et al., 1997). However, this method is very limited for a full dynamics analysis because it only covers area generalized in two dimensions. It does not consider the additional molecular-scale surface area caused by vertical elevation (see **Figure 1** on the following page).

The vertical component of surface area becomes especially important when a lipid bilayer's interactions with an external body may cause some phospholipids to move perpendicular to the plane of the membrane (i.e. the lipids are attracted to or repelled by the external body).

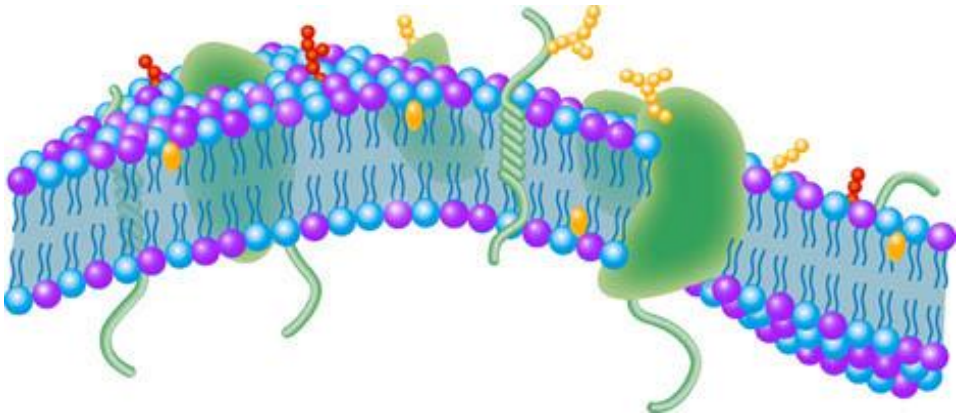

Figure 1: A lipid bilayer does not have a flat, static surface. Because the surface is vertically curved, accounting for only a two-dimensional horizontal plane would underestimate the surface area per lipid. Image source: http://www.scs.illinois.edu/~mlkraft/Kraft_group_research-2.html (Kraft).

The method proposed by this paper takes this vertical component into account by using a triangle mesh instead of a flat plane to calculate surface area. The mesh is constructed by tessellating a weighted three-dimensional grid into two triangles per vertical grid column. The grid is loaded using precise position data of individual phospholipid heads from a molecular dynamics simulation before it is tessellated. The surface area of the triangle mesh takes into account surface fluctuations over any period of time at a molecular scale and in three dimensions.

Mathematical Description and Solution Approach

Algorithm Summary. The following is a summary of the weighted-grid tessellation algorithm:

Inputs- simulation trajectory and grid cell width in trajectory spatial units

Output- average surface area per particle

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

1) Use the spatial limits of the trajectory and the given cell width to define a three-dimensional cubic grid that bounds all the coordinates in the trajectory.
2) Load the vertices of the grid defined in step 1 with weights based on the number of trajectory points contained in each vertex's grid cells and the distances from those points.
3) For each vertical column of vertices in the grid, find the vertex with the maximum weight in its column.
4) For each vertical column of cells in the grid, calculate the sum of the areas of two triangles formed by the four maximum-weight vertices of that cell column. The two triangles should be non-overlapping and their combined projection in the horizontal plane should be the square base of the grid cells. Cell columns that have one or more maximum weight vertices of zero weight are given an area of zero.
5) Sum all of the areas calculated in step 4 and divide this sum by the number of particles in the trajectory system to obtain the surface area per particle.

Defining the Grid. The weighted-grid tessellation algorithm takes as input a simulation trajectory and a grid cell width, and outputs the average surface area per particle in the trajectory. The trajectory consists of the position of each particle in the system at each time frame as three-dimensional coordinates. The grid cell width specifies the width of each cubic cell in the weighted grid that will be generated for the trajectory, determining the amount of grid spacing. A higher cell width will produce a lower resolution triangular mesh.

The three-dimensional weighted grid is fixed (i.e. it does not change shape, dimensions, or position throughout the trajectory). It is large enough to contain all of the trajectory points in all of the time frames. All of the cells in the grid are cubic and are equal in volume. Based on these constraints, a grid can be defined by the coordinates of its origin point in trajectory space, its cell width in the trajectory's spatial units, and the number of grid vertices in each dimension.

