ST-Analyzer: A Web-Based User Interface for Simulation Trajectory Analysis

Jong Cheol Jeong,[a] Sunhwan Jo,[a] Emilia L. Wu,[a] Yifei Qi,[a] Viviana Monje-Galvan,[b] Min Sun Yeom,[c] Lev Gorenstein,[d] Feng Chen,[d] Jeffery B. Klauda,[b] and Wonpil Im*[a]

Molecular dynamics (MD) simulation has become one of the key tools to obtain deeper insights into biological systems using various levels of descriptions such as all-atom, united-atom, and coarse-grained models. Recent advances in computing resources and MD programs have significantly accelerated the simulation time and thus increased the amount of trajectory data. Although many laboratories routinely perform MD simulations, analyzing MD trajectories is still time consuming and often a difficult task. ST-analyzer, http://im.bioinformatics.ku.edu/st-analyzer, is a standalone graphical user interface (GUI) toolset to perform various trajectory analyses. ST-analyzer has several outstanding features compared to other existing analysis tools: (i) handling various formats of trajectory files from MD programs, such as CHARMM, NAMD, GROMACS, and Amber, (ii) intuitive web-based GUI environment—minimizing administrative load and reducing burdens on the user from adapting new software environments, (iii) platform independent design—working with any existing operating system, (iv) easy integration into job queuing systems—providing options of batch processing either on the cluster or in an interactive mode, and (v) providing independence between foreground GUI and background modules—making it easier to add personal modules or to recycle/integrate pre-existing scripts utilizing other analysis tools. The current ST-analyzer contains nine main analysis modules that together contain 18 options, including density profile, lipid deuterium order parameters, surface area per lipid, and membrane hydrophobic thickness. This article introduces ST-analyzer with its design, implementation, and features, and also illustrates practical analysis of lipid bilayer simulations. © 2014 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.23584

Introduction

The capability of exploring biomolecular structure, dynamics, and interactions using various levels of descriptions, such as all-atom, united-atom, and coarse-grained models, makes molecular dynamics (MD) simulation one of the key tools to obtain deeper insights into biological systems. For example, MD simulations have been applied to many general areas of biology (proteins[1–3] and membrane-associated proteins[4–6]) with some specific examples in revealing an activation mechanism for the β2-adrenergic receptor (a prototypical GPCR),[7] understanding permeation and gating in ion channels,[8,9] characterizing conformational changes of kinases,[10,11] identifying enzyme inhibitors,[12] and designing drugs.[13]

In particular, recent advances in computing resources[14] and various MD programs[15–19] and associated packages[20,21] make MD simulation easier to use and thus increase its popularity. However, analyzing a huge amount of trajectory data with many atoms is still a difficult task, even for the experts in the field, especially because of the following three reasons. First, analyzing MD simulation requires understanding various analysis methods and being familiar with programming languages in addition to the simulation program. Second, even for well-qualified users, a task as simple as recycling trajectory analysis scripts of other systems is time consuming. This requires a user to keep track of and understand all existing scripts and change the script based on new simulation systems. Finally, although this may not be the case for all users, occasionally it is necessary to analyze third party trajectories. Since most MD users have their own favorite simulation package, the analysis of trajectory data produced from different package becomes very challenging. Indeed, due to increasing importance of referencing studies in modern sciences, there already exist public data depositories for sharing MD simulation trajectories[22–25]; thus, analyzing trajectory produced by unfamiliar simulation packages could bring major challenges.

In this context, we propose minimum guidelines for a general-purpose MD trajectory analysis program: (i) the program should be easy to use and able to handle diverse trajectories produced

[a] *Jong Cheol Jeong, S. Jo, E. L. Wu, Y. Qi, W. Im*
Department of Molecular Biosciences and Center for Bioinformatics, The University of Kansas, Lawrence, Kansas 66047
E-mail: wonpil@ku.edu

[b] *V. Monje-Galvan, J. B. Klauda*
Department of Chemical and Biomolecular Engineering, The University of Maryland, 2113 Chemical and Nuclear Engineering Bldg., College Park, Maryland 20742

[c] *M. S. Yeom*
Korean Institute of Science and Technology Information, Yuseong-gu, Daejeon 305-806, Korea

[d] *L. Gorenstein, F. Chen*
Research Computing (RCAC), Purdue University, West Lafayette, Indiana 47097

Contract/grant sponsor: NSF; contract/grant numbers: ABI-1145987 and DBI-1145652; Contract/grant sponsor: XSEDE resources; contract/grant number: TG-MCB070009

from different simulation packages along with reliable and extensively tested analysis methods, (ii) source codes must be open and easy to read, so that users can verify the algorithm and modify them for their own purpose, (iii) the ideal program should be able to trace and maintain the analysis results, and (iv) the program is required to have high portability ranged from personal computer to community-wide cluster machines (e.g., XSEDE, https://xsede.org, and DiaGrid, https://diagrid.org), so users can have more choices to use computer resources without demanding personalized system environment generally required for trajectory analysis using users' own packages or scripts. In this way, users can avoid not only the necessary preparations of analyzing trajectories, but also transfer of huge trajectory data from one machine to another.

