Development of an Open Biomolecular Mesoscopic Simulation Environment for the Study of Cyclotide/Membrane Interactions

Dissertation

Zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

– Dr. rer. nat. –

vorgelegt von

M. Sc. Karina van den Broek

geboren in Oberhausen

Fakultät für Chemie

Anorganische Chemie

Universität Duisburg-Essen

&

Institut für biologische und chemische Informatik

Westfälische Hochschule

2021

DOI: 10.17185/duepublico/74767
URN: urn:nbn:de:hbz:464-20210915-081812-2

“Life is an unfoldment, and the further we travel the more truth we can comprehend. To understand the things that are at our door is the best preparation for understanding those that lie beyond.”

- Hypatia

To my family

Die vorliegende Arbeit wurde im Zeitraum von März 2016 bis Februar 2021 durch eine Kooperation des Arbeitskreises von Prof. Dr. Matthias Epple, Anorganische Chemie der Universität Duisburg-Essen, und des Arbeitskreises von Prof. Dr. Achim Zielesny am Institut für biologische und chemische Informatik der Westfälischen Hochschule erstellt. Der Tag der mündlichen Prüfung war der 27.08.2021.

Erstgutachter:	Prof. Dr. Matthias Epple
Zweitgutachter:	Prof. Dr. Achim Zielesny
Vorsitzender:	Prof. Dr. Gebhard Haberhauer
Tag der Disputation:	27.08.2021

Overview

1 Zusammenfassung ..... - 13 -

2 Summary ..... - 15 -

3 Preface ..... - 17 -

4 Introduction and R&D objectives ..... - 18 -

4.1 Molecular structure representation ..... - 18 -

4.1.1 Development objective ..... - 20 -

4.2 Mesoscopic simulation kernel ..... - 22 -

4.2.1 Development objective ..... - 26 -

4.3 Mesoscopic simulation environment ..... - 26 -

4.3.1 Development objective ..... - 27 -

4.4 Cyclotide/membrane interactions ..... - 27 -

4.4.1 Research objective ..... - 29 -

5 Results ..... - 31 -

5.1 SPICES molecular structure representation ..... - 31 -

5.2 Mesoscopic simulation kernel Jdpd ..... - 39 -

5.3 Mesoscopic simulation environment MFsim ..... - 45 -

5.3.1 Implementation ..... - 45 -

5.3.2 Simulation job design ..... - 49 -

5.3.3 Simulation job execution ..... - 60 -

5.3.4 Simulation job result evaluation ..... - 60 -

5.3.5 MFsim use cases ..... - 65 -

5.4 Cyclotide/membrane interactions ..... - 66 -

5.4.1 Particle data ..... - 66 -

5.4.2 Open simulation software ..... - 66 -

5.4.3 Simulation details ..... - 67 -

5.4.4 Molecular fragmentation ..... - 68 -
5.4.5 Plasma membrane models ..... - 74 -
5.4.6 Cyclotide-membrane interaction model ..... - 75 -
5.4.7 Plasma membrane simulations ..... - 77 -
5.4.8 Cyclotide backbone flexibility ..... - 79 -
5.4.9 Cyclotide-membrane interaction ..... - 80 -
5.4.10 Quantitative analysis of the membrane disruption process ..... - 81 -
5.4.11 Precision of the membrane disruption evaluation procedure ..... - 84 -
5.4.12 Kalata B1 membrane interaction ..... - 84 -
5.4.13 Kalata B1 – hydrophobic patch mutants ..... - 85 -
5.4.14 Kalata B1 – differently charged mutants ..... - 86 -
5.4.15 Cycloviolacin O2 membrane interaction ..... - 87 -
5.4.16 Cycloviolacin O2 – hydrophobic patch mutants ..... - 87 -
5.4.17 Cycloviolacin O2 – differently charged mutants ..... - 88 -
5.4.18 Cyclotide activity and membrane composition ..... - 88 -
5.4.19 Summary of previous findings ..... - 90 -
5.4.20 Additivity of cyclotide mixtures ..... - 91 -
5.4.21 Pore formation ..... - 92 -
6 Discussion ..... - 94 -
6.1 SPICES molecular structure representation ..... - 94 -
6.2 Mesoscopic simulation kernel Jdpd ..... - 94 -
6.3 Mesoscopic simulation environment MFsim ..... - 95 -
6.4 Cyclotide/membrane interactions ..... - 95 -
7 Appendices ..... - 97 -
7.1 Appendix I – SPICES syntax rules ..... - 97 -
7.2 Appendix II – DPD conversion formulas ..... - 100 -
7.3 Appendix III – Data objects, preferences and re-usable settings ..... - 103 -

7.4 Appendix IV – Simulation kernel integration ..... - 104 -

7.5 Appendix V – Tutorial MFsim –
Cyclotide-membrane sandwich interaction model ..... - 106 -

8 Tables ..... - 153 -

9 References ..... - 160 -

10 Attachment ..... - 174 -

10.1 List of publications ..... - 174 -

10.2 Poster presentations ..... - 175 -

10.3 Curriculum Vitae ..... - 176 -

10.4 Danksagung ..... - 178 -

10.5 Eidesstattliche Erklärung ..... - 179 -

1 Zusammenfassung

SPICES (Simplified Particle Input ConnEction Specification) ist eine partikelbasierte Molekülstrukturdarstellung, die von der atomaren SMILES-Zeilennotation abgeleitet ist. Sie zielt darauf ab, langwierige und fehleranfällige Molekülstrukturdefinitionen für partikelbasierte mesoskopische Simulationstechniken wie die Dissipative Partikeldynamik zu unterstützen, indem ein Zusammenspiel verschiedener molekularer Kodierungsebenen ermöglicht wird, die von topologischen Zeilennotationen und entsprechenden Partikel-Graph-Visualisierungen bis hin zu 3D-Strukturen mit Unterstützung ihrer räumlichen Positionierung in einer Simulationsbox reichen.
Eine offene Java-Bibliothek für die Handhabung von SPICES-Strukturen und die Unterstützung mesoskopischer Simulationen in Kombination mit einer offenen Java-Viewer-Anwendung für die visuelle topologische Inspektion von SPICES-Definitionen wird bereitgestellt.

Jdpd ist ein offener Java-Simulationskern für die Dissipative Partikeldynamik mit parallelisierbarer Kraftberechnung, effizienten Caching-Optionen und schnellen Eigenschaftsberechnungen. Er zeichnet sich durch ein schnittstellen- und mustergetriebenes Design für einfache Code-Änderungen aus und kann helfen, Probleme der polyglotten Programmierung zu vermeiden.
Detaillierte Ein-/Ausgabe-Kommunikation, Parallelisierung und Prozesssteuerung sowie interne Protokollierungsmöglichkeiten für Debugging-Zwecke werden unterstützt. Der neue Kernel kann in verschiedenen Simulationsumgebungen eingesetzt werden, die von flexiblen Skripting-Lösungen bis hin zu voll integrierten "All-in-one"-Simulationssystemen reichen.

MFsim ist eine offene Java-All-in-one-Rich-Client-Computerumgebung für mesoskopische Simulationen mit Jdpd als Standard-Simulationskern für die Dissipative Partikeldynamik. Die neue Umgebung umfasst die komplette Triade Vorbereitung-Simulation-Auswertung einer mesoskopischen Simulationsaufgabe und ermöglicht insbesondere biomolekulare Simulationsaufgaben mit Peptiden und Proteinen. Schwerpunkte sind ein SPICES-Molekülstruktureditor, ein PDB-zu-SPICES-Parser für partikelbasierte Peptid-/Proteindarstellungen, eine Unterstützung von Polymerdefinitionen, ein Kompartiment-Editor für komplexe Simulationsbox-Startkonfigurationen, interaktive und flexible Simulationsbox-Ansichten einschließlich Analytik, Erzeugung von Simulationsfilmen oder animierten

Diagrammen. Als offenes Projekt erlaubt MFsim spezifische Erweiterungen für verschiedene Forschungsbereiche.

Die Wechselwirkung von Zyklotiden mit Zweischichtmembranen wird für die Zyklotide Kalata B1, Cycloviolacin O2 und ausgewählte Mutanten mit Dissipativer Partikeldynamik untersucht. Für eine semi-quantitative Bioaktivitätsschätzung wird eine kinetische Geschwindigkeitskonstante für die anfängliche Membranlipid-Extraktion aus einem "Sandwich"-Wechselwirkungsmodell abgeleitet, bei dem zwei Doppelschichtmembranen ein Zyklotid/Wasser-Kompartiment umschließen. Die Struktur-Aktivitäts-Beziehungen weisen auf eine komplexe Bioaktivitätslandschaft mit Trends hin, die mit den experimentellen Befunden übereinstimmen: Cholesterin führt im Vergleich zu cholesterinfreien Membranen zu einer verminderten membranzerstörenden Aktivität, die Oberfläche des "hydrophoben Flecks" der Cyclotide ist wichtig für die Membraninteraktion und -zerstörung, der Austausch von oder mit geladenen Aminosäureresten kann zu Supermutanten mit einer über der nativen Bioaktivität liegenden Bioaktivität führen, und phosphoethanolaminreiche Membranen weisen eine erhöhte Membranzerstörung auf. Zyklotidgemische zeigen linear additive Bioaktivitäten. Eine Bildung von stabilen zyklotidinduzierten Membranporen mit definierten Durchmessern konnte nicht nachgewiesen werden.

2 Summary

SPICES (Simplified Particle Input ConnEction Specification) is a particle-based molecular structure representation derived from straightforward simplifications of the atom-based SMILES line notation. It aims at supporting tedious and error-prone molecular structure definitions for particle-based mesoscopic simulation techniques like Dissipative Particle Dynamics by allowing for an interplay of different molecular encoding levels that range from topological line notations and corresponding particle-graph visualizations to 3D structures with support of their spatial mapping into a simulation box.
An open Java library for SPICES structure handling and mesoscopic simulation support in combination with an open Java Graphical User Interface viewer application for visual topological inspection of SPICES definitions are provided.

Jdpd is an open Java simulation kernel for Molecular Fragment Dissipative Particle Dynamics with parallelizable force calculation, efficient caching options and fast property calculations. It is characterized by an interface and factory-pattern driven design for easy code changes and may help to avoid problems of polyglot programming.
Detailed input/output communication, parallelization and process control as well as internal logging capabilities for debugging purposes are supported. The new kernel may be utilized in different simulation environments ranging from flexible scripting solutions up to fully integrated “all-in-one” simulation systems.

MFsim is an open Java all-in-one rich-client computing environment for mesoscopic simulation with Jdpd as its default simulation kernel for Molecular Fragment (Dissipative Particle) Dynamics. The new environment comprises the complete preparation-simulation-evaluation triad of a mesoscopic simulation task and especially enables biomolecular simulation tasks with peptides and proteins. Productive highlights are a SPICES molecular structure editor, a PDB-to-SPICES parser for particle-based peptide/protein representations, a support of polymer definitions, a compartment editor for complex simulation box start configurations, interactive and flexible simulation box views including analytics, simulation movie generation or animated diagrams. As an open project, MFsim allows for customized extensions in different fields of research.

The disruptive interaction of cyclotides with bilayer membranes is studied for the cyclotides Kalata B1, Cycloviolacin O2 and selected mutants with Molecular Fragment (Dissipative Particle) Dynamics. For a semi-quantitative bioactivity estimate a kinetic rate constant for initial membrane lipid extraction is derived from a “sandwich” interaction model where two bilayer membranes enclose a cyclotide/water compartment. The structure-activity relationships indicate a complex bioactivity landscape with trends that are in agreement with experimental findings: Cholesterol leads to a decreased membrane-disrupting activity compared to cholesterol-free membranes, the cyclotide “hydrophobic patch” surface area is important for membrane interaction and disruption, the replacement of or with charged amino acid residues may lead to super-mutants with above-native bioactivity and phosphoethanolamine-rich membranes exhibit an increased membrane disruption. Cyclotide mixtures show linearly additive bioactivities. A formation of stable cyclotide-induced membrane pores with defined diameters could not be detected.

3 Preface

Most of the presented results are derived from the so far four publications that were released throughout the scientific working time of this thesis The forth manuscript is currently in press. These publications are namely: “SPICES: a particle-based molecular structure line notation and support library for mesoscopic simulation” [vandenBroek2018], “Jdpd - an open java simulation kernel for molecular fragment dissipative particle dynamics” [vandenBroek2018b], “MFsim - an open Java all-in-one rich-client simulation environment for mesoscopic simulation” [vandenBroek2020] and “Quantitative Estimation of Cyclotide-Induced Bilayer Membrane Disruption by Lipid Extraction with Mesoscopic Simulation” [vandenBroek2021].

Chapters 4.1, 4.2, 4.3, 4.4, 5.1, 5.2, 5.3, 5.4, 6.1, 6.2 6.3 and 6.4 are composed of extensive citations of the stated publications and abstracts were not marked separately.

The content of the above-mentioned papers was not used or published in any other way than for this dissertation.

4 Introduction and R&D objectives

4.1 Molecular structure representation

A molecular simulation task comprises three successive steps: The definition of a simulation job with all necessary input information (preparation step), the actual loop over discrete integration time steps to numerically solve the equations of motion (the actual simulation step) and the analysis of the simulation record with all calculated results (evaluation step). The first (preparation) step of this triad has to provide data structures that can be leveraged by the algorithms of the second (simulation) step in an optimized manner to allow for a maximum performance of their interplay. This is commonly achieved by definition of adequate sets of arrays that encode all necessary molecular information like spatial positions or bonds of the interacting entities. The content of these arrays is usually provided by large tabular ASCII files that are often (at least partly) edited by hand. An example of these ASCII files may be found at [PositionBonds2018] for 1,2-Dimyristoyl-sn-glycero-3-phosphocholine (DMPC) phospholipid molecules of a bilayer-membrane simulation task where each line contains an interacting entity, its spatial x,y and z coordinates, line offsets to bonded entities and specific indices for additional force assignments. The manual creation of these machine-oriented contents is not only a tedious but an error-prone type of work: For all but the simplest molecular ensembles errors are likely to be generated that may spoil the whole simulation process. Thus there is a valid necessity to prevent mistakes by safeguarded operations and to reduce manual preparation overhead by adequate automation.

Cheminformatics aims at supporting efficient and errorless human-machine interfaces where adequate molecular structure representations (line notations, connection tables, XYZ tables or Z-matrices, fragment codes or fingerprints, file formats like MOL file or PDB file) are at heart of the discipline [Engel2018]. The majority of existing structure representations are atom-based descriptions that comprise characteristic properties and topological or spatial aspects concerning a molecule's atomic composition [Engel2018, Engel2018b] with additional approaches towards fragment-based molecular representations especially for polymers [Siani1994, Siani1995, Drefahl2011, Zhang2012, Dufresne2015]. In order to support the preparation step of a molecular simulation task cheminformatics methods allow for an effective interplay of different levels of molecular encoding that are constitutive for a comfortable and safe human-machine interface (see figure 1): The topological structural formula is a common way used by molecular scientists to represent a chemical compound (e.g. drawn by hand with a structure editor or manually selected from structure repositories).

Alternatively the compound may be represented by a textual line notation – where the interplay between structural formula and line notation may be realized by mutual conversion methods like an adequate structure diagram layout. The following transition from topological representations to 3D structures allows for the final mapping to their spatial positions within a simulation box which completes the preparation step. All prepared information may then be stored in form of the tabular ASCII files sketched above as an input for the actual simulation step.

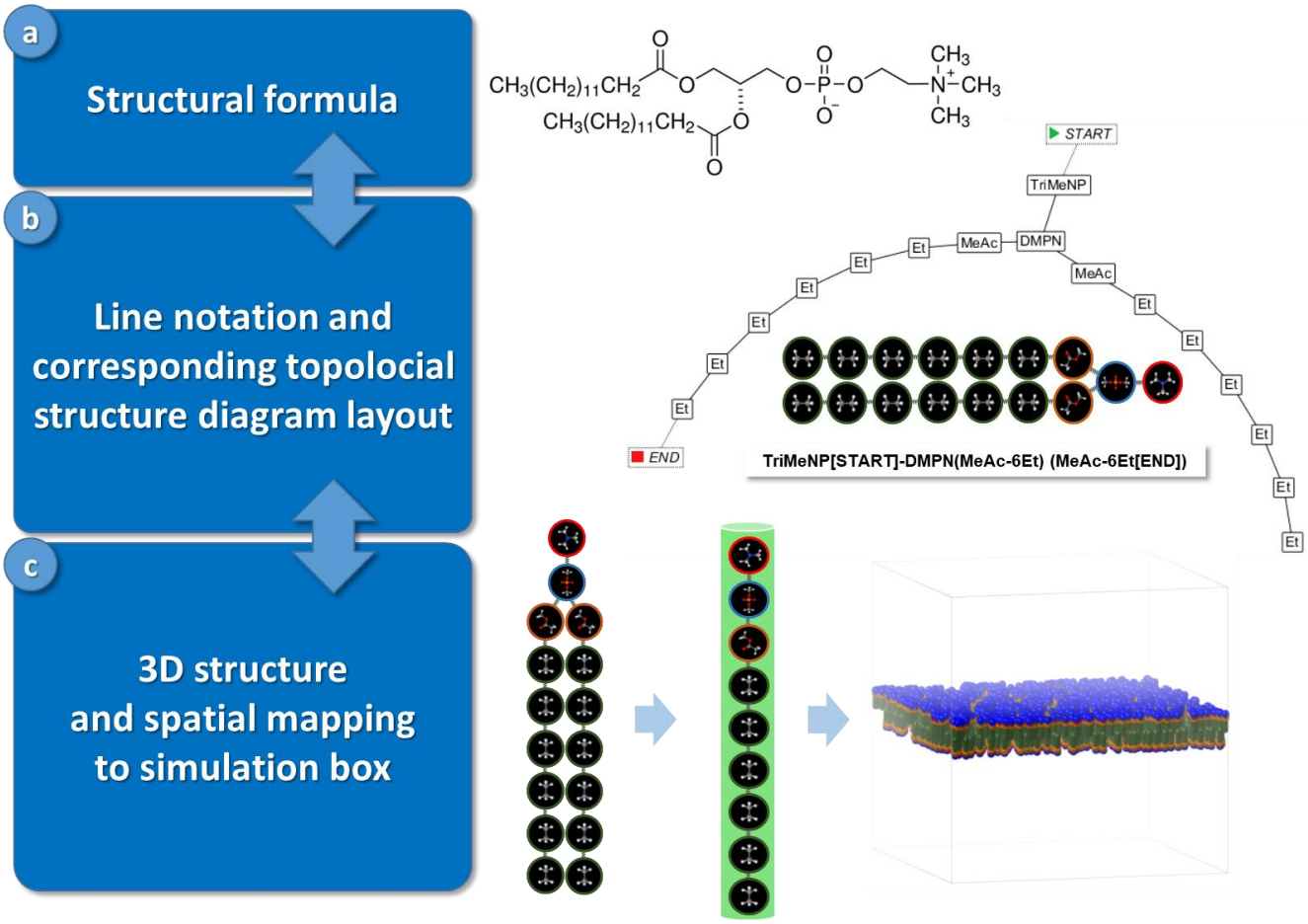


Figure 1. Interplay between different encoding levels of molecular structures for a preparation step of a molecular simulation task (with examples of this work, compare figure 2, figure 5 and figure 6): (a) Structural formula of a DMPC phospholipid. (b) SPICES line notation of the particle-based topological DMPC structure with its corresponding structure diagram layout/particle graph and illustration of the particle bonds. (c) Conversion of the topological particle structure to a compressed 3D tube geometry plus spatial mapping into an oriented bilayer compartment of the simulation box.

4.1.1 Development objective

In order to contribute to the realization of a molecular fragment cheminformatics roadmap [Truszkowski2014] this work tries to alleviate molecular structure handling and encoding for particle-based mesoscopic simulation techniques like Dissipative Particle Dynamics (DPD). Since no unique molecular fragmentation scheme exists for the various mesoscopic simulation approaches there is nothing like a universal particle set. An adequate decomposition of a chemical compound into appropriate "molecular fragment" particles is a kind of artisan craftwork which is guided by experience, empirical rules and field of application. Figure 2 demonstrates a possible fragmentation for a DMPC phospholipid that successfully preserves its amphiphilic characteristics [Truszkowski2015]. Key part of this work is a set of methods

operating on an intuitive line notation for particle-decomposed molecular structures denoted SPICES (Simplified Particle Input ConnEction Specification).

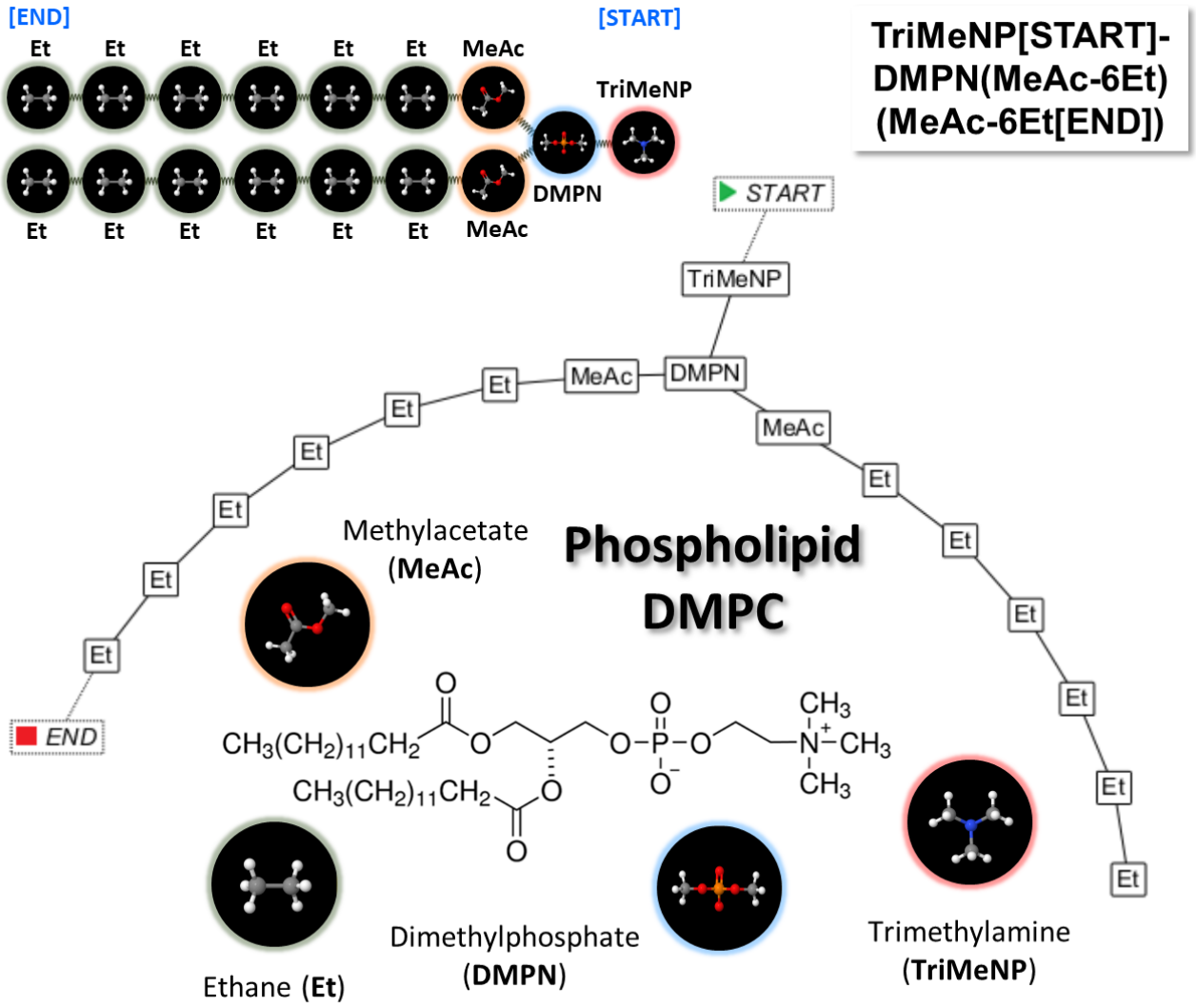


Figure 2. Decomposition of the DMPC phospholipid into "molecular fragment" particles [Truszkowski2015] and illustration of the resulting bonded particles (upper left) with corresponding SPICES line notation (upper right): The *SpicesViewer* GUI generated visual particle graph surrounds the "molecular fragment" particle identification.

4.2 Mesoscopic simulation kernel

Mesoscopic simulation aims at describing supramolecular phenomena at the nanometer (length) and microsecond (time) scale for large interacting physical ensembles (representing millions of atoms) within comparatively short computational time frames (hours) by "coarse grained" neglect of uninteresting degrees of freedom. Dissipative Particle Dynamics (DPD) is a mesoscopic simulation technique for isothermal complex fluids and soft matter systems that combines features from Molecular Dynamics (MD), Langevin Dynamics and Lattice-Gas Automata [Hoogerbrugge1992, Koelman1993, Espanol1995, Espanol1995b, Groot1997]. It satisfies Galilean invariance and isotropy, conserves mass and momentum and achieves a rigorous sampling of the canonical NVT ensemble due to soft particle pair potentials that diminish molecular entanglements or caging effects [Hoogerbrugge1992, Koelman1993, Groot1997]. DPD is expected to show correct hydrodynamic behavior and to obey the Navier-Stokes equations [Hoogerbrugge1992, Koelman1993].

DPD particle trajectories are guided by Newton's equation of motion where the total force on a particle exerted by other particles consists of a conservative, a dissipative (frictional) and a random part:

$$m_i \frac{d^2 \underline{r}_i}{dt^2} = \underline{F}_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\underline{F}_{ij}^C + \underline{F}_{ij}^D + \underline{F}_{ij}^R \right)$$

$m_i, \underline{r}_i$, mass and position vector of particle i; t, time; $\underline{F}_i$, total force on particle i exerted by other particles j; N, number of particles in simulation; $\underline{F}_{ij}^C, \underline{F}_{ij}^D, \underline{F}_{ij}^R$, conservative, dissipative and random force on particle i exerted by particle j.

Dissipative (frictional) and random forces oppose each other and act as a thermostat conserving the total momentum and introducing Brownian motion into the system. The conservative forces comprise soft DPD particle repulsions (with a common cut-off length of 1 DPD unit) as well as harmonic springs between bonded particles and electrostatic interactions between charged particles (the model to be implemented for the latter is an ad-hoc approach to take electrostatic long-range interactions between "a few" charged particles in the simulation box into account – details are outlined in [Truszkowski2015]

where an application to biological membranes and protein models is described. A theoretically more sound treatment of electrostatic interactions like those proposed in [Groot2003, González-Melchor2006, Ibergay2009] may be addressed by future developments with the current technical implementation of electrostatics interactions as a useful blueprint to alleviate coding efforts):

$$\underline{F}_{ij}^{C} = \underline{F}_{ij}^{C,DPD} + \underline{F}_{ij}^{C,Bond} + \underline{F}_{ij}^{C,Estat}$$

$$\underline{F}_{ij}^{C,DPD}\left(\underline{r}_{ij}\right) = \begin{cases} a_{ij}\left(1-r_{ij}\right)\underline{r}_{ij}^{0} & \text{for } r_{ij} < 1 \\ 0 & \text{for } r_{ij} \geq 1 \end{cases}$$

$$\underline{F}_{ij}^{C,Bond} = -k_{Bond}\left(r_{ij} - r_{Bond}\right)\underline{r}_{ij}^{0}$$

$\underline{F}_{ij}^{C,DPD}, \underline{F}_{ij}^{C,Bond}, \underline{F}_{ij}^{C,Estat}$, soft repulsive DPD force, harmonic bond force and electrostatic force on particle i exerted by particle j; a_{ij}, maximum isotropic repulsion between particles i and j; $\underline{r}_{ij} = \underline{r}_i - \underline{r}_j = r_{ij}\,\underline{r}_{ij}^{0}$; $\underline{r}_{ij}^{0}$, unit vector; k_{Bond}, spring constant of bond; r_{Bond}, bond length.

Dissipative force

$$\underline{F}_{ij}^{D}\left(\underline{r}_{ij}, \underline{v}_{ij}\right) = -\gamma\,\omega^{D}\left(r_{ij}\right)\left(\underline{r}_{ij}^{0} \cdot \underline{v}_{ij}\right)\underline{r}_{ij}^{0}$$

γ, friction coefficient; $\omega^{D}\left(r_{ij}\right)$, dissipative force distance variation; $\underline{v}_i$, velocity of particle i; $\underline{v}_{ij} = \underline{v}_i - \underline{v}_j$.

and random force

$$\underline{F}_{ij}^{R}\left(\underline{r}_{ij}\right)=\sigma\,\omega^{R}\left(r_{ij}\right)\,\frac{\zeta_{ij}}{\sqrt{\Delta t}}\,\underline{r}_{ij}^{0}$$

σ, noise amplitude; $\omega^{R}\left(r_{ij}\right)$, random force distance variation; ζ_{ij}, random number with zero mean and unit variance; Δt, integration time step [Groot1997].

depend on each other in a canonical NVT ensemble (again a cut-off length of 1 DPD unit is applied):

$$\gamma=\frac{\sigma^{2}}{2\,k_{B}T}$$

$$\omega^{R}\left(r_{ij}\right)=\sqrt{\omega^{D}\left(r_{ij}\right)}=\begin{cases}1-r_{ij} & \text{for}\quad r_{ij}<1\\ 0 & \text{for}\quad r_{ij}\geq 1\end{cases}$$

k_B, Boltzmann constant; T, thermodynamic temperature.

Since dissipative DPD forces depend on relative particle velocities, the common MD Velocity-Verlet integration of the equations of motion has to be modified [Groot1997].

Whereas DPD particles in general may be arbitrarily defined “fluid packets” [Koelman1993], the Molecular Fragment “bottom-up” DPD variant identifies each particle with a distinct small molecule of molar mass in the order of 100 Da. Larger molecules are composed of adequate smaller “molecular fragment” particles that are bonded by harmonic springs to mimic covalent connectivities and spatial 3D conformations (compare figure 2 and figure 19). Thus this variant may be regarded as a chemically intuitive and molecular accurate “fine grained” version of the intrinsically “coarse grained” DPD technique [Groot1997, Groot1998, Ryjkina2002, Schulz2004, Truszkowski2013, Vishnyakov2013, Truszkowski2014, Truszkowski2015].

The a priori choice of specific particles (the analogue of the atom types of an atomistic molecular force field) in combination with their repulsive a_{ij} interaction parameters (the

analogue of the force field parameters) approximate the detailed interactions within the molecular ensemble under study, and thus are essential for an adequate description of its dynamical behavior. In a previous publication [Truszkowski2015] a preliminary particle set for biomolecular simulations was proposed. The derivation procedure for adequate estimates of the particle pair repulsion parameters a_{ij} followed the approach developed in [Groot1997]

$$a_{ij}(T) = 75\frac{k_B T}{\rho_{DPD}} + 3.4965\, k_B T\, \chi_{ij}(T)$$

$$\chi_{ij}(T) = \frac{Z_{ij}\, E_{ij} + Z_{ji}\, E_{ij} - Z_{ii}\, E_{ii} - Z_{jj}\, E_{jj}}{2k_B T}$$

with k_B, Boltzmann constant; T, thermodynamic temperature; ρ_{DPD}, DPD density; χ_{ij}, Flory-Huggins interaction parameter between particles i and j; Z_{ij}, coordination number of particles j around particle i; E_{ij}, interaction energy between particles i and j;

which related the isotropic particle pair repulsions a_{ij} with Flory-Huggins interaction parameters χ_{ij}. An estimate of the Flory-Huggins interaction parameters χ_{ij} requires corresponding estimates of the coordination numbers Z_{ij} and inter-particle interaction energies E_{ij}. In [Truszkowski2015] the coordination numbers Z_{ij} were estimated by a static "packaging" approach of particles j around a particle i. The inter-particle interaction energies E_{ij} were approximated with MD simulations by averaging the non-binding interactions of the particle molecules which need to be adequately approximated by the underlying force field: For this purpose the *Condensed-phase Optimized Molecular Potentials for Atomistic Simulation Studies* (COMPASS) force field [Sun1998] was chosen which had been parameterized for the most accurate reproduction of enthalpies of evaporation of fluid systems so that the crucial non-binding electrostatic and van der Waals forces are taken into account properly. Despite its simple three-point water model, it satisfactorily describes important water properties: With parametrization to reproduce the properties of aqueous systems under ambient conditions, it provides reliable density

predictions [Rigby2004]. For a pure water system it leads to an inter-molecular potential energy arising from non-bonding interactions that compares favorably with experiment, and the average O-H bond length is similar to its empirical value. The average H-O-H bond angle is found to be too small, which is an unfavorable bias for the formation of H-bonds – although this geometric deficiency is not expected to inhibit the formation of H-bonds between water and organic molecules [Sutton2005]. Mesoscopic particle-particle interactions derived with the COMPASS force field have been successfully applied to simulate aqueous systems in accordance with experimental results [Ryjkina2002, Schulz2004, Truszkowski2013].