The cell width is already provided by the user, and the other two properties needed to define the grid are determined based on the cell width and the trajectory. The algorithm iterates through all of the particle positions in all of the frames of the trajectory to determine the minimum and maximum x, y, and z coordinates in the trajectory. The origin of the grid is

$$\textbf{origin} = (\min(x), \min(y), \min(z)) \tag{1}$$

and the dimensions (the number of grid vertices in each dimension) of the grid are

$$\left(\left\lceil\frac{\max(x)-\min(x)}{l}\right\rceil+1, \left\lceil\frac{\max(y)-\min(y)}{l}\right\rceil+1, \left\lceil\frac{\max(z)-\min(z)}{l}\right\rceil+1\right) \tag{2}$$

where ⌈ ⌉ indicates the ceiling function, and l is the width of a grid cell. The ceiling of (max(*dimension)* – min(*dimension*))/l gives the number of grid cells needed to contain all the trajectory coordinates in that dimension. Adding one to this value yields the number of grid vertices in that dimension. The number of grid vertices is used instead of the number of grid cells to define the grid dimensions since the data that will be stored in the grid, i.e. the weight of each grid point, is stored per grid vertex and not per grid cell.

Loading Weights. In order for the grid to be a weighted grid, a set of weights must be added to its definition. A three-dimensional array of weights in the dimensions of the grid is initialized to zeros. To load the grid with weights, all of the points in the trajectory are iterated through again. For each particle point in the trajectory, the grid cell containing the point is determined by calculating the (i, j, k) grid index of the cell's lowest corner vertex:

$$(i,j,k) = \left\lfloor\frac{\mathbf{x}-\mathbf{o}}{l}\right\rfloor \tag{3}$$

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

where ⌊ ⌋ indicates use of the floor function, *i*, *j*, and *k* are integers, **x** is the position vector of the trajectory point, **o** is the origin of the grid from eq. 1, and *l* is the width of a grid cell. From the cell origin indices (*i*,, *j*, *k*), the indices of the other seven grid vertices that form the grid cell can be determined by incrementing *i*, *j*, *k* and all combinations of the three by 1. The trajectory space coordinates of a vertex can be found by simply multiplying its indices by the cell width. After determining a trajectory point's containing grid cell, the weights of the eight grid vertices are increased by weight w_i based on the following formula:

$$w_i = D^2 - ||\mathbf{x_i} - \mathbf{g}||^2 \tag{4}$$

where w_i is the weight contributed to a grid vertex by trajectory point *i* (which must be a point contained by one of the grid cells adjacent to the vertex), *D* is the magnitude of the diagonal (distance between opposite vertices) of a cube in the grid, $\mathbf{x_i}$ is the position vector of trajectory point *i*, and **g** is the position vector of the grid vertex in trajectory space. D^2 is pre-computed as

$$D^2 = l^2 + l^2 + l^2 = 3l^2 \tag{5}$$

where *l* is the width of a grid cell.

In (4), $||\mathbf{x_i} - \mathbf{g}||$ is the distance between trajectory point *i* and a grid vertex. Because point *i* must be contained by a grid cell containing the grid vertex, its distance from the grid vertex can be no greater than *D*. Therefore, subtracting the distance $||\mathbf{x_i} - \mathbf{g}||$ from *D* yields a positive value that increases in magnitude with closeness of point *i* to the grid vertex. These properties also hold true for D^2 and $||\mathbf{x_i} - \mathbf{g}||^2$; the distance squared between point *i* and the grid vertex can be no greater than D^2, and therefore subtracting $||\mathbf{x_i} - \mathbf{g}||^2$ from D^2 yields a positive value that increases

in magnitude with closeness of point *i* to the grid vertex. Therefore, we can also use the squared version to weigh by closeness. We choose to use the squared version instead of the more intuitive version because the square root operation needed to compute $||\mathbf{x_i} - \mathbf{g}||$ is computationally expensive and can increase numerical error.