ST-analyzer is a web-based graphical user interface (GUI) designed for analyzing MD trajectories with the aforementioned guidelines in mind. Since there have been previous efforts aiming to achieve similar outcomes,[26–28] these approaches are discussed briefly in terms of three categories based on their functionality: application programming interface (API) libraries, command line application, and GUI application. MDAnalysis[26] and LOOS[27] are distributed as API containing various analysis libraries. Therefore, programming skills and understanding of the libraries are the minimum requirement to use these APIs, and significant efforts are required for users who are not familiar with these APIs. Command line applications including ptraj,[17] crama,[29] and CHARMM[30] are other popular analysis tools with various built-in analysis options. However, users need to be familiar with command line environments, and extending or adding modules is usually difficult due to complicated codebase.[31] GUI-based approaches such as VMD[21] and Grcarma[28] are the closest ones toward the goal of our guidelines for developing analysis tools. However, these applications are typically designed to work interactively and not suitable for some analysis functions that require significant runtime (e.g., per lipid area in mixed bilayers and density profiles) or loading long simulation trajectory data. Because MD simulations are typically performed in a remote computing environment and more sophisticated analysis functions are used in a daily basis, it is becoming more important to provide an analysis interface that can operate in a remote computing environment yet user-friendly and extendable. Finally, the absence of tracking and maintaining inputs and outputs is also common issues for current analysis tools.

ST-analyzer is designed to overcome the shortcomings of existing tools while emphasizing its design principles to reduce user's learning curve and maximize the software flexibility. The details about ST-analyzer including its design scheme, implementation, and features are described and discussed in the following sections along with some practical examples of analyzing trajectories of complex membrane simulations.

ST-analyzer Design, Implementation, and Features

ST-analyzer is a standalone GUI toolset to provide a variety of modules to perform various analyses of MD simulation trajectories. ST-analyzer has three important features in its design: (i) user-friendly GUI, (ii) integrated analysis environment (IAE), and (iii) process-oriented design (POD). The assorted manuals are available and classified as the comprehensive system manual, tutorials for quick use, and personal module implementation (see below for details).

ST-analyzer GUI

ST-analyzer provides a user-friendly GUI that can significantly reduce user's efforts to adapting new software environments through intuitive menus and supporting multiwindow workspace commonly found in most desktop applications. As shown in Figure 1, the GUI contains three main menus in the workspace: *ST-analyzer* ("A"), *USER accounts* ("U"), and *Result Viewer* ("R"). The menu, *ST-analyzer*, prepares analysis job submission in two steps. STEP 1 of *ST-analyzer* icon defines system environments including the locations of trajectories and output directories, and STEP 2 provides simple GUI for preparing MD analysis with the default parameters and/or template queries that can be modified anytime by users. These two STEPs are designed to minimize the interaction between a user and analysis GUI while maximizing the flexibility of modules. *Result Viewer* makes it easy to track the job status and outputs by listing a summary of analysis, job status, thumbnail figures, and links for downloading outputs. It also provides simple sorting, searching, and editing tools to help maintaining large amount of outputs and ongoing processes. Finally, the *USER accounts* menu makes it possible for ST-analyzer to be used as groupware by isolating the outputs based on authorized users, so each user can work independently.

ST-analyzer IAE

ST-analyzer provides IAE by accomplishing an all-in-one system consisting of GUI, analysis modules, and database (DB) for preparing and maintaining MD analysis inputs and outputs. The schematic system design is shown in Figure 2. ST-analyzer achieves IAE via integration of four subsystems: (i) Django—providing web-based GUI supported by most operating system and web browsers, (ii) SQLite—embedded DB maintaining inputs and outputs of analysis modules, (iii) File server—storing analysis results, and (iv) Cluster/Workstation—computer resources for analysis calculations. Although these subsystems cooperate with each other to become a total solution of MD trajectory analysis, each subsystem works as an individual process, so that they do not need to be dependent on computer resources or applications.

ST-Analyzer POD

To describe the general approach toward POD and its implementation, the above four subsystems are categorized into two processes, the foreground process that interacts with users and the background process that takes over the responsibility from the foreground process after analysis jobs are submitted. The schematic of workflow is shown Figure 3. Although the background process is initiated by the job submission at the foreground GUI, individual processes are

