Bioinformatics, 39(2), 2023, btad083
https://doi.org/10.1093/bioinformatics/btad083
Advance Access Publication Date: 8 February 2023
Applications Note

Structural bioinformatics

APL@voro—interactive visualization and analysis of cell membrane simulations

Martin Kern[1], **Sabrina Jaeger-Honz**[1], **Falk Schreiber**[1,2,*] **and Bjorn Sommer**[3,*]

[1]Department of Computer and Information Science, University of Konstanz, Konstanz 76484, Germany, [2]Faculty of Information Technology, Monash University, Clayton, VIC 3800, Australia and [3]Royal College of Art, School of Design, London SW7 2EU, UK

*To whom correspondence should be addressed.

Associate Editor: Lenore Cowen

Received on May 31, 2022; revised on November 11, 2022; editorial decision on December 15, 2022; accepted on February 7, 2023

Abstract

Summary: Molecular dynamics (MD) simulations of cell membranes allow for a better understanding of complex processes such as changing membrane dynamics, lipid rafts and the incorporation/passing of macromolecules into/through membranes. To explore and understand cell membrane compositions, dynamics and processes, visual analytics can help to interpret MD simulation data. APL@Voro is a software for the interactive visualization and analysis of cell membrane simulations. Here, we present the new APL@Voro, which has been continuously developed since its initial release in 2013. We discuss newly implemented algorithms, methodologies and features, such as the interactive comparison of related simulations and methods to assign lipids to either the upper or lower leaflet.

Availability and implementation: The current open-source version of APL@Voro can be downloaded from http://aplvoro.com.

Contact: bjoern@cellmicrocosmos.org or falk.schreiber@uni-konstanz.de

1 Introduction

Cell membranes are central biological structures that separate the interior of a cell from the outside, thereby protecting the cell from its environment (Campbell, 2002). They consist of a lipid bilayer and contain various membrane proteins and small molecules. The properties of a cell membrane can change over time, for example, during different stages of the development of the cell or due to changes in its environment. Molecular dynamics (MD) simulations of cell membranes are a common approach to understand membranes and related processes. This includes using MD simulations to understand and prioritize cell-penetrating peptides (Tran *et al.*, 2021), cell membrane pore sealing (Zhang *et al.*, 2019), membrane permeability (Venable *et al.*, 2019) and many more processes.

Software tools for the analysis of cell membrane simulations play an important role when trying to understand the simulations. Examples are GridMAT-MD (Allen *et al.*, 2009), FATSLiM (Buchoux, 2017), MEMBPlugin (Guixà-González *et al.*, 2014), MemSurfer (Bhatia *et al.*, 2019) and APL@Voro (Lukat *et al.*, 2013). GridMAT-MD, like most tools, is a command-line tool that can calculate *membrane thickness* (*MT*) and *area per lipid* (*APL*) using algorithms similar to those used in APL@Voro. FATSLiM, another command-line tool, offers advanced algorithms to handle strongly curved membranes and vesicles. The output of both tools can then be visualized as 2D graphs with external packages like *Xmgrace* or Matplotlib (Hunter, 2007), but they do not provide direct internal visualization of membrane structures. To visualize the membrane itself additional software is required, such as VMD (Humphrey *et al.*, 1996) or UCSF Chimera (Pettersen *et al.*, 2004). MEMBPlugin is a plugin for VMD that can calculate APL, MT, order parameter, APL distribution and cholesterol tilt angle distribution. MEMBPlugin offers a user interface and includes the option to plot generated data. MemSurfer is a Python API that uses 3D point coordinates for Delaunay triangulations and surface parameterization to represent membrane surfaces. This procedure gives the user direct access to the membrane surface to enable calculations, such as MT and APL. The visualizations are however non-interactive which limits the exploration of the results. APL@Voro is developed with the goal of offering a software that unifies the analysis, interactive visualization, exploration and comparison of multiple membrane simulations. Table 1 contains a comparison of the different tools and their features.

2 Methods and implementation

APL@Voro uses data obtained from GROMACS simulations (Abraham *et al.*, 2015) which are processed and then visualized for the analysis. It supports .PDB, .NDX, .XTC and .TRR formats. Other formats can be converted by using external tools such as MDAnalysis (Gowers *et al.*, 2016; Michaud-Agrawal *et al.*, 2011).