For any grid vertex at position **g**, its total weight w_{total} is the sum of all the weights w_i contributed by all the trajectory points *i* at positions $\mathbf{x_i}$ contained in the grid cells adjacent to the grid vertex. Substituting eq. 4 and eq. 5:

$$w_{total} = \sum_{i=0}^{n} w_i = \sum_{i=0}^{n} (3l^2 - ||\mathbf{x_i} - \mathbf{g}||^2) \tag{6}$$

where *n* is the number of such particles *i*. This weighting system is illustrated in **Figure 2** on the following page.

We can now create a complete definition for any weighted grid $W_{X,l}$ of a trajectory *X* and cell width *l*:

$$W_{X,l} = \left\{ w_{i,j,k} \in \mathbb{R} \,\middle|\, w_{i,j,k} = \textstyle\sum_{m=0}^{n} (3l^2 - ||\mathbf{x_{i,j,k,m}} - \mathbf{g_{i,j,k}}||^2) \right\} \tag{7}$$

where *i, j, k* are integer grid indices such that

$$(0,0,0) \leq (i,j,k) \; < \; (\left\lceil \frac{\max(x) - \min(x)}{l} \right\rceil + 1, \left\lceil \frac{\max(y) - \min(y)}{l} \right\rceil + 1, \left\lceil \frac{\max(z) - \min(z)}{l} \right\rceil + 1)$$

where max(x), max(y), max(z) are the maximum independent x, y, and z coordinates respectively, of all points in the trajectory X. Also, $\{\boldsymbol{x}_{\mathbf{i,j,k,0}}, \boldsymbol{x}_{\mathbf{i,j,k,1}}, \boldsymbol{x}_{\mathbf{i,j,k,2}}, \ldots, \boldsymbol{x}_{\mathbf{i,j,k,n-1}}, \boldsymbol{x}_{\mathbf{i,j,k,n}}\}$ is

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

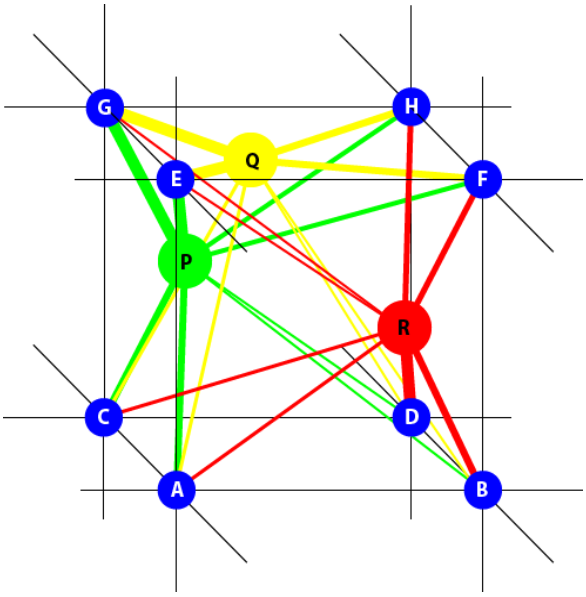

Figure 2: Grid vertices A, B, C, …, H are weighted by particles P, Q, and R. Because particle P is closer to vertex E than particle R, E gains a higher weight from P than it does from R. Because vertex B is closer to particle R than vertex E, B gains a higher weight from R than E does. However, vertex E has two particles closer to itself than B, causing it to have a higher total weight within the system shown than vertex B. Similar relative weights exist for the other vertices. Besides the grid cell shown, each vertex also gains weights from particles in seven other grid cells adjacent to it (unless the vertex is on an edge or corner of the grid).

the position coordinate set of the n trajectory points in X that are contained in grid cells adjacent to the grid where max(*x*), max(*y*), max(*z*) are the maximum independent *x*, *y*, and *z* coordinates respectively, of all points in the trajectory X. Also, $\{\boldsymbol{x}_{\mathbf{i,j,k,0}}, \boldsymbol{x}_{\mathbf{i,j,k,1}}, \boldsymbol{x}_{\mathbf{i,j,k,2}}, \ldots, \boldsymbol{x}_{\mathbf{i,j,k,n-1}}, \boldsymbol{x}_{\mathbf{i,j,k,n}}\}$ is the position coordinate set of the *n* trajectory points in *X* that are contained in grid cells adjacent to the grid vertex at index (i, j, k). In addition, $\mathbf{g}_{\mathbf{i,j,k}}$ is the position vector in the trajectory space of the grid vertex at index (i, j, k) such that

$$\mathbf{g}_{\mathbf{i,j,k}} = (\min(x), \min(y), \min(z)) + (il, jl, kl)$$

(with $\min(x), \min(y), \min(z)$ being the minimum independent *x*, *y*, and *z* coordinates, respectively, of all points in the trajectory *X*).