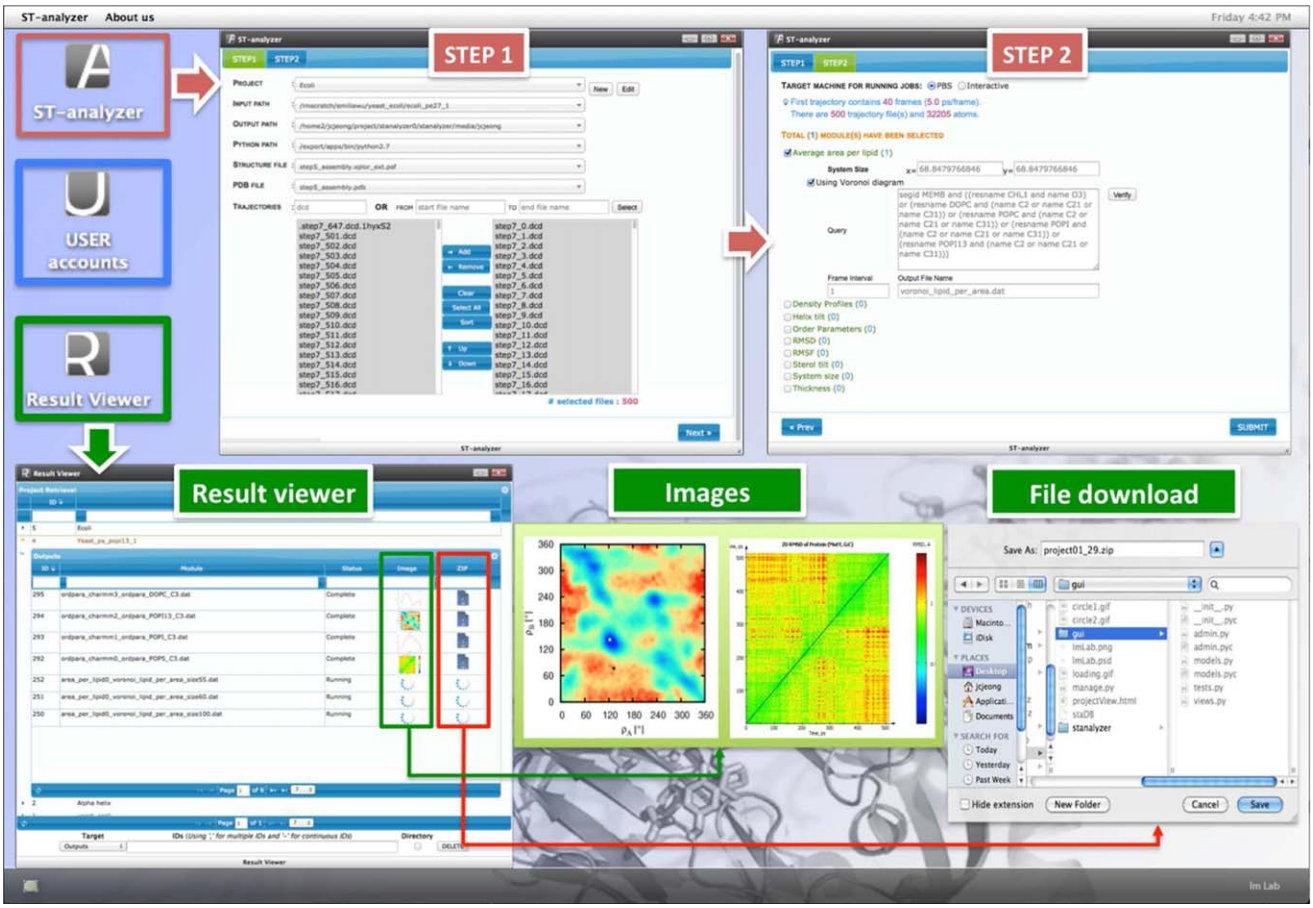

Figure 1. ST-analyzer GUI. The desktop icon *ST-analyzer* prepares simulation trajectory analysis in two steps. In STEP 1, users define the locations of trajectories and computer environments (e.g., PBS scripts for job submission and Python containing all necessary modules). In STEP2, users can select and define the built-in analysis modules. *USER accounts* icon creates and modifies user's accounts—maintaining multiple accounts are essential when ST-analyzer is used as groupware. *Result Viewer* icon provides an interface to maintain input and output of analysis results, providing three main functions: (i) retrieving results via built-in simple search engine, (ii) displaying sample images of analysis results, and (iii) downloading files containing the inputs (e.g., parameters and selection queries for selected analysis modules) and outputs (e.g., graph images, gnuplot scripts, and data files in text format).

implemented independently. For example, the foreground GUI is only responsible for providing information about the MD trajectory (i.e., the *system information* in Fig. 3) and parameters of analysis modules (i.e., the *setup analysis modules* in Fig. 3) to the background modules. Once the jobs are submitted from the foreground, a background process creates an independent job script for each analysis task and assigns it to computer resources (denoted as *cluster or workstation* in Fig. 3) chosen by a user at the foreground GUI. Each script works independently with DB and file servers to update job status and store job results, so that users can retrieve the job status and results anytime through the foreground GUI. To maintain POD among the background modules, they communicate each other through a specifically designed data file independent of programming language and keep each process independent. Therefore, it is a user's decision to choose programming languages to develop their own background processes, although current background processes are implemented mainly with Python and MDAnalysis.[26]

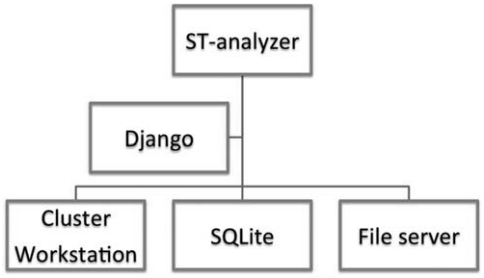

Figure 2. ST-analyzer IAE. ST-analyzer provides IAE via integration of four subsystems: (i) Django—establishing communication between web-based GUI and background modules via HTTP, (ii) SQLite—embedded DB maintaining inputs and outputs of analysis modules, (iii) File server—storing analysis results, and (iv) Cluster/Workstation—computer resources for analysis calculations. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