A simulation kernel software comprises the fundamental data structures and numerical calculation algorithms that are necessary to approximate the temporal evolution of a defined particle ensemble. DPD (as well as similar MD) code consists basically of a main loop over (non-parallelizable) successive simulation steps in which (parallelizable) particle pair force evaluations are the most time-consuming part [Allen1987, Frenkel2002]. Thus parallelization efforts focus mainly on these force calculations in order to speed up simulations.

4.2.1 Development objective

This work aims at developing a general open simulation kernel for Molecular Fragment DPD in Java – denoted Jdpd – that especially takes the requirements for efficient biomolecular simulations into account.

4.3 Mesoscopic simulation environment

Setting up a molecular simulation involves three coordinated steps: First of all a preparation step in which the simulation job is defined and all required input information is given. Subsequently the actual simulation step has to be performed by approximate numerical integration of the equations of motion. Finally the last step – the evaluation step – follows where the simulation record with all calculated results has to be analyzed. A computational all-in-one rich-client environment aims at supporting this triad in a unified and comprehensive manner to allow for productive applicability with minimum effort, minimized training periods and usability without programming skills. An additional goal is the prevention of common problems like inadequate or ill-defined parameter settings by extensively safeguarding simulation job definitions and operations. In general, these desirable features require a highly integrated monolithic architecture: Its pre-defined optimized workflows are comfortable and

fast but inflexible in comparison with scripting or pipelining-workflow approaches. For the latter the implementation of a new feature may be achieved within minutes whereas its rich-client realization may involve days and weeks of complex software development. Thus, the different approaches have their intrinsic strengths and weaknesses.

4.3.1 Development objective

A general open "all-in-one" rich-client simulation environment on top of the Jdpd simulation kernel – denoted MFsim – is to be developed that especially enables biomolecular simulations with Molecular Fragment DPD containing peptides and proteins. The intended computational environment will especially enable the study of cyclotide/membrane interactions that could not be performed otherwise.

4.4 Cyclotide/membrane interactions

Cyclotides are plant-derived macrocyclic peptides of about 30 amino acids with their N- and C-termini being connected by a peptide bond. They contain a characteristic structural motif, the cyclic cystine knot (CCK), and can be divided into the Möbius and Bracelet subfamilies by a particular twist of their cyclic backbone (as well as a third main subgroup of trypsin inhibitor cyclotides which are not taken into account in this work). Due to their cyclic structure with three additional disulfide bridges, cyclotides possess extraordinarily stable spatial conformations [Craik1999]. Over the last two decades cyclotides attracted increasing attention due to their interesting range of biological activities (like cytotoxic, hemolytic, anti-HIV, anti-cancer, anthelmintic, anti-fouling, antimicrobial or insecticidal effects) with a potential use as drug scaffolds and therapeutic agents. An overview of relevant studies was recently provided by a comprehensive review [deVeer2019].

Cyclotide bioactivities may at least partially be traced to specific interactions with lipid bilayer membranes [Lindholm2002, Barry2003, Herrmann2006, Shenkarev2006, Svangard2007, Simonsen2008, Wang2009, Wang2009b, Huang2009, Göransson2009, Gerlach2010, Huang2010, Pränting2010, Wimley2010, Burman2011, Burman2011b, Henriques2011, Wimley2011, Wang2012, Henriques2012b, Henriques2012, Göransson2012, Henriques2014, Burman2014, Henriques2015, Weidmann2016, Cranfield2017, Ghani2017, Grage2017, Henriques2017, Henriques2019] which have also been demonstrated by simulation models [Nawae2014, Nawae2014b, Truszkowski2015, Nawae2017]: Whereas a single or a few cyclotide peptides already interact with membranes

they do neither impair their structural integrity nor their segregating function. However, a collective interaction of multiple cyclotides is able to distort or to disrupt the bilayer structure. Different modes of action depending on the cyclotide/membrane system are to be expected: These comprise cyclotide membrane binding [Herrmann2006, Shenkarev2006, Simonsen2008, Wang2009, Wang2009b, Göransson2009, Huang2010, Pränting2010, Burman2011, Burman2011b, Henriques2011, Wang2012, Henriques2012, Göransson2012, Henriques2014, Nawae2014b] and intrusion [Henriques2015, Grage2017, Henriques2017] as well as cell penetration [Shenkarev2006, Wang2012, Henriques2015, Henriques2017, Ghani2017] up to the final ability to disrupt a membrane by a postulated formation of pores [Lindholm2002, Herrmann2006, Shenkarev2006, Svangard2007, Wang2009, Huang2009, Gerlach2010, Huang2010, Wimley2010, Burman2011, Henriques2011, Wimley2011, Henriques2012, Wang2012, Henriques2012b, Göransson2012, Burman2014, Nawae2014, Nawae2014b, Truszkowski2015, Nawae2017, Cranfield2017, Grage2017] or by distinct lipid extraction [Burman2011, Nawae2014, Truszkowski2015, Nawae2017].

Lipid extraction as a particular mechanism for membrane disruption was experimentally shown in [Burman2011] as well as detected by coarse-grained Molecular Dynamics (MD) [Nawae2014, Nawae2017] and mesoscopic [Truszkowski2015] simulations. In [Nawae2014], a detailed interfacial bioactivity model is described for cyclotide Kalata B1 where two configurations of Kalata B1 oligomers, denoted tower-like and wall-like clusters, are formed. Conjugation between wall-like clusters results in the formation of a ring-like hollow on the membrane surface. This interfacial membrane binding of Kalata B1 induces a positive membrane curvature with a subsequent lipid extraction from the membrane through the ring-like pore into the Kalata B1 cluster.

These promising findings should not hide the fact that the detailed biophysical aspects of cyclotide/membrane bioactivity as a whole are still poorly understood. To complement the experimental studies, it may be helpful to establish fast approximate simulation models that are able to reflect at least aspects of the observed bioactivity trends to generate productive directions for further research and practical applications. Unfortunately, these simulation models require to study large chemical ensembles on the microsecond timescale in order to account for the collective cyclotide-membrane interactions.

Dynamic simulations of biomolecular processes on the nanometer lengthscale and microsecond timescale are challenging. The system size of the corresponding molecular ensembles comprises millions of atoms so that atomistic Molecular Dynamics (MD) studies of

their temporal evolution are sophisticated and time consuming – despite their substantial recent speed-up due to GPU computing [Kutzner2019] or coarse-grained MD techniques with a reduced number of interacting physical units [Marrink2007]. Mesoscopic simulations are comparatively easy to set up and orders of magnitude faster: A molecular ensemble with a million particles (representing several million atoms) may be investigated with "bottom-up" Dissipative Particle Dynamics (DPD) for several microseconds per day on a common multicore processor workstation [vandenBroek2018b]. However, the increased speed inevitably implies a loss of resolution: The detailed atomistic description has to be replaced by a coarse-grained "blurred" bead/particle representation with at best adequately averaged isotropic particle-particle interactions. Stable molecular conformations like rigid ring systems or peptide/protein secondary structures (α-helices, β-sheets) must be specifically imposed on connected particles to prevent their spatial collapse. Thus, mesoscopic simulation approaches mainly focus on studying the temporal evolution of comparatively large nanoscale supramolecular structures that emerge from non-bonded molecule-molecule interactions [Moeendarbary2009, Espanol2017].

4.4.1 Research objective

This work is based on preliminary findings in [Truszkowski2015] on cyclotide/membrane interactions which qualitatively demonstrated that lipid extraction may be characterized by a mesoscopic simulation approach with Molecular Fragment DPD. The geometrical simulation setup is extended towards a "sandwich" cyclotide/membrane interaction model where two bilayer membranes enclose a cyclotide/water compartment that allows for evaluations of the interaction phenomena in a quantitative manner: It attempts to evaluate an approximate initial rate of membrane lipid extraction (abbreviated r_{MD}) as a certain aspect of the kinetics of membrane disruption on the microsecond timescale in order to obtain corresponding structure-activity relationships (SAR) that can be compared with experimental findings and tendencies. The proposed "sandwich" model is applied to a range of different cyclotide/membrane systems containing the Möbius cyclotide Kalata B1 (abbreviated kB1), the Bracelet cyclotide Cycloviolacin O2 (abbreviated cO2) as well as some of their mutants. All necessary operations to realize the sketched setup, to perform the necessary simulations and to enable the evaluation procedures are supported by MFsim and Jdpd.

The SAR evaluations can support cyclotide comparison or ranking tasks, allow for an identification of relevant structural features or provide hints for possible molecular interaction

mechanisms. Finally they may contribute to a more elaborated description of the complex cyclotide/membrane bioactivity landscape and at best lead to predictive assumptions for further progress (see figure 3).

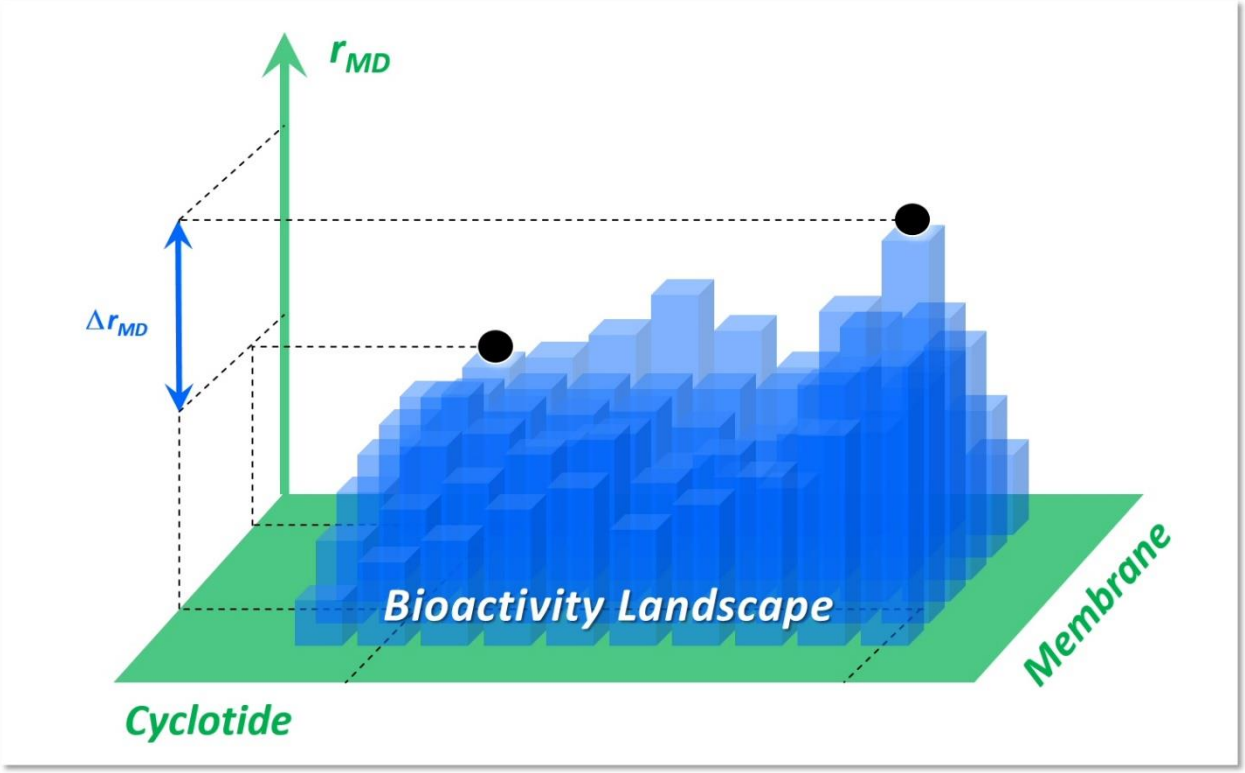


Figure 3. Schematic view of the cyclotide/membrane bioactivity landscape targeted by this work where each bar represents the membrane disruption activity of a specific cyclotide/membrane system and each row (column) of bars corresponds to a specific membrane (cyclotide). The dynamical rate of cyclotide-induced membrane disruption by lipid extraction is abbreviated with r_{MD}.

5 Results

5.1 SPICES molecular structure representation

The SPICES design is derived from straightforward simplifications of the well-established SMILES representation for atom-based molecular connectivity [Weininger1988, Weininger1989, Weininger1990]. The set of SPICES-related methods supports the interplay of structural encoding levels (see figure 1) as well as structure-based calculations for mesoscopic simulations e.g. length and time scales, simulation box size or compound concentrations: It allows for parsing and (graphically) analyzing the line notations, topological calculations (e.g. particle frequencies, particle neighbors or particle paths) as well as the generation of corresponding 3D particle structures with support of their spatial mapping into the simulation box and the final output of tabular ASCII files with molecular information for the following simulation step (the construction of the tabular ASCII file at [PositionBonds2018] was in fact supported by the SPICES-related code of this work).

The SPICES implementation extends the fragment structure representation proposal from [Truszkowski2014]. The syntax rules for a correct SPICES line notation together with some helpful comments are outlined in appendix 1. These rules allow arbitrary topological particle connections with branches and ring closures but do not comprise attributes like electric charges or chiral centers since these are intrinsic particle properties (i.e. differently charged states or different enantiomers of a “molecular fragment” particle have to be coded with different particles where each particle has a specific charge and a specific stereochemistry). Particles may possess a “backbone” label which may be utilized to assign specific particle pair forces e.g. for spatial 3D structure constraints of ring structures (see figure 4), the tail stiffness of surfactants and lipids or the backbone conformation of macromolecules like proteins. The intention here is to retain the characteristics of the molecule of interest, e.g. preserving the stiffness of the steroid scaffolding via specific inter-particle forces to prevent collapsing. This kind of labeling could be performed in an automated manner by attaching a tagging label to each particle. Nevertheless, the user control of the “backbone” label distribution within a molecule alleviated possible manual force assignments as well as the interplay between the textual line notation and the corresponding visual particle graph. In addition the concrete force assignments were chosen to be not a part of the line notation itself due to their intrinsic differences (from simple springs to e.g. complicated polygonal force chains) and possible automated conditional assignments according to various criteria. Thus the manual “backbone”

labels allow for a flexible post-processing for different purposes in the aftermath of molecular definitions.

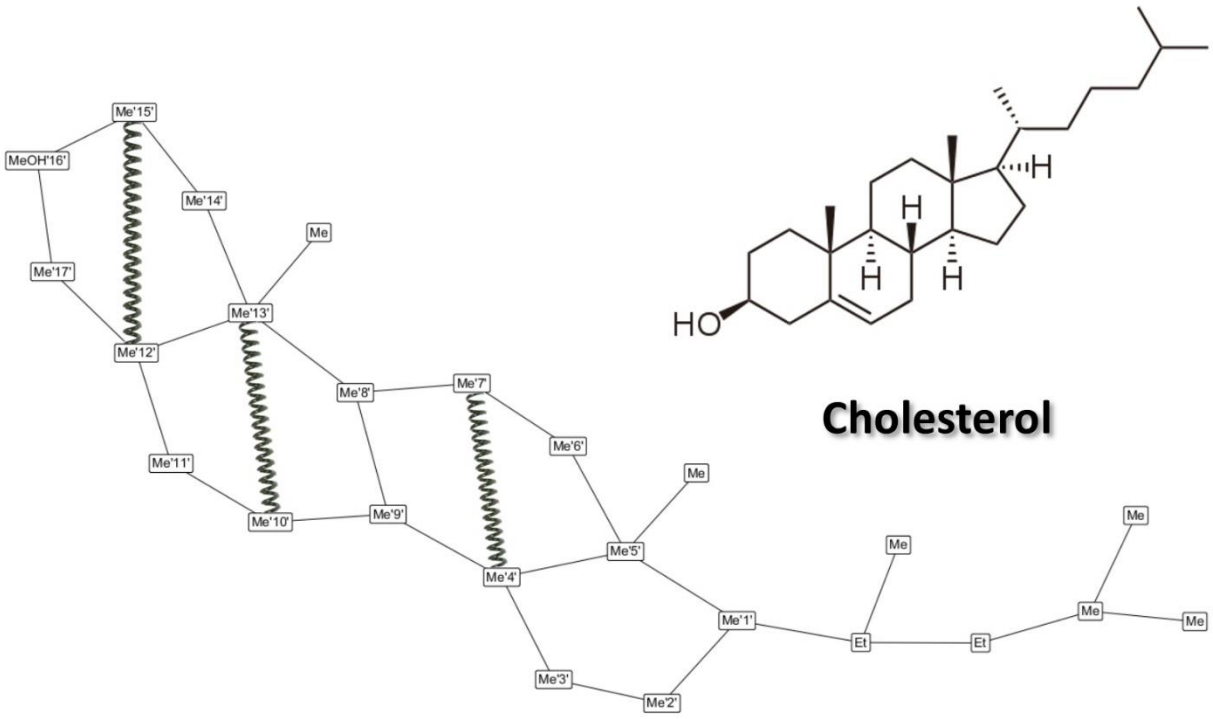


Figure 4. Cholesterol fragmentation scheme with SPICES line notation (at the bottom) [vandenBroek2018]. The specified backbone labels '1' to '17' allow for an assignment of specific inter-particle forces (e.g. the shown harmonic springs between particles Me'12' and Me'15', Me'10' and Me'13' and Me'4' and Me'7') in order to control the stiffness of molecular structure elements like the cholesterol ring structure. This choice of setting the harmonic springs is just one possibility, a more favorable version is presented in figure 19 (right).

A SPICES representation may contain multiple independent parts (with each part being a valid molecule), e.g. to represent aggregated molecular structures like the quaternary structure of proteins. Finally a [START] and an [END] tag may be attributed for spatial orientation in the simulation box, see figure 2, figure 5 and figure 6.

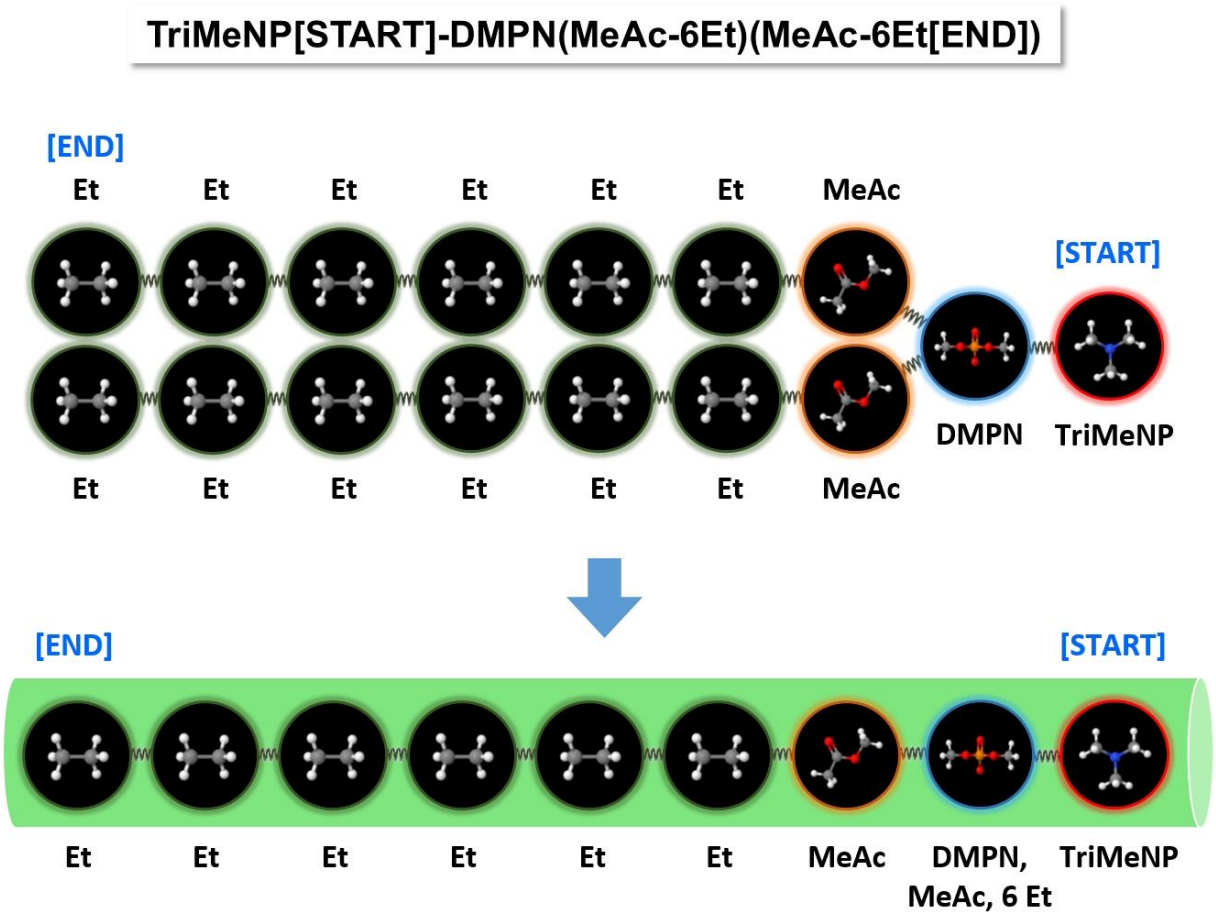


Figure 5. Top: Phospholipid DMPC fragmentation scheme [Truszkowski2015] with 16 particles connected by harmonic springs (compare to figure 2). Bottom: For spatial mapping into the simulation box the topological DMPC particle structure is converted to a linear 3D tube along the [START]/[END] tagged main chain where side-chain particles are collapsed onto the spatial positions of their neighbored main-chain particles, i.e. the second spatial position to the right contains 8 particles with the exact same position: The main-chain particle DMPN and the side-chain particles MeAc and 6 Et [vandenBroek2018]. This approach is just an initial procedure to position a molecule into the simulation box. In advance of every simulation run energy minimization steps are performed, in which - in the case of a phospholipid – the second side-chain unfolds again into the chemically more sensible form.

The *Spices.jar* library supports all aspects of SPICES definition and handling. A *Spices* object may be created with at least an input structure string or in combination with additional information like a map of available particles. A syntax parser analyzes the provided line notation and returns detailed syntax error information if necessary by the methods *isValid* and *getErrorMessage*. SPICES properties like the frequency of particles or complete lists of particle neighbors are evaluated upon user request by the methods *getParticleFrequencies* or *getNextNeighbors*.

A function of specific importance is the spatial projection of topological SPICES into a simulation box to set up adequate start geometries. Since a mesoscopic simulation is driven by soft particle potentials (in contrast to atomic hard core repulsions for e.g.

molecular dynamics), different particles may occupy the same exact spatial position (which would lead to infinite forces for hard atomic potentials) as well as penetrate each other. Thus the possibly severe problems of particle entanglements or caging effects due to inadequate start geometries are considerably attenuated [Groot2003]. Larger occurring empty spaces during the setup of start geometries are negligible, because they are filled immediately within the initial minimization steps before every simulation run (the volume is defined in a manner, that the box has the correct density). Nonetheless, a more favorable initial configuration may considerably reduce the necessary simulation period.

A straightforward approach is a spatial linear tube representation [Truszkowski2014] as shown in figure 5 and figure 6: The longest linear particle chain in the molecule is determined and its particles are consecutively lined up along a straight line according to the specified bond length (which may be squeezed to fit into specific compartments like simulation box layers, see below and figure 6). Then all branched side particles are collapsed onto their nearest-neighbor particle on this line.

For a fast determination of a sufficiently long linear particle chain, the Depth-First Search (DFS) algorithm is used [Wayne2011]. Starting from the first particle of the SPICES line notation the maximum-distant particle A is evaluated by a first DFS run. With a second DFS run, the maximum-distant particle B from particle A is determined. Finally the particle chain between A and B is chosen for the spatial tube representation. If a [START]/[END] tag pair is defined, the longest (oriented) linear chain between the tagged particles is evaluated. The sketched algorithm leads to true longest chains for acyclic SPICES but not necessarily for cyclic particle structures (long chains will be found, but possibly not the mathematically longest).

For a distinct fragmentation scheme of a molecule there may be several different but equally valid SPICES line notations since the proposed line notation is not canonically unique. For acyclic SPICES with a defined [START]/[END] tag pair, the sketched 3D tube construction process will lead to a single distinct spatial 3D tube representation for all these possible different line notations (without a defined [START]/[END] tag pair there may be two possible orientations). For cyclic particle structures this may not be the case, i.e. different but equally valid SPICES line notations may lead to different spatial 3D tube representations and corresponding different start geometries of a simulation.

This shortcoming is of minor practical relevance since the possibly different 3D tube representations for small molecules seem to be sufficiently similar for convergent mesoscopic simulation results. On the other hand, for large complex molecules like cross-

linked (bio)polymers the simple linear 3D tube representation is questionable in principle so that specific conversion tools like a PDB-to-SPICES parser for peptides and proteins would be advised which would take the known molecular 3D structure into account.

The sketched spatial projection (see figure 6) is accomplished by the interplay of the methods *setCoordinates* and *getParticlePositionsAndConnections*: After creation of a *Spices* object from a SPICES line notation string (which is rapidly performed within a fraction of a second for small molecules like DMPC) arrays for the first (start) and the last (end) particle positions of all spatial linear 3D tubes as well as the bond length may be provided via the *setCoordinates* method. The first (start) particles of the linear chains always have the defined start positions whereas the last (end) particles may not necessarily reach the defined end positions if the length of the defined start/end straight line is longer than the accumulated bond lengths of the particles on the longest linear chain so that a 3D tube may be smaller than defined.

On the other hand a 3D tube may be squeezed (with equally reduced bond lengths) if the length of the defined start/end straight line is smaller than the accumulated bond lengths. Thus the calling code (e.g. a compartment editor that allows for flexible compartment definitions within the simulation box like the bilayer compartment shown right in figure 6) must only define correctly-oriented and valid lines within an arbitrary compartment (which is comparatively simple to realize) without the necessity to calculate and pre-check every individual length (which could be more difficult).

The method *getParticlePositionsAndConnections* then provides all corresponding particle positions within the simulation box where in addition all particle-particle bonds are coded with specific offsets which are commonly used by simulation kernels (compare to the tabular ASCII file at [PositionBonds2018]). The sketched interplay of the methods *setCoordinates* and *getParticlePositionsAndConnections* performs sufficiently fast for true on-the-fly calculations, e.g. a spatial projection of 50,000 DMPC molecules (with 800,000 particles) into the simulation box performs in less than a second using an ordinary scientific workstation or even a standard notebook computer.

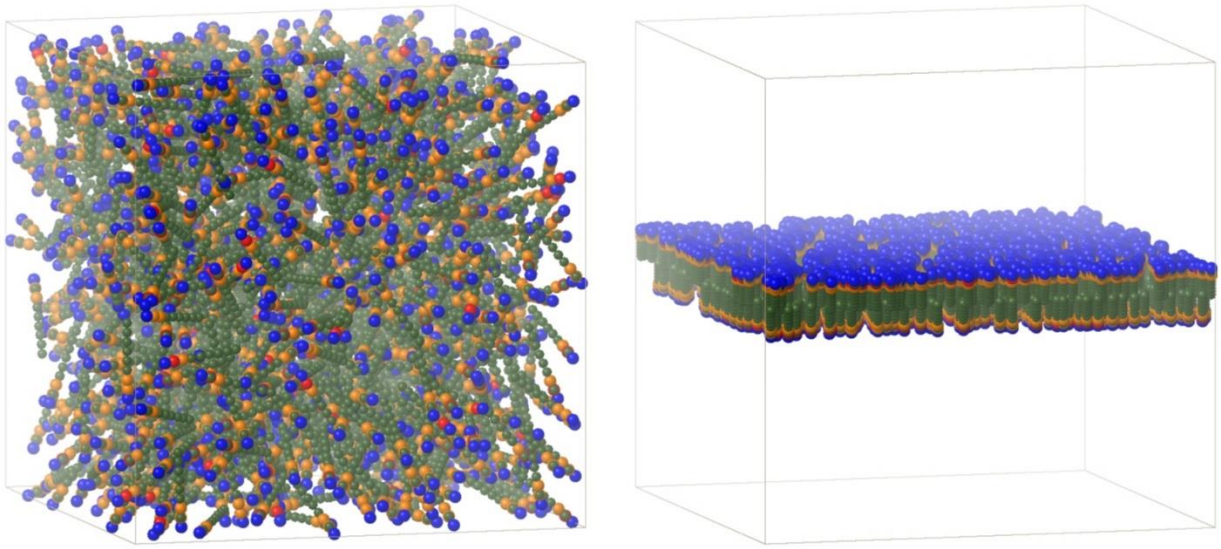
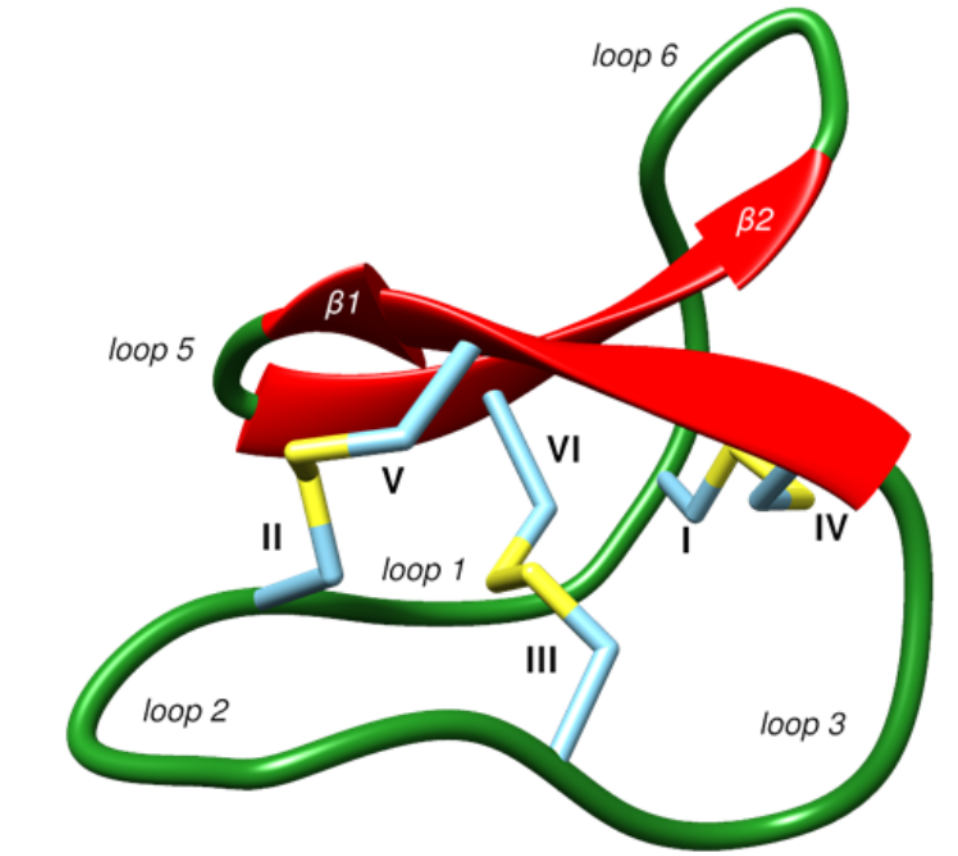


Figure 6. Simulation box start geometry with random distribution (left) or bilayer orientation (right) of phospholipid DMPC molecules as linear 3D tubes (see figure 1, figure 2 and figure 5) [vandenBroek2018]. Color code of particles: Et (olive), MeAc (orange), DMPN (red), TriMeNP (blue).

Whereas line notations may be regarded as a reasonable compromise for a human-machine interface (readable by human beings, decomposable by machine), their definitions are error-prone for complex branched or ring structures. A visual display of the topological particle graph with all its particle-particle connections may considerably alleviate a correct SPICES definition, see figure 7.



MeAcNHBB[4](MeSHSS[1])-MeAcNHBB-MeAcNHBB(Me-HAc)-MeAcNHBB(PrOH)-MeAcNHBB(MeSHSS[2])-MeAcNHBB(Pr)-MeAcNHBB-MeAcNHBB-MeAcNHBB(PrOH)-MeAcNHBB(MeSHSS[3])-MeAcNHBB(AcNH2)-MeAcNHBB(PrOH)-AzolidBB-MeAcNHBB-MeAcNHBB(MeSHSS[1])-MeAcNHBB(PrOH)-MeAcNHBB(MeSHSS[2])-MeAcNHBB(MeOH)-MeAcNHBB(Me-Pyrrole-Ph)-AzolidBB-MeAcNHBB(Pr)-MeAcNHBB(MeSHSS[3])-MeAcNHBB(PrOH)-MeAcNHBB(Pr-Guanidine)-MeAcNHBB(AcNH2)-MeAcNHBB-MeAcNHBB(Me-Pr)-AzolidBB-MeAcNHBB[4](Pr)

Figure 7. The cyclotide Kalata B1 with 29 amino acids [vandenBroek2018] according to the fragmentation scheme in [Truszkowski2015].