Maximizing Weights. The weighting routine aggregates all the position data from all the frames of the trajectory. As a consequence, the regions in the grid with higher weights represent regions in trajectory space that are visited more frequently by a greater number of particles throughout the frames of the trajectory than areas with lower weights. The higher weight points are essentially "hotspots" of particle activity. Therefore, the grid vertices with highest weight serve as good representatives of the trajectory's spatial distribution integrated over time.

To consider these hotspots, the maximum weight vertex is determined for each vertical column of vertices in the grid (in the case of a lipid bilayer, vertical should be the axis most perpendicular to the approximate horizontal plane of the bilayer). The set of these maximum weights W_{max} is defined as a subset of all the vertex weights in the grid defined in eq. 7 as follows:

$$W_{max} = \left\{ w_{i,j,k} \in W_{X,l} \;\middle|\; \forall w_{i,j,h} \in W_{X,l}, w_{i,j,k} \geq w_{i,j,h} \right\} \tag{8}$$

where *i*, *j*, *k*, and *h* are integer grid indices of the grid vertex corresponding to a weight. In other words, each element $w_{i,j,k}$ of W_{max} is the maximum of all the weights in the vertical column at horizontal index (*i*, *j*) of the weighted grid $W_{X,l}$. We can define a relation *K*(*i*, *j*) from the set of horizontal indices (*i*, *j*) to the set of vertical indices *k* of the grid where *K*(*i*, *j*) yields the vertical indices of the vertices with maximum weight at horizontal index (*i*, *j*):

$$K(i,j) = \left\{ k_{i,j} \in \mathbb{Z} \,\middle|\, W(i,j,k) \in W_{max}(i,j) \wedge W(i,j,k) > 0 \right\} \tag{9}$$

Where $W(i,j,k)$ yields the weight of the grid vertex at indices (i,j,k) and $W_{max}(i,j)$ yields the maximum weight(s) in grid column (*i*, *j*). We add the extra condition $W(i,j,k) > 0$ because we do not want to include any grid column of zero total weight when tessellating the grid.

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

Tessellating the Grid. For each vertical cell column in the grid $W_{X,l}$, we produce two triangles from the four vertices that have the maximum weight of each vertex column of that cell column. A vertical cell column can be defined by its lowest horizontal indices (*i*, *j*), and the other three vertex columns of the cell column can be obtained by incrementing *i*, *j*, or both by 1. Using eq. 9 as a definition for $K(i, j)$, when any of the four vertex columns is the empty set, then the cell column is not tessellated. Otherwise, the trajectory space coordinates **g** of the grid vertex with highest weight at index (*i*, *j*) are calculated as follows:

$$\mathbf{g_{i,j}} = l(i, j, \min(K(i, j))) \tag{10}$$

Recall that *l* is the width of a grid cell, and $K(i, j)$ is a set that can have multiple *k* indices which all correspond to grid vertices with equal but maximum weights. In order to have only one point per vertex column in the triangle mesh, the lowest *k* index in $K(i, j)$ is taken arbitrarily (and hence the min function in eq. 10). Then, the area $A_{i,j}$ for a cell column at (*i*, *j*) is calculated as the sum of the areas of two triangles formed by the four **g** vectors of the cell column as follows:

$$A_{i,j} = \frac{\left\|\left(\mathbf{g_{i,j+1}} - \mathbf{g_{i,j}}\right) \times \left(\mathbf{g_{i+1,j+1}} - \mathbf{g_{i,j}}\right)\right\|}{2} + \frac{\left\|\left(\mathbf{g_{i+1,j+1}} - \mathbf{g_{i,j}}\right) \times \left(\mathbf{g_{i+1,j}} - \mathbf{g_{i,j}}\right)\right\|}{2} \tag{11}$$