ST-analyzer manuals

ST-analyzer comes with task-oriented manuals that can be freely downloaded from http://im.bioinformatics.ku.edu/st-analyzer/tutorials.html. Users can choose the current version of tutorials from three categories: (i) quick tutorial—consisting of brief explanation of installation to simple usages of built-in analysis modules, (ii) full manual—delivering detailed information helping both nonexperts and experts to efficiently utilize ST-analyzer, and (iii) implementation tutorial—targeting experts

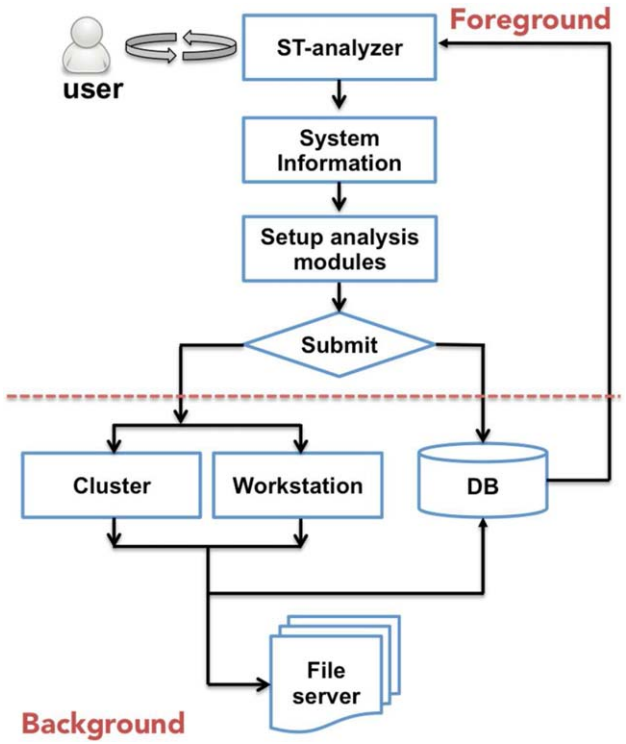

Figure 3. ST-analyzer POD and workflow. All processes in ST-analyzer can be grouped into two processes: foreground process taking inquires from users through web-based GUI and background process (below the dashed red line) analyzing trajectory based on user's demands defined by foreground GUI. *System Information* and *Setup analysis modules* correspond to STEP 1 and STEP 2 of *ST-analyzer* icon in Figure 1, respectively. Once jobs are submitted, all background processes work independently, report their status to DB, and store results into a file server based on the progress of each individual job. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

who want to implement their own analysis modules and add them to ST-analyzer. To help users' understanding, all manuals include detailed explanations with screenshots.

ST-Analyzer analysis modules and key features

The current ST-analyzer contains nine main modules consisting of 18 options, and the roles of each module are summarized in Table 1. The built-in modules are cross-validated with other available analysis tools. In addition to built-in modules, the independent design scheme and intuitive script-based programming language (i.e., HTML and Python) make it easier for users to develop personal modules in the ST-analyzer framework. There are tutorials and templates that guide developments of personal add-ons (see *ST-analyzer Manuals*). ST-analyzer also comes with unique features shown in Figure 4: (i) selection GUI—helping query preparation for selecting atoms, (ii) verification—verifying query before sending actual jobs, and (iii) membrane centering—options for placing the bilayer center at $Z = 0$ (for each frame) when the bilayer has shifted during the simulation.

Application

An *Escherichia coli* membrane system is selected to illustrate the usage of ST-analyzer. This system consists of 32,205 atoms with five different lipid types (Fig. 5): 1,2-dipalmitoyl-phosphatidylethanolamine (DPPE), 1,2-dipalmitoleic-phosphatidylethanolamine (DXPE), 1-palmytoil-2-cis-9,10-methylenehexadecanoyl-phosphatidylethanolamine (PMPE), 1,2-dipalmitoyl--1′-palmytoil-2′-cis-9,10-methylenehexadecanoyl-cardiolipin (PMCL2), and 1,2-1′,2′-tetrahexadecenoyl-cardiolipin (TXCL2). There are 3 DPPE, 6 DXPE, 3 PMPE, 24 PMCL2, and 9 TXCL2 in each leaflet, 132 neutralizing K^+ ions, and 4665 water molecules. The system was prepared by Membrane Builder[32,33] in CHARMM-GUI[20] and simulated with NAMD[18] for 150 ns with the standard Membrane Builder nonbonded interaction options and simulation options including a time-step of 2 fs and a coordinate saving frequency of 5.0 ps. For trajectory analysis of this system, ST-analyzer was installed in our local cluster machine where the trajectory and structure information files (i.e., PSF and PDB files) were located.

System size

The time-series of the system size can be used as a quick check for convergence of a membrane simulation. Supporting Information Figure S1 shows the input (STEP 2 in Fig. 1) and output (*Result Viewer* in Fig. 1) screenshots as well as the system size time-series along *X*-, *Y*-, and *Z*-axis. For most analysis modules, a user can choose a frame interval and the output filenames to write out the raw calculated data. When multiple analysis

Table 1. ST-Analyzer built-in modules.