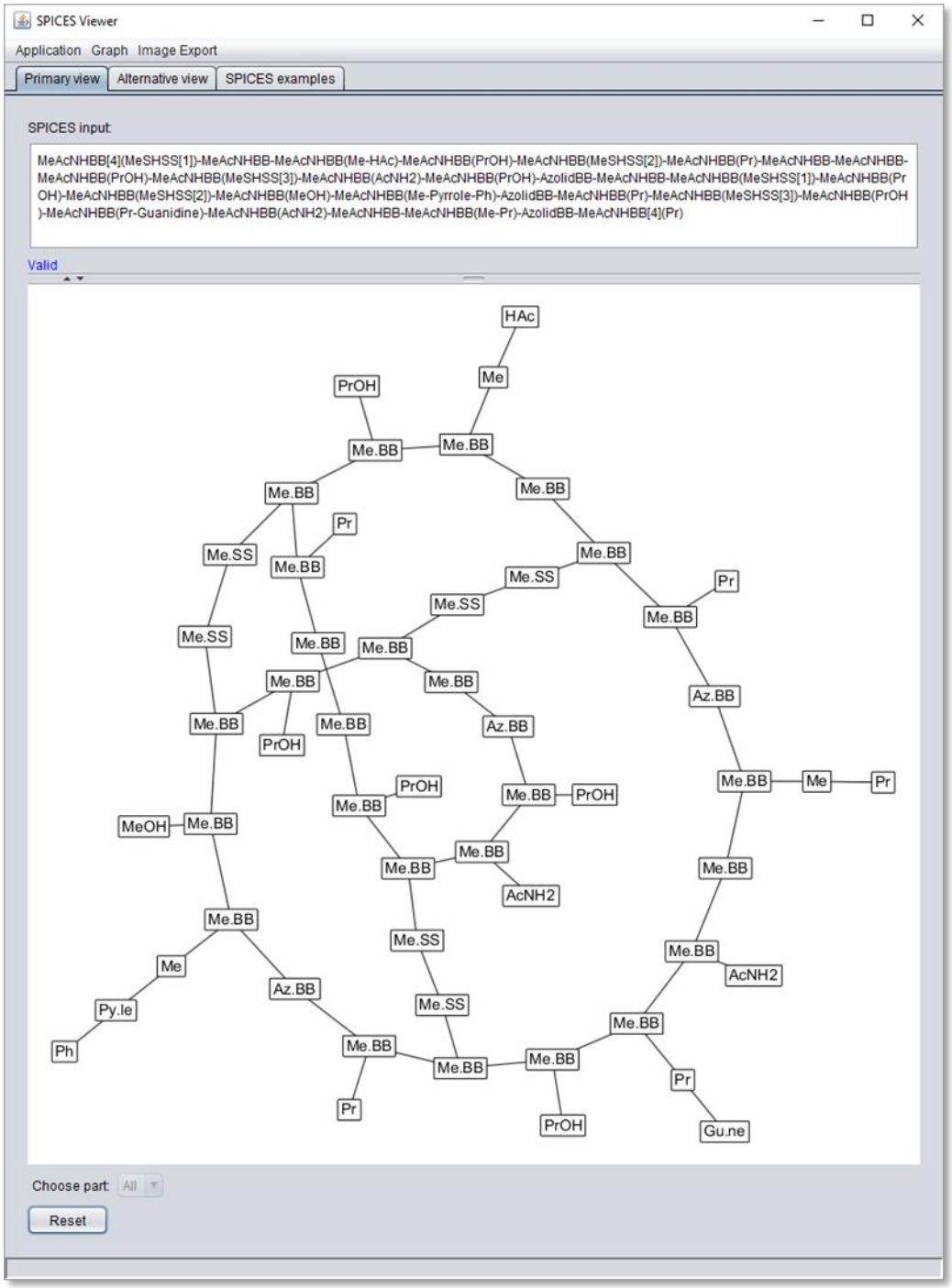


Figure 8. *SpicesViewer* graph display of the cyclotide Kalata B1 (figure 7) [vandenBroek2018].

A graphical visualization may be achieved by adequate application of open-source projects that provide chemical structure drawing capabilities. For instance the structure-diagram layout of the *Chemistry Development Kit* (*CDK*) [Steinbeck2003, Steinbeck2006, Willighagen2017] can be customized to display SPICES instead of atom-based connection topologies [Truszkowski2014]. A principal problem of this (mis)use of atom-based layouts is the inappropriateness of its layout elements and templates: Particle graphs do

not follow common patterns of atomic connections (see figure 7) so that topological visualizations may result in incomprehensible graphs. Thus a more general graph visualization approach with e.g. the *GraphStream* library [GraphStream2020] is necessary. In addition this library allows individually tailored changes of the produced graph by manual displacement of node positions to remove unwanted node or edge overlaps. *SpicesViewer.jar* is a GUI application (on top of *Spices.jar* and connection library *SpicesToGraphStream.jar*) for a topological SPICES display with the *GraphStream* library to analyze the influence of different graph settings and to demonstrate computational functions like zooming or graph image generation. Figure 7 shows the *SpicesViewer.jar* GUI with a manually tailored SPICES graph visualization of the cyclic peptide Kalata B1 with 29 amino acids.

5.2 Mesoscopic simulation kernel Jdpd

The new Jdpd library [JdpdRepository2020] enriches the small set of existing commercial [BIOVIA2020, CULGI2018], open general [ESPResSo2020, LAMMPS2020, Gromacs2020, DL_MESO2020, DPDmacs2020, SYMPLER2020] and hardware-specific DPD kernels [USER-MESO2020, LAMMPS-GPU2020]. It is the first pure Java implementation for Molecular Fragment DPD leveraging inherent Java strengths like cross-platform portability, parallelized computations, automatic memory management or object-oriented plus functional programming capabilities, see figure 8. In addition Jdpd may help to tackle problems of polyglot programming where a simulation kernel may be coded in C/C++ or FORTRAN with an intermediate software layer written in a different language (like “universal” Java).

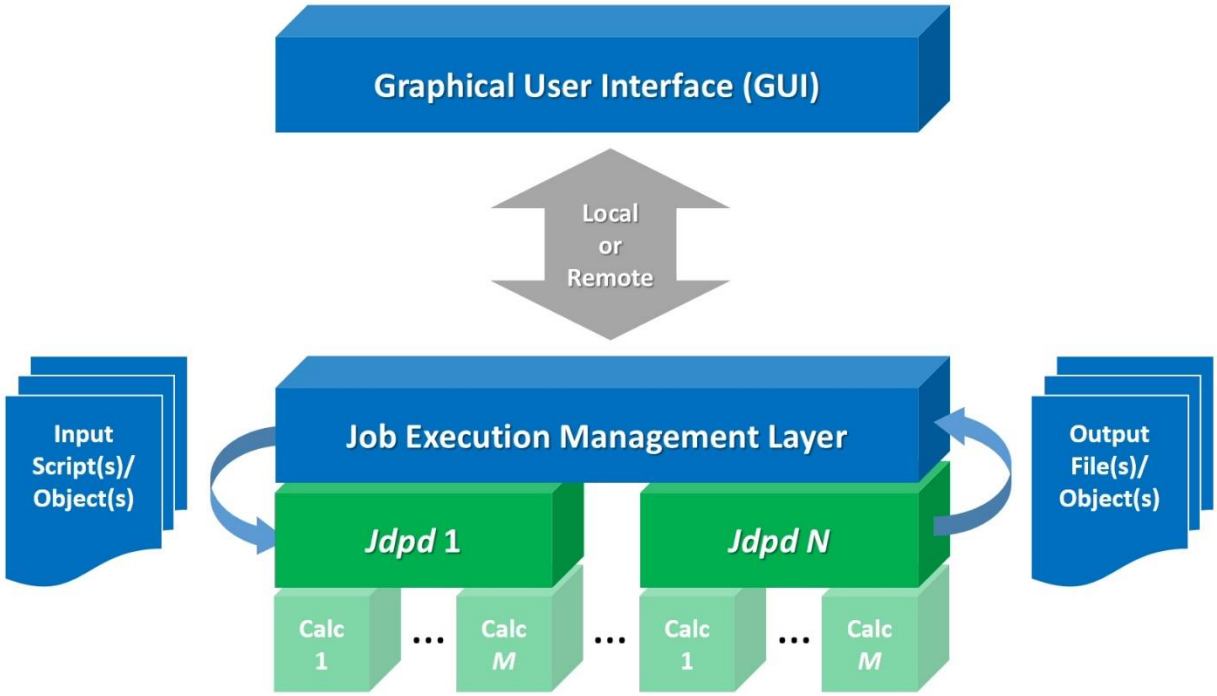


Figure 9. DPD simulation architecture(s) with Jdpd: A job execution management layer may run several parallelized Jdpd instances with different simulation jobs (denoted 1 to *N*) where every Jdpd instance itself may use several parallelized internal calculation threads (denoted Calc 1 to *M*) [vandenBroek2018b].

DPD forces and corresponding potentials are implemented in calculation classes of the packages *harmonicBonds*, *dpdCutoff1* and *electrostatics*. An additional gravitational acceleration that acts on particle masses may be defined for every direction (this function may be interesting for very large mesoscopic simulation systems). All calculation classes for DPD and electrostatics forces and potentials extend abstract class *ParticlePairInteractionCalculator* of package *interactions* which itself implements a cut-off length-based simulation box cell partitioning [Allen1987]: The resulting cell linked-list method allows a (near) linear scaling detection of interacting particle pairs within neighboring cells – a task that otherwise would be quadratically scaling with the number of particles. In addition the simulation box cells are grouped in (3·3·2=)18 parallelizable chunks (each cell is surrounded by 26 neighbor cells, every third cell in each of the three spatial directions has no common neighbor cell but due to symmetric forces every second cell can be used in one of the three directions, see method *getParallelisationSafeCellChunks* of class *CellBox* in package *utilities*; for an in-depth description of this method a far-reaching graphical explanation would be necessary and is not shown in this work, nevertheless it is a known technique in the field of computer simulation): These chunks guarantee a separated non-overlapping access of internal force

array elements from parallelized calculation threads so that fast lock-free array manipulations become possible (all particle-related arrays are located in class *ParticleArrays* of package *parameters*). An analogue lock-free parallelization feature with separated parallelizable bond chunks is realized within the bond-related calculation classes (see package *harmonicBonds*).

Another significant performance improvement is achieved by efficiently caching the already evaluated interacting particle pairs for reuse throughout different force calculations within a single simulation step (see the use of caching class *ParticlePairDistanceParameters* in package *utilities*). This caching avoids the time-consuming recalculation of particle pair distances (see relative performance factors of the different implemented integration schemes below): The saved number of expensively safe-guarded particle pair distance calculations may be (empirically) estimated to about 7 times the total number of particles in simulation for a common DPD particle density of 3. An upper limit for this empirical number may be deduced from the average number of all neighbor-cell particle pairs, which is 40 times the number of particles in the simulation (which overestimates the number of relevant particle pair distances since – incorrectly – particle pairs with a distance above the cut-off length of 1 DPD unit are included):

$$\left\langle N_{ij} \right\rangle_{\rho_{DPD}} = \underbrace{\frac{N}{\rho_{DPD}}}_{\text{Number of cells}} \left(\underbrace{\frac{\rho_{DPD}^2 - \rho_{DPD}}{2}}_{\text{Number of particle pairs within single cell}} + \underbrace{13\rho_{DPD}^2}_{\substack{\text{Particle pairs of single cell} \\ \text{with 13 of its 26 neighbor cells} \\ \text{(no double counts)}}} \right) = \frac{27\rho_{DPD} - 1}{2} N$$

$$\left\langle N_{ij} \right\rangle_3 = 40N$$

N, number of particles; ρ_{DPD}, DPD density; $\left\langle N_{ij} \right\rangle_{\rho_{DPD}}$, average number of particle pairs for specific DPD density.

Since (unlike MD) dissipative DPD forces depend on relative particle velocities, the common Velocity-Verlet (VV) integration of the equations of motion [Allen1987, Frenkel2002] has to be modified (abbreviated MVV). Jdpd consists of optimized implementations of four DPD integration schemes (located in package *integrationType*) that cover different integration techniques from the literature: (I) The original Groot-Warren scheme (GWMVV) [Groot1997, Besold2000, Vattulainen2002] which depends on a tuning parameter where GWMVV equals VV integration for a value of 0.5, (II) the self-consistent scheme (SCMVV) [Besold2000, Vattulainen2002, Pagonabarraga1998] with an adjustable number of self-consistent dissipative force iterations where a single iteration leads to the DPDMVV variant, (III) Shardlow's S1 scheme (S1MVV) [Shardlow2003, Nikunen2003] and (IV) the Nonsymmetric Pairwise Noose-Hoover-Langevin thermostat (PNHLN) [Leimkuhler2015] that requires the definition of an additional coupling parameter.

A single simulation task is performed by activation (instantiating) a *DpdSimulationTask* (which implements a Callable interface, see package *jdpd*) that may be submitted to an appropriate thread executor service to be invoked. The *DpdSimulationTask* constructor requires six configuration objects: (I) An (optional) *RestartInfo* instance that contains information about a possible simulation job restart (all Jdpd jobs may be restarted with altered settings which allows flexible job execution chains), (II) an input instance that implements the *IInput* interface with all simulation settings (e.g. particle types and their interactions, initial particle positions within the simulation box, additional properties like molecule boundaries etc.), (III) an output instance implementing the *IOutput* interface for all output data, (IV) a progress monitor instance implementing the *IProgressMonitor* interface for real-time simulation progress information, (V) an *ILogger* instance for log-level dependent accumulation of detailed internal calculation progress information and (VI) a *ParallelizationInfo* instance that describes internal settings for parallelized calculations. All interfaces are located in package *interfaces* and possess concrete sample implementations like classes *FileInput* and *FileOutput* of package *samples* that implement file-based I/O methods. The *RestartInfo* and *ParallelizationInfo* classes are found in package *parameters*.

An *IInput* object contains a *Factory* instance (located in package *utilities*) that implements enumerated type definitions for random number generation, DPD and electrostatics calculations, harmonic bonds or integration schemata. The factory pattern allows a simple extension or replacement of computational algorithms used throughout

the whole simulation kernel, e.g. the implemented random numbers generators [ApacheRNG2020, PCG2018, Nishimura2018] may be supplemented by alternative methods with a few additional lines of code in the *Factory* class. An *IInput* object also provides detailed molecular information (see class *MoleculeDescription* in package *utilities*): These molecular descriptions differentiate between topological bonds of a molecule's particles for an adequate description of covalent connectivity and bonds between backbone particles that maintain a defined spatial 3D structure, e.g. ring systems or secondary and tertiary structures of proteins. As a matter of fact, both bond types are treated equally throughout calculations.

For software programs with extensive parallelized numerical calculations, the debugging facilities of current integrated development environments are often not sufficient to tackle subtle errors. Thus Jdpd comprises (extensible) loggers (see classes *FileLogger* and *MemoryLogger* in package *logger*) with different (extensible) log-levels to obtain detailed information about the simulation progress.

Jdpd efficiently calculates kinetic and potential energies, average kinetic temperatures [Frenkel2002], surface tensions along the simulation box axes [Frenkel2002, Walton1983] or particle-based radii of gyration [Flory1953]. In addition a fast parallelized particle- and molecule-based nearest-neighbor analysis for the whole simulation process is implemented which allows a detailed temporal monitoring of changing averaged particle and molecule vicinities. Initial force steps for potential energy minimization to improve start geometries may be defined and velocities may be scaled to keep a desired temperature. A molecule-specific position or velocity fixation is provided with the additional option to define molecule-specific boundaries in form of reflective virtual walls within the simulation box to confine molecules to a desired subspace. Molecule-specific periodic "force kicks" may be defined to "smoothly drive" molecules into a desired direction. Periodic boundary conditions as well as reflective box walls are supported. Implemented safeguards try to tackle common problems like unfavorable simulation box start configurations with improbable (high or low) particle densities.

For testing purposes Jdpd contains two appropriate packages *tests* and *tests.interactions* that contain a Unit test code e.g. for a command file driven Jdpd usage.

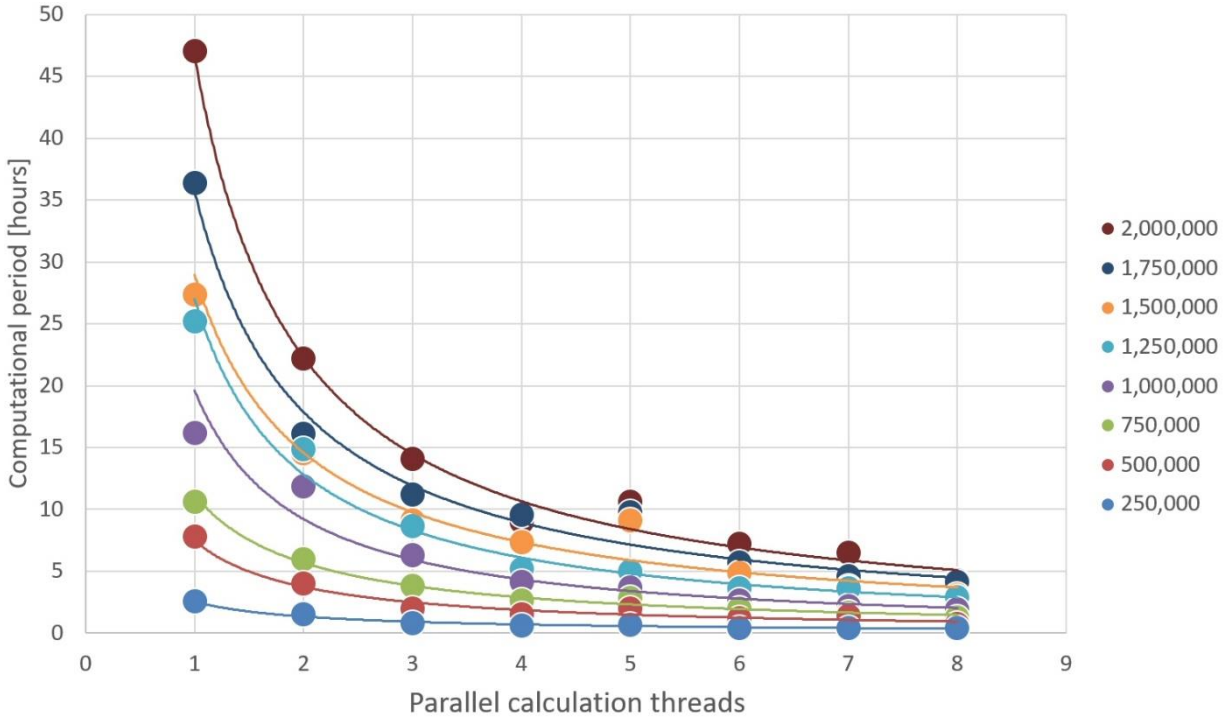


Figure 10. Jdpd performance snapshots in dependence of the number of simulated particles (see legend on the right) and the number of parallelized calculation threads for 13,000 simulation steps (corresponding to a physical time period of one microsecond) with the integration type after Groot-Warren (GWMVV) [vandenBroek2018b].

Jdpd offers a satisfactory performance for various scientific applications with workstation computers that are commonly available in scientific institutions (where the computational speed of Java just-in-time compiled Jdpd is in the order of comparable ahead-of-time compiled FORTRAN or C/C++ code) but does not explicitly exploit specific hardware environments or devices like graphics processing units (thus Jdpd may be orders of magnitude slower than hardware-specific high performance computing implementations – but future Java Virtual Machines may address these hardware acceleration options [Ishizaki2015, Sumatra2018]): As a rule of thumb, a one-million-particle/one-microsecond simulation using 8 parallelized calculation threads consumes about one gigabyte of memory and takes less than 3 hours to finish on standard multicore processors, e.g. an Intel Xeon E5-2697 v2 CPU [XeonE5-2697-2018] used for the performance snapshots shown in figure 9. Up to the studied 8 parallelized calculation threads, Jdpd computational periods scale inversely proportional with the thread number exhibiting scaling exponents between -0.92 and -1.09. The corresponding scaling with a growing simulated particle number is near-proportional with scaling exponents between 1.17 and 1.39 (thus well below quadratic scaling). The relative performance factors of the

different implemented integration schemes (see above) is GWMVV (1.0) < S1MVV (1.1) < DPDMVV (1.4) < S1MVV without cache (1.8) < PNHLN (1.9) < DPDMVV without cache (2.2) < SCMVV with 5 iterations (2.3) < PNHLN without cache (2.8) < SCMVV with 5 iterations without cache (4.8). These findings correspond to those reported in [Besold2000, Vattulainen2002, Nikunen2003, Leimkuhler2015].

5.3 Mesoscopic simulation environment MFsim

The new MFsim project provides the first open Java all-in-one rich-client mesoscopic simulation environment and complements the few available commercial systems [BIOVIA2020, CULGI2020] with specific support for biomolecular applications containing peptides and proteins. By default, MFsim is integrated with the Jdpd simulation kernel, an open MFD Java code [vandenBroek2018b, Jdpd2020]. As an open approach, MFsim is not restricted to a specific mesoscopic simulation engine but may be customized to communicate with any particle-based simulation code [ESPResSo2020, LAMMPS2020, Gromacs2020, DL_MESO2020, DPDmacs2020, SYMPLER2020, USER-MESO2020, LAMMPS-GPU2020]: Appendix 4 outlines the corresponding simulation kernel integration details.

Usage of MFsim does not require programming skills and supports the complete preparation-simulation-evaluation triad of a mesoscopic simulation task. It comprises features like a SPICES line notation [vandenBroek2018, SPICES2020] based chemical structure editor, a PDB file parser [ProteinDataBank2020] for particle-based peptide/protein representations, a peptide and protein editor, support of polymer definitions, a compartment editor for complex molecular start configurations and interactive simulation box views including analytics, on-the-fly movie generation and animated diagrams. Parameter settings are supported by reusable schemata and filtered bulk operations, inter-parameter dependencies are controlled by directed internal update cascades to avoid ill-defined settings. MFsim parallelizes operations to effectively exploit multi-core processor hardware.

5.3.1 Implementation

The object-oriented Java architecture (see figure 10) follows a Model-View-Controller (MVC) pattern [Reenskaug2020]: Graphical user interface (GUI) view classes (based on the Swing GUI Toolkit [Swing2020] of the Java platform [Java2020]) are governed by corresponding controller classes (*gui* packages *control*, *dialog* and *main*). The controllers communicate with a layer of Molecular Fragment Cheminformatics (MFC) models

(*model* packages *changeNotification*, *graphics*, *jmolViewer*, *job*, *message*, *particle*, *particleStructure*, *peptide*, *preference*, *util* and *valueItem*) that provide all core functions where the MFC layer itself controls the particle simulation kernel (Jdpd as a default).

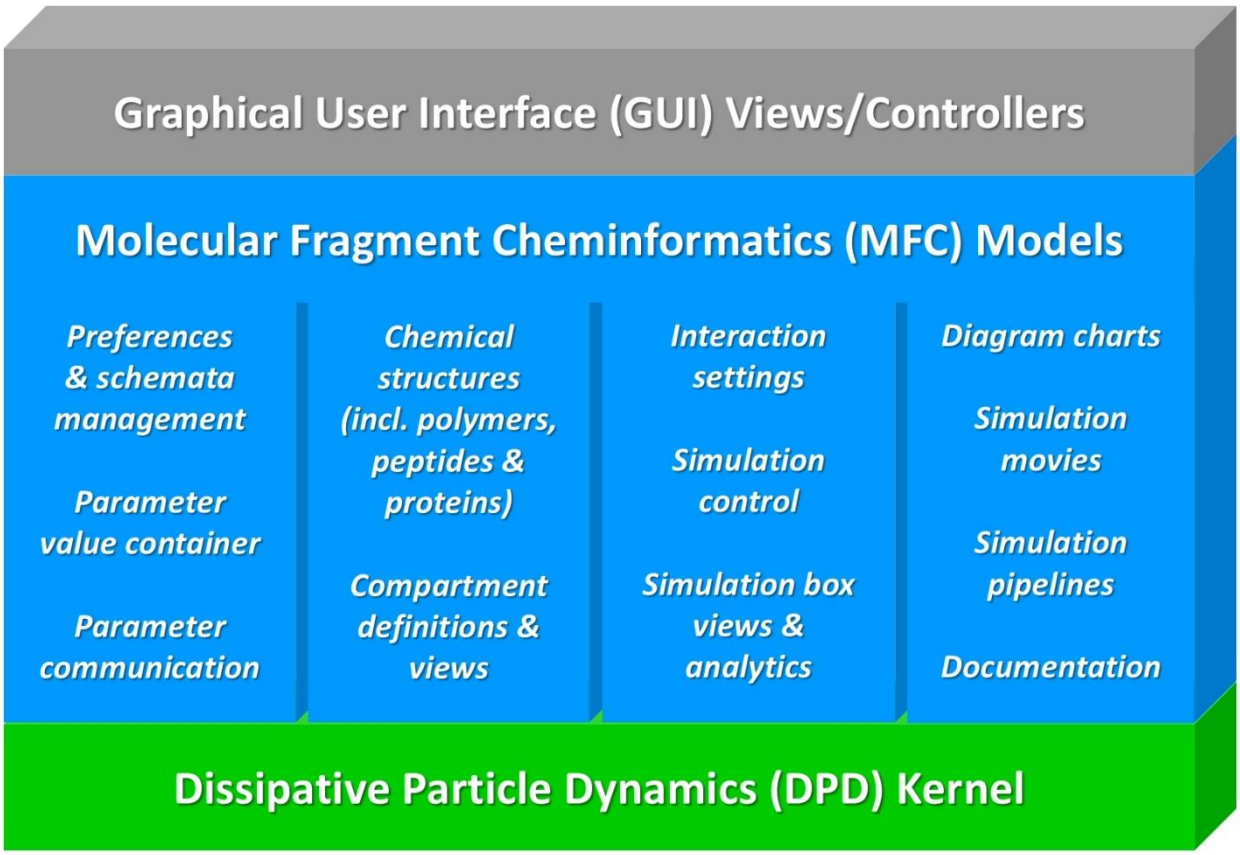


Figure 11. MFsim architecture with underlying MVC pattern: Graphical user interface (GUI) views/controllers communicate with a layer of Molecular Fragment Cheminformatics (MFC) models that provide all core functions where the MFC layer itself controls the DPD simulation kernel layer [vandenBroek2020].

The basic GUI frame is organized by three management tabs for simulation job *Design*, job *Execution* and job *Results* evaluation that operate on a defined workspace directory of the file system (see figure 11): This workspace directory contains a *JobInputs* folder (where each subfolder corresponds to a single *Job Input* definition) and a *JobResults* folder (with subfolders corresponding to *Job Result* instances). During job execution a temporary directory is used for file storage which may be located on a fast hardware/memory device for maximum performance. All specific GUI functions are provided by successive levels of modal dialogs that can be opened above the basic GUI frame.

MFsim automatically logs internal problems to support the detection of possibly subtle errors which are likely to occur in complex architectures with hundred thousands of code lines. The log file entries may be viewed via menu entry *Help/MFsim log/Browse* of the basic GUI frame.

Details about internal data objects, preferences and convenient re-usable settings are comprised in appendix 3.

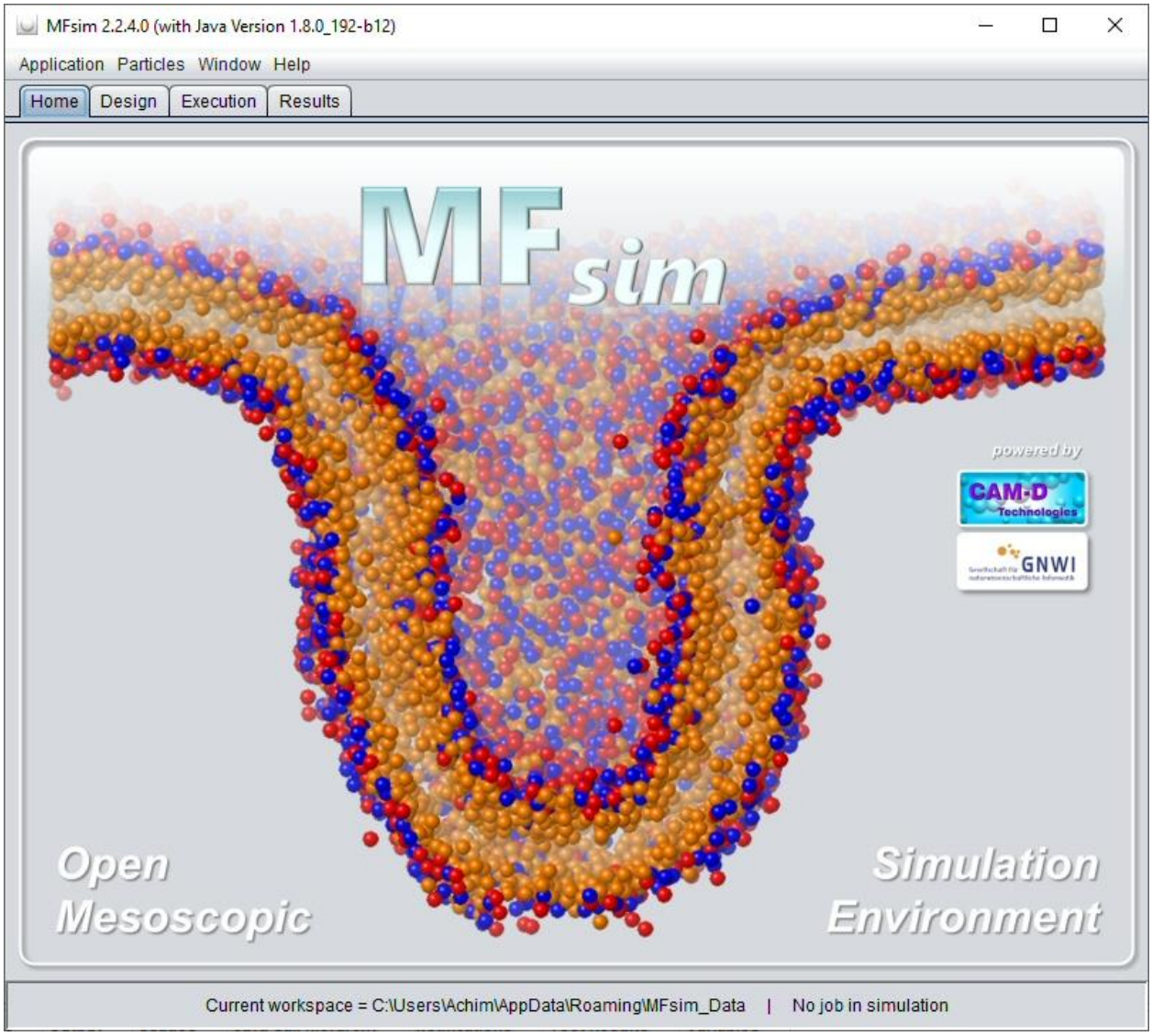


Figure 12. Basic MFsim GUI frame with three management tabs for simulation job *Design*, job *Execution* and job *Results* evaluation operating on a defined workspace directory (see status line at the bottom). The displayed *Home* tab depicts graphical project information and links to the MFsim GitHub repository [vandenBroek2020].

MFsim supports concurrent calculations to exploit the capabilities of multi-core processors for performance improvements that (almost) linearly scale with the number of available processor cores (and additionally benefit from technologies like Hyper-Threading). To arrive at an overall optimum balance, the global preference section *Parallel computing* provides options for fine-tuning of concurrent operations. As an example the number of parallel calculation threads for force calculations of the (default Jdpd) particle simulation engine may be specified in combination with the number of concurrent jobs in simulation to achieve an adequate workload of an octa-core processor with 2 concurrent simulations and 4 force calculation threads each – whereas for a maximum single-job performance the number of force calculation threads could be increased to 8 with the number of parallel simulations decreased to 1. Additional parallel computing options address the number of concurrent particle position writers for output file creation, the minimum number of simulation box cells and particle-particle bonds for parallelization (to avoid ineffective parallelization efforts due to the necessary computational overhead) and the number of concurrent graphical simulation box slice and diagram generators (abbreviated slicers – their number should at best correspond to the number of available processor threads including Hyper-Threading).

Another performance-related feature is global caching: SPICES line notations (class *SpicesPool* of package *model.particleStructure*) and PDB file-related protein definitions (class *PdbToDpdPool* of package *model.peptide*) are kept in memory to speed up chemical structure-related operations by avoidance of extensive reevaluations during calculations. The size of the SPICES and protein cache may be inspected via menu entry *Application/Cache/Show* of the basic GUI frame. In-memory caching also improves the performance of simulation box slice image-related operations: *Preference/Simulation box/Slicer graphics/Image storage* in the global preferences dialog may be used to activate the *Memory uncompressed* (fastest) or *compressed* (smallest) image cache, otherwise slice images are stored as image files which is comparatively slow.

MFsim slicer graphics is realized with Java2D [Java2D2020]. 3D displays of the simulation box are parallel projections that allow for versatile through-space measurements (but appear somewhat distorted compared to central projections). The spatial 3D impression is additionally supported by a configurable fog generation (spatially deeper lying particles are presented in an attenuated color) to intensify the depth perception. Simulation movie generation is realized by merging box images into a movie clip utilizing the open FFmpeg software [FFmpeg2020].