when $K(i, j)$ is not empty and $A_{i,j} = 0$, if $K(i, j) = \{\emptyset\}$. This triangulation of cell columns is illustrated in **Figure 3** on the following page. The magnitude of the cross product divided by 2: **a**×**b**/2, is the area of the area of triangle **a**, **b**, O. The total surface area A_{total} of the entire weighted grid is the sum of the triangulated areas $A_{i,j}$ of all the vertical cell columns in the grid:

$$A = \sum_{\substack{0 \le i < m \\ 0 \le j < n}} A_{i,j} \tag{12}$$

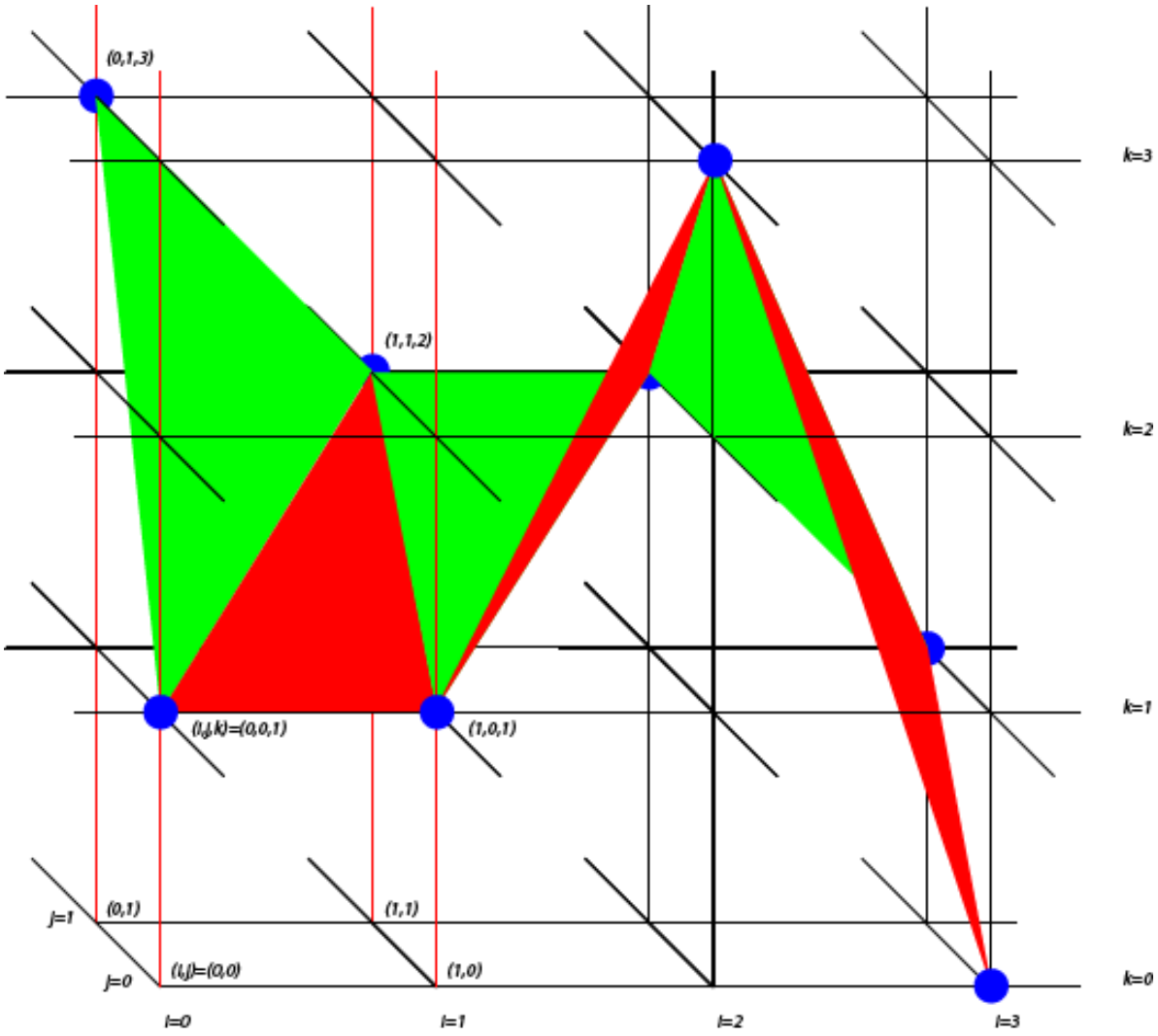

Figure 3: The tessellation of three vertical cell columns (i, j): columns (0, 0), (1, 0), and (2, 0). The vertical vertex columns that make up cell column (0, 0) are highlighted in red. For each vertex column (i, j), a highest weight vertex of that column with indices (i, j, k ∈ *K*(i, j)) is shown as a solid blue circle. Two triangles, one shown in green and the other in red, are formed for each cell column between its four highest weight vertices.

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

where $m =$ number of grid cells in dimension $x = \left\lceil \frac{\max(x) - \min(x)}{l} \right\rceil$ and

$n =$ number of grid cells in dimension $y = \left\lceil \frac{\max(y) - \min(y)}{l} \right\rceil$ (see eq. 2).

The final output of the algorithm is

$$Area\ per\ particle = \frac{A}{n_X} = \frac{\sum_{\substack{0 \le i < m \\ 0 \le j < n}} A_{i,j}}{n_x} \tag{13}$$

where n_x is the number of particles in the trajectory. See eq. 11 for $A_{i,j}$.

DISCUSSION

An implementation of this algorithm, lltessellator, was written in C (see Appendix) (Siddiqui, 2015). Version 1.0.0-beta of this program was run on data from a lipid bilayer system (Duro, 2015) that was simulated using Gromacs 4 (Berendsen et al., 1995). The lipid bilayer was simulated on a charged nano-porous substrate and its trajectory was split into two trajectories, one for the distal leaflet (the leaflet further from the substrate) and one for the proximal leaflet (the leaflet closer to the substrate) of the bilayer. The program was run separately on each leaflet trajectory. Each trajectory consisted of 256 phosphate atoms over 15000 frames.