Modules	Description
Density profile	Analyzing the density of selected atoms along a given axis. Sub-modules are available based on atom selections including all atoms, lipid head group, lipid acyl tail, water, water dipole, a two-atom vector, and custom selections.
Helix tilt angle	Analyzing the changes of tilt angle of user defined helix during simulation
Order parameters	Calculating deuterium order parameters for user selected atoms.
RMSD	Analyzing root-mean-square deviation with user selected atoms
RMSF	Analyzing root-mean-square fluctuations with user selected atoms
Sterol tilt angle	Calculating the tilt angle of sterols. Two submodules, ring tilt and tail tilt angle are supported.
Surface area per lipid	Calculating the surface area of individual lipids and their average by utilizing Voronoi diagram.
System size	Analyzing the variations of box size during simulations
Thickness	Calculating membrane thickness. Submodules are available based on atom selections including phosphate, carbon, and custom selection. Providing an option for adjusting bilayer shift.

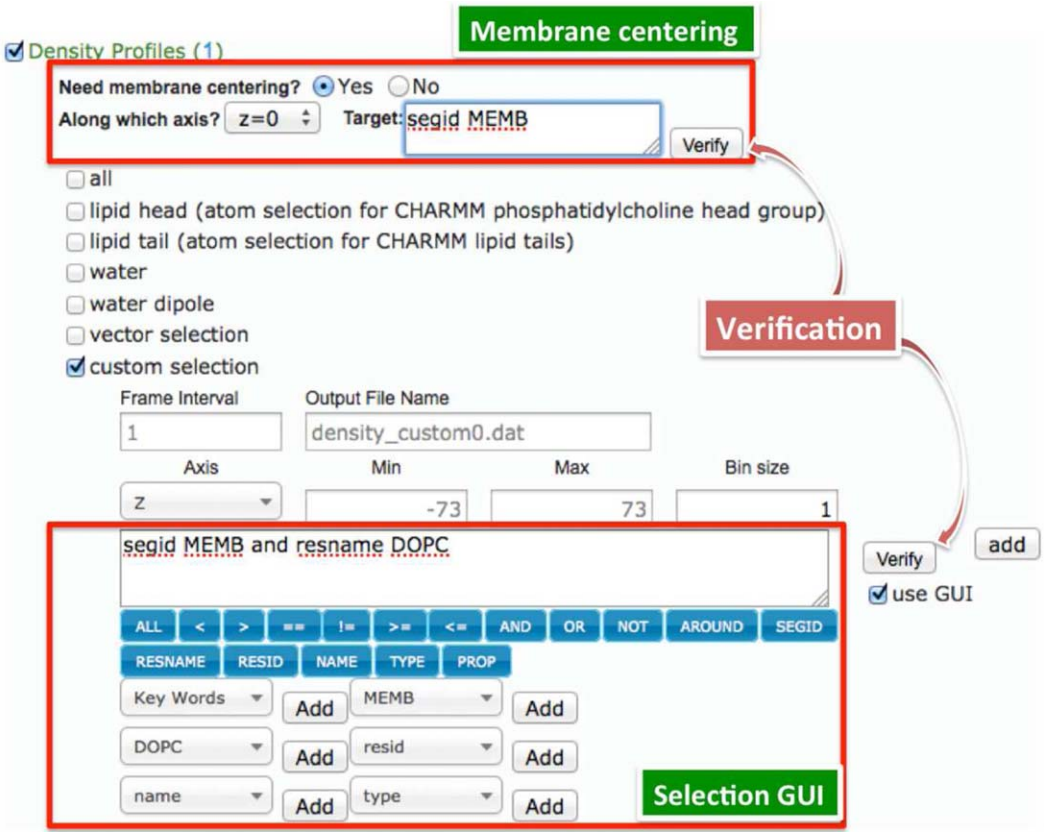

Figure 4. Two key features in ST-analyzer: membrane centering and atom selection. Membrane centering: to avoid the influence of membrane drifting during the simulation (i.e., having one leaflet on the top and the other on the bottom of the system or having a few lipid molecules on the top and the rest on the bottom of the system), ST-analyzer provides an option to place the bilayer center at $Z = 0$ (for each frame) by double image centering: one after translating the COM of the first residue of lipid molecules defined in "*Target*" to $Z = 0$ and the other after placing the bilayer center at $Z = 0$. Atom selection: while ST-analyzer provides predefined atom selections for certain properties based on the CHARMM lipid force field naming scheme, a user can freely select any atoms and validate the selections. The selection grammar is based on MDAnalysis.[26] As shown in the figure, a user can use the selection GUI using the pull-down selections such as segment names, residue names, residue IDs, atom names, and atom types, which are automatically provided based on the protein structure file (PSF) information. Note that the selection GUI appears when "use GUI" is selected. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

options are chosen, *Result Viewer* lists all the chosen filenames and the status of each analysis, as shown in Supporting Information Figure S1B. Clearly, for this system the system dimensions are well equilibrated (and fluctuate around an average value).