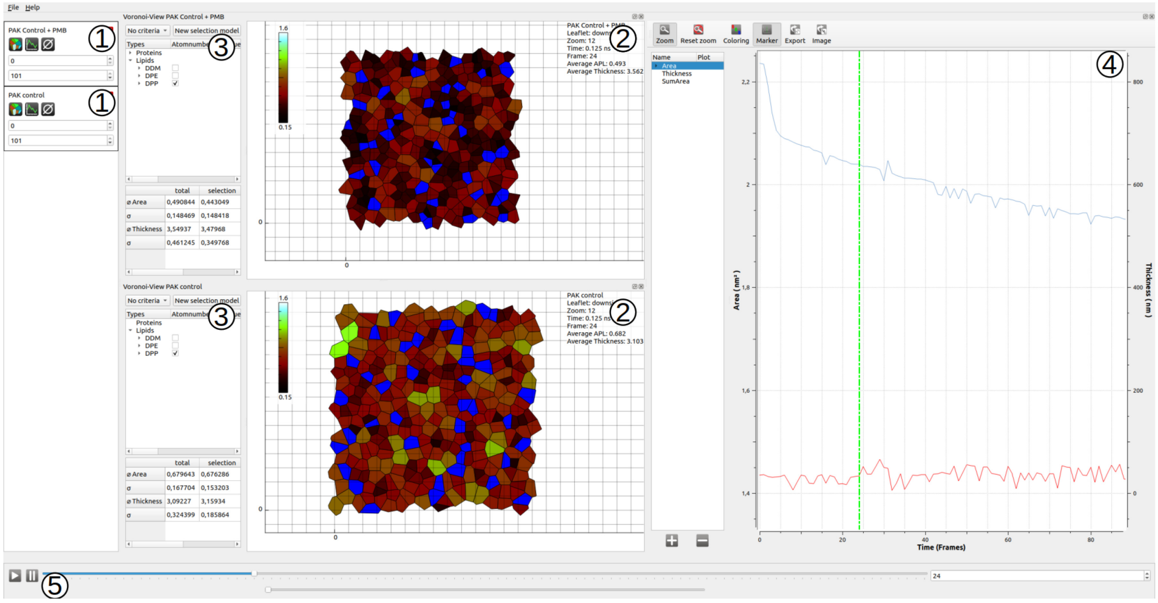
The data processing can be separated into the following steps:

1. leaflet detection,
2. Delaunay triangulation for each leaflet,

Table 1. A comparison of different tools and their features

Category	Feature	MDAnalysis	FATSLiM	GridMAT-MD	MEMB-Plugin	MEMSurfer	APL@Voro
Membrane types	Flat membranes	✓	✓	✓	✓	✓	✓
	Curved membranes	✓	✓	✗	✗	✓	✗
	Vesicles	✓	✓	✗	✗	✗	✗
Comparisons	Multiple membrane comp.	✗	✗	✗	✗	✓	✓
Access	GUI	✗	✗	✗	✓	✗	✓
	Command line	✓	✓	✓	✓	✗	✓
	API	✗	✗	✗	✗	✓	✗
Visualizations	Voronoi	✗	✓	✗	✓	✓	✓
	Internal plotting	✓	✗	✗	✓	✗	✓
Interactions	Mouse interaction	✗	✗	✗	✓	✗	✓
	Linked views	✗	✗	✗	✗	✗	✓
	Interactive molecule selection	✗	✗	✗	✓	✗	✓

tick: supported by the tool; cross: not supported by the tool.



PAK Control + PMB

0

101

PAK control

0

101

Voronoi-View PAK Control + PMB

No criteria | New selection model

Types	Atomnumber
Proteins	
Lipids	
DDM	□
DPE	□
DPP	☑

	total	selection
⌀ Area	0,490844	0,443049
σ	0,148469	0,148418
⌀ Thickness	3,54937	3,47968
σ	0,461245	0,349768

PAK Control + PMB
Leaflet: downs
Zoom: 12
Time: 0.125 ns
Frame: 24
Average APL: 0.493
Average Thickness: 3.562

Voronoi-View PAK control

No criteria | New selection model

Types	Atomnumber
Proteins	
Lipids	
DDM	□
DPE	□
DPP	☑

	total	selection
⌀ Area	0,679643	0,676286
σ	0,167704	0,153203
⌀ Thickness	3,09227	3,15934
σ	0,324399	0,185864