Since there is no unique particle set for mesoscopic simulation, a specific field of research requires a specific particle set – a situation that is similar to Molecular Mechanics/Dynamics

with different force fields and specific atom types. All mandatory particle information and particle-particle interactions must be provided by a particle set text file where particle definitions are to be described in mandatory section *[Particle description]*, particle-particle repulsions in mandatory section *[Particle interactions]* and peptide/protein decomposition related particle information in optional section *[Amino acids]*. MFsim comes with *ParticleSet_H2O.txt*, a minimal single-particle set for test purposes, and *ParticleSet_AA_V02.txt*, a basic biomolecular particle set for an approximate fragmentation of phospholipids and peptide/proteins. The latter is based on the particle data in [LAMMPS-GPU2020] with rescaled molecular volumes and the water molecule being the smallest particle of volume 30 $Å^3$. MFsim contains some functions for particle set manipulation (e.g. particle duplication, see menu *Particles* of the basic GUI frame) and supports the automated update of a *Job Input* definition for a new particle set.

Last but not least – as an open project itself, MFsim uses several other open libraries: Apache Commons IO [ApacheIO2020]/Lang [ApacheLang2020]/RNG [ApacheRNG2020], BioJava [Prlic2012, BioJava2020], FFmpeg (currently with the Windows OS only), GraphStream [GraphStream2020], Jama [JAMA2020], JCommon [JCommon2020], Jdpd, JDOM [JDOM2020], JFreeChart [JFreeChart2020], Jmol [Jmol2020], PCG [PCG-Java2020, PCG-RNG2020], SPICES [SPICESRepository2020] and Vecmath [3DVecMath2020].

The MFsim simulation system aims at supporting the complete preparation-simulation-evaluation triad in form of an integrated all-in-one workflow which is sketched step-by-step in the following paragraphs.

5.3.2 Simulation job design

The simulation *Design* tab of the basic GUI frame manages all *Job Input*-related operations. It comprises an optionally filtered list of all available *Job Input* definitions of the specified workspace where a single *Job Input* definition may be viewed, edited, re-used (as a start for a new definition), removed, imported from an archive file or itself archived to a file (e.g. for exchange purposes). For a new *Job Input* definition, a corresponding modal dialog is opened above the basic GUI frame.

A new *Job Input* definition is already a complete and valid simulation job definition with a single particle type (the water particle). The *Job Input* features have to be defined in a top-down manner where a change in a higher-level feature leads to an immediate update of subordinate features in an adequate manner – e.g. a change in the number of

particles via the *Quantity* feature automatically changes the subordinate *Box size* feature to be consistent with the new higher-level *Quantity* settings according to the even higher *DPD density* setting. These successive top-down updates (which are realized by directed *ValueItem* update cascades, see class *UtilityJobUpdate* in package *model.job* as well as appendix 3) alleviate error-free *Job Input* definitions with an overall logical integrity. A complete *Job Input* definition is organized in form of a feature tree that consists of four main sections (see figure 12): The *General job description*, the *Chemical system description*, the *Interaction description* and the *Simulation description*. Every single *Job Input* feature contains a local *Description*, a *Hint* and an *Error* tab: The *Description* tab provides a descriptive outline of the feature in question and summarizes its possible settings. The *Hint* tab informs about possible shortcomings of the current setting (e.g. possibly unwanted identical colors for different molecules so that the molecules cannot be distinguished in a simulation box display) whereas an activated *Error* tab signals a severe problem which has to be resolved (an erroneous *Job Input* definition is not allowed to be executed). Since the list of *Job Input* features should exploit the range of capabilities of the underlying particle simulation kernel the current MFsim *Job Input* feature set especially addresses the default Jdpd simulation kernel (see class *JdpdValueItemDefinition* in package *model.job*). For other particle simulation kernels the feature set would have to be customized accordingly (see appendix 4).

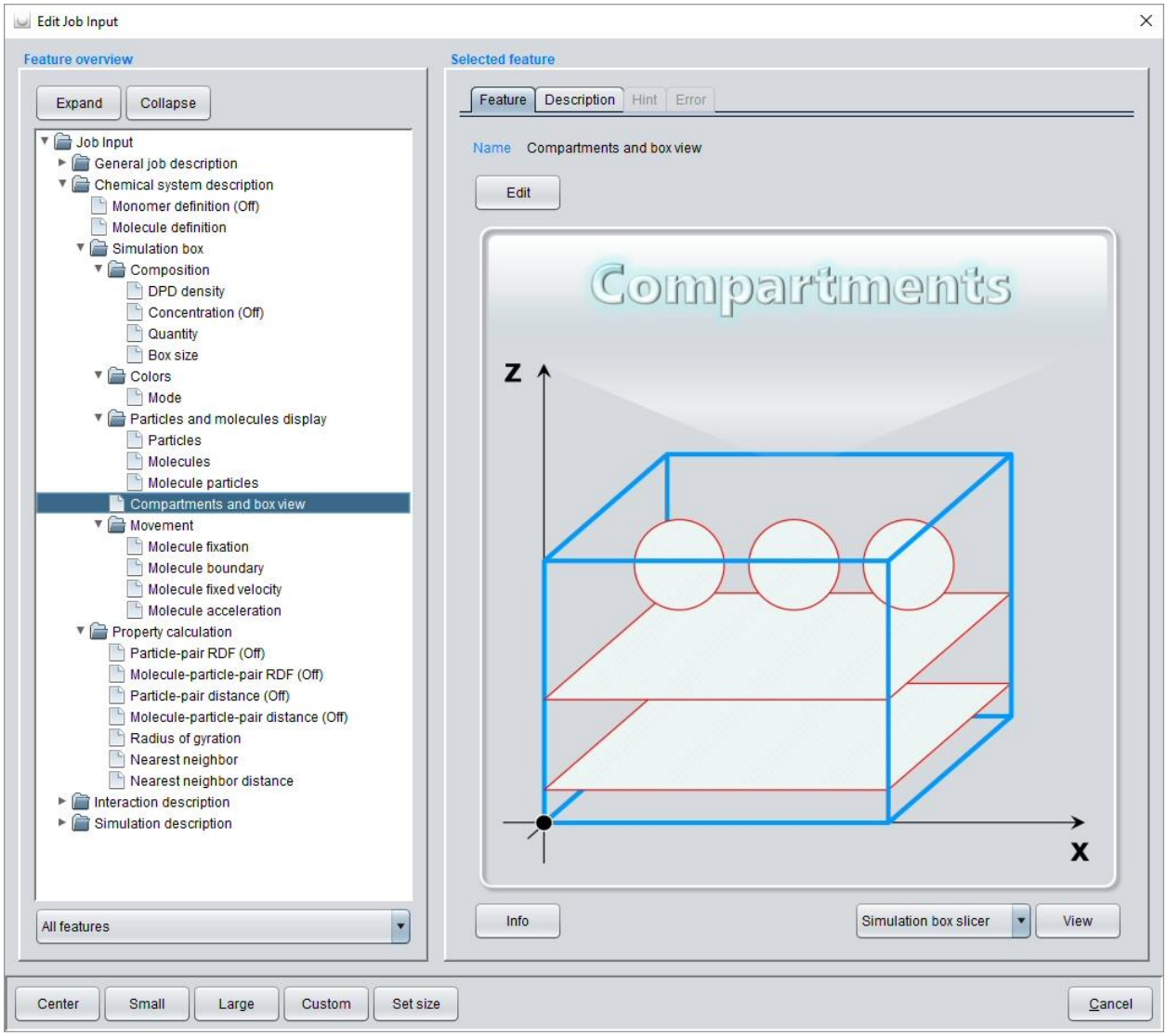


Figure 13. Modal dialog for a Job Input definition with main sections General job description (collapsed), Chemical system description (expanded), Interaction description (collapsed) and Simulation description (collapsed). Feature Compartments and box view of subsection Simulation box in main section Chemical system description is selected [vandenBroek2020].

The *General job description* comprises job-related information like a *Description* line or a creation *Timestamp*.

The *Chemical system description* addresses all features that characterize the chemical ensemble to be simulated. *Monomer definition* and *Molecule definition* are realized with chemical structure editors plus specific editors for peptides and proteins. The editor for monomers and molecular structures is based on the SPICES and GraphStream libraries [vandenBroek2018] and supports the input of a SPICES line notation. A SPICES string can be manually coded (or composed by clicking on available structure elements like particles, brackets or tags) and is immediately parsed with its manipulative topological structure displayed below (for a valid definition) – thus the input of a forbidden SPICES

line notation is (hopefully) impossible. If a particle set with amino acids definition is chosen the *PDB structure* tab and the *Peptide* button are activated. The additional modal peptide dialog allows for the input of a one-letter-code peptide sequence (manually or by clicking on the available amino acids and structure elements like disulfide bridges, charges etc.) that is afterwards converted to a SPICES line notation in the structure editor (see figure 13).

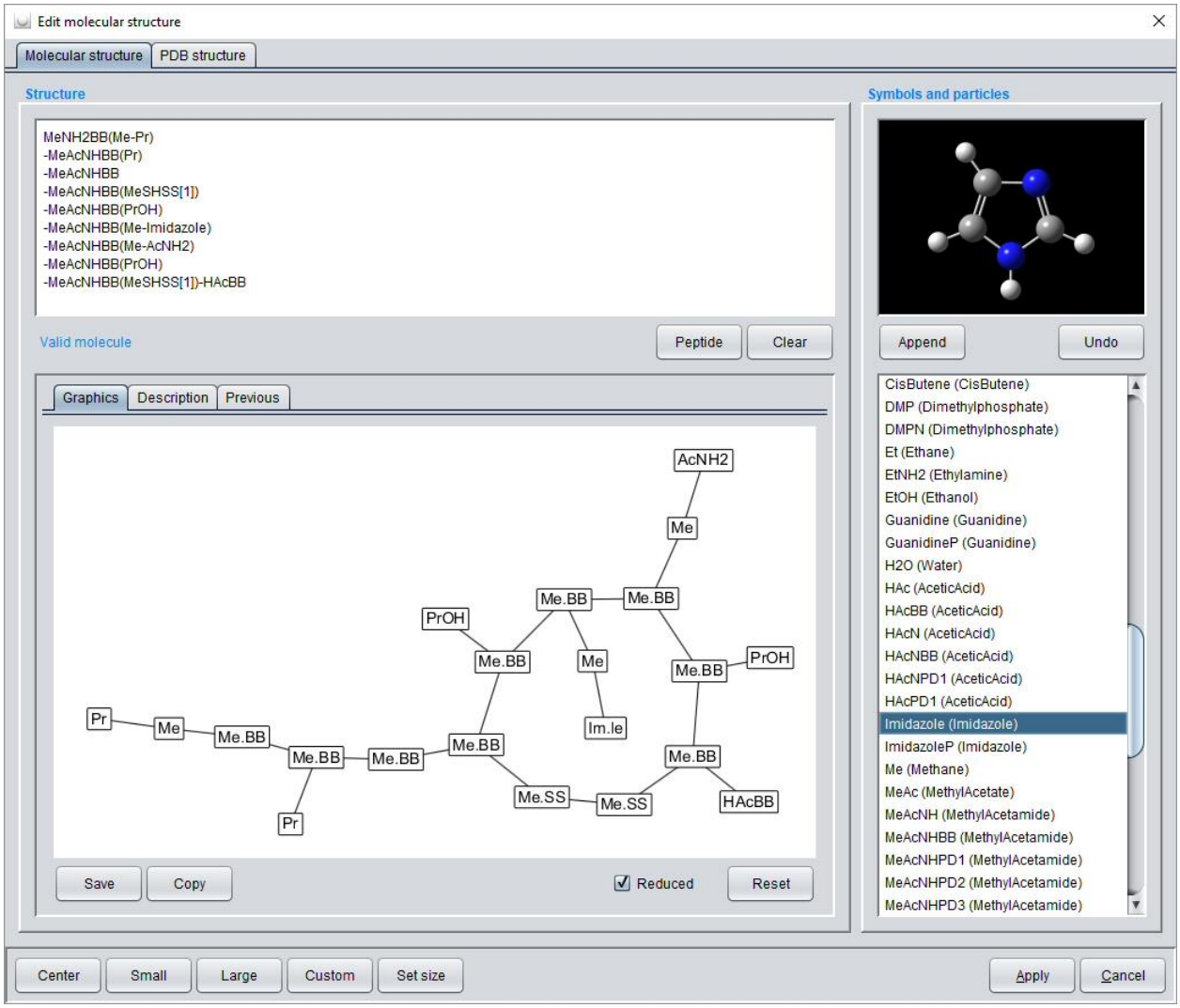


Figure 14. *Molecular structure* tab of the modal structure editor dialog: The SPICES line notation (upper left) of a cyclic peptide was prepared with the modal peptide dialog (available via the active *Peptide* button) thus the line notation is automatically split into several lines for each amino acid to alleviate comprehension. The notation is valid (confirmed by the *Valid molecule* label) and its cyclic topological structure is displayed below. Available structure elements (particles, brackets, tags etc.) are listed on the right [vandenBroek2020].

For biomolecular simulations MFsim provides a PDB-to-SPICES parser which is available via the *PDB structure* tab (see figure 14). An opened PDB file is rendered with the Jmol library and internally evaluated with the BioJava library to create a corresponding SPICES line notation. Different biological assemblies may be chosen from the PDB definition via a drop-down box. The modal *Chain* dialog may be used to exclude protein chains and to a priori assign a segment to an amino acid according to its protein chain. The modal *Mutant* dialog allows the specific change of amino acids to create in-silico mutants (with the same 3D structure of the backbone chain). A pH value may be set with the modal *pH* dialog to charge corresponding particles. The modal *Property* dialog offers a number of settings for the amino acids of all

chains: Protein backbone particles may be replaced by model (probe) particles with identical physical properties but different labels for e.g. selective display of specific amino acids of a chain. In addition, a status and a segment may be defined for each amino acid backbone particle which can be used for intra-protein force assignments in order to achieve specific control concerning the flexibility or stiffness of the protein backbone (whereas amino acid side chain particles are always flexible). The complete protein information is encapsulated in a *PdbToDpd* object (package *model.peptide*) which is globally stored in the protein cache for inexpensive reuse in protein-related calculations. A *PdbToDpd* object also comprises an automated mapping of the 3D protein structure to the relative positions of its DPD backbone particles which supports realistic 3D protein start geometries in the simulation box.

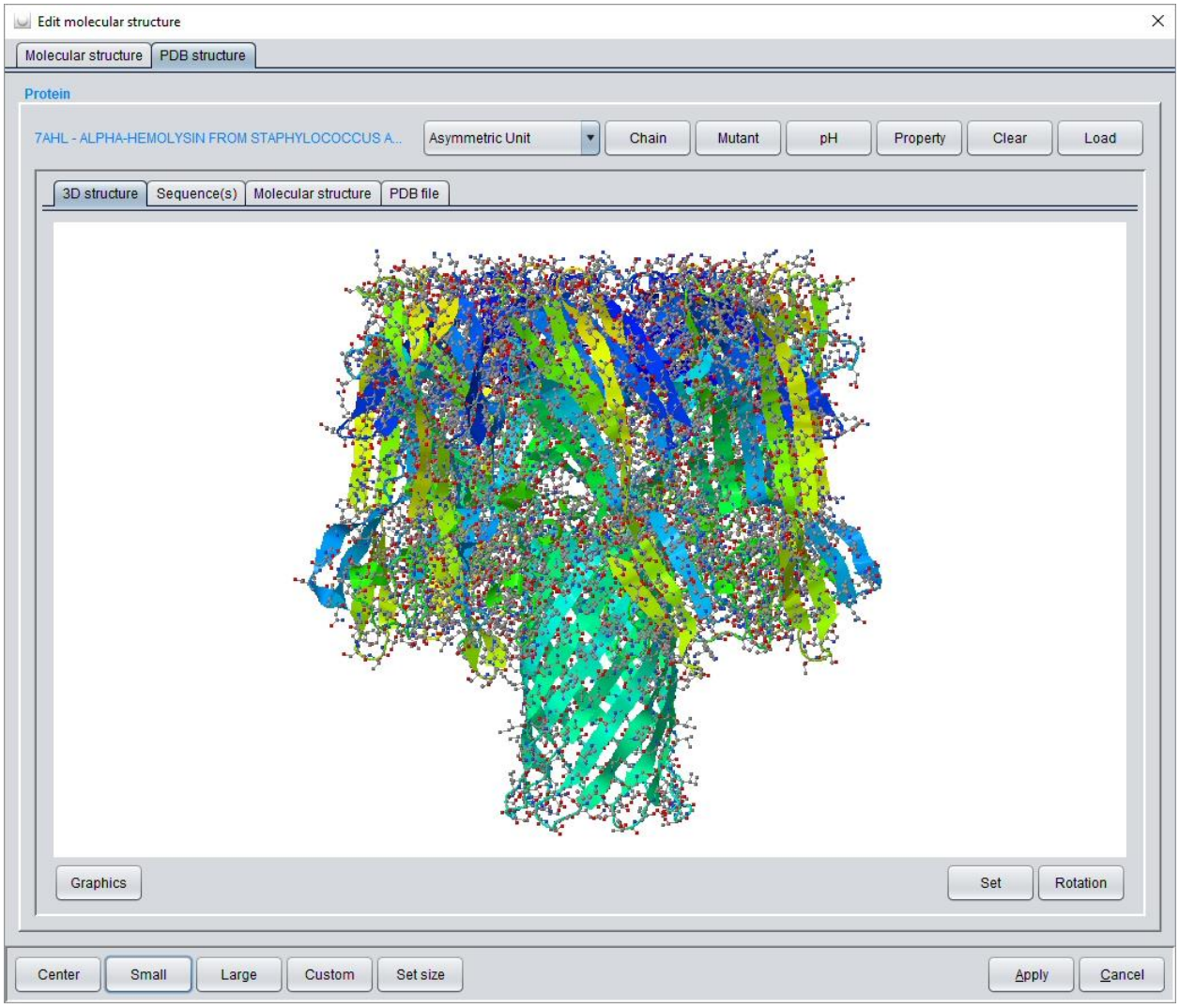


Figure 15. *PDB structure* tab of the modal structure editor dialog: A PDB file can be loaded with automated conversion to a SPICES line notation. Different tabs show the rendered protein structure, its amino acid sequence, the corresponding SPICES line notation and the text of the PDB file. Several additional functions and dialogs are available at the top (see text) [vandenBroek2020].

The SPICES-related polymer construction exploits the syntax characteristics of the SPICES definition [vandenBroek2018] and is realized by an interplay of the SPICES monomer editor (feature *Monomer definition*) and the SPICES structure editor (feature *Molecule definition*) for polymer construction from monomers.

The *Simulation box* subsection of the *Chemical system description* (see figure 12 for the expanded feature tree) provides detailed setting options for the molecular ensemble. The *Composition* sub-subsection allows the definition of the number of molecules in the box where *Concentration* settings in *gram*, *mol*, *mol-percent* or *weight-percent* are also possible. Feature *Box size* may modify the box geometry with a cube as a default. The *Colors* sub-subsection defines different color display modes with *Molecule-particle color* as the most fine-grained setting where an individual color may be chosen for each particle type within a molecule. The *Particles and molecules display* sub-subsection then allows for the corresponding individual default display settings for all molecules/particles (which may be arbitrarily changed at a later stage). The generated default molecular simulation box configuration consists of randomly-oriented spatial SPICES 3D tubes (described in [vandenBroek2018]) at random box positions (see figure 15 which also describes the treatment of PDB-derived 3D peptide/protein structures). The *Compartments and box view* feature (selected in figure 12) may be used to set up specific compartments with specific molecule orientations and to view the resulting start configuration of the simulation box. The compartment editor enables various settings as a combination of sphere, layer or rectangular cuboid compartments where different molecular orientations may be defined within each compartment (see figure 16). Layers with exclusively single particle molecules allow for a simple cubic lattice particle positioning which may generate a solid surface in combination with the *Molecule fixation* function (see below). The editor also supports a correct a priori DPD density of particles inside a compartment (which may be arbitrarily changed).

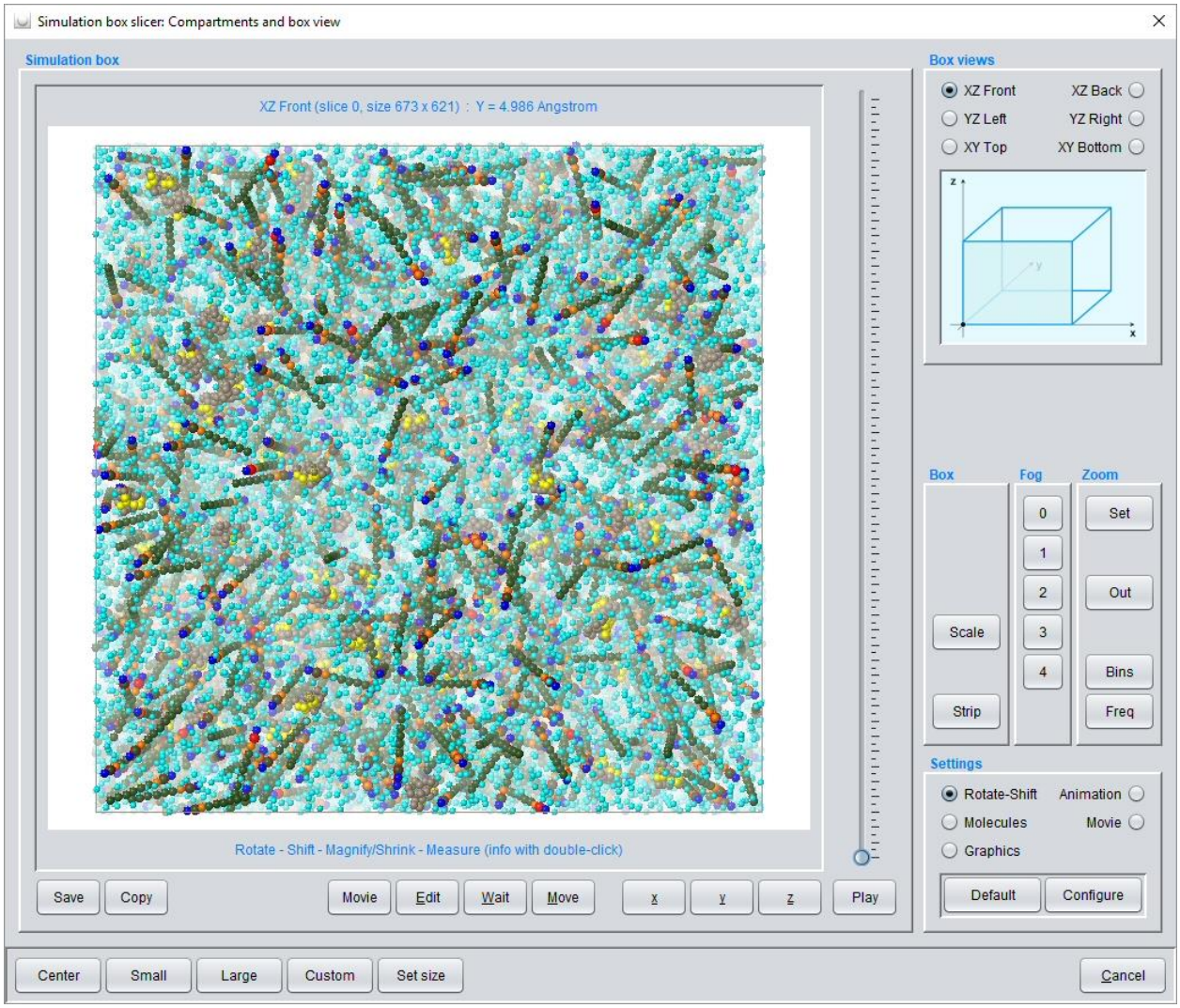
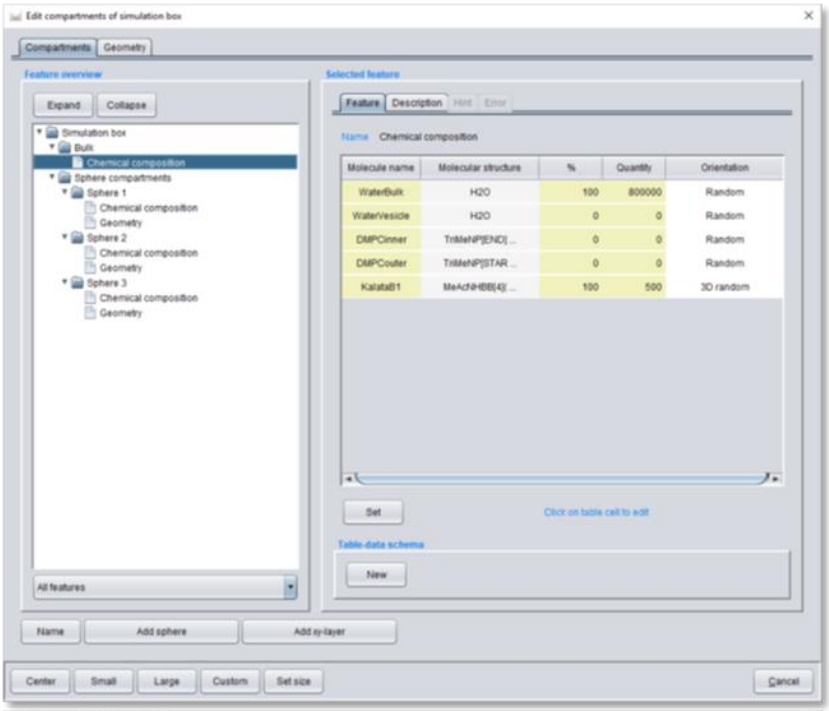
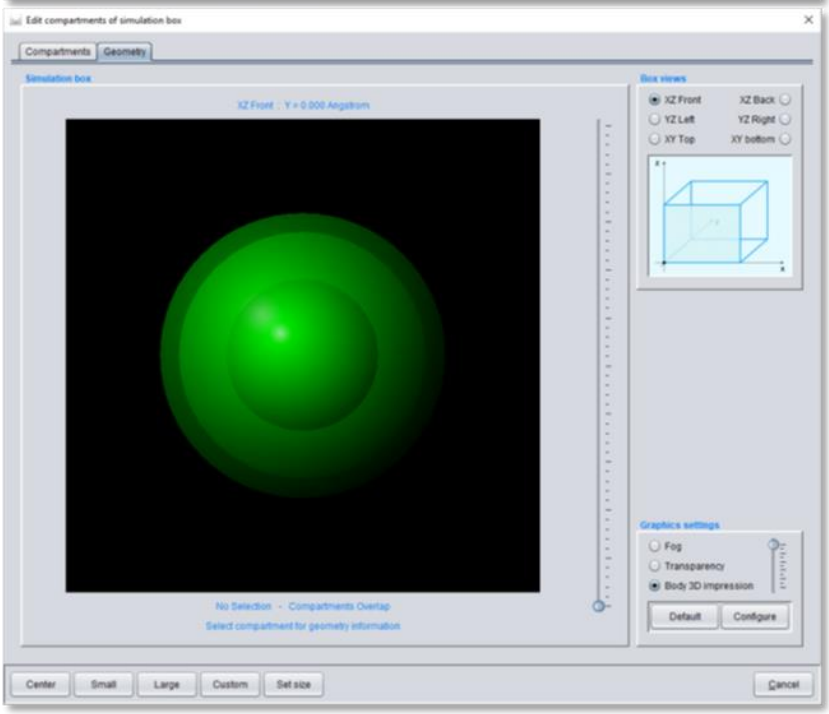


Figure 16. Default random simulation box start configuration (looking at the xz-cross-section at the position y=4.986 Å) without compartment definition (compare to figure 16) of a chemical ensemble that comprises 800,000 (bulk) water particles (excluded from display), 40,000 water particles (to be located inside a vesicle, colored cyan), 2,200 (oriented outer-vesicle) DMPC molecules (with SPICES line notation: TriMeNP[START]-DMPN(MeAc-6Et)(MeAc-6Et[END]) [vandenBroek2018]), 1,800 (oriented inner-vesicle) DMPC molecules (with SPICES line notation: TriMeNP[END]-DMPN(MeAc-6Et)(MeAc-6Et[START]) [vandenBroek2018]) and 500 Kalata B1 cyclotides [vandenBroek2020]. All molecules – except the Kalata B1 cyclotides which are derived from PDB file 1NB1 [KalataB12020] – are generated as randomly-oriented spatial SPICES 3D tubes at random positions in the box. The 500 Kalata B1 cyclotides are also randomly distributed throughout the box but their individual amino acid backbone particle positions are set according to their spatial 3D structure (with the side chain particles collapsed onto their neighboring backbone particles). In addition, each Kalata B1 cyclotide is overall shrunk into a virtual sphere (with a volume that corresponds to its particles' DPD volume) from which all other particles are excluded – thus a Kalata B1 cyclotide start size is somewhat smaller than its actual size during simulation. DMPC particle colors: DMPN (red), Et (olive), MeAc (orange), TriMeNP (blue). Kalata B1 backbone particles are shown in beige (all amino acid side chain particles are excluded from display) with the backbone particles that correspond to the characteristic hydrophobic spot of these peptides displayed in yellow.

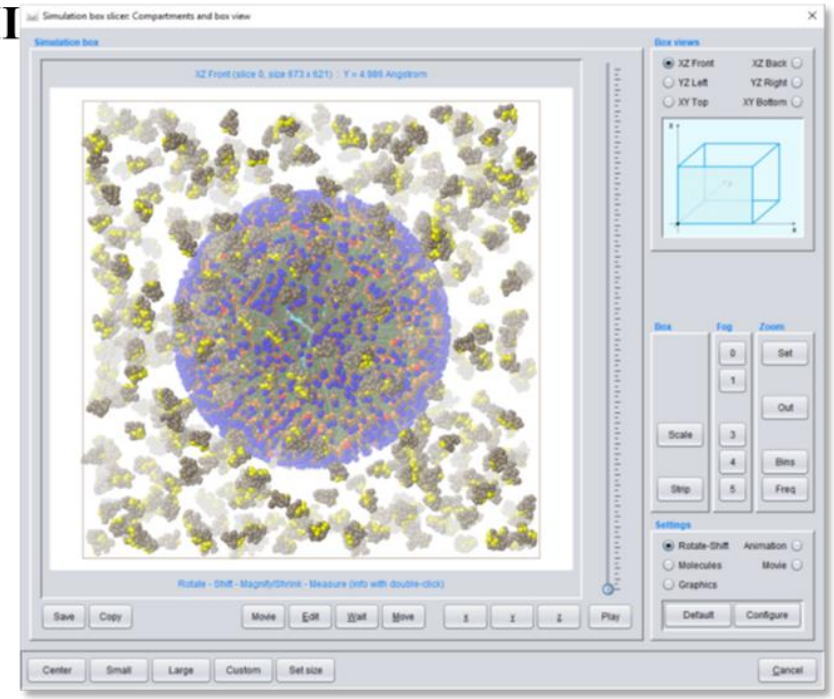
I


Edit compartments of simulation box
Compartments
Geometry
Feature overview
Expand
Collapse
Simulation box
Bulk
Chemical composition
Sphere compartments
Sphere 1
Chemical composition
Geometry
Sphere 2
Chemical composition
Geometry
Sphere 3
Chemical composition
Geometry
All features
Selected feature
Feature
Description
Hint
Error
Name
Chemical composition
Molecule name
Molecular structure
%
Quantity
Orientation
WaterBulk
H2O
100
800000
Random
WaterVesicle
H2O
0
0
Random
DMPCinner
TriMeNP[END]...
0
0
Random
DMPCouter
TriMeNP[STAR...
0
0
Random
KalataB1
MeAcNHBB[4]:...
100
500
3D random
Set
Click on table cell to edit
Table-data schema
New
Name
Add sphere
Add xy-layer
Center
Small
Large
Custom
Set size
Cancel

II


Edit compartments of simulation box
Compartments
Geometry
Simulation box
XZ Front : Y = 0.000 Angstrom
No Selection - Compartments Overlap
Select compartment for geometry information
Box views
XZ Front
XZ Back
YZ Left
YZ Right
XY Top
XY bottom
Graphics settings
Fog
Transparency
Body 3D impression
Default
Configure
Center
Small
Large
Custom
Set size
Cancel

III



IV

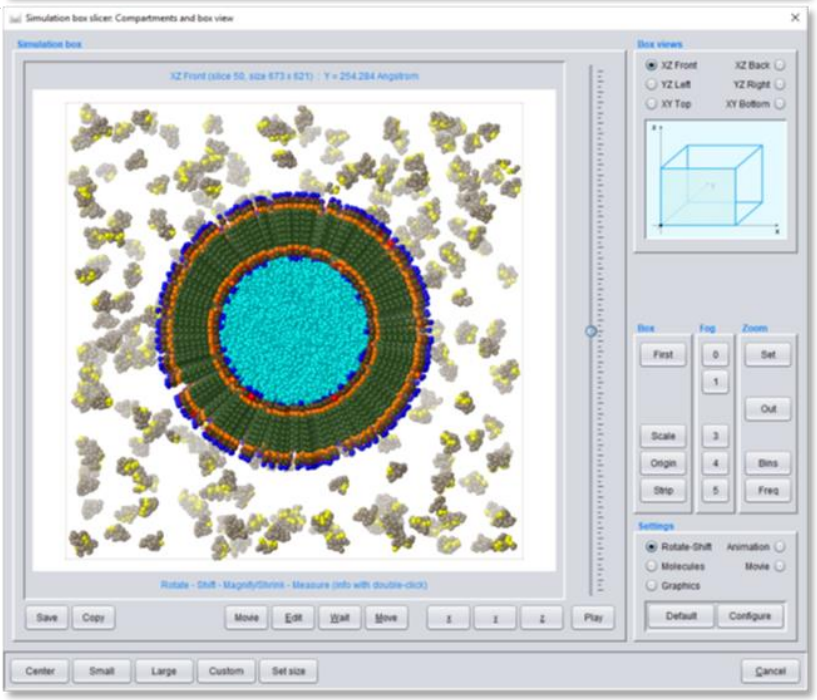


Figure 17. Definition of compartments (a tutorial for other compartments can be found in appendix V) for a 30 nm DMPC phospholipid double-layer vesicle and view of the resulting simulation box start configuration of the chemical ensemble described in figure 15 [vandenBroek2020]. I: Modal compartment editor dialog with detailed settings of the bulk phase (with all Kalata B1 cyclotides and the 800,000 bulk water particles) outside any compartment and the 3 sphere compartments for oriented spatial positioning of the inner/outer DMPC phospholipids and the inner vesicle water particles. II: Graphical display of size and spatial position of the 3 overlaid sphere compartments. III: Corresponding *Simulation box slicer* view of the simulation box start configuration. IV: Cross-section through the simulation box for a more detailed view of the DMPC molecule orientation inside the vesicle.