An attempt was made through trial and error to use the smallest cell width possible for the highest triangular mesh resolution without leaving any undesired gaps in the grid (i.e. vertex columns with zero weight). Because the particles in the trajectories used take up the entire horizontal plane of the trajectory bounds, any gaps in the grid would be inside the lipid leaflet and would therefore create a hole when calculating surface area. A cell width of 8 Å (angstroms) was used for both trajectories without any resulting gaps. The final outputs of the program are shown in **Table 1**, and more detailed output data is provided in the Appendix. The first thing to

notice is that the program produces reasonable values for surface area per lipid, within the range of 60-80 $Å^2$ that is typical of membrane models in the literature (Barz et al., 2008). This suggests that the algorithm is numerically viable in the magnitude of its outputs. In addition, it was expected that the proximal leaflet would have a more curved surface fluctuation and therefore a higher surface area due to its closer interactions with the charged substrate than the distal leaflet. The results do agree with this hypothesis, with about a 2.47 % difference in the surface areas per lipid.

	Distal leaflet	**Proximal leaflet**
Total surface area ($Å^2$)	17476	17907
Area per lipid ($Å^2$)	68.265	69.948

Table 1: The total surface area and area per lipid of the triangular tessellation of the weighted grid for the distal leaflet and proximal leaflet trajectories. Å denotes angstrom.

Using the other data produced by the program in the Appendix, a more traditional approach can be used to calculate the surface per lipid as

$$\frac{(\max(x) - \min(x))(\max(y) - \min(y))}{number\ of\ particles} \tag{14}$$

Using this method, results of 61.449 A^2 and 61.385 A^2 are obtained for distal and proximal respectively, with a percent difference of about 0.1% that is much smaller than the discriminability yielded by the tessellation method. This is the case because there is surface fluctuation in the *z*-axis, which is not accounted for by eq. 14. The tessellation method factors in vertical fluctuation, and is therefore more representative of and better at identifying differences in phospholipid dynamics.