PAK control
Leaflet: downsi
Zoom: 12
Time: 0.125 ns
Frame: 24
Average APL: 0.682
Average Thickness: 3.103

Name	Plot
Area	
Thickness	
SumArea	

Fig. 1. APL@Voro with two different membrane simulations. (1) Local simulation control panels which are used to open new views and synchronize simulations, (2) Voronoi views visualizing the APL using the linear optimized color scale (LOCS) (Herman and Levkowitz, 1992) (maximum: White—1.6 nm^2, minimum: Black—0.15 nm^2) and the blue cells represent user-selected areas (selected in 3., DPP), (3) selection panel with detailed information on each lipid, including area, local thickness and the key atom position, (4) 2D plotting view showing the average APL of the outer leaflet over time for both simulations and (5) global simulation control

3. use of Delaunay triangulation to insert non-membrane atoms (e.g. protein) and calculate membrane thickness,
4. Voronoi diagram construction from Delaunay triangulation, and
5. use of Voronoi diagram to calculate APL.

The new version of APL@Voro is now capable of loading multiple simulations to compare them side by side. Each imported simulation appears in a list from where views can be opened. New views appear in a docking area that can be arranged by the user. Figure 1 shows an example of two imported simulations. Each simulation has a Voronoi view showing the lower leaflet. The 2D view contains the average area per lipid of the outer leaflet over time for both simulations. APL@Voro can synchronize simulations along frames which can be used to line up certain events (e.g. a substance entering the membrane) or account for different simulation lengths or time steps.

APL@Voro originally detected leaflets only by estimating the orientation of lipids (Lukat *et al.*, 2013). This approach is imprecise in some cases, e.g. for complex lipids such as lipid A or around membrane proteins. The newly added position-based approach fits a surface to the membrane using polynomial regression and uses the position of key atoms relative to that surface to assign lipids to a leaflet. Its runtime complexity is $\mathcal{O}(nd^2)$, where n is the number of lipid atoms, and d is the degree of the polynomial. This approach is in practice a bit slower than the orientation-based method (around 4%), but it proved more reliable.

The trajectory needs to be loaded into computer memory to enable responsive interactions, requiring a decent amount of memory space. Therefore, the new version allows the user to exclusively load relevant parts of a trajectory, as well as skip frames in order to manage memory usage. Loading 300 frames of a membrane with 23 764 atoms takes an average of 3.5 s on a system running Ubuntu 20.04 with a AMD Ryzen 7 5800X CPU and 32 Gb of 3200 MHz RAM.

The Voronoi view has been overhauled. The legend has now a fixed size and will be displayed in the top left corner. The top right corner is used to display additional information on the visualized leaflet. Also the grid rendering has been improved, the view can be dragged around by holding the middle mouse button. Hovering over a Voronoi cell will display information on the associated lipid next to the pointer. The original color scale options (rainbow or none) have been extended by three other color scale options.

APL@Voro can also be used as a pure command-line tool. Due to a lack of interactivity, there is no need to load trajectories into memory. Using the command-line functions will now only load one frame for analysis at a time to save memory.

To summarize the previous paragraphs, APL@Voro 3.3.3 offers the following new features in comparison to the old APL@Voro version:

- load multiple simulations for side-by-side comparison,
- frame synchronization,
- position-based leaflet detection,
- overhauled Voronoi view: improved visualization, easier navigation, larger area,
- possibility to load only parts of the trajectory,
- additional color scales (rainbow, LOCS, heated object and linear grey),
- optimization of memory usage (i.e. code optimization and 3D view removed) and
- optimized memory usage (by limiting the amount of memory used during parsing) when using console.

3 Usage

APL@Voro calculates APL and MT and creates an interactive visualization of membrane simulations. The average APL is related to other membrane properties like acyl chain ordering, compressibility and molecular packaging (Patra *et al.*, 2003). Changes in the average APL and MT, which are strongly related (Moradi *et al.*, 2019), can be indicative of various processes taking place in the membrane, e.g. phase changes.