Membrane thickness

The time-series of the membrane thickness is an indicator for convergence of a membrane simulation and is inversely proportional to the per-lipid area. There are several ways of defining the membrane thickness for different purposes. Supporting Information Figure S2 shows two conventional options using phosphorus atoms in glycerophospholipid bilayers to define an overall membrane thickness and using C22 and C32 carbon chain atoms (just below the carbonyl) to define a thickness of the bilayer hydrophobic core. In ST-analyzer, with the conventional definition of using the *Z*-axis as the membrane normal, the selected atoms' average *Z* value is calculated for each leaflet and the difference is used for the membrane thickness. To avoid the influence of membrane drifting during the simulation, ST-analyzer provides an option to place the bilayer center at zero in user's desired axis (e.g., $Z = 0$ for each frame) by double image centering (Fig. 4): one after translating the center of mass (COM) of the first residue of lipid molecules defined in Supporting Information Figure S2A *Target* to $Z = 0$ and the other after placing the bilayer center at $Z = 0$. Note that the (predefined) phosphorus and carbon tail atom selections in Supporting Information Figure S2A are based on the CHARMM lipid force field.[34] A user can modify these selections or use the "*Using custom group*" option for different atom name schemes. Supporting Information Figure S2B, *Result Viewer*, shows retrieving results via built-in search engine with a simple query "thickness." Note that search engine is attached to every field in *Result Viewer*, so users can have more choices for retrieving results.

Per-lipid area

Together with the membrane thickness, the time-series of the per-lipid area is another indicator for convergence of a membrane simulation. When the membrane system has only one lipid type, the per-lipid area can be easily obtained from the system size in the *XY* dimension. When multiple lipid types are used (i.e., for a

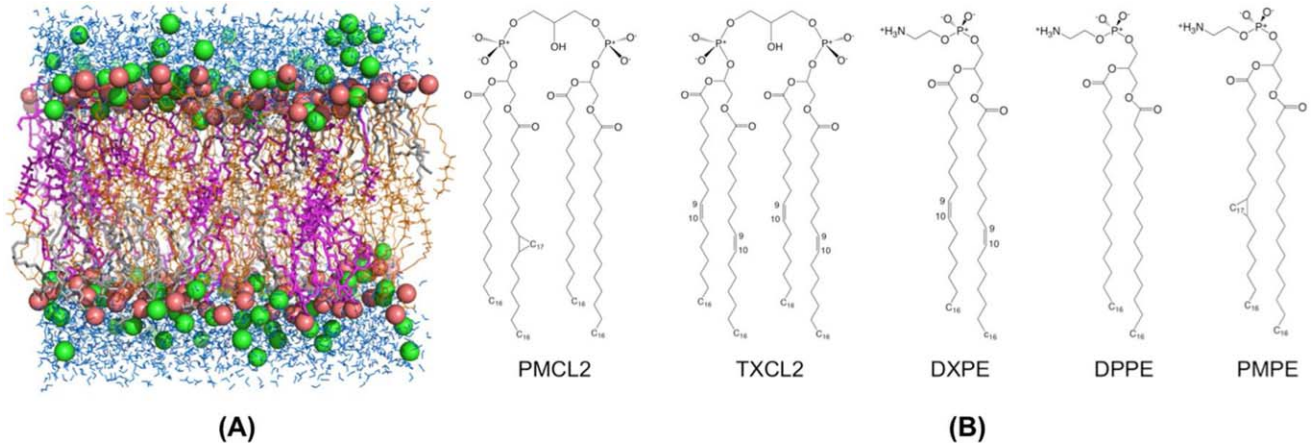

Figure 5. *E. coli* membrane system. (A) The system structure: DPPE, PMPE, and DXPE (gray), TXCL2 (magenta), PMCL2 (yellow), K^+ ions (green spheres), phosphate atoms (red spheres), and water atoms (blue). (B) The chemical structure of each lipid type in the system.

mixed or heterogeneous bilayer), Voronoi tessellation is commonly used. ST-Analyzer uses the qhull program[35] for Voronoi tessellation, and a user can choose atoms to define a lipid. In Supporting Information Figure S3, following the approach of Pandit et al.,[36] three atoms were used to define a phospholipid, that is, the two carbonyl carbon atoms (C21 and C31) and the middle carbon (C2) of the glycerol backbone. Note that the cardiolipin lipids (PMCL2 and TXCL2) have four lipid tails (Fig. 5), so their per-lipid areas are roughly twice larger than typical phospholipid area, as shown in Supporting Information Figure S3C. Similar to past analysis, individual lipid areas of two-chain phosphatidylethanolamine lipids all average about similar values.[37]

Lipid order parameters

Lipid deuterium order parameters (S_{CD}) are a common metric for bilayer fluidity (liquid ordered vs. liquid disordered). The order parameter of each C—D bond vector is defined as $S_{CD}=\langle 3\cos^2\theta_{CH}-1\rangle/2$ where θ_{CH} (CH because we simulate undeuterated lipids) is the time dependent angle between the C—H bond vector and the bilayer normal, and the angular bracket denotes a time and ensemble average. Supporting Information Figure S4 shows the $|S_{CD}|$ of *sn*-2 chains of DPPE and DXPE (Fig. 5). The general trends for saturated (DPPE *sn*-2) and unsaturated (DXPE *sn*-2) acyl chains are similar to previous simulation studies,[34,38] including the fact that local disordering is found around the double bond (C9 and C10) in unsaturated DXPE *sn*-2 chain. In general, a maximum S_{CD} below 0.3 indicate liquid disordered states. Although there are predefined atoms selections for some lipid molecules in the CHARMM lipid force field as shown in Supporting Information Figure S4A, a user can modify the atom selection or use different atom selections for other lipid force fields.