The settings of the final *Movement* sub-subsection of the *Simulation box* subsection address molecule movement control capabilities of the Jdpd simulation kernel: Molecules may be spatially fixed (*Molecule fixation*) or their movement restricted to virtual cages surrounded by "reflective walls" within the simulation box (*Molecule boundary*). On the other hand, molecules may possess a fixed velocity (*Molecule fixed velocity*) or may be kicked with a specified frequency (*Molecule acceleration*).

The final *Property calculation* subsection of the *Chemical system description* offers several calculation options during simulations like particle-pair radial distribution functions (RDF) and distances, particle-based molecular radii of gyration or a detailed particle/molecule nearest-neighbor analysis to monitor changing particle/molecule vicinities with temporal evolution of the simulated ensemble.

Interaction description as the third main *Job Input* section addresses the fundamental physics of the DPD particle-particle interactions like the *Temperature* setting, *Random DPD force magnitude*, electrostatic interactions of charged particles, gravitational acceleration or the characteristics of bonds between particles and their neighbor particles within molecules. Since mesoscopic DPD simulations are based on isotropic particle-particle repulsions (editable via feature *Particle interactions* with the default repulsion parameters taken from the selected particle set) it may be necessary to impose preferential molecular conformations, e.g. to stabilize the 3D backbone structure of proteins by adequate spring forces between backbone particles to prevent a structural collapse. Thus, additional particle-particle spring forces may be specifically defined between indexed particles in SPICES line notations (*Molecule backbone forces*), the amino acid backbone particles of peptides and proteins (*Protein backbone forces*) and the backbone particles of peptides and proteins within assigned chain segments (*Protein distance forces*).

The final main *Simulation description* section completes the *Job Input* settings with definitions for control of the simulation itself. The number of *Simulation steps* defines the temporal physical period to be simulated by integration of the equations of motion with a defined *Time step length* and a specific *Integration type*. *Output frequency* controls the intermediate output (e.g. the creation of particle position files for specific simulation steps) for later evaluation of the simulation record. *Initial minimization steps* may be defined to improve the initial simulation box start configuration with *Minimization step output* as a control setting for later inspection. Remaining definitions address *Periodic boundaries* along the box axes, the use of a *DPD unit mass*, possible *Initial velocity scaling steps* for temperature control and the choice of the *Random number generator* for the random DPD force.

A completely defined job may be finally saved (*Apply* button of the modal *Job Input* dialog) with an automated addition to the list of available *Job Input* definitions of the current workspace.

5.3.3 Simulation job execution

The *Execution* tab of the basic GUI frame manages the simulation of defined (error-free) *Job Input* definitions and a restart of already simulated *Job Result* instances (with a defined number of additional simulation steps): After addition to the *Job Execution Queue*, jobs may be executed in parallel according to the global settings defined in *Preferences/Parallel computing*. For job execution MFsim must convert a *Job Input* definition to a Jdpd command file with corresponding particle position files (see class *JobUtilityMethods* in package *model.job* as well as appendix 4). Completed simulations are stored as new *Job Result* instances for further evaluation (see below). Job execution is always performed in background so it does not interfere with the concurrent usage of any other MFsim function (with the exception that a currently simulated *Job Input* definition is not allowed to be changed by editing operations or removal). For every job in simulation its progress in percent and the estimated remaining simulation period is displayed and constantly updated. The *Job Execution Queue* may be altered in an arbitrary manner after execution start where a job in simulation cannot be immediately killed but only stopped with a delay to guarantee a valid *Job Result*. All *Job Result* evaluation functions are also available for jobs in execution which allows for a detailed inspection and analysis of simulation progress.

The *Job Result* restart feature offers the possibility of successive simulation pipelines since underlying *Job Input* definitions may be changed before a restart. As an example, a possible molecule fixation or caging defined in the *Movement* sub-subsection (see above) could be removed before a restart of an initial relaxation simulation to allow for new modes of interaction between the molecular species. Changes of the molecular composition or the spatial configuration are not possible since they would affect the overall physical state of a simulation.

5.3.4 Simulation job result evaluation

The *Results* tab of the basic GUI frame comprises a list of all *Job Result* instances of the current workspace where a *Job Result* may be viewed, removed, imported from or exported to an archive file.

The *View* of a *Job Result* opens a modal dialog that consists of a feature tree which offers substantial simulation related evaluations (see figure 17). The *General information* section

summarizes general settings and results like simulation end point and period, number of simulation steps and corresponding physical time, parallelization settings or the used particle set file name. The *Simulation progress* section provides time-step data and corresponding graphical 2D diagram charts of relevant physical quantities like temperature, kinetic and potential energy or surface tension. The 2D diagram charts may be converted to movies that are in accordance with simulation movies (see below) to allow for a combined display. If specific property calculations throughout the simulation were defined in the *Property calculation* subsection of the underlying *Job Input* definition, the corresponding result evaluations appear as additional sections in the *Job Result* feature tree, e.g. the *Nearest neighbor* section in figure 17. The *Distribution movie* section (see figure 17) provides movies with animated 2D diagram charts of a particle/molecule frequency along a selected axis for a defined simulation period.

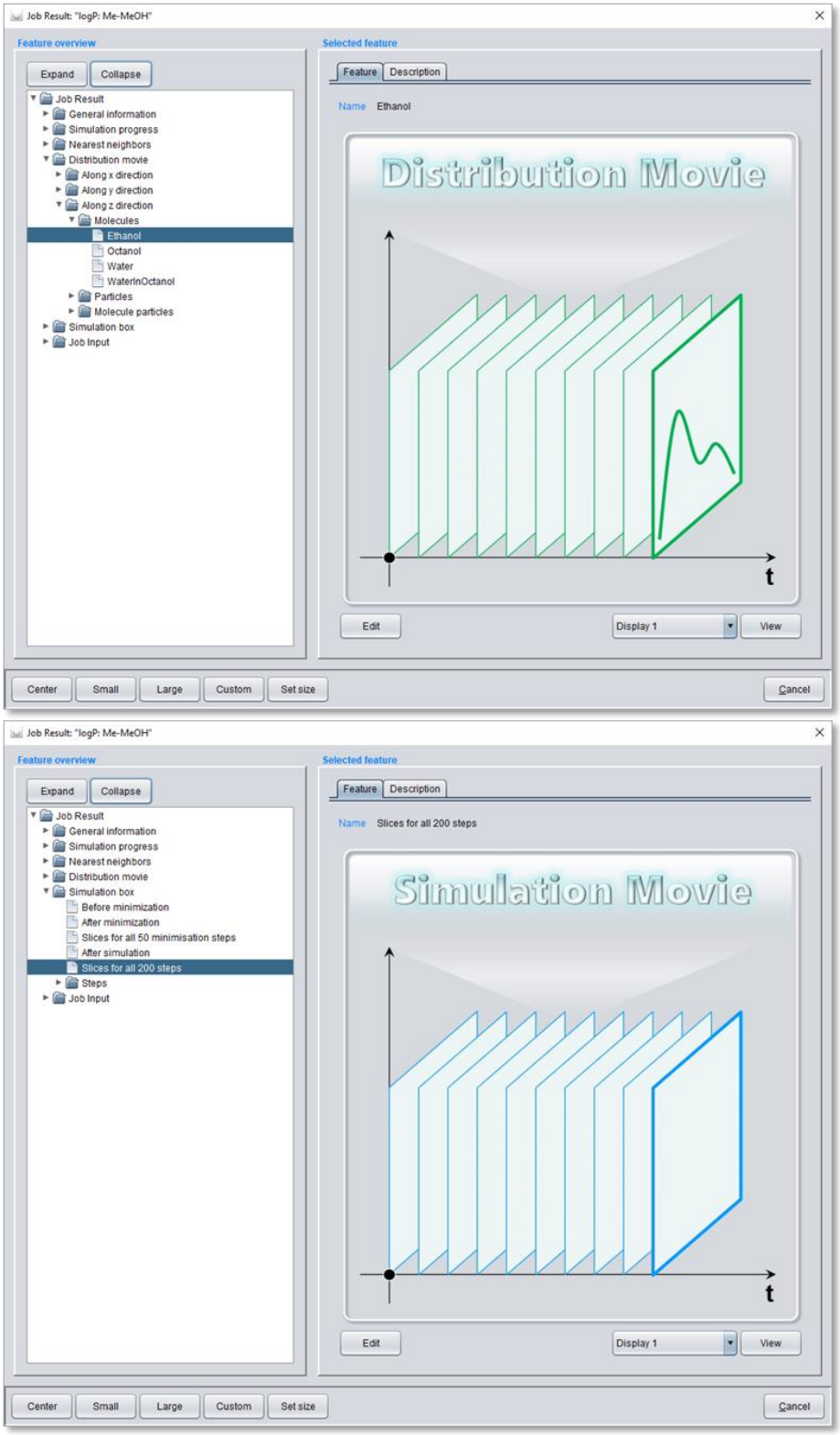


Figure 18. Feature tree of the modal *View* dialog of a *Job Result* with selected *Distribution Movie* feature (top) and *Simulation Movie* feature (bottom) [vandenBroek2020].

The *Simulation box* section provides (animated) simulation box views of all (output) steps of the simulation process including the a priori minimization steps (if defined in the *Job Input* definition) and the generation of simulation movies for defined simulation periods (see figure 17). For simulation box view and analysis, MFsim consists of two types of viewers for arbitrarily rotated or shifted box inspections: The *Simulation box viewer* (mis)uses the graphical Jmol capabilities designed for atom-based molecule representations to display particle structures whereas the *Simulation box slicer* generates successive slice images of the simulation box using Java2D. Moreover, the *Simulation box slicer* provides extensive box analysis features like through-space measurements (compare appendix 2), (individual) particle selections or the definition of zoom volumes with a corresponding particle/molecule frequency analysis (see figure 18). Molecular color, size, visibility or transparency settings may be arbitrarily changed for alternative views of the molecular ensemble of interest.

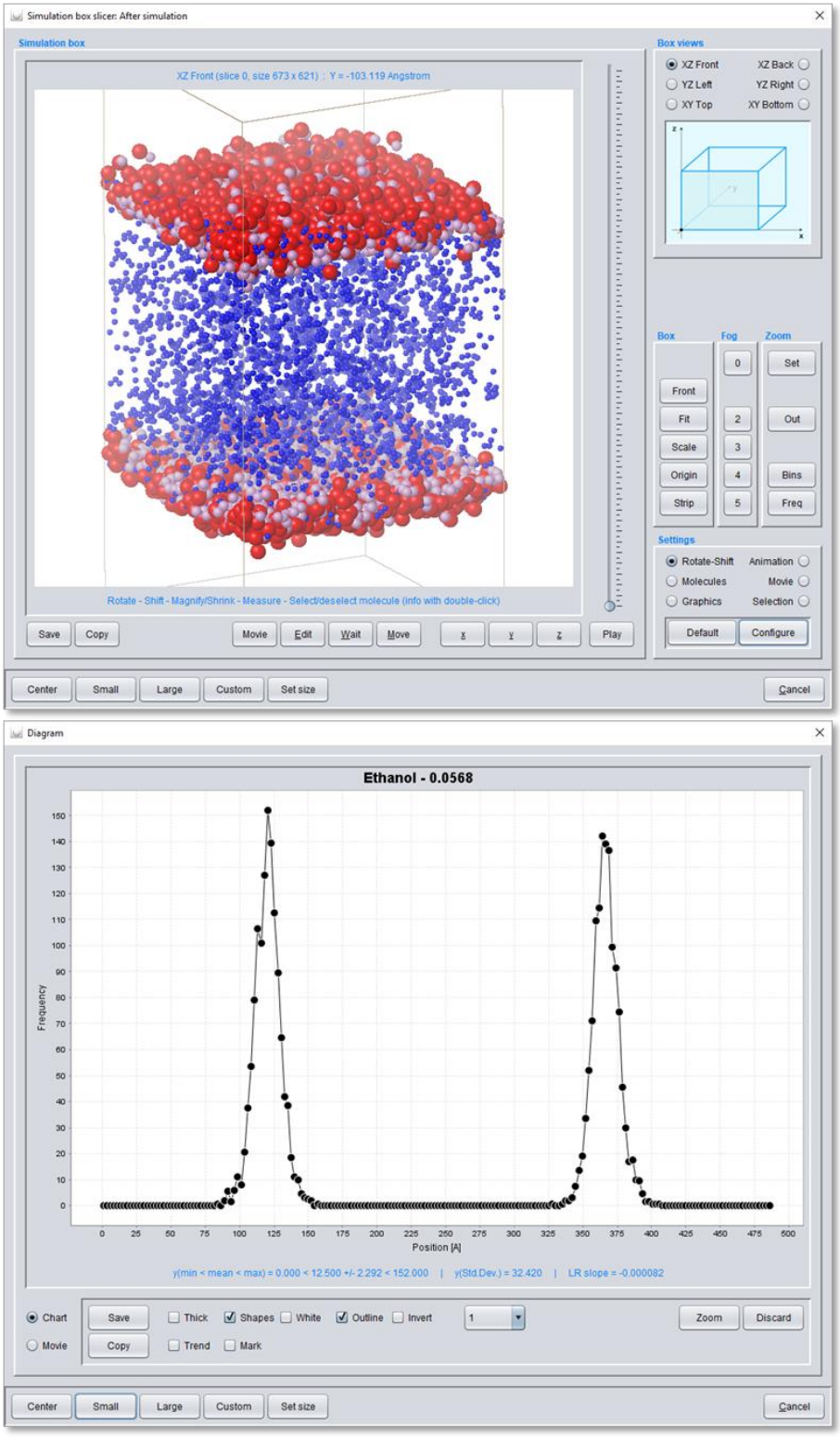


Figure 19. *Simulation box slicer* display of ethanol molecules (SPICES line notation: Me-MeOH) from an octanol-water mixture in a simulation box [vandenBroek2020]. The diagram at the bottom quantifies the ethanol frequencies along the z-axis with two distinct layers and corresponds to the ethanol distribution in the simulation box at the top. Ethanol particle colors: Me (plum), MeOH (red). All other molecules of the ensemble are discarded from display (octanol not visible) except water particles (blue) between the layers.

Simulation movies for a defined simulation period are generated with the slicer functionality. Slicer-related animation settings (e.g. box spinning, box slicing or box zooming) may be used to contribute to the movie generation in an additive manner: A simulation movie may start with a spinning box, then show the minimization steps, again stop with a spinning box after minimization, exclude specific particles/molecules, zoom-in, then show a defined part of the simulation process within the zoomed box volume etc. All slicer functions produce sequences of images which may finally be merged into a MP4 movie clip with the open FFmpeg software (automated integration of FFmpeg is currently only available for the Windows operating system).

Last but not least the final *Job Input* section just shows all features of the underlying *Job Input* definition.

5.3.5 MFsim use cases

The new MFsim environment may be beneficial for different fields of research. Theoretical physicists and chemists that contribute to the foundations of mesoscopic simulation with new methods/models/applications or improved/extended particle parameter sets may choose the open environment for an exemplification of their work so that their new insights and developments can be more easily adapted and utilized by a wider scientific community where especially scientific end-user communities appreciate availability within a rich-client system.

Chemo- and bioinformaticians may likewise be interested in the extended possibilities to popularize their new and improved algorithmic solutions by offering a versatile usage in a convenient environment with only comparatively small additional development efforts.

As an open all-in-one rich-client system with (modest) hardware requirements that are commonly available in scientific institutions, MFsim especially targets scientific end-users without programming skills. If a theoretically sound simulation model is available, specific alterations and extensions can be performed by end-users themselves. Tutorials *Simulation of a DMPC bilayer membrane model* [MFsimPDF04, MFsimClip09] and *Cyclotide-membrane sandwich interaction model* [MFsimPDF01, MFsimClip04] illustrate details of biomolecular simulation setups in a step-by-step fashion (with the corresponding MFsim Job Inputs being available in the MFsim GitHub repository tutorials subfolder) to show that the inevitable training hurdle for productive usage is manageable with tolerable effort. The MFsim GitHub repository also contains links to MFsim-generated simulation clips (see README in subfolder

2020 Cyclotide-membrane interaction study) which demonstrate an especially attractive feature for visual communication of research findings.

In summary, MFsim may broaden the applicability of mesoscopic approaches (at best in close collaboration with experimental research) and stimulate the collaboration between the different disciplines from theory over computing to end-user application.

5.4 Cyclotide/membrane interactions

5.4.1 Particle data

The particle-related data of this work are taken from [Truszkowski2015] (with rescaled molecular volumes and water being the smallest particle of volume 30 $Å^3$) and collected in particle set text file *ParticleSet_AA_V02.txt* [MFsimParticleSet2020]: Within this text file, all particle-related definitions are specified in section *Particle description*, the particle-particle repulsions a_{ij} are listed in section *Particle interactions* and the peptide/protein decomposition-related particle information is given in section *Amino acids*.

5.4.2 Open simulation software

All simulations of this work were performed with the open mesoscopic simulation environment MFsim [vandenBroek2020] using the default open MFD engine Jdpd [vandenBroek2018b] where all illustrations and animations were derived from MFsim graphics. The MFsim project provides the preliminary open biomolecular particle set sketched in the previous section for an approximate representation of phospholipids and amino acids, implements the SPICES particle structure line notation [vandenBroek2018] and provides a peptide/protein editor with automated SPICES decomposition from PDB files [ProteinDataBank2020]. A compartment editor supports the initial setup of interaction models by defined molecule placement in the simulation box including spatial peptide/protein 3D structures.

A detailed tutorial for the setup and analysis of MFsim simulation jobs used for this publication may be downloaded [MFsimPDF03] or viewed/downloaded in animated form [MFsimClip10]. All simulation job definitions are openly available at [MFsimStudy2020]. Thus, the complete setting details may be reviewed with MFsim (e.g. running on a standard notebook computer), and each simulation job may be anywhere

reproduced with MFsim running on a modest multicore workstation using 8 parallelized Jdpd calculation threads (defined in MFsim via *Preferences/Parallel computing/Parallel calculators*). A similar open documentation of the simulation records is still prohibitive due to their size (multiple gigabytes each). In order to support more detailed inspections of the performed simulations, MFsim-generated animations are referenced throughout the text to be viewed or downloaded.

5.4.3 Simulation details

For all simulations MFsim default settings are used if not denoted otherwise. All molecular species are manually defined by their SPICES strings. The cyclotide SPICES strings are automatically derived from their corresponding PDB files and additional settings: Mutations and amino acid sequence loop closures are manually defined, disulfide bridges and pH dependent charges are assigned in an automated manner. All amino acid side chains are charged for pH 7.4 according to the particle set information. The "sandwich" model start geometry (see below) is constructed with xy-layer compartments: Single water particles are randomly positioned within their compartments (but outside of exclusive cyclotide spheres, see below). Phospholipids and cholesterol are collapsed and compressed into linear SPICES tubes according to their start/end tagged particles [vandenBroek2018] and randomly positioned in the xy-plane perpendicular to their tube orientation with their start particles being located at the compartment's upper or lower z-surface. All amino acid sidechain particles of the cyclotides are collapsed onto their corresponding backbone particles which themselves are relatively positioned according to the C_α-coordinates defined in their PDB 3D structures. These collapsed cyclotide 3D particle structures are then squeezed and randomly rotated within exclusive spheres (with a DPD volume according to the cyclotide particle count) and randomly positioned in their compartment without overlap. Different random start geometries are generated with different *Geometry random seed* values. *Protein distance forces* for preservation of the cyclotide backbone 3D structure during simulation are completely activated with force constants (denoted k_{BB}) of 4 DPD units for all cyclotides. *Molecule backbone forces* are defined for cholesterol with a force constant of 4 DPD units (see cholesterol below). For integration of the equations of motion Shardlow's S1 scheme [Shardlow2003, Nikunen2003] is used (denoted *S1MVV* in MFsim setting *Integration type*). *Periodic boundaries* are turned on in x- and y-direction with reflective box walls

in z-direction. Each simulation is preceded by a conservative energy minimization procedure with 100 *Initial minimization steps* ("force steps") to relax the initial "high energy" box geometry (particularly due to the collapsed and squeezed SPICES tubes and cyclotide 3D structures). For additional initial system relaxation – now in contact with a "293 K thermostat" – each simulation starts with 100 *Initial velocity scaling steps* that also remove "excess momentum". The molecular ensembles for the SAR studies comprise more than 600,000 particles representing more than two million atoms and are studied with up to 125,000 simulation steps which correspond to a physical simulation time of 7.5 µs. As a rule-of-thumb, a single simulation run with 8 parallelized Jdpd calculation threads allows to study the dynamics for several microseconds in less than a day with an ordinary multicore workstation.

5.4.4 Molecular fragmentation

Figure 19 and figure 20 sketch the molecular structure decompositions of the bilayer membrane constituent phospholipids and cholesterol into connected "molecular fragment" particles using the SPICES line notation. In general, adequate "molecular fragment" particles have to be determined by experience and chemical intuition due to the lack of an objective fragmentation procedure (e.g. see their "circular identification" in figure 19 on the left). In order to keep spatial characteristics of a molecular species, it may be necessary to impose additional intramolecular "backbone forces" between specific particles, e.g. the "rod-like" geometry of cholesterol due to its rigid ring system is reflected by an additional "molecule backbone force" spring as sketched in figure 19 on the right (where the SPICES line notation supports the corresponding backbone labels).

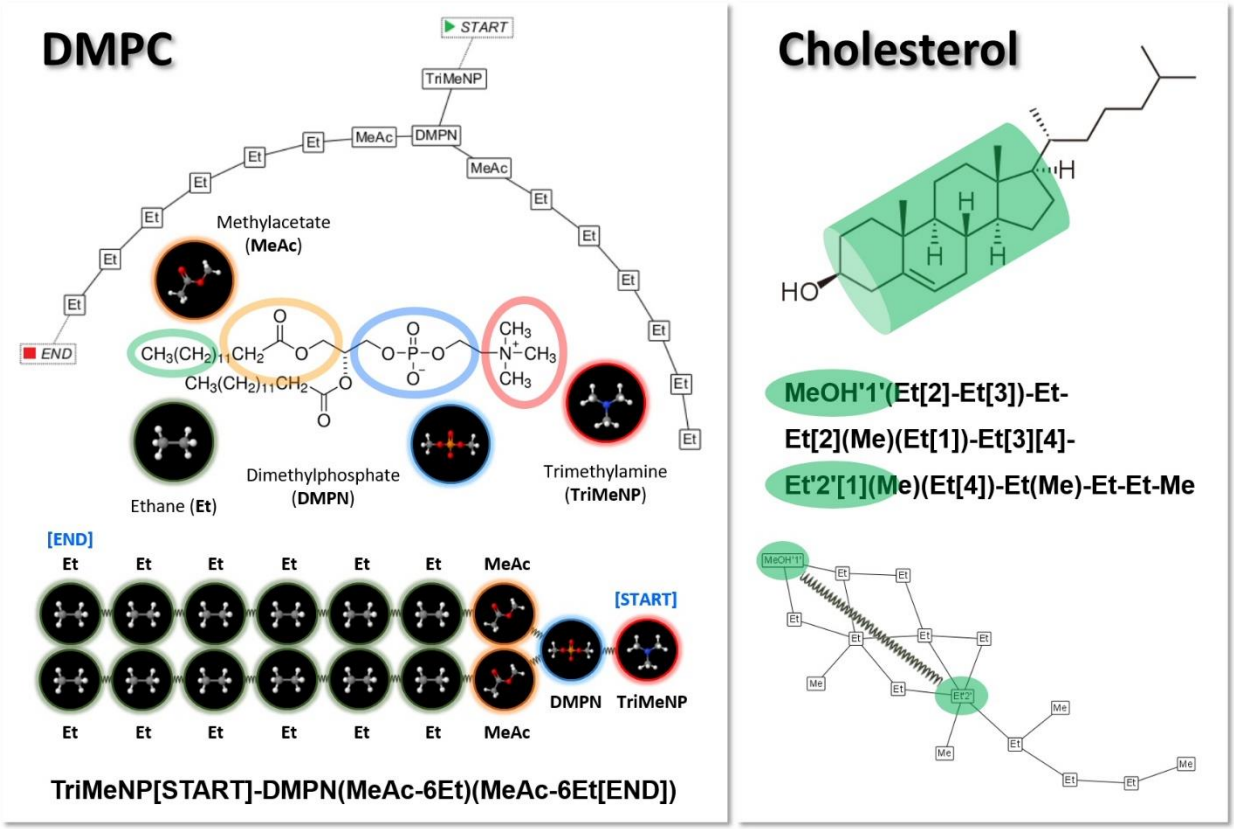
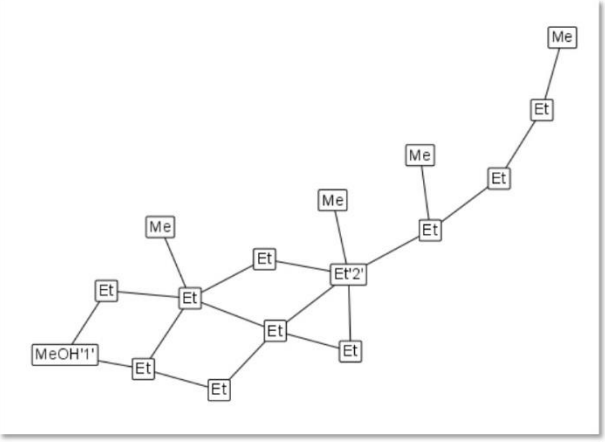


Figure 20. Left: Decomposition of phospholipid DMPC into "molecular fragment" particles (ethane, abbreviated with "particle acronym" Et, methyl acetate, MeAc, negatively charged dimethyl phosphate, DMPN, and positively charged trimethylamine, TriMeNP) and their adequate topological connection with harmonic springs. At the top the topological particle connection illustration generated by MFsim is depicted with the corresponding SPICES line notation shown at the bottom (the included start and end tags support adequate spatial positioning in the simulation box for an initial bilayer geometry [vandenBroek2018]). Right: "Molecular fragment" particle decomposition of cholesterol allowing for an adequate spring force definition between the highlighted particles MeOH and Et (with backbone labels '1' and '2') to account for the spatial rigidity of the cholesterol ring system. Particle acronyms follow the conventions defined in the particle set text file [MFsimParticleSet2020].

Cholesterol

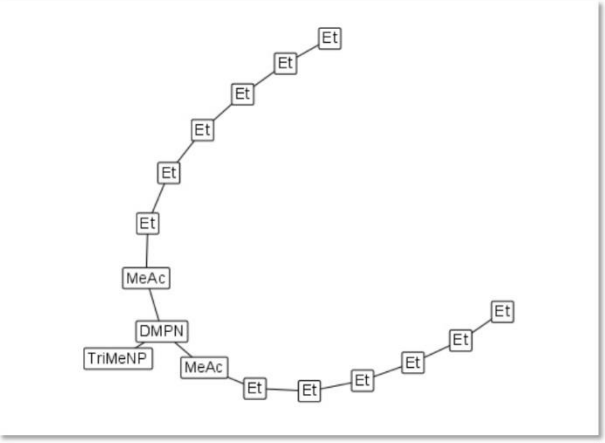
(3ß)-Cholest-5-en-3-ol



MeOH'1'(Et[2]-Et[3])-Et-Et[2](Me)(Et[1])-Et[3][4]-Et'2'[1](Me)(Et[4])-Et(Me)-Et-Et-Me

DMPC

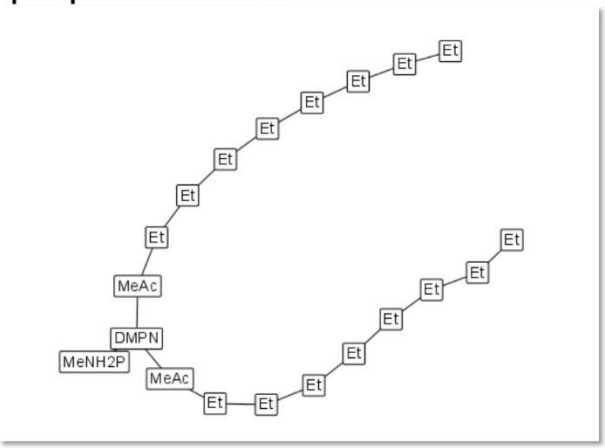
1,2-Dimyristoyl-sn-glycero-3-phosphocholine



TriMeNP-DMPN(MeAc-6Et)(MeAc-6Et)

DOPE

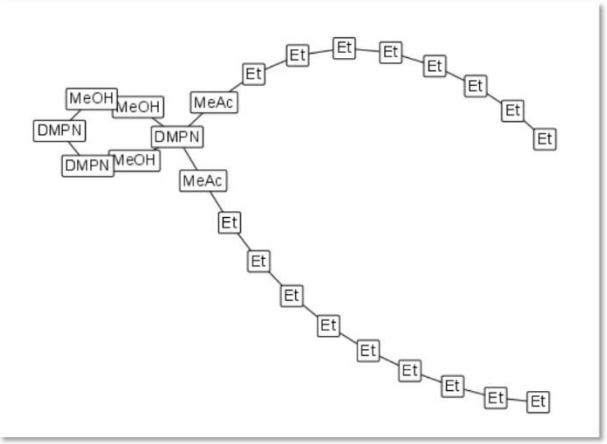
1,2-Dioleoyl-sn-glycero-3-phosphoethanolamine



MeNH2P-DMPN(MeAc-8Et)(MeAc-8Et)

PIP_2

Phosphatidylinositol 4,5-bisphosphate



DMPN[1](MeAc-8Et)(MeAc-9Et)-MeOH-DMPN-DMPN-MeOH-MeOH[1]

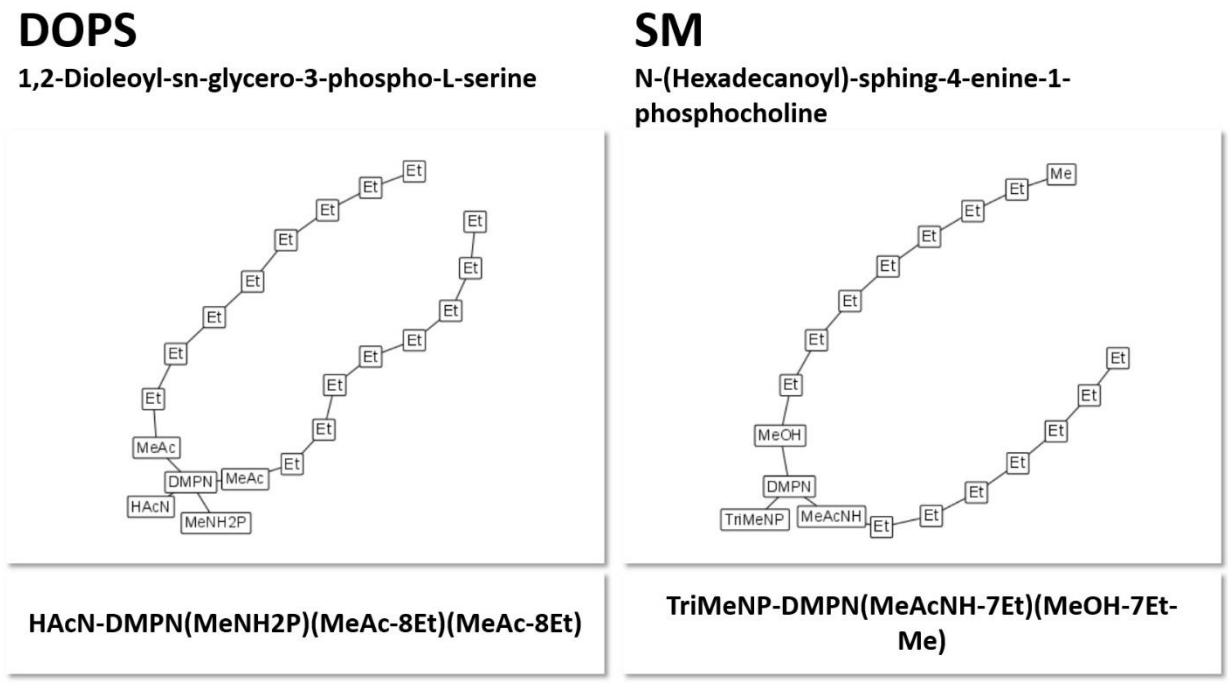


Figure 21. "Molecular fragment" particle decomposition of the constituent membrane lipids (acronym, chemical name, topological particle connection illustration generated by MFsim and corresponding SPICES line notation). Particle acronyms follow the conventions defined in the particle set text file [MFsimParticleSet2020].