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

The weighted-grid tessellation algorithm can be further analyzed by visualizing the tessellation. On the following page **Figure 4** shows surface plots of the heightmap, or the maximum-weight z-indices (min($K(x, y)$), (K is as in (9)) for each trajectory's weighted grid. These surface plots roughly represent the tessellated surface for which the area is computed by lltessellator. The figures reveal that the proximal leaflet's heightmap has more bumps and a wider valley, causing its surface area to be higher. This representation can be interpreted as a more curvaceous spatial distribution of the proximal leaflet's trajectory. The most popular hotspots of activity of phosphate heads in the proximal leaflet are more spread along the vertical axis than in the distal leaflet.

This difference in spatial distribution implies that the aggregate movement of lipids in a lipid leaflet will change perceivably in response to external bodies. If the horizontal component is kept constant, as it mostly is here, the magnitude of the surface area per lipid produced by this analysis correlates to the amount by which the movement of lipids in a leaflet is biased to either side of the membrane plane. This value can then be taken as a quantitative representation of the extent by which a lipid leaflet is attracted to or repulsed by an external body. In this case, because of the depressed region in the heightmap, it appears that the phospholipids in the system are attracted to the nano-porous substrate, with the lipids closer to the substrate feeling a stronger attraction.

Conclusion and Recommendations

These results support the hypothesis that this weighted-grid tessellation algorithm can uniquely and appropriately quantify the average surface area over a period of time of a nearly planar system of particles that fluctuate in three dimensions. However, a single comparison trial using a pre-release implementation is not enough to satisfyingly verify the integrity of this algorithm.

More rigorously applied studies must be conducted, involving comparisons with experimental and literature data, to properly test the reliability of this method.

The algorithm's demonstrated ability to discriminate between the dynamics of a system under different conditions should be explored further, such as with lipid bilayers under other condition parameters as well as other biological systems with similar fluctuating surfaces. While this paper mostly focused on the mathematical aspect of the algorithm, its biological applications and the biological significance of its surface area data should be evaluated in studies with more biological depth. The algorithm's potential applications in non-biological fields should also be explored.

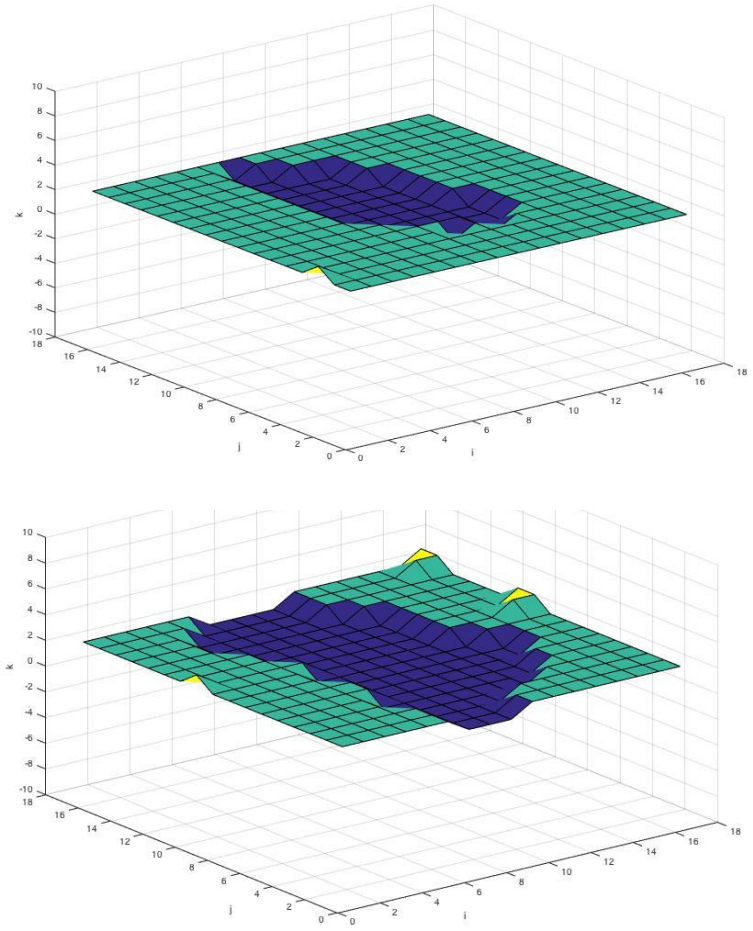

Figure 4: Distal leaflet. Bottom: Proximal leaflet. Surface graphs of (i, j) vs. k where (i, j, k) are indices in the weighted grid and k ∈ *K* defined in eq. 9, for the tessellation of the distal leaflet and proximal leaflet of a lipid bilayer simulation on a charged nano-porous substrate.