Density profiles along *Z*-axis

Density profiles of various system components along the bilayer normal provide their distributions during the simulations. Supporting Information Figure S5 shows the ST-analyzer selections of typical components for the density/distribution profiles, such as lipid head groups, lipid carbon tails, water, and water dipole. While Supporting Information Figure S5C shows the electron-weighted density [$e\text{Å}^{-3}$] profiles, the output data file also contains the number density [Å^{-3}] and mass-weighted density in the second and third columns, respectively. The water dipole (μ) distribution is also shown in Figure S5D; $\boldsymbol{\mu}=q_o\mathbf{r}_o+q_H\mathbf{r}_{H1}+q_H\mathbf{r}_{H2}$ where $q_o=-0.834e$ and $q_H=0.417e$ based on the TIP3P water model. It should be noted that, since the example *E. coli* membrane system has highly negative surface charge distribution due to abundant PMCL2 and TXCL2 (Fig. 5), the diploe distribution is much wider than in a typical bilayer system with neutral lipid types,[39] that is, the water molecular orientation is more ordered in the current system than in the neutral lipid bilayer system. Although not illustrated here, a user can also define any vector (e.g., P-N head group atoms in phosphatidylethanolamine) to get their distribution along the membrane normal (Table 1).

Summary and Future Developments

ST-analyzer is developed based on three underlying principles, that is, to be simple, clear, and useful. The simple and intuitive GUIs are designed to help users easily adapt new software environments, a clear design for setting up analysis modules aims to minimize user interactions to complete user's demands for trajectory analysis, and assured built-in modules and personal add-ons guarantee the maximum flexibility of ST-analyzer to cover personal inquires on diverse trajectory analysis. The current version contains nine main modules that together contain 18 options, and additional 20 modules are under development (http://im.bioinformatics.ku.edu/st-analyzer/modules.html). ST-analyzer will reflect users' demands and keep adding options to built-in analysis modules upon its further development. As part of our ongoing efforts to make ST-analyzer available for a user in community computational resources, ST-analyzer is currently available on National Institute of Supercomputing and Networking in South Korea (http://www.nisn.re.kr) as well as campus wide clusters at the University of Kansas and the University of Maryland. To provide easy access of

ST-analyzer, we will keep working with researchers in large multicampus distributed research computing network facilities including DiaGrid (http://diagrid.org) and XSEDE (https://www.xsede.org), and users in these networks will be able to use ST-analyzer in near future. We expect that ST-analyzer can facilitate improving computational biology experiences for experts and nonexperts with its flexibility and easy control on assured analysis modules that can significantly reduce the burdens of handling various trajectory analyses and speed up rigorous analysis.

Acknowledgment

Testing of ST-analyzer was performed on the High Performance Computing Cluster (HPCC) at the University of Maryland (Deepthought) through J.B.K.

Keywords: molecular dynamics · membrane bilayer · lipid order parameter · hydrophobic thickness

How to cite this article: J. C. Jeong, S. Jo, E. L. Wu, Y. Qi, V. Monje-Galvan, M. S. Yeom, L. Gorenstein, F. Chen, J. B. Klauda, W. Im. *J. Comput. Chem.* **2014**, *35*, 957–963. DOI: 10.1002/jcc.23584

Additional Supporting Information may be found in the online version of this article.