Figure 21 summarizes the decomposition of cyclotides Kalata B1 (PDB ID: 1NB1) and Cycloviolacin O2 (PDB ID: 2KNM) into "molecular fragment" particles. The cyclotides consist of a loop structure with 29 (Kalata B1)/30 (Cycloviolacin O2) amino acids where the "hydrophobic patch" as a specific structural feature is highlighted. The "molecular fragment" particle decomposition is generated by the automated PDB-file-based fragmentation procedure of MFsim. For this purpose, section *[Amino acids]* of the particle set text file defines SPICES strings for all 20 proteinogenic amino acids and pH-dependent (charged) particle replacements for the C- and N-terminus as well as the sidechains (note that in figure 21 only the sidechain replacements for pH 7.4 signaled by the amino acid tags {S-}/{S+} are considered since both cyclotides contain a loop structure). Additional intramolecular "protein distance forces" between backbone particles are defined for control of the rigidity/flexibility of the spatial 3D backbone structure. The number label of a single cyclotide amino acid is derived from Jmol [Jmol2020] labelling which is used by MFsim.

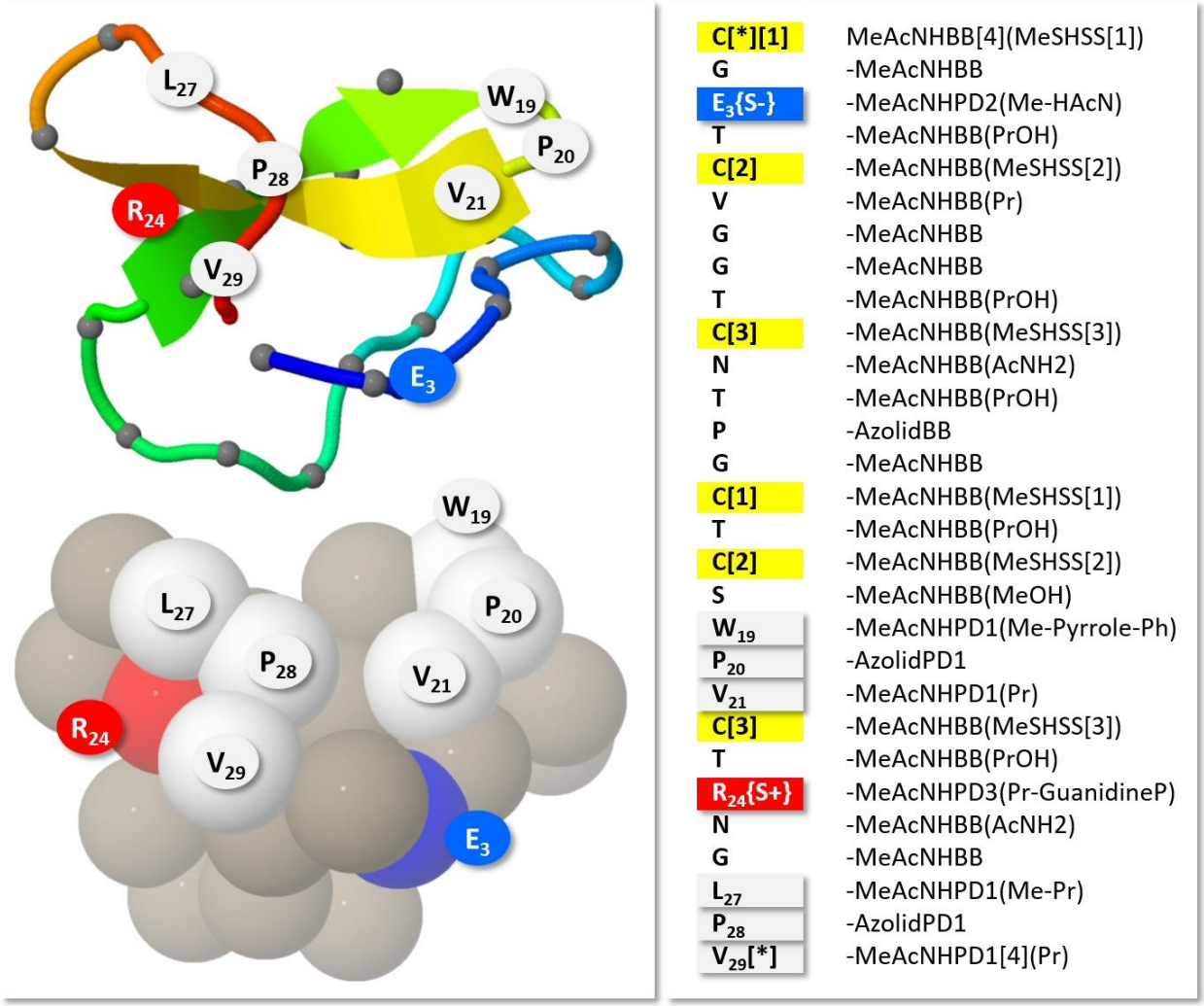


Figure 22. The cyclotide Kalata B1. Upper left: Cartoon representation of the C_α-backbone of Kalata B1 with the 6 highlighted amino acids of the "hydrophobic patch" (W_{19}, P_{20}, V_{21}, L_{27}, P_{28}, V_{29}) and the two charged sidechain amino acids (E_3, R_{24}). Lower left: Corresponding backbone particle 3D structure generated by MFsim from the 3D coordinate information of the PDB file (side chain particles are not shown) with the backbone particles of the hydrophobic patch highlighted in white and the positively/negatively charged amino acid backbone particles colored in red/blue. Top, right: Amino acid sequence (bold font type from top to bottom) with corresponding parts of the SPICES line notation generated by the automated MFsim "molecular fragment" particle decomposition for peptides and proteins. The amino acid tags {S+}/{S-} denote the positively/negatively charged side chain (particle) at pH 7.4 for R_{24}/E_3, bracketed indices indicate connections and the bracketed asterisk signals a closed loop. "Hydrophobic patch" and charged amino acids are highlighted and labelled, cysteine in disulfide bridges is indicated by a yellow background color. All particle acronyms follow the conventions in the particle set text file [MFsimParticleSet2020] (suffix BB denotes backbone particles, suffix SS particles in disulfide bridges, suffix PD<index> defines a "probe definition" that may be used for particle highlighting with specific colors. e.g. to visualize the "hydrophobic patch").

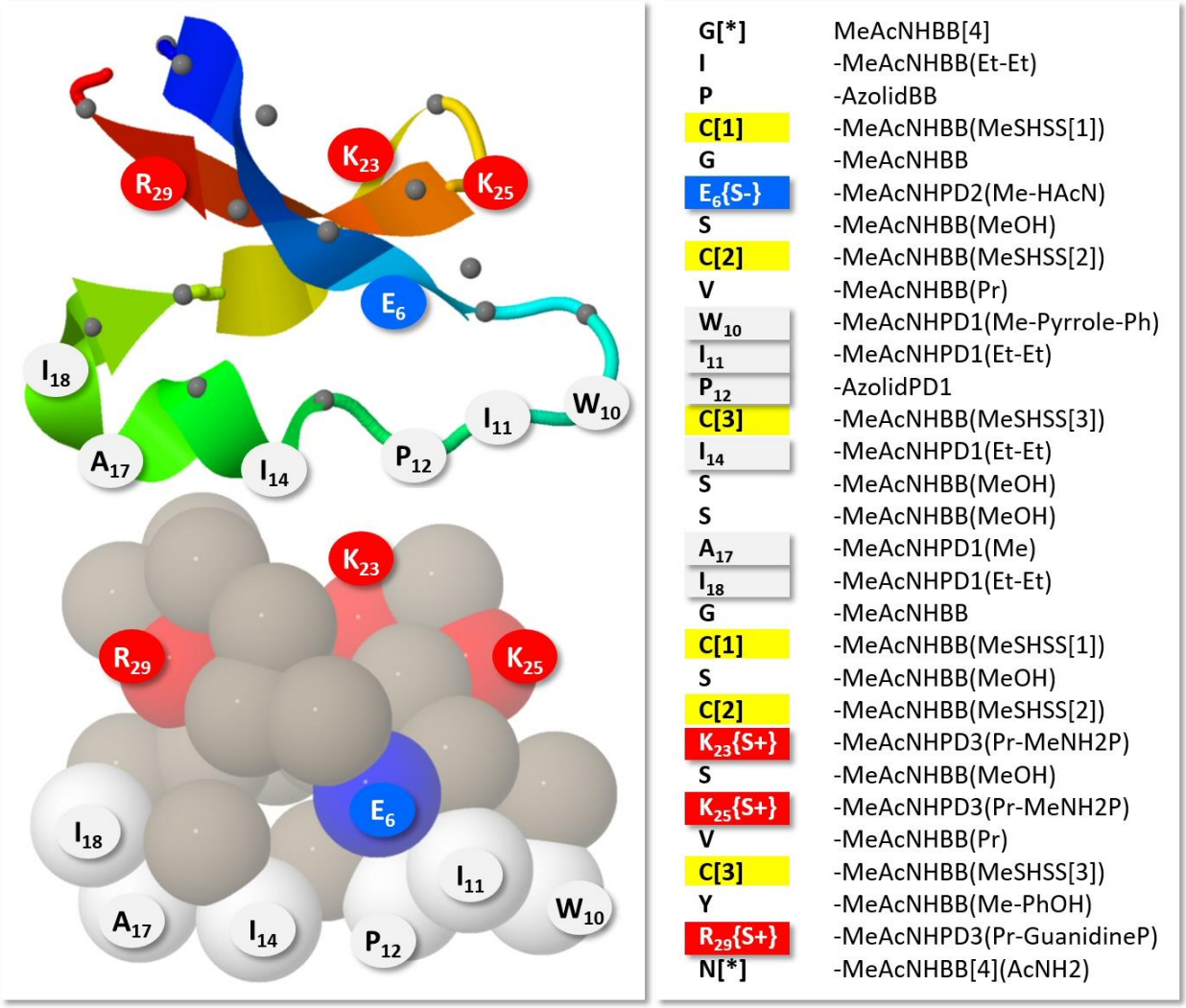


Figure 23. The corresponding information for cyclotide Cycloviolacin O2. Upper left: Cartoon representation of the C_α-backbone of Cycloviolacin O2 with the 6 highlighted amino acids of the "hydrophobic patch" (W_{10}, I_{11}, P_{12}, I_{14}, A_{17}, I_{18}) and the four charged sidechain amino acids (E_6, K_{23}, K_{25}, R_{29}). Lower left: Corresponding backbone particle 3D structure generated by MFsim from the 3D coordinate information of the PDB file (side chain particles are not shown) with the backbone particles of the hydrophobic patch highlighted in white and the positively/negatively charged amino acid backbone particles colored in red/blue. Top, right: Amino acid sequence (bold font type from top to bottom) with corresponding parts of the SPICES line notation generated by the automated MFsim "molecular fragment" particle decomposition for peptides and proteins. The amino acid tags {S+}/{S-} denote the positively/negatively charged side chain (particle) at pH 7.4 for E_6/K_{23}/K_{25}/R_{29}, bracketed indices indicate connections and the bracketed asterisk signals a closed loop. "Hydrophobic patch" and charged amino acids are highlighted and labelled, cysteine in disulfide bridges is indicated by a yellow background color. All particle acronyms follow the conventions in the particle set text file [MFsimParticleSet2020] (suffix BB denotes backbone particles, suffix SS particles in disulfide bridges, suffix PD<index> defines a "probe definition" that may be used for particle highlighting with specific colors. e.g. to visualize the "hydrophobic patch").

All cyclotide mutants are constructed with the MFsim peptide/protein editor which supports specific amino acid replacements. These replacement operations do not influence the spatial 3D backbone structure which is in general likely to cause structural inaccurateness in the backbone conformation. But cyclotides Kalata B1 and Cycloviolacin O2 exhibit an extraordinary stable spatial conformation (with each containing a loop structure plus 3 disulfide bridges) so that only negligible conformational backbone changes are expected – note that all sidechain particles are flexible being only constrained by standard "spring force" bonds.

Cyclotide mutants follow the notation <*original amino acid*><*number label of amino acid*><*amino acid of mutant*> where multiple mutations are concatenated, e.g. mutant kB1-W19Y has a single replacement with amino acid tryptophan (W) at position 19 being replaced by amino acid tyrosine (Y), mutant kB1-W19Y-P20S-V21T contains three amino acid replacements at consecutive positions 19 (tryptophan – W – replaced by tyrosine – Y), 20 (proline – P – replaced by serine – S) and 21 (valine – V – replaced by threonine – T).

5.4.5 Plasma membrane models

Each bilayer membrane is constructed with 1600 phospholipid/cholesterol molecules. The phospholipid composition of plasma membrane models is chosen to be 40% DMPC, 20% DOPE, 5% PIP_2, 10% DOPS and 25% SM (see figure 20) with an inner to outer leaflet distribution of 24 to 76 for DMPC, 77.5 to 22.5 for DOPE, 75 to 25 for PIP_2, 1 to 0 for DOPS (only inside) and 22 to 78 for SM [Daleke2008, vanMeer2008]. The proportion of cholesterol is varied from 0% to 50%. If cholesterol molecules are present, the sum of phospholipid and cholesterol molecules are equalized for the inner and outer membrane leaflet. The start geometry consists of a bilayer of Start/End-tag oriented phospholipid and cholesterol molecules (compare figure 19 and figure 22) being manually built with the compartment editor of MFsim. The xy-area of the simulation box can be chosen to allow for an area per lipid molecule (ApL) of 40, 50 or 60 $Å^2$ [Yamamoto1963, Bar1966, Engelman1969, Kiselev2006, Kučerka2011, Ingólfsson2014]. Pure DMPC or DOPE bilayer membranes without cholesterol are abbreviated NoC-DMPC-M or NoC-DOPE-M respectively. A plasma membrane without cholesterol is abbreviated NoC-PM, a plasma membrane with 50% cholesterol molecules 50C-PM. The complete definitions of the membrane models are available at [MFsimStudy2020].

5.4.6 Cyclotide-membrane interaction model

The proposed "sandwich" cyclotide/membrane interaction model is an artificial construct to estimate lipid extraction, designed for simplicity without any additional forces or spatial constraints as well as a clear connection to experimental findings. The choice of the cyclotide number in the cyclotide/water compartment is driven by maximizing the disruptive effect with a minimum number of cyclotides – where the extremes would be a single cyclotide within the compartment (with possible membrane interaction but no disruptive lipid extraction effect) and a biologically unrealistic compartment which solely consists of cyclotides without any water. The initial simulation box setup is sketched in figure 22: There are two water compartments at top and bottom of the simulation box with 200,000 water particles each in contact with the inner membrane leaflets. The two membrane compartments are composed as described above. The outer membrane leaflets of both membranes are in contact with two buffer compartments containing 5,000 water particles each to prevent initial contact of membranes and cyclotides. The enclosed cyclotide/water compartment consists of 190,000 cyclotide and water particles distributed according to the defined cyclotide number. Within their compartment the cyclotides are randomly oriented and positioned with their spatial 3D backbone structure (option *3D random* in compartment editor of MFsim), all sidechain particles are initially collapsed onto their corresponding backbone particles by default.

The number of cyclotides is set to 1,000 to obtain evaluable effects within simulation times of a few microseconds. This corresponds to an initial cyclotide to water ratio within the cyclotide/water compartment of 1 to 145 for Kalata B1 (a single Kalata B1 cyclotide is represented by 55 particles, so 1,000 cyclotides lead to 55,000 particles; the compartment size without buffer is fixed to 190,000 particles, so that 190,000 – 55,000 = 135,000 water particles are added; 135,000 + 10,000 buffer water particles = 145,000 water particles for 1,000 cyclotides result in 145 water particles for every single cyclotide in the whole cyclotide/water compartment) and 1 to 134 for Cycloviolacin O2 (where a single Cycloviolacin O2 cyclotide is represented by 66 particles) with similar values for their mutants (due to slightly different numbers of the specific cyclotide particles). The aim is to create a high local concentration of cyclotides.

Since the physical length of a DPD simulation box depends on the particle types and their relative numbers, the DPD lengths of the model systems containing cyclotides are adjusted to the DPD lengths of the pure membrane DPD models. The complete settings of all interaction models are available at [MFsimStudy2020].

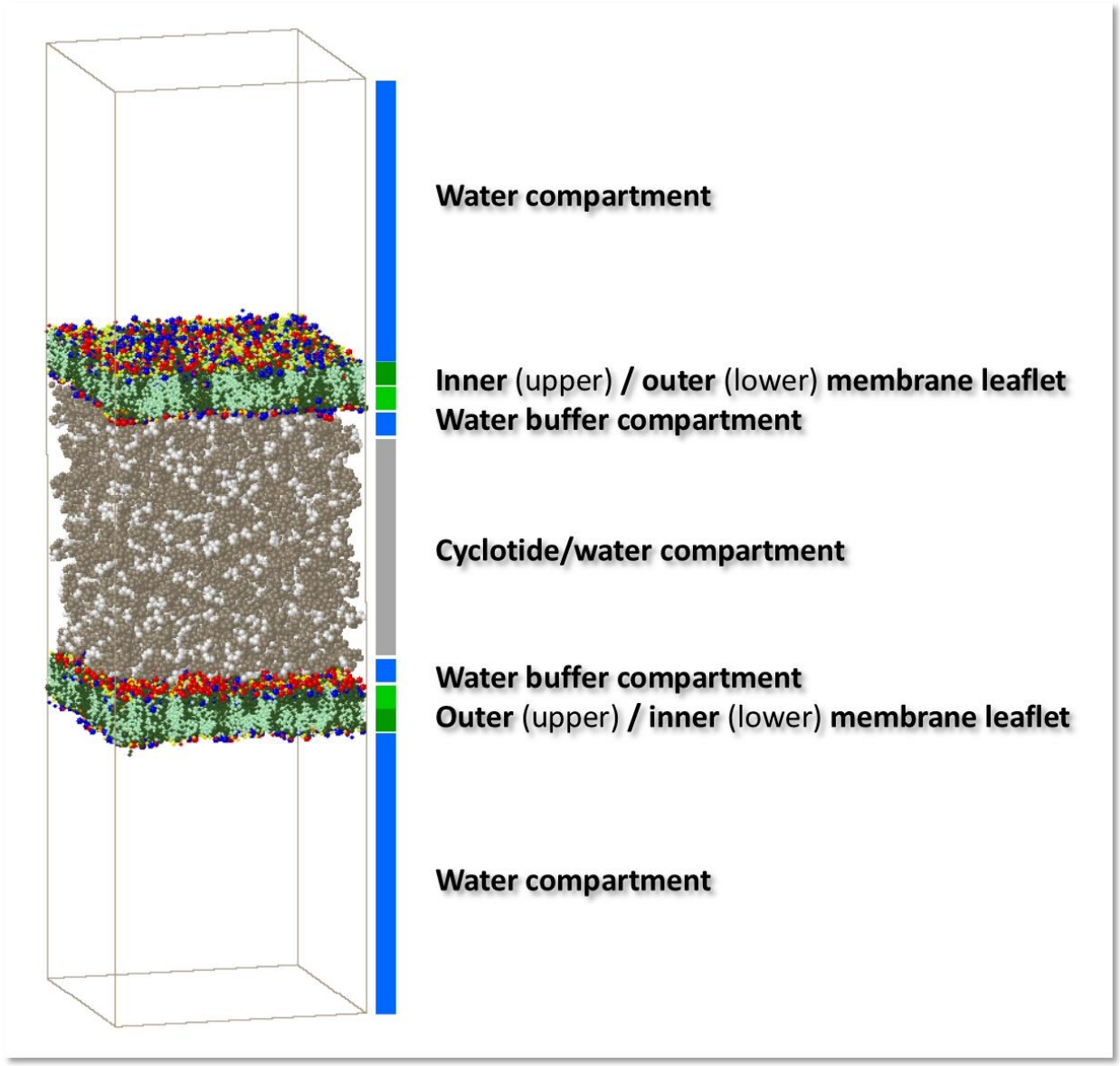
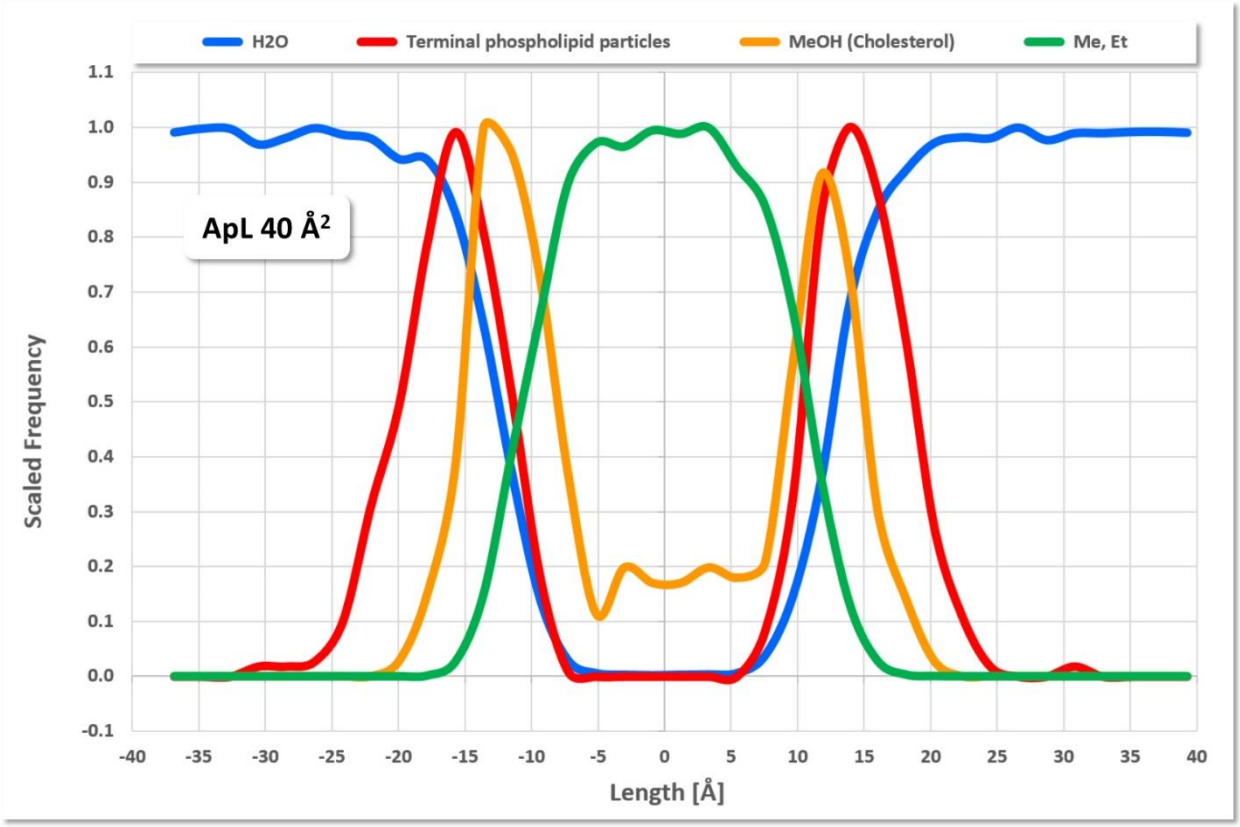
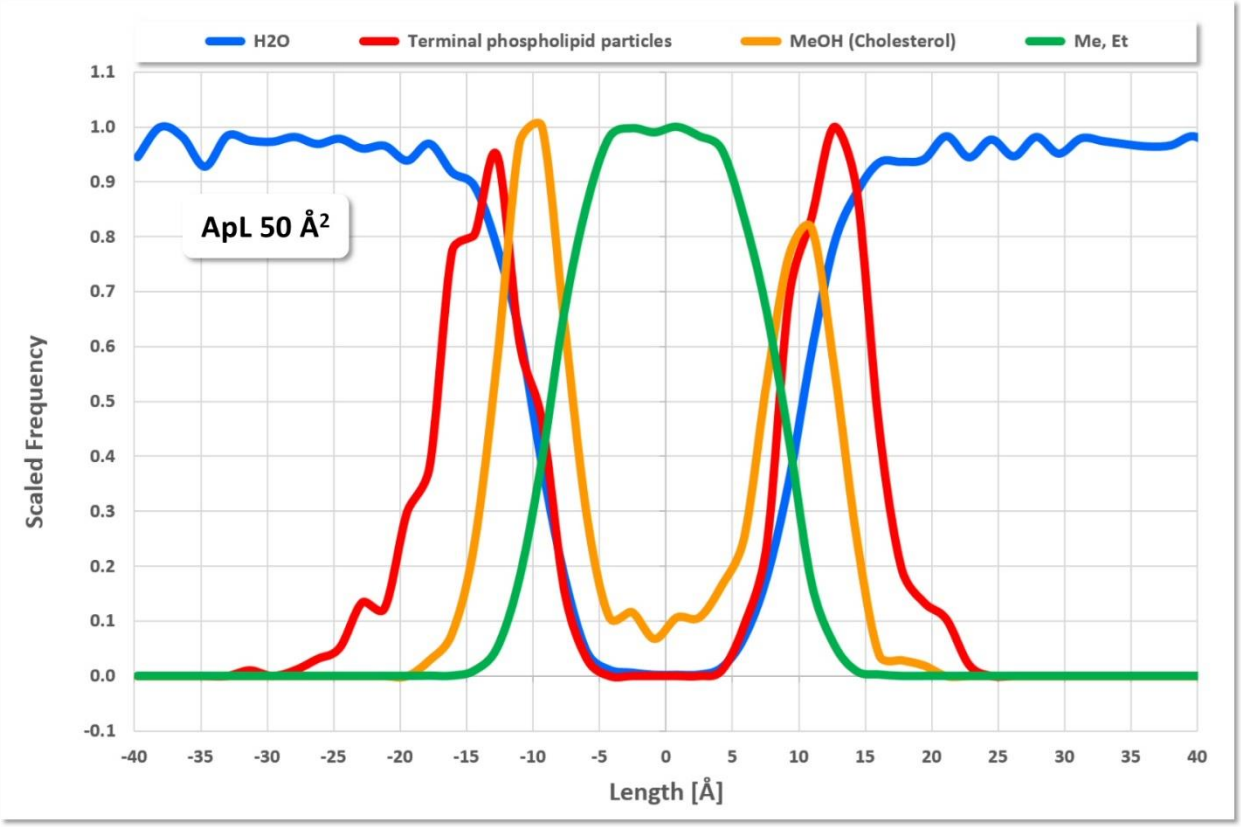


Figure 24. Initial simulation box setup of the "sandwich" cyclotide/membrane interaction model: The backbone particles of the Kalata B1 "hydrophobic patch" are colored in white, all other backbone particles are shown in beige. The positively/negatively charged particles of the phospholipids are colored in red/blue, MeOH in orange and "hydrophobic" particles in olive. MeOH particles of cholesterol are displayed in yellow, the "hydrophobic" particles in mint. Kalata B1 sidechain particles are not shown. All water particles are omitted.

5.4.7 Plasma membrane simulations

The plasma membrane models with areas per lipid molecule (ApL) of 40, 50 or 60 $Å^2$ build stable bilayers on the microsecond timescale. In contrast to the phospholipids which mainly stick to their initial leaflet with only rare inter-leaflet flips, cholesterol shows an extremely high inter-leaflet mobility with a steady-state distribution reached after a few hundred nanoseconds. The cholesterol mobility is demonstrated at [MFsimClip02] for a 50% cholesterol plasma membrane (50C-PM) with an ApL of 40 $Å^2$. An extremely high rate of cholesterol translocation between the leaflets is also experimentally indicated [Lange1981, Hamilton2003]. The overall thickness of the plasma membranes corresponds to their different areas per lipid to be about 40 to 50 Å with a hydrophobic thickness of approximately 20 to 30 Å, see figure 23: The hydrophobic thickness is approximated with the width of the distribution of the hydrophobic Me and Et particles (green), the overall thickness with the distance between the inner and outer distributions of the terminal phospholipid particles (red). The charged terminal phospholipid particles from the outer water contact layer, followed by the cholesterol MeOH particle layer. Me- and Et-particles of all cholesterol and phospholipids constitute the hydrophobic membrane core with a low permeability for water particles. These findings agree with experimental as well as alternative simulation results [Zaccai1975, Zaccai1982, Lewis1983, Marsh2002, Mitra2004, Bermúdez2004, Kiselev2006, Gao2007, Kučerka2009, Castillo2013, Kiselev2013]. For the SAR studies, the thickest membrane with an area of 40 $Å^2$ per lipid is chosen.


H2O
Terminal phospholipid particles
MeOH (Cholesterol)
Me, Et
ApL 40 Å²
Scaled Frequency
Length [Å]


H2O
Terminal phospholipid particles
MeOH (Cholesterol)
Me, Et
ApL 50 Å²
Scaled Frequency
Length [Å]

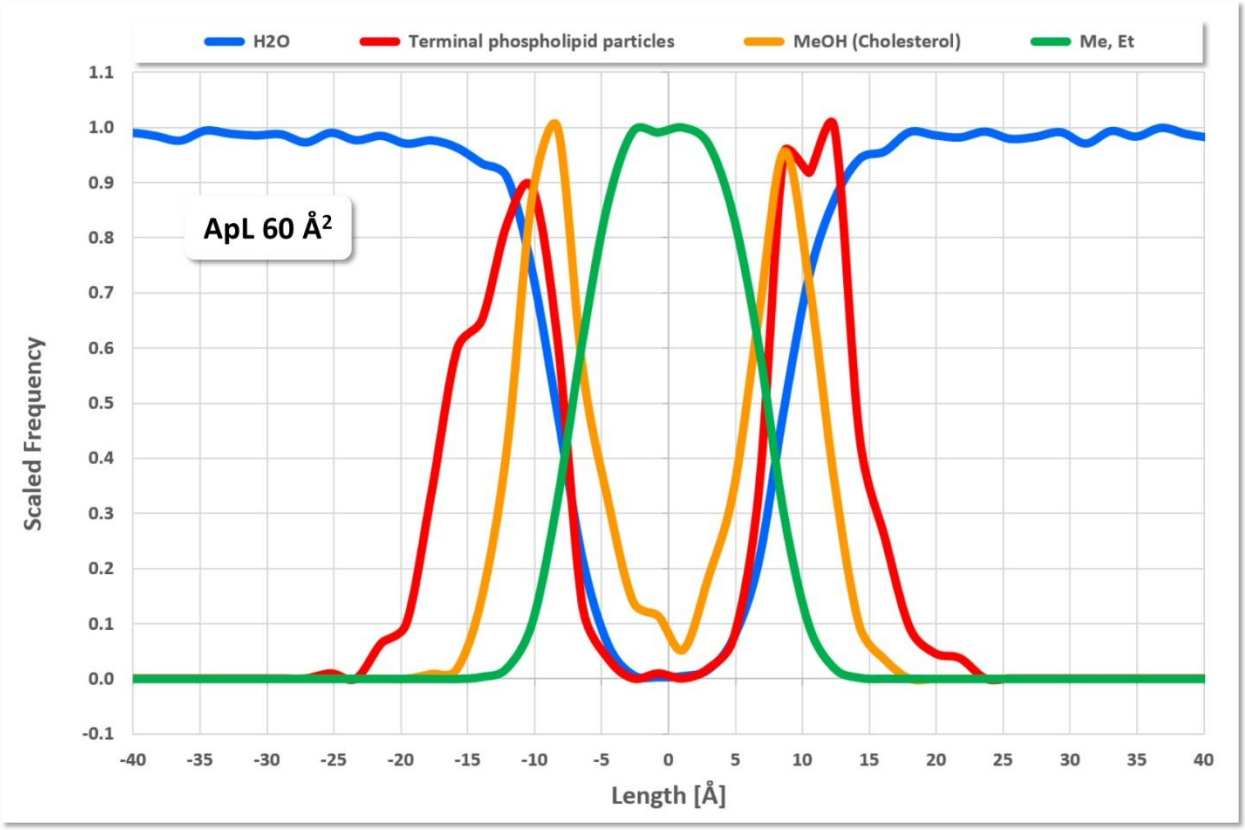


Figure 25. Snapshots of particle distributions across 50C-PM and an area per lipid (ApL) of 40, 50 and 60 Å^2 (simulation time: 2.5 µs). The depicted length (z) axis is perpendicular to the bilayer membrane (xy) plane with its origin being adjusted in the middle of the membrane, so that negative lengths correspond to the inner membrane leaflet and positive lengths to the outer one.