APL@Voro visualizes membrane leaflets as a Voronoi diagram in the so-called Voronoi views where APL, MT and lipid neighborhood can be mapped to one of several color scales. Colors can also be manually assigned to lipid types. The Voronoi view is complemented by a table that contains detailed information on each lipid as well as averages for the whole leaflet and the current selection (see Fig. 1(3)). The Voronoi view can reveal lipid rafts, protein aggregation, local anomalies, etc. The user can define a selection based on a combination of conditions such as lipid type, MT, APL and neighbors. This can be used to track lipids with certain properties, e.g. lipids that are in the gel phase. These selections are also used for 2D views where the user can plot the average membrane thickness and area per lipid for all open simulations over time (Fig. 1(4)).

The results can be exported in various ways for further analysis: Voronoi views and 2D views as image file, simulation frames as either .TXT or .XML file, and 2D plots as .XVG file.

APL@Voro is available as executable versions for Linux as well as Windows. For Mac OS X (including M1 chipset), Windows versions can be used via CrossOver (https://www.codeweavers.com/). In the future, we are planning to provide native Mac OS X versions as well.

4 Discussion and future development

APL@Voro provides analysis and interactive visualization of cell membrane simulations. The new developments enable direct comparison of multiple membrane simulations within one session, better algorithms (e.g. for leaflet detection) and an updated intuitive graphical user interface. APL@Voro therefore allows for easy analysis of MD simulations and helps in exploring changing membrane dynamics, lipid rafts and membrane interactions of macromolecules. It is a good basis for MD simulation analysis but also allows for future developments such as exploring density profiles, diffusion coefficients, membrane curvature and deuterium order parameters.

Funding

This work was partly supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) [Project-ID 251654672—TRR 161].

Conflict of Interest: none declared.

References

Abraham,M.J. *et al.* (2015) GROMACS: high performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, **1**, 19–25.

Allen,W. *et al.* (2009) GridMAT-MD: a grid-based membrane analysis tool for use with molecular dynamics. *J. Comput. Chem.*, **30**, 1952–1958.

Bhatia,H. *et al.* (2019) MemSurfer: a tool for robust computation and characterization of curved membranes. *J. Chem. Theory Comput.*, **15**, 6411–6421.

Buchoux,S. (2017) FATSLiM: a fast and robust software to analyze MD simulations of membranes. *Bioinformatics*, **33**, 133–134.

Campbell,N.A. (2002) *Biology*. The Benjamin-Cummings Publishing Company, Reading, Massachusetts, USA.

Gowers,R.J. *et al.* (2016) MDAnalysis: a Python package for the rapid analysis of molecular dynamics simulations. In: S. Benthall and S. Rostrup (ed.) *Proceedings of the 15th Python Science Conference, Austin, Texas, USA*, pp. 98–105.

Guixà-González,R. *et al.* (2014) MEMBPLUGIN: studying membrane complexity in VMD. *Bioinformatics*, **30**, 1478–1480.

Herman,G.T. and Levkowitz,H. (1992) Color scales for image data. *IEEE Comp. Graph. Appl.*, **12**, 72–80.

Humphrey,W. *et al.* (1996) VMD – visual molecular dynamics. *J. Mol. Graphics*, **14**, 33–38.

Hunter,J.D. (2007) Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.*, **9**, 90–95.

Lukat,G. *et al.* (2013) APL@voro: a Voronoi-based membrane analysis tool for GROMACS trajectories. *J. Chem. Inf. Model.*, **53**, 2908–2925.

Michaud-Agrawal,N. *et al.* (2011) MDAnalysis: a toolkit for the analysis of molecular dynamics simulations. *J. Comput. Chem.*, **32**, 2319–2327.

Moradi,S. *et al.* (2019) Shedding light on the structural properties of lipid bilayers using molecular dynamics simulation: a review study. *RSC Adv.*, **9**, 4644–4658.

Patra,M. *et al.* (2003) Molecular dynamics simulations of lipid bilayers: major artifacts due to truncating electrostatic interactions. *Biophys. J.*, **84**, 3636–3645.

Pettersen,E. *et al.* (2004) UCSF chimera–a visualization system for exploratory research and analysis. *J. Comput. Chem.*, **25**, 1605–1612.

Tran,D. *et al.* (2021) Using molecular dynamics simulations to prioritize and understand AI-generated cell penetrating peptides. *Sci. Rep.*, **11**, 10630.

Venable,R. *et al.* (2019) Molecular dynamics simulations of membrane permeability. *Chem. Rev.*, **119**, 5954–5997.

Zhang,L. *et al.* (2019) Molecular dynamics simulation of cell membrane pore sealing. *Extreme Mech. Lett*, **27**, 83–93.