[1] M. Karplus, J. A. McCammon, *Nat. Struct. Biol.* **2002**, *9*, 646.
[2] M. K. Gilson, H. X. Zhou, *Annu. Rev. Biophys. Biomol. Struct.* **2007**, *36*, 21.
[3] K. A. Dill, S. B. Ozkan, M. S. Shell, T. R. Weikl, *Annu. Rev. Biophys.* **2008**, *37*, 289.
[4] S. A. Shaikh, J. Li, G. Enkavi, P.-C. Wen, Z. Huang, E. Tajkhorshid, *Biochemistry* **2013**, *52*, 569.
[5] F. Khalili-Araghi, J. Gumbart, P.-C. Wen, M. Sotomayor, E. Tajkhorshid, K. Schulten, *Curr. Opin. Struct. Biol.* **2009**, *19*, 128.
[6] P. J. Stansfeld, M. S. P. Sansom, *Structure* **2011**, *19*, 1562.
[7] R. Nygaard, Y. Zou, R. O. Dror, T. J. Mildorf, D. H. Arlow, A. Manglik, A. C. Pan, C. W. Liu, J. J. Fung, M. P. Bokoch, F. S. Thian, T. S. Kobilka, D. E. Shaw, L. Mueller, R. S. Prosser, B. K. Kobilka, *Cell* **2013**, *152*, 532.
[8] B. Roux, K. Schulten, *Structure* **2004**, *12*, 1343.
[9] B. Roux, T. Allen, S. Berneche, W. Im, *Q. Rev. Biophys.* **2004**, *37*, 15.
[10] J. D. Faraldo-Gomez, B. Roux, *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 13643.
[11] Y. Shan, M. A. Seeliger, M. P. Eastwood, F. Frank, H. Xu, M. O. Jensen, R. O. Dror, J. Kuriyan, D. E. Shaw, *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 139.
[12] J. D. Durrant, R. Cao, A. A. Gorfe, W. Zhu, J. Li, A. Sankovsky, E. Oldfield, J. A. McCammon, *Chem. Biol. Drug Des.* **2011**, *78*, 323.
[13] D. B. Kitchen, H. Decornez, J. R. Furr, J. Bajorath, *Nat. Rev. Drug Discov.* **2004**, *3*, 935.
[14] D. E. Shaw, M. M. Deneroff, R. O. Dror, J. S. Kuskin, R. H. Larson, J. K. Salmon, C. Young, B. Batson, K. J. Bowers, J. C. Chao, M. P. Eastwood, J. Gagliardo, J. P. Grossman, C. R. Ho, D. J. Ierardi, I. Kolossvary, J. L. Klepeis, T. Layman, C. Mcleavey, M. A. Moraes, R. Mueller, E. C. Priest, T. B. Shan, J. Spengler, M. Theobald, B. Towles, S. C. Wang, *Commun. ACM* **2008**, *51*, 91.
[15] S. Pronk, S. Pall, R. Schulz, P. Larsson, P. Bjelkmar, R. Apostolov, M. R. Shirts, J. C. Smith, P. M. Kasson, D. van der Spoel, B. Hess, E. Lindahl, *Bioinformatics* **2013**, *29*, 845.
[16] B. R. Brooks, C. L. Brooks, A. D. Mackerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V. Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, M. Karplus, *J. Comput. Chem.* **2009**, *30*, 1545.
[17] R. Salomon-Ferrer, D. A. Case, R. C. Walker, *Wires Comput. Mol. Sci.* **2013**, *3*, 198.
[18] M. T. Nelson, W. Humphrey, A. Gursoy, A. Dalke, L. V. Kale, R. D. Skeel, K. Schulten, *Int. J. Supercomput. Ap.* **1996**, *10*, 251.
[19] B. FrantzDale, S. J. Plimpton, M. S. Shephard, *Eng. Comput-Germany* **2010**, *26*, 205.
[20] S. Jo, W. Im, *Biophys. J.* **2011**, *100*, 156.
[21] W. Humphrey, A. Dalke, K. Schulten, *J. Mol. Graph. Model.* **1996**, *14*, 33.
[22] D. A. Beck, A. L. Jonsson, R. D. Schaeffer, K. A. Scott, R. Day, R. D. Toofanny, D. O. Alonso, V. Daggett, *Protein Eng. Des. Sel.* **2008**, *21*, 353.
[23] C. Kehl, A. M. Simms, R. D. Toofanny, V. Daggett, *Protein Eng. Des. Sel.* **2008**, *21*, 379.
[24] A. Leftin, M. F. Brown, *Biochim. Biophys. Acta* **2011**, *1808*, 818.
[25] T. Meyer, M. D'Abramo, A. Hospital, M. Rueda, C. Ferrer-Costa, A. Perez, O. Carrillo, J. Camps, C. Fenollosa, D. Repchevsky, J. L. Gelpi, M. Orozco, *Structure* **2010**, *18*, 1399.
[26] N. Michaud-Agrawal, E. J. Denning, T. B. Woolf, O. Beckstein, *J. Comput. Chem.* **2011**, *32*, 2319.
[27] T. D. Romo, A. Grossfield, *Conf. Proc. IEEE Eng. Med. Biol. Soc.* **2009**, 2009, 2332.
[28] P. I. Koukos, N. M. Glykos, *J. Comput. Chem.* **2013**, *34*, 2310.
[29] N. M. Glykos, *J. Comput. Chem.* **2006**, *27*, 1765.
[30] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, M. Karplus, *J. Comput. Chem.* **1983**, *4*, 187.
[31] D. R. Roe, T. E. Cheatham, *J. Chem. Theory Comput.* **2013**, *9*, 3084.
[32] S. Jo, T. Kim, W. Im, *PLoS One* **2007**, *2*, e880.
[33] S. Jo, J. B. Lim, J. B. Klauda, W. Im, *Biophys. J.* **2009**, *97*, 50.
[34] J. B. Klauda, R. M. Venable, J. A. Freites, J. W. O'Connor, D. J. Tobias, C. Mondragon-Ramirez, I. Vorobyov, A. D. MacKerell, R. W. Pastor, *J. Phys. Chem. B* **2010**, *114*, 7830.
[35] C. B. Barber, D. P. Dobkin, H. Huhdanpaa, *ACM Trans. Math. Softw.* **1996**, *22*, 469.
[36] S. A. Pandit, S. Vasudevan, S. W. Chiu, R. J. Mashl, E. Jakobsson, H. L. Scott, *Biophys. J.* **2004**, *87*, 1092.
[37] K. R. Pandit, J. B. Klauda, *Biochim. Biophys. Acta,* **2012**, *1818*, 1205.
[38] J. B. Klauda, R. M. Venable, J. A. Freites, J. W. O'Connor, C. Mondragon-Ramirez, I. Vorobyov, D. J. Tobias, A. D. MacKerell, R. W. Pastor, *J. Phys. Chem. B* **2010**, *114*, 7830.
[39] J. B. Klauda, X. W. Wu, R. W. Pastor, B. R. Brooks, *J. Phys. Chem. B* **2007**, *111*, 4393.

Received: 17 January 2014
Revised: 7 February 2014
Accepted: 15 February 2014
Published online on 17 March 2014