5.4.8 Cyclotide backbone flexibility

The backbone particle flexibility of a single Kalata B1 and a single Cycloviolacin O2 cyclotide within a water box is demonstrated at [MFsimClip01] for the specified "Protein distance forces" setting with force constants (k_{BB}) of 4 DPD units. This choice allows for fast transient oscillations from the "squeezed" start geometry (due to DPD density constraints of the MFsim compartment editor) towards the 3D geometry derived from the PDB structure during the minimization process. The simulation shows an acceptable degree of backbone flexibility with an overall conservation of the imposed 3D structure.

5.4.9 Cyclotide-membrane interaction

For the SAR studies the number of cyclotides is chosen to be 1,000 to obtain evaluable effects within simulation times of a few microseconds. This corresponds to an initial cyclotide to water ratio within the cyclotide/water compartment of 1 to 145 for Kalata B1 (a single Kalata B1 cyclotide is represented by 55 particles, so 1,000 cyclotides lead to 55,000 particles; the compartment size without buffer is fixed to 190,000 particles, so that 190,000 – 55,000 = 135,000 water particles are added; 135,000 + 10,000 buffer water particles = 145,000 water particles for 1,000 cyclotides result in 145 water particles for every single cyclotide in the whole cyclotide/water compartment) and 1 to 134 for Cycloviolacin O2 (where a single Cycloviolacin O2 cyclotide is represented by 66 particles) with similar values for their mutants (due to slightly different numbers of the specific cyclotide particles).

Qualitative simulation characteristics

The qualitative cyclotide/membrane interaction behavior from the mesoscopic point of view may be summarized as follows – compare animation of the interaction of cyclotide Kalata B1 with cholesterol-free (NoC-PM) and 50% cholesterol plasma membranes (50C-PM) at [MFsimClip07], figure 24 and upper row of figure 26: From the minimized start geometry, the two membranes exhibit a fast crossover to the particle distributions described above. While near-membrane cyclotides may immerse into the membranes, the majority of randomly distributed cyclotides form oligomers (also described by experimental [Nourse2004, Herrmann2006, Simonsen2008, Wang2009, Huang2009, Wang2012, Rosengren2013] and alternative simulation [Nawae2014] studies) that themselves aggregate towards a “network structure” in the cyclotide/water compartment with emerging “hydrophobic vicinities” particularly created by “hydrophobic patch” amino acids. These “hydrophobic vicinities” are able to accommodate the hydrophobic parts of membrane lipids. Thus, a membrane lipid distribution outside the initial membranes is initiated by extraction of phospholipid/cholesterol compounds from the membranes (see highlighted circular area in figure 24). With ongoing interaction, membrane lipids successively advance into the cyclotide/water compartment which increasingly disrupts the membranes. This process is accompanied by a constant decrease of the total potential energy of the system, i.e. the membrane lipid/cyclotide interaction is overall energetically preferred.

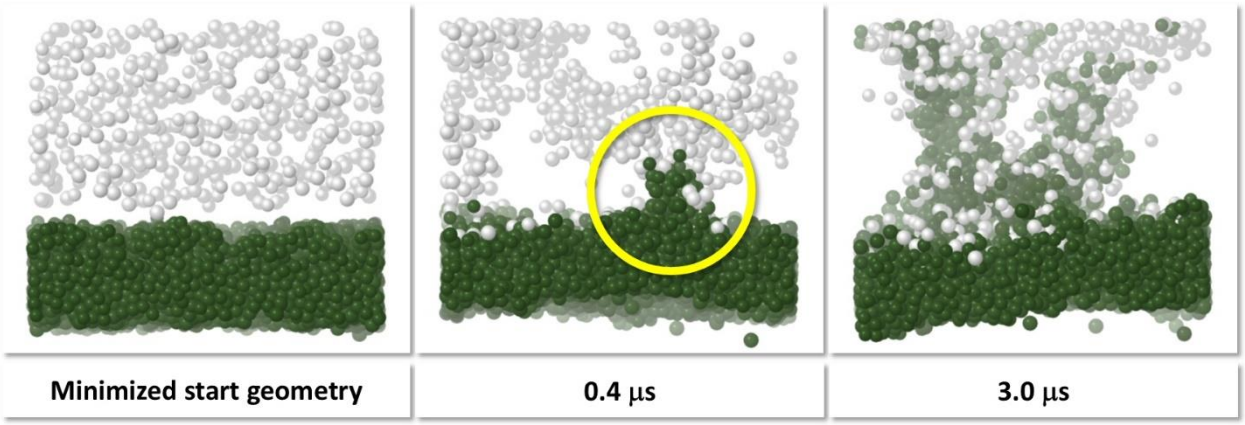


Figure 26. Interaction snapshots of Kalata B1 with a NoC-PM at different simulation times for the same magnified simulation box cutout. Only backbone particles of the Kalata B1 "hydrophobic patch" (white color) and "hydrophobic" phospholipid particles (dark green color) are shown.

5.4.10 Quantitative analysis of the membrane disruption process

A quantitative analysis of the sketched membrane disruption process may be performed by evaluating the temporal Et particle distribution of cholesterol and phospholipids in the cyclotide/water compartment which is initially equal to zero, i.e. there are no membrane lipids in this compartment. For that purpose, the cyclotide/water compartment is partitioned into 10 equal bins along the z-axis with bins 2 to 9 being evaluated to avoid the influence of possible membrane curvature, which may affect bins 1 and 10. An animation of the temporal Et-particle-distribution evaluation is available at [MFsimClip08]. Figure 25 (top) summarizes the procedure where the percentages on the y-axis refer to the corresponding total numbers of Et particles in the membranes which allows for comparability between different cyclotide/membrane systems. The phospholipid extraction into bins 2 to 9 is evaluated every 0.3 µs over 7.5 µs. Figure 25 (bottom) shows the results for 10 simulation runs with different random start geometries for each of three cyclotide/membrane systems (kB1-W19Y-P20S-V21T/NoC-PM, kB1/NoC-PM and cO2-E6Q/NoC-PM) which span the range from low to high bioactivities: For simulation times of 3.0 µs and above a "linear regime" appears that may be well approximated by a straight line from the fixed point of origin (no extracted Et particles at simulation start).

Thus, the total extracted Et percentages (the "areas" under the Et distribution "curves" in figure 25, top) can be characterized by a single rate constant for membrane disruption (abbreviated r_{MD}) being the slope of this linear regression line as an averaged percentage of extracted Et particles per microsecond: A higher r_{MD} value corresponds to a higher membrane disruption activity with a value of zero indicating no membrane disruption by lipid extraction.

Rate constants denoted $r_{MD}(6.0)$ refer to an evaluation of data between 3.0 and 6.0 µs whereas $r_{MD}(7.5)$ rate constants are derived from data between 3.0 and 7.5 µs.

From the tenfold repeated simulation runs with different random start geometries the standard error of a single-run r_{MD} value can be estimated to be about 0.2 units so that for a 10-fold repetition average the error drops below 0.1 units. The sketched evaluation procedure itself (with $25 \cdot 10 = 250$ simulation box distributions) – denoted full evaluation – can be considerably reduced without any significant loss by just evaluating distinct simulation times at 3.0, 4.5, 6.0 and 7.5 µs (with $4 \cdot 10 = 40$ simulation box distributions) for $r_{MD}(7.5)$ rate constants or 3.0, 4.5 and 6.0 µs (with $3 \cdot 10 = 30$ simulation box distributions) for $r_{MD}(6.0)$ rate constants respectively – denoted reduced evaluation. In addition, the resulting $r_{MD}(6.0)$ and $r_{MD}(7.5)$ values do not exhibit significant differences for the cyclotide/membrane systems studied. For minimal effort a 4-fold repetition with different random start geometries and reduced $r_{MD}(6.0)$ rate constant evaluation (with $4 \cdot 3 = 12$ simulation box distributions) may be regarded as a reasonable compromise which leads to an error of the resulting averaged r_{MD} rate constant of about 0.1 units. To ease comprehension the bioactivity of r_{MD} rate constants between 0 and 1 will be characterized as "low", values between 1 and 2 as "medium" and values above 2 as "high". Note, that a further "standardization" could be introduced by defining a relative bioactivity $r_{rel} = r_{MD} / r_{MD,standard}$ that is related to an arbitrarily chosen "standard" cyclotide membrane system. All r_{MD} rate constants of the cyclotide/membrane systems of this study and their type of evaluation are collected in tabular form in the appendix.

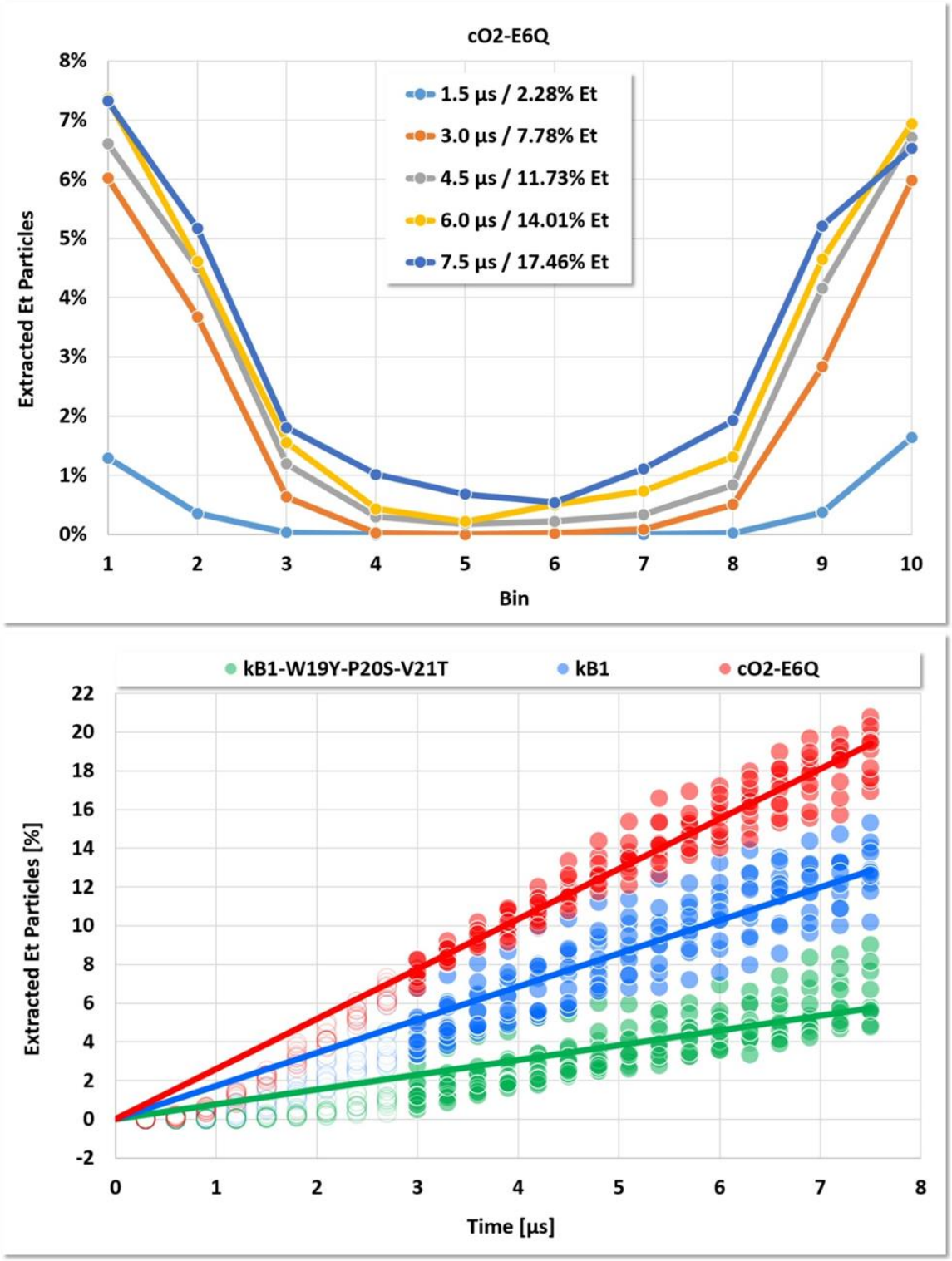


Figure 27. Top: Distribution of extracted Et-particles in the cyclotide/water compartment at different simulation times for quantitative analysis of the membrane-disrupting activity of Cycloviolacin O2 with NoC-PM. Bottom: Total inside-bins-2-to-9 extracted-Et percentages evaluated at every 0.3µs over 7.5µs for 10 simulation runs with different random start geometries for each of the cyclotide/membrane systems kB1-W19Y-P20S-V21T/NoC-PM, kB1/NoC-PM and cO2-E6Q/NoC-PM. The straight regression lines are fixed at the origin with their slopes being fit to the data for simulation times from 3.0 to 7.5 µs.

5.4.11 Precision of the membrane disruption evaluation procedure

The outlined quantitative evaluation procedure consists of a rough, linear averaging of intrinsically nonlinear kinetic processes within a model being dependent on numerous parameter settings. Thus, the obtained r_{MD} values should be regarded to be at the edge of the semi-quantitative and be utilized for trend assessments only. The precision of an r_{MD} value obtained with a single simulation run can be estimated to be around 0.2 units (table 1 provides a snapshot for the Kalata B1 interaction with NoC-PM and 50C-PM with different random start geometries/seeds). Thus, the found r_{MD} value range of 2.5 units for the results of this work can hardly correspond to more than half a dozen different "activity classes". In addition, there is the (rare) possibility for a “sandwich” interaction model with two membranes that the cyclotide interaction with both membranes becomes extremely asymmetrical so that one membrane shows a predominant disruptive interaction with phospholipid/cholesterol extraction (e.g. “Job 2” for Kalata B1 interaction with 50C-PM in table 1). Outliers of this kind can be detected by visual inspection of the simulation progress and the corresponding simulation job should be repeated with a different random start geometry/seed.

All reported r_{MD} values of this study refer to a single simulation run: While averaging of multiple simulation repetitions with different random start geometries/seeds may decrease the random error to any desired precision, a misleading impression could be generated which overestimates the capabilities of the whole approach. In addition, multiple repetitions convert a fast method inevitably into a slower one. For application, the sketched aspects have to be balanced where a single (overnight) simulation run with a visual outlier inspection should be sufficient in most cases.

5.4.12 Kalata B1 membrane interaction

The interaction of Kalata B1 with NoC-PM and 50C-PM exhibits a decreased membrane lipid extraction for the latter (table 1). This finding can be traced to the difference between phospholipid and cholesterol extraction: Whereas the phospholipids of a 50C-PM are extracted equally to a NoC-PM, the cholesterol extraction is significantly reduced. Thus, the extraction of phospholipids is distinctly preferred.

5.4.13 Kalata B1 – hydrophobic patch mutants

The study of cyclotide mutants with specific amino acid replacements may indicate the importance of single or multiple amino acids for the membrane disruption process (table 2). Kalata B1 mutant kB1-W19Y consists of a single amino acid replacement in the "hydrophobic patch" (compare figure 21) where the amino acid tryptophan (W) at position 19 is replaced by the more polar amino acid tyrosine (Y): This single spot mutation already leads to a decreased membrane disruption activity for both plasma membrane types compared to Kalata B1. Multiple polar/hydrophilic mutations of amino acids in the "hydrophobic patch" of Kalata B1 further decrease the membrane disruption activity: Mutant kB1-W19Y-P20S-V21T (with half of the amino acids in the "hydrophobic patch" being replaced) performs even worse and the final kB1-W19Y-P20S-V21T-L27T-P28S-V29T mutant (where the six "hydrophobic patch" amino acids are being completely replaced towards a "hydrophilic patch") does no longer show any significant membrane disruption for both membrane types. The animation at [MFsimClip06] and figure 26 demonstrate the different behaviors: The increasingly polar/hydrophilic transformation of the amino acids in the "hydrophobic patch" still allows for cyclotide oligomerization but leads to a decreased tendency to aggregate into networks: This in turn diminishes the "hydrophobic vicinities" so that the collective lipid extraction abilities are more and more reduced. The importance of the "hydrophobic patch" structure for the bioactivity of Kalata B1 has also been emphasized by experimental studies [Simonsen2008, Huang2010, Henriques2011].

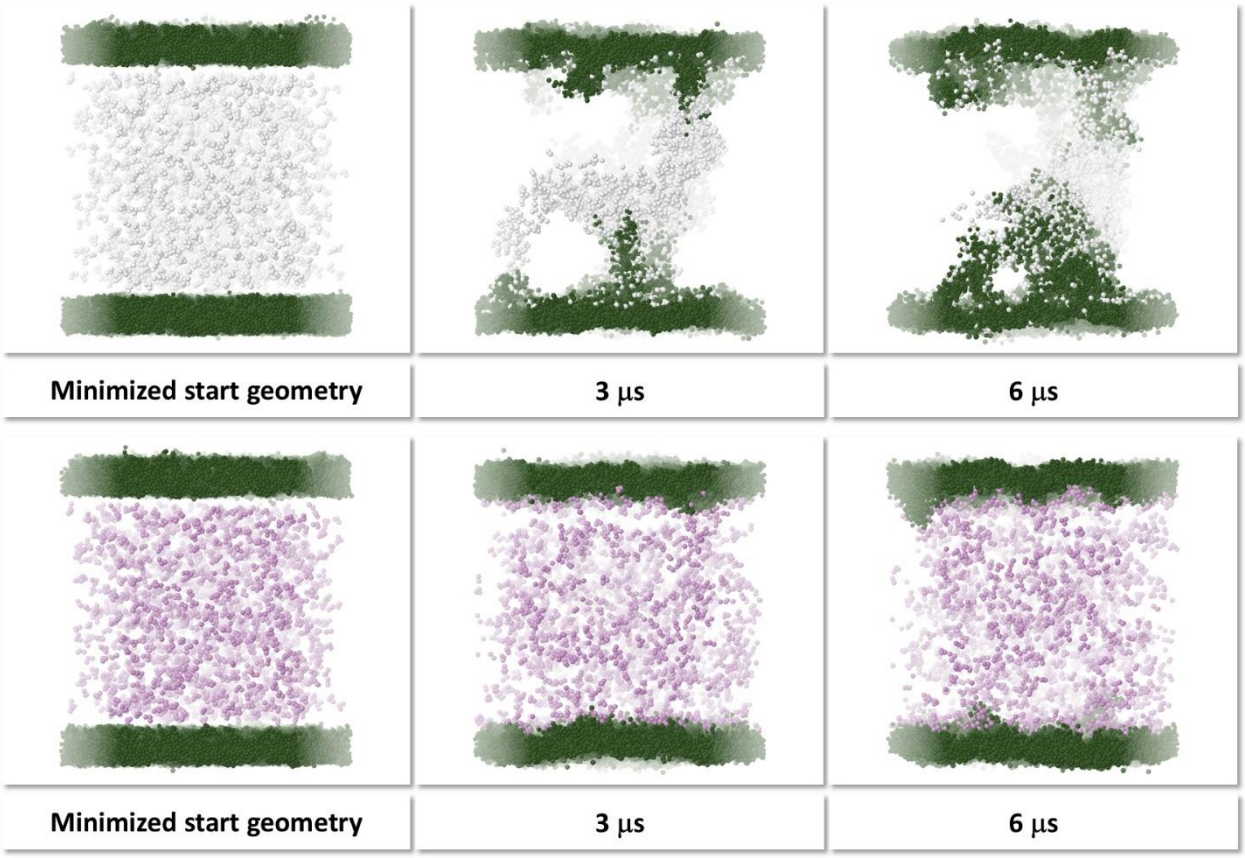


Figure 28. Interaction snapshots of Kalata B1 (upper row) and the "hydrophilic patch" mutant kB1-W19Y-P20S-V21T-L27T-P28S-V29T (lower row) with NoC-PM at different simulation times for the same simulation box cross-section view. The backbone particles of the Kalata B1 "hydrophobic patch" are colored in white (upper row), the mutated backbone particles of the "hydrophobic patch" in plum (lower row). All other cyclotide backbone and sidechain particles are omitted. "Hydrophobic" phospholipid particles are shown in olive. All water particles are omitted.

5.4.14 Kalata B1 – differently charged mutants

Kalata B1 contains two charged amino acids (compare figure 21, table 3 provides an overview of the charge state of the studied cyclotides): Negatively charged glutamic acid (E) at position 3 and positively charged arginine (R) at position 24. Mutant kB1-E3Q replaces glutamic acid with uncharged glutamine (Q) to obtain a net charge of +1 which leads to an increased membrane disruption with NoC-PM but a decreased disruption with 50C-PM (table 4). The "opposite" kB1-R24W mutant with apolar tryptophan at positively charged arginine (R) position 24 and net charge of -1 is more bioactive for NoC-PM but compares to native Kalata B1 for 50C-PM. Interestingly, the uncharged "double mutant" kB1-E3Q-R24W performs slightly better than mutants kB1-E3Q and kB1-R24W in NoC-PM disrupting activity and compares to kB1-R24W and native Kalata B1 for 50C-PM. These results may not be satisfactorily summarized with simple additive tendencies but indicate a more intricate balance of differential contributions.

In Kalata B1 mutant kB1-W19K the amino acid tryptophan (W) at position 19 is not only replaced with the more polar amino acid tyrosine (Y) in mutant kB1-W19Y but with the positively charged amino acid lysine (K): For NoC-PM and 50C-PM this leads to a membrane disrupting activity distinctly below Kalata B1 and even slightly below mutant kB1-W19Y. In [Henriques2011] mutant kB1-W19K (denoted "W23K" in this publication) was experimentally studied in different membrane systems and characterized to be "inactive". In Kalata B1 mutant kB1-T16K-N25K two amino acids are replaced with positively charged lysine (K): Compared to native Kalata B1 these mutations lead to an increased membrane disrupting activity for NoC-PM and a comparable one for 50C-PM. The experimental study of this mutant with different membrane systems leads to a "more potent" characterization in comparison to Kalata B1 [Henriques2011] (where this mutant is denoted T20K/N29K).

5.4.15 Cycloviolacin O2 membrane interaction

In comparison to Kalata B1 the Cycloviolacin O2 cyclotide exhibits a slightly enhanced membrane-disrupting activity for NoC-PM and 50C-PM – with again a distinct preference of phospholipid over cholesterol extraction (see table 5): This slight enhancement is not primarily deduced from their difference in r_{MD} values (cO2: 1.5/1.1, kB1: 1.3/0.9 – these differences alone would be assessed as comparable) but indicated by the mixture study described below (figure 28 exhibits slightly positive slopes towards higher cO2 concentrations for both membranes). The experimental study in [Burman2011] characterizes Cycloviolacin O2 as the "most potent cytotoxic and antimicrobial cyclotide" in comparison to the Möbius cyclotides Kalata B1 and Kalata B2, in [Pränting2010] Cycloviolacin O2 is "the most active cyclotide" compared to the Möbius ones.

5.4.16 Cycloviolacin O2 – hydrophobic patch mutants

A mutation of the six Cycloviolacin O2 amino acids in the "hydrophobic patch" analog to the Kalata B1 "hydrophobic patch" mutations leads to a corresponding decrease in bioactivity for the 50C-PM (see table 5). Singly mutated cO2-W10Y is already less active than Cycloviolacin O2 with triply mutated cO2-W10Y-I11T-P12S exhibiting a further decreased activity up to the inactive "hydrophilic patch" mutant cO2-W10Y-I11T-P12S-I14T-A17S-I18T with complete "hydrophobic patch" replacement. For NoC-PM the

Cycloviolacin O2 mutants behave differently compared to their Kalata B1 analogs: Whereas cO2-W10Y shows a decreased activity compared to Cycloviolacin O2 the triply mutated cO2-W10Y-I11T-P12S becomes even slightly more active. The final "hydrophilic patch" mutant cO2-W10Y-I11T-P12S-I14T-A17S-I18T exhibits the smallest but still distinct lipid extraction activity. These findings do not only indicate differences between the two cyclotide families but also "non-linear" contributions regarding successive amino acid mutations in a designated surface area like the "hydrophobic patch".

5.4.17 Cycloviolacin O2 – differently charged mutants

In mutant cO2-E6Q the negatively charged glutamic acid (E) at position 6 is replaced with the uncharged amino acid glutamine (Q) so that a total cyclotide net charge of +3 results. This mutant exhibits the highest membrane disrupting activity of all studied cyclotides for NoC-PM and is among the highest for 50C-PM (see table 6) which agrees with experimental findings in [Burman2011] for the interaction of DOPC/DOPA liposomes with a chemically modified cO2 derivate where the glutamic acid was esterified by acetyl chloride – while the opposite is reported for other membrane types [Herrmann2006, Pränting2010, Burman2011b].

The electrostatically inverted mutant cO2-K23Q-K25Q-R29W has all positively charged amino acids replaced with uncharged alternatives so that a total cyclotide net charge of -1 results. For NoC-PM and 50C-PM this mutant performs slightly below the activity of native Cycloviolacin O2 (see table 6) which agrees with experimental results in [Herrmann2006].

Mutants cO2-K23Q-K25Q and cO2-E6Q-K23Q-K25Q-R29W with zero net charge behave similarly for both membrane types and show a decreased membrane disrupting activity compared to Cycloviolacin O2 where the decrease for 50C-PM is more pronounced (see table 6). Again, there are no simple tendencies regarding the influence of charge – a finding that is also indicated by experimental studies [Burman2011].

5.4.18 Cyclotide activity and membrane composition

The influence of the membrane composition on the cyclotide activity is exemplarily studied by cholesterol-free pure DMPC (NoC-DMPC-M) and cholesterol-free pure DOPE membranes (NoC-DOPE-M) as well as their 50% cholesterol analogs (50C-DMPC-M and

50C-DOPE-M) with the two native cyclotide types (see table 7). For NoC-DMPC-M Kalata B1 and Cycloviolacin O2 exhibit activities comparable to NoC-PM. In interaction with NoC-DOPE-M both cyclotides considerably increase their activity. The replacement of phospholipids with cholesterol leads to an overall significantly decreased membrane disrupting activity of both cyclotide types in comparison to the cholesterol-free pure DMPC and DOPE membranes. In comparison to 50C-PM, the 50C-DMPC-M are less sensitive to Kalata B1 attacks with reduced cholesterol and DMPC-specific extractions whereas the Cycloviolacin O2 activity is comparable to 50C-PM. The membrane disrupting activity of both cyclotide types with 50C-DOPE-M is similar to 50C-PM. An increased membrane-disrupting activity with PE-rich membranes was also shown in experimental setups [Henriques2012].

Since the phospholipid composition of the inner and outer leaflet of the plasma membrane differs, an inversion of the plasma membrane (the outer leaflet becomes the inner leaflet and vice versa) on the cyclotide activity is studied with the two native cyclotide types so that they primarily interact with the inner membrane leaflets instead of the outer ones. For cholesterol-free inverted plasma membranes (NoC-invPM) Kalata B1 and Cycloviolacin O2 exhibit a considerably increased activity compared to the non-inverted ones: The membrane disruption is the highest detected and compares to pure NoC-DOPE-M. The increased activity with the inner membrane leaflet may be partly attributed to the higher (77.5 to 22.5) DOPE concentration of the inner leaflet which is in agreement with the detected increased membrane disrupting activity with PE-rich membranes above. The replacement of phospholipids with cholesterol (50C-invPM) again leads to an overall considerably decreased membrane disrupting activity of both cyclotide types in comparison to the cholesterol-free inverted plasma membranes where the Kalata B1 activity and the Cycloviolacin O2 activity match the activity for the non-inverted plasma membranes and 50C-DOPE-M.

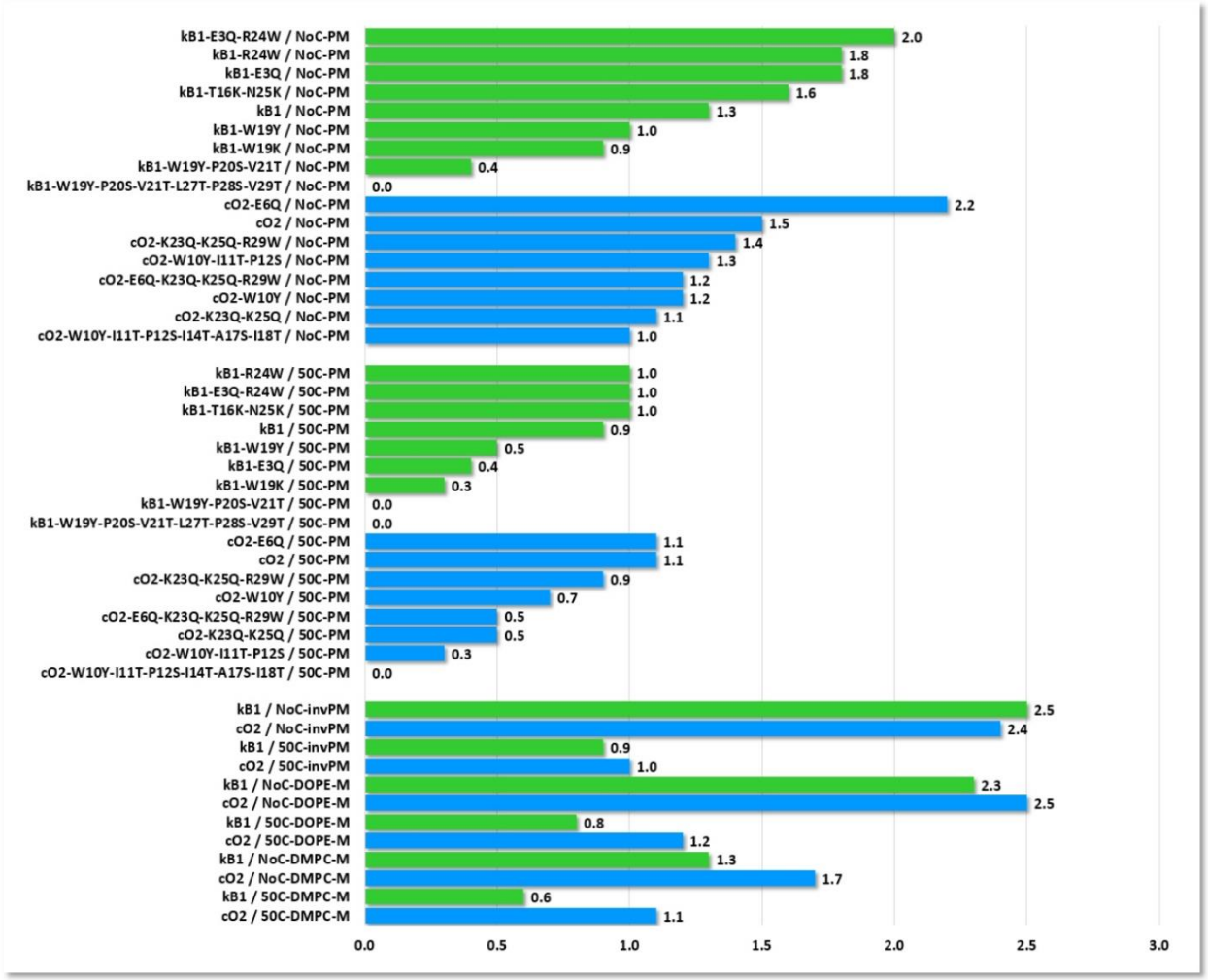


Figure 29. Rate constants r_{MD} for membrane disruption (averaged percentage of extracted Et particles per microsecond) of studied cyclotide/membrane systems. Kalata B1 related results are illustrated with green bars, blue bars refer to Cycloviolacin O2 correspondingly.

5.4.19 Summary of previous findings

Figure 27 summarizes the findings of this work: The preferred phospholipid over cholesterol extraction for the 50% cholesterol membranes is found for all studied cyclotides and membrane systems. A decrease of cyclotide membrane disrupting activity due to cholesterol was also experimentally observed [Burman2011] as well as detected by coarse-grained MD simulations [Nawae2017]. The “hydrophobic patch” is an important surface area for membrane disruption of both cyclotide types where successive “hydrophilic” amino acid replacements lead to decreased activities up to inactivity but do not necessarily follow a simple linear trend which is in agreement with experimental findings [Simonsen2008, Huang2010, Henriques2011]. Replacements of/with charged amino acids caused considerable changes in activity including super-active mutants but without any simple patterns. The same holds true for the influence of the total net charge.