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

In addition, the algorithm itself can be developed and improved in numerous ways, such as by adding new weighting and tessellation methods. Alternative tessellation procedures could use shapes other than triangles, a non-cubic grid, or triangles with a different triangulation algorithm such as a three-dimensional Delaunay triangulation (Joe, 1991). It is also important to evaluate the performance of the algorithm, particularly in terms of its time complexity. Although an average case complexity of $O(n)$ is expected due to the linearity of the data processing, input trajectories of unexpected spatial configurations or unexpected cell widths can potentially cause variations in performance.

NOMENCLATURE

Symbol	Description
x, y, z	It is assumed that in a lipid bilayer trajectory, x and y are the horizontal axes in the major plane of the bilayer and z is the vertical axis perpendicular to that plane
i, j, k	Integer indexes of vertices in a 3D grid in the x-axis, y-axis, and z-axis directions, respectively
max() and min()	Represents the maximum and minimum value, respectively, of a variable or set
ceiling() or $\lceil \quad \rceil$	Represents the smallest integer following a real number
floor() or $\lfloor \quad \rfloor$	Represents the largest integer preceding a real number
$\mathbb{Z}$	The set of all integers
$\mathbb{R}$	The set of all real numbers
Å	Angstrom

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886

REFERENCES

Alberts, B. (1989). *Molecular Biology of the Cell* (Garland Pub.). Barz, B., Wong, T.C., and Kosztin, I. (2008). Membrane curvature and surface area per lipid affect the conformation and oligomeric state of HIV-1 fusion peptide: A combined FTIR and MD simulation study. Biochimica et Biophysica Acta (BBA) - Biomembranes *1778*, 945-953.

Berendsen, H.J.C., van der Spoel, D., and van Drunen, R. (1995). GROMACS: A message-passing parallel molecular dynamics implementation. Computer Physics Communications *91*, 43-56.

Duro, N. (2015). 512 POPC with CH4 support surface hydrolyated 5/nm2 in water (University of South Florida).

Feller, S.E., Venable, R.M., and Pastor, R.W. (1997). Computer Simulation of a DPPC Phospholipid Bilayer: Structural Changes as a Function of Molecular Surface Area. Langmuir *13*, 6555-6561.

Joe, B. (1991). Construction of three-dimensional Delaunay triangulations using local transformations. Computer Aided Geometric Design *8*, 123-142.

Kraft, M.L. *Plasma Membrane Organization and Function* (University of Illinois at Urbana-Champaign).

Siddiqui, A. (2015). lltessellator.

Waheed, Q., and Edholm, O. (2009). Undulation Contributions to the Area Compressibility in Lipid Bilayer Simulations. Biophysical Journal *97*, 2754-2760.

APPENDIX

The source code for lltessellator, a C implementation of the algorithm presented in this paper, can be found at https://github.com/luky1971/lltessellator.

The following is output from lltessellator when run on the simulation trajectories described in the paper (see Discussion section).

Distal

Grid cell width: 8 Å

min(x) = 0.000000, min(y) = 0.000000, min(z) = 79.01000

max(x) = 124.77000, max(y) = 126.08000, max(z) = 107.45000

Grid dimensions (x × y × z vertices): 17 ×17 × 5

Total tessellated surface area: 17475.8026

Tessellated surface area per particle: 68.2649

Proximal

Grid cell width: 8 Å

min(x) = 0.000000, min(y) = 0.000000, min(z) = 40.90000

max(x) = 124.62000, max(y) = 126.10000, max(z) = 71.04000

Grid dimensions (x × y × z vertices): 17 ×17 × 5

Total tessellated surface area: 17906.7169

Tessellated surface area per particle: 69.9481

http://scholarcommons.usf.edu/ujmm/vol8/iss1/5
DOI: http://doi.org/10.5038/2326-3652.8.1.4886