These charge-related observations are in line with experimental results [Herrmann2006, Pränting2010, Henriques2011, Burman2011, Burman2011b]. The phospholipid membrane composition significantly influences the disrupting activity with a trend towards higher activities for phosphatidylethanolamine-rich membranes which was experimentally shown in [Henriques2011] [Henriques2012, Wang2012, Henriques2012b, Henriques2014, Cranfield2017, Grage2017].

5.4.20 Additivity of cyclotide mixtures

Figure 28 demonstrates the influence of kB1-W19Y-P20S-V21T/cO2-E6Q cyclotide mixtures with NoC-PM on the membrane disrupting activity which suggests a linear additivity without significant sub- or over-additive effects which is in agreement with experimental findings [Burman2011b]. Note, that an analysis of cyclotide mixtures may also be utilized for comparative assessments of cyclotide/membrane systems with similar bioactivity with the sign of the regression line slope as a threshold for attributing lower (negative sign) or higher activity (positive sign) to the cyclotide with an increasing fraction.

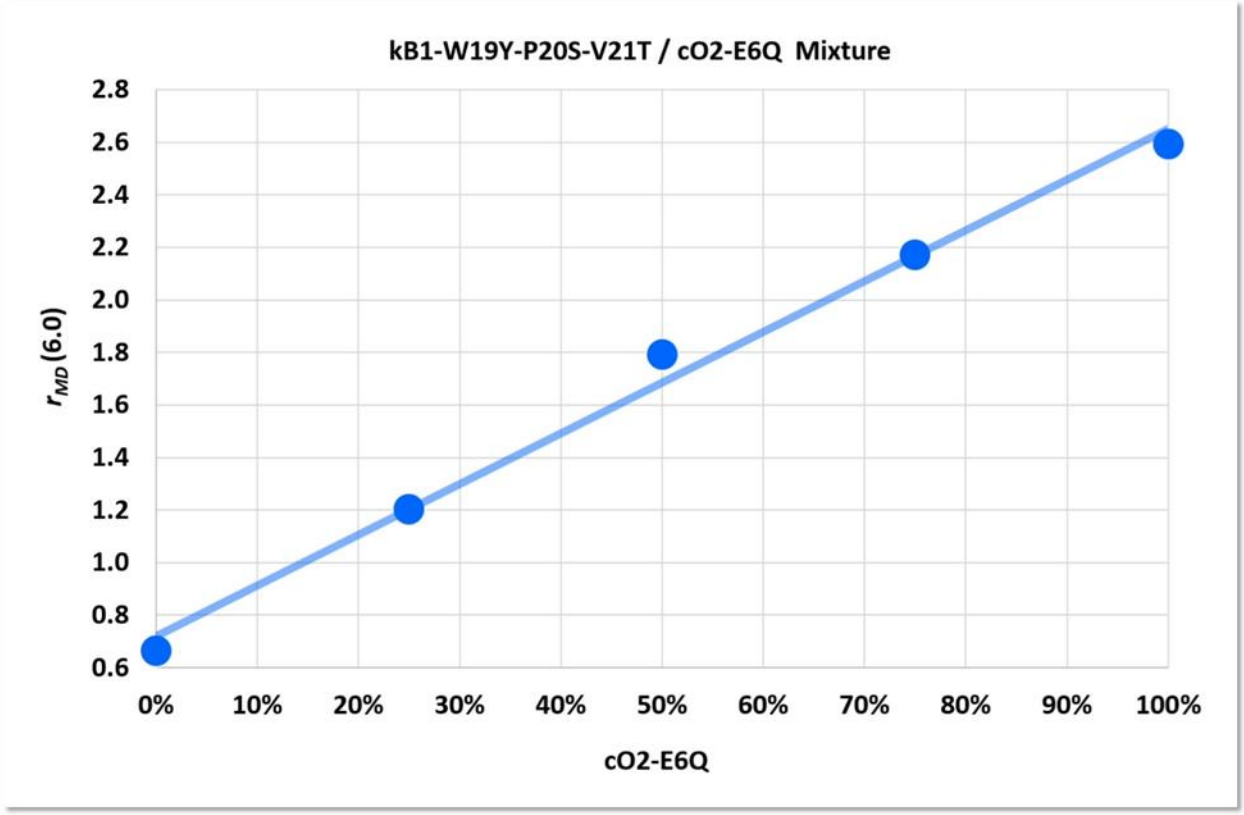


Figure 30. Rate constants $r_{MD}(6.0)$ for membrane disruption of different kB1-W19Y-P20S-V21T/ cO2-E6Q mixtures with NoC-PM. The coefficient of determination (R^2) for the sketched linear regression line is above 0.99.

5.4.21 Pore formation

The role of a possible pore formation for the membrane disruption process caused by cyclotides is still a contentious issue [Lindholm2002, Herrmann2006, Shenkarev2006, Svangard2007, Wang2009, Huang2009, Gerlach2010, Huang2010, Wimley2010, Burman2011, Henriques2011, Wimley2011, Henriques2012, Wang2012, Henriques2012b, Göransson2012, Burman2014, Nawae2014, Nawae2014b, Truszkowski2015, Nawae2017, Cranfield2017, Grage2017]. The pore formation described by mesoscopic simulation in [Truszkowski2015] should be regarded as a qualitative snapshot only, which focused on a pore's molecular composition and not its stability. Subsequent simulation trials for construction of stable pores with mesoscopic simulation have not been successful: Whereas pores like those described in [Truszkowski2015] could be generated, they always exhibited continuous growth without any plateau of stability.

The simulation models with ApL 40 A^2 membranes studied in this work also do not exhibit any membrane pores due to cyclotide interaction over the studied time span – a

single exception was found for mutant cO2-W10Y with 50C-PM which formed a small growing pore after 5 µs. A "sandwich" interaction model of Kalata B1 with a thinner ApL 60 Å^2 NoC-PM membrane showed a larger non-stable and continuously growing toroidal [Wang2012] membrane pore (see animation at [MFsimClip03] and figure 29). Thus, a hint for stable pore formation with a defined diameter could not be found.

Combining the findings in [Truszkowski2015], the sketched subsequent investigations and the findings of this work, it may be deduced that stable cyclotide-induced pores either do not exist or that a mesoscopic approach principally fails to describe the directed atomistic interactions that stabilize them. It should be noted that the question of pore stability by no means excludes a pore formation effect that contributes to membrane disruption – membrane pores would merely be an intermediate dynamic phenomenon.

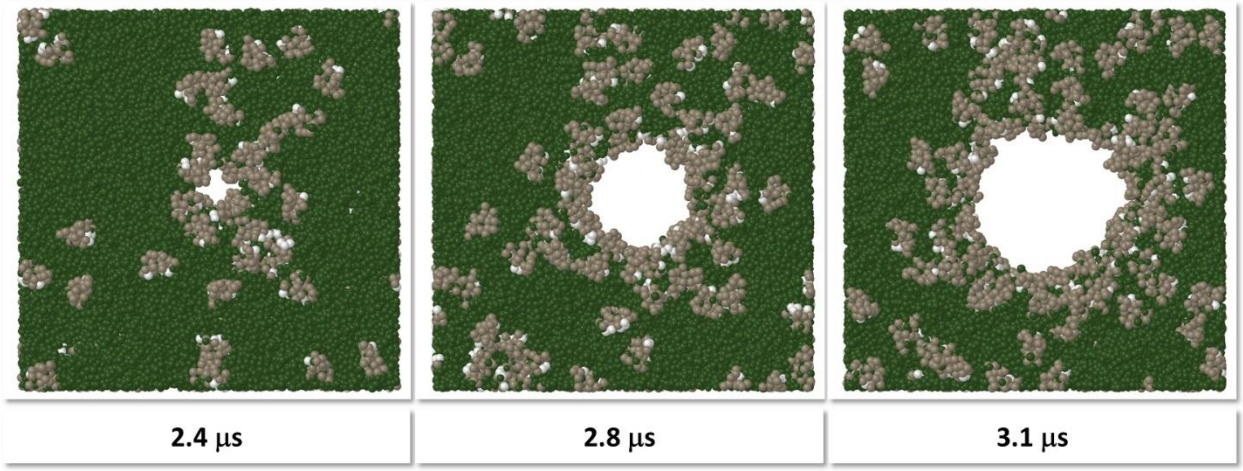


Figure 31. Snapshots of a pore formation for cyclotide/membrane interaction of Kalata B1 with cholesterol-free ApL 60 Å^2 plasma membranes at different simulation times (bottom view of simulation box). The backbone particles of the Kalata B1 "hydrophobic patch" are colored in white, all other backbone particles are shown in beige. "Hydrophobic" phospholipid particles are shown in olive. All water particles are omitted.

6 Discussion

6.1 SPICES molecular structure representation

This work provides a Java library for SPICES handling and mesoscopic simulation support (*Spices.jar*) in combination with a connection library (*SpicesToGraphStream.jar*) and a Java Graphical User Interface (GUI) viewer application (*SpicesViewer.jar*) for visual topological inspection and manipulation of SPICES molecule definitions [SPICESRepository2020]. All libraries/applications are publicly available as open source published under the GNU General Public License version 3 [GNU-GPL2020]. The SPICES GitHub repository contains the Java bytecode libraries, a Windows OS installer for the *SpicesViewer* GUI application, all Javadoc HTML documentations [Javadoc2020] and the Netbeans [NetBeans2020] source code packages including Unit tests.

The presented set of methods may alleviate molecular structure definitions for mesoscopic simulation tasks. The *SpicesViewer* GUI application demonstrates relevant use cases in detail with corresponding sample code. The new libraries may be utilized within scripting environments or become part of integrated mesoscopic simulation systems.

Future developments may address SPICES parsers that especially support the more difficult preparation of polymer systems, e.g. a PDB-to-SPICES parser for peptides and proteins provided in form of PDB files (actually, the SPICES string of the Kalata B1 peptide in figure 7 was generated from its PDB file with a prototype parser that uses the amino acid fragmentation schemes and connection rules outlined in [Truszkowski2015]). Another promising challenge would be a conversion between particle and all-atom representations for an interplay of atomistic and mesoscopic simulation.

6.2 Mesoscopic simulation kernel Jdpd

Jdpd is a new open DPD simulation kernel completely written in Java that complements the small available set of general purpose DPD kernels [JdpdRepository2020]. It especially supports molecular fragment structures and offers parallelizable force calculation plus efficient caching options with an interface and factory-pattern driven design for comfortable and low expenditure code extensions, customizations or replacements. Detailed input/output communication, parallelization and process control as well as internal logging capabilities for debugging purposes are supported. The new kernel may be utilized in different simulation environments ranging from flexible scripting solutions up to fully integrated “all-in-one”

simulation systems described in [Truszkowski2014] where it may help to avoid polyglot programming.

The Jdpd library [JdpdRepository2020] uses the Apache Commons RNG libraries [ApacheRNG2020] and is publically available as open source published under the GNU General Public License version 3 [GNU-GPL2020]. The Jdpd repository on GitHub comprises the Java bytecode libraries (including the Apache Commons RNG libraries), the Javadoc HTML documentation [Javadoc2020] and the Netbeans [NetBeans2020] source code packages including Unit tests.

6.3 Mesoscopic simulation environment MFsim

With the MFsim project, an open simulation environment for mesoscopic simulation is provided with the default Jdpd simulation kernel for Dissipative Particle Dynamics [MFsimRepository2020]. MFsim supports polymer and especially biomolecular simulations containing peptides and proteins aiming at pushing the mesoscopic simulation frontiers towards these areas of research which often require the study of large systems on the microsecond time scale. As an open rich-client all-in-one approach MFsim targets theoretical, computational as well as end-user scientists (without programming skills), thus it contributes to making computational tools more widespread in the scientific community. For the chemo- or bioinformatician the project may serve as a starting point for specific customizations where the reusable functionality may outweigh the initial training hurdle.

MFsim is publicly available as open source published under the GNU General Public License version 3 [GNU-GPL2020]. The MFsim GitHub repository contains all Java bytecode libraries, a Windows OS installer and a corresponding installation tutorial [MFsimPDF02, MFsimClip05], Javadoc HTML documentations [Javadoc2020] and the Netbeans [Netbeans2020] source code packages including Unit tests. A growing number of tutorials that outline more specific features and use-cases as well as an MFsim based biomolecular research study concerning the interaction of cyclotides and bilayer membranes are in preparation.

6.4 Cyclotide/membrane interactions

The interactions of cyclotides and bilayer membranes span a complex bioactivity landscape with multiple contributing factors which may act in opposed directions. In this work an attempt was made to quantify a cyclotide mode-of-action for bilayer membrane

disruption in form of membrane lipid extraction by studying a “sandwich” interaction model with mesoscopic simulation at the microsecond timescale to obtain hints and tendencies for structure-activity relationships. The reported simulation results agree with experimentally observed trends with the virtue of being compressible into simple numbers that allow for comparison and ranking tasks. Nevertheless, the approach may be regarded as a start only that requires substantial future improvements.

From a “bottom-up” DPD simulation point of view, more elaborated biomolecular particle sets with more realistic particle-particle repulsions in accordance with improved biomolecular fragmentation schemes are required: The approach may be regarded as an initial series of deliberate choices where every single part (the definition of appropriate particles, the coordination number estimation procedure, the particle-particle interaction energy averaging method or the chosen force field) is in need of improvement. As an example, progress could be achieved with open evaluations that systematically study the influence of different force fields/water models on the resulting mesoscopic particle-particle repulsions since there is clear evidence that improved water models improve atomistic biomolecular simulations (e.g. [Piana2015]). Unfortunately, a quantum-mechanical estimation of the non-binding particle-particle interactions with the most accurate available DFT-SAPT calculations is still prohibitive. With regard to the proposed cyclotide/membrane model, additional alternative interaction models should be explored and comparatively studied. However, a principal limit of the appropriateness of mesoscopic approaches should always be taken seriously: Whenever the cyclotide/membrane disruption is driven by specific atomic interactions that cannot be adequately abstracted in mesoscopic particle-particle interaction averages, the approach will inevitably fail and – at worst – be misleading. The combined efforts of improved simulation setups and corresponding experimental studies may finally lead to a proper assessment of the possibilities as well as the limits of mesoscopic approaches.

An often undervalued feature of mesoscopic simulations with open rich-client end-user systems is their speed in combination with their ease of use (compare the animated tutorial for setup and evaluation of the outlined “sandwich” interaction model with MFsim [MFsimPDF03, MFsimClip10]): Presumed that a suitable simulation model is well established, simulation studies and lab experiments can be performed in a concurrent manner by the experimental scientists themselves since there are no extravagant hardware devices nor specific programming skills required. The open-source simulation system additionally allows for its general availability as well as continuous improvements.

7 Appendices

7.1 Appendix I – SPICES syntax rules

SPICES monomer and molecular structures are constructed according to the following syntax rules:

1) Particle names consist of a maximum of 10 characters (a-z, A-Z, 0-9, the first character is not allowed to be a digit and must be upper case) and an optionally prefixed frequency number.

2) Particle names in molecular structures (but NOT monomers) may be followed by a backbone label (i.e. a number between apostrophe characters, e.g. '1', '2' etc.) for later definition of spring forces between particle pairs where particles must be labeled in a consecutive manner.

3) The connection character '-' is used for bonding between particles.

4) Round brackets '(' and ')' indicate branches. They may be nested for arbitrary levels of branches.

5) Square brackets '[' and ']' with an enclosed number which follow a particle indicate a ring closure. Multiple bonds between two particles are counted only once.

6) Curly brackets '{' and '}' include a monomer definition. Monomers are defined as molecular structures but must contain at least 1 particle with a [HEAD] and [TAIL] attribute: Structure elements that precede the monomer connect to the HEAD particle, structure elements that follow the monomer connect to the TAIL particle. Monomers are not allowed to be nested and backbone labels are forbidden in monomers.

7) Monomer labels start with a '#' character followed by a sequence of characters (first character is not allowed to be a digit and must be upper case).

8) Monomer labels may be preceded by a frequency number (to construct polymers).

9) A molecule may consist of multiple independent parts (i.e. parts are not allowed to be connected in any way). Each part must be framed by angle brackets '<' and '>'. Parts are not allowed to be nested. Monomers are not allowed to contain parts. A part may have an optional prefixed frequency number.

10) A particle (that is not within a monomer) may optionally contain a [START] or an [END] tag which may be used for orientation purposes. There is only one [START]/[END] pair allowed per independent part.

Comments:

"A-B-C" defines a connection of particle A with particle B and particle B with particle C.

"3A-B" is a shortcut notation for "A-A-A-B".

"A-2B(E-F)-D" is identical to "A-B-B(E-F)-D", "3A(B)-D" is a shortcut for "A-A-A(B)-D". The shorter initial string should be preferred in both cases.

"A-B(D-E)-F" defines a main chain "A-B-F" with a side chain "D-E" where particle D is connected to particle B.

"A-B[1]-C-C-C-D-E[1]" defines a ring closure between particles B and E.

"A[1]-B[1]" is equal to "A-B", "A[1][2]-B-C-D[1][2]" is equal to "A[1]-B-C-D[1]": Multiple bonds between two particles are counted only once.

"A-B(D-E(G-H[1])-F)-I-A-K[1]-B" defines a main chain "A-B-I-A-K-B" with a side chain "D-E-F" (connected to particle B of the main chain) and another side chain "G-H" (connected to particle E of the first side chain). In addition there is a ring closure between particle H of the second side chain and particle K of the main chain.

"3A[1]-B-B-C[1]" is a shortcut for "A-A-A[1]-B-B-C[1]".

"A'1'-B-C-D-E'2'" has two backbone particles. Note that backbone particles must be labeled in a consecutive manner, i.e. "A'1'-B-C-D-E'3'" or "A'1'-B-C-D-E'1'" are forbidden, but "A'1'-B-C'3'-D-E'2'" is valid.

The shorter (preferred) string "3A'1'-B-C-D-E'2'" is identical to "A-A-A'1'-B-C-D-E'2'".

Multiple ring closures at one particle are marked by successive use of ring-closure brackets, e.g. particle B in "A-B[1][2]-4C-D[1]-4C-E[2]" is connected to particles D and E.

The simplest structure of a monomer consists of a single particle A with attributes [HEAD] and [TAIL], i.e. "{A[HEAD][TAIL]}".

In "E-#MyMonomer-F" with #MyMonomer equal to "{A[HEAD]-B-C[TAIL]-D}" particle E is connected to particle A (the head) of the monomer and particle C (the tail) of the monomer is connected to particle F. This definition is equivalent to "E-A-B-C(D)-F".

"2{A[HEAD]-B-C[TAIL]-D}" defines a structure of 2 monomers where particle A (the head) of the second monomer is connected to particle C (the tail) of the preceding first monomer. This definition is equivalent to "A-B-C(D)-A-B-C-D".

Definitions like "{A[HEAD]-{A[HEAD]-B-B[TAIL]-C}-B[TAIL]-C}" with nested monomers are forbidden.

"A[START]-B-C[END]" defines orientation information.

"A[START][END]-B-C" is syntactically correct but makes no sense.

"A[START]-B[START]-C[END]" is forbidden: There is only one [START]/[END] pair allowed per structure.

The shorter (preferred) string "3A[START]-B-C[END]" is identical to "A-A-A[START]-B-C[END]".

"<A-B-C><A-D>" defines a molecule which consists of two independent parts "A-B-C" and "A-D".

"<A-B[1]-C><A-D[1]>" is forbidden since parts are not allowed to be connected in any way. The correct definition in this case would be "(A-B[1]-C)(A-D[1])" or "A-B(C)-D-A".

"3<A-B>" is equal to "<A-B><A-B><A-B>".

7.2 Appendix II – DPD conversion formulas

The appendix comprises conversion formulas between physical and DPD units [Groot1997, Groot1998, Groot2003] as well as formulas for molecule concentration related calculations implemented in MFsim.

The conversion between DPD lengths and physical lengths is based on the conversion radius r_c ("radius of interaction") in physical units

$$r_c = \sqrt[3]{V_{\min}\ \rho_{DPD}\ \frac{\sum_{i=1}^{N_{particles}} N_{particle,i}\ \frac{V_{particle,i}}{V_{\min}}}{\sum_{i=1}^{N_{particles}} N_{particle,i}}}$$

$$l_{phys} = l_{DPD}\ r_c$$

$V_{\min}$, volume of smallest particle in physical units; ρ_{DPD}, DPD (number) density; $N_{particles}$, number of different particle types; $N_{particle,i}$, number of particles of type i; $V_{particle,i}$, volume of particle of type i in physical units; l_{phys}, length in physical units; l_{DPD}, length in DPD units.

with the conversion between DPD time and physical time being approximated by

$$t_{phys} = t_{DPD}\ f_{soft}\ r_c \sqrt{\frac{1}{R\,T}\ \frac{\sum_{i=1}^{N_{particles}} N_{particle,i}\ M_{particle,i}}{\sum_{i=1}^{N_{particles}} N_{particle,i}}}$$

t_{phys}, time in physical units; t_{DPD}, time in DPD units; f_{soft}, factor for increased particle diffusivity due to soft potentials ($f_{soft} \approx 1000$); R, gas constant; T, thermodynamic temperature; $M_{particle,i}$, molar mass of particle of type i.

Molecule concentration calculations are based on the relations

$$N_{molecule,1}^{scaled} = \frac{\rho_{DPD} V_{box,DPD}}{\sum_{k=1}^{N_{particles}} N_{1k} + \sum_{i=2}^{N_{molecules}} \left(\frac{n_{molecule,i}^{scaled}}{n_{molecule,1}^{scaled}} \sum_{k=1}^{N_{particles}} N_{ik} \right)} \quad ; \quad N_{molecule,i\neq 1}^{scaled} = N_{molecule,1}^{scaled} \frac{n_{molecule,i\neq 1}^{scaled}}{n_{molecule,1}^{scaled}}$$

$$n_{molecule,i}^{scaled} = \frac{n_{molecule,i}\ s_{molecule,i}}{\sum_{j=1}^{N_{molecules}} n_{molecule,j}\ s_{molecule,j}}$$

$$s_{molecule,i} = \frac{\sum_{k=1}^{N_{particles}} N_{ik} \frac{V_{particle,k}}{V_{\min}}}{\sum_{k=1}^{N_{particles}} N_{ik}}$$

$$n_{molecule,i} = \frac{\frac{w_{molecule,i}}{\sum_{k=1}^{N_{particles}} N_{ik}\ M_{particle,k}}}{\sum_{j=1}^{N_{molecules}} \left(\frac{w_{molecule,j}}{\sum_{k=1}^{N_{particles}} N_{ik}\ M_{particle,k}} \right)}$$

$$w_{molecule,i} = \frac{N_{molecule,i} \sum_{k=1}^{N_{particles}} N_{ik}\ M_{particle,k}}{\sum_{j=1}^{N_{molecules}} \left(N_{molecule,j} \sum_{k=1}^{N_{particles}} N_{jk}\ M_{particle,k} \right)}$$

$N_{molecule,i}^{scaled}$, Number of molecules of type i in simulation box; $V_{box,DPD}$, volume of simulation box in DPD units; N_{ik}, number of particles of type k in single molecule of type i; $N_{molecules}$, number of different molecule types; $n_{molecule,i}^{scaled}$, volume-scaled relative number of molecules of type i; $n_{molecule,i}$, relative number of molecules of type i; $s_{molecule,i}$, volume scaling

factor of molecule of type i; $w_{molecule,i}$, relative weight of molecule of type i; $N_{molecule,i}$, number of molecules of type i.

7.3 Appendix III – Data objects, preferences and re-usable settings

All data items for GUI display are stored in *ValueItem* objects (package *model.valueItem*) which can be configured for data structures like scalars, vectors or matrices (see enumeration *ValueItemEnumBasicType* in package *model.valueItem*) that consist of basic data types like texts, numbers, directory paths or links to specific dialogs (see enumeration *ValueItemEnumDataType* in package *model.valueItem*). A *ValueItem* object may contain default values as well as specific data checks like min/max boundaries for numeric values, selection texts or regular expressions for control of textual input in order to prevent input errors. Related *ValueItem* objects may be encapsulated by a *ValueItemContainer* object (package *model.valueItem*) which itself may be used for a tree view of its contents via GUI display. Moreover, a *ValueItemContainer* object allows for an interplay of its *ValueItem* objects like directed update cascades after value changes (e.g. see class *UtilityJobUpdate* in package *model.job* for update cascades concerning *Job Input* definitions). *ValueItem* and *ValueItemContainer* objects can be made persistent via XML serialization.

Global preferences are realized by a modifiable *Preferences* singleton (package *model.preference*) which provides adequate *ValueItem* objects for all settings. The global preferences dialog is available via menu entry *Application/Preferences/Edit* of the basic GUI frame.

A particular feature for efficient usage is the implemented table-data schema management. A table-data schema is a pattern of values for vectors or matrices of *ValueItem* objects which can be named and internally stored for reuse: Schemes alleviate complex setting operations e.g. for the graphical display of multiple particles. A table-data schema manager is available via menu entry *Application/Schemata/Manage* of the basic GUI frame and provides view, edit, remove or clear functions with an additional file export of schemes lists for exchange. Persistent schemata files may be reloaded or merged.

The definition of *ValueItem*-related vectors or matrices is supported by bulk functions which can be refined by column-based filters. Thus, the tedious input of multiple values (e.g. for force constants or colors) may be realized by a few manual operations.

7.4 Appendix IV – Simulation kernel integration

Jdpd is integrated as a command text file driven simulation kernel: All settings are comprised in an *Input.txt* text file with references to compound related *PositionsBonds<index>.txt* text files which contain all initial particle xyz positions and particle-particle spring-force definitions for all molecular species. During simulation Jdpd produces (partly compressed) output text files in form of simple xy table data for e.g. relevant physical quantities versus simulation step (like *T.txt* for temperature, *UpotDpd.txt* for conservative DPD energy etc.), particle position files which provide all particle positions at a specific simulation step in the simulation box (e.g. *PP2400.gz* for particle positions of simulation step 2400) as well as structurally more complex text files like *M_M_TUPLE.txt* for nearest-neighbor data. Thus, an integration of Jdpd requires the automated generation of the necessary input text files, the programmatic control of Jdpd during simulation (with intermediate simulation progress inspections or possible exception/error handling) and the automated analysis of the simulation record contained in the Jdpd-generated output text files: The code for these tasks is mainly located in package *model.job* for input/output text file generation/analysis and in class *MainFrameController* (section *Job execution related command methods*) in package *gui.main* for Jdpd control during simulation.

Class *JdpdValueItemDefinition* (package *model.job*) defines in method *initializeJdpdValueItems()* the *ValueItem* instances for all job input settings (MFsim stores data in *ValueItem* objects, see previous appendix) which are collected in a *ValueItemContainer* instance. This job input *ValueItemContainer* instance is then modified during job design with a cascaded top-down interplay of the *ValueItem* instances coded in class *JobUpdateUtils* (package *model.job*) to prevent later ill-defined settings. A completed job design results in a corresponding job input *ValueItemContainer* instance (which is then made persistent via XML file output). Before activation (instantiation) of a Jdpd kernel instance the information of a job input *ValueItemContainer* instance is parsed to Jdpd input and position/bonds text files via method *getJdpdInputText()* of class *JobUtilityMethods* (package *model.job*). Then a Jdpd kernel instance is created with the generated text files as its main input and submitted to a thread-pool for concurrent execution (methods *startJobsInJobExecutionQueue()* and *startRemainingJobExecutionTasks()* in class *MainFrameController*, package *gui.main*). During as well as after simulation the Jdpd generated simulation record is mapped back

to corresponding *ValueItem* instances (collected in a job result *ValueItemContainer* instance) via method *getResultValueItemContainerForJobResult()* of class *JobResult* (package *model.job*) which are visualized by MFsim for job result analysis.

To integrate an alternative simulation kernel software the sketched classes and methods have to be customized accordingly. In addition, the kernel control structure has to be adjusted for polyglot programming since scientific kernel software is usually written in languages like C/C++ or FORTRAN and (ahead-of-time) compiled to executables. To realize the integration task the elaborated capabilities of the Java platform can be utilized. Thus, from a software development point of view, the integration into the MFsim environment allows for an extremely flexible response to a wide range of requirements – where the existing Jdpd integration code may serve as a productive blueprint. It can be estimated from experience with previous integration tasks that – a skilled software developer presumed – a simulation kernel integration requires at least several weeks (for comparatively simple and less demanding solutions) up to several months (for more complex, comprehensive and thoroughly safe-guarded solutions like the Jdpd integration).

7.5 Appendix V – Tutorial MFsim – Cyclotide-membrane sandwich interaction model

This tutorial demonstrates details of the preparation-simulation-evaluation triad of a cyclotide/membrane sandwich interaction model (compare to figure 22). It consists of two plasma bilayer membranes enclosing a cyclotide/water compartment with the cyclotide Kalata B1 (kB1) single-spot mutant kB1-W19Y. This mutant exhibits a replacement of amino acid tryptophan (W) at position 19 by amino acid tyrosine (Y). Please not that some of the following screenshots were truncated due to saving space.

Start MFsim and select the menu item ***Help / MFsim data / Browse*** to browse the MFsim data directory (figure 30).

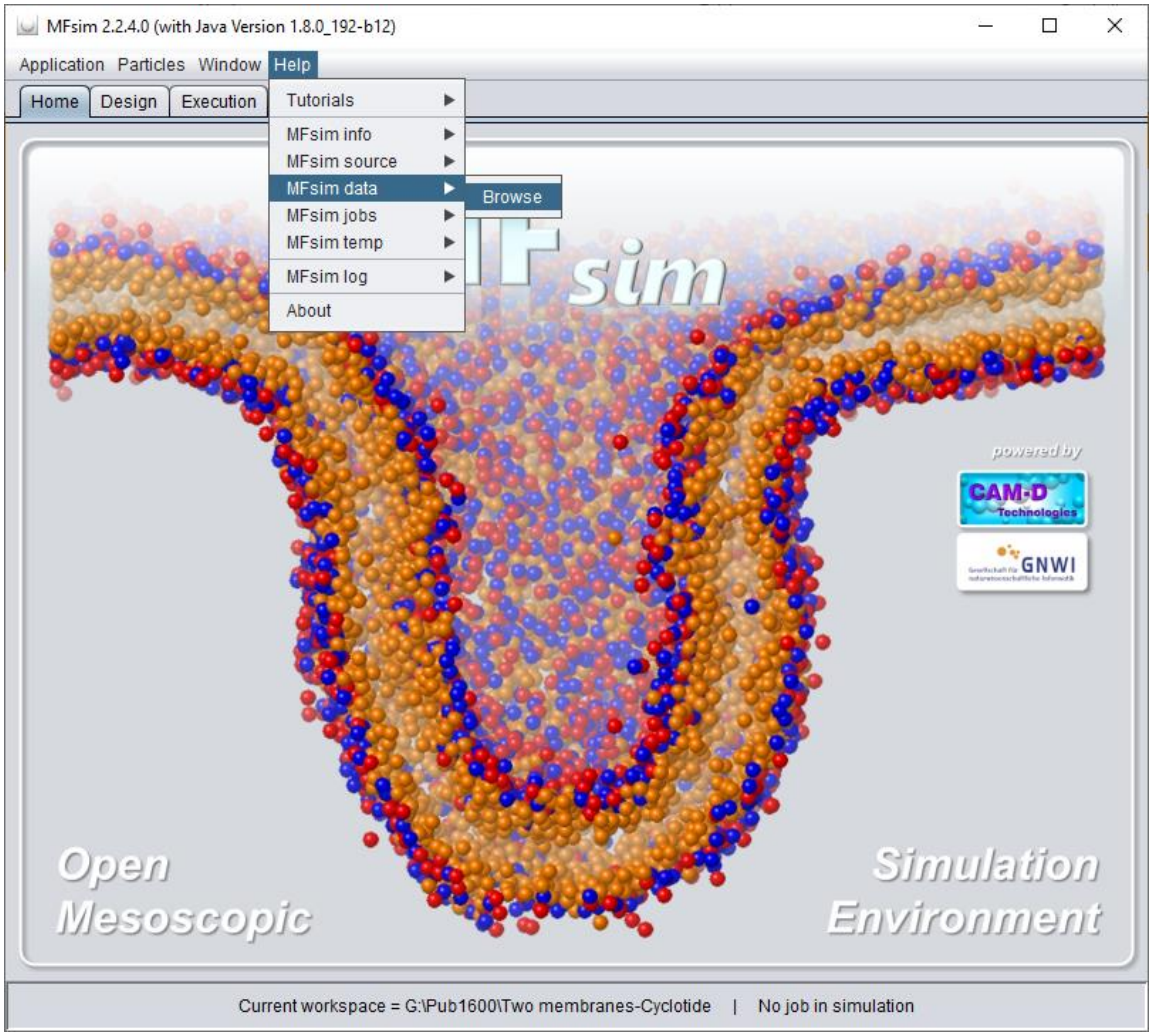


Figure 32. Starting the application and choosing the particle set.

Then download the particle set text file *ParticleSet_AA_V03_pm.txt* from the MFsim GitHub repository (located in subfolder *tutorials/Supplement*) into the *CustomParticles* subfolder (figure 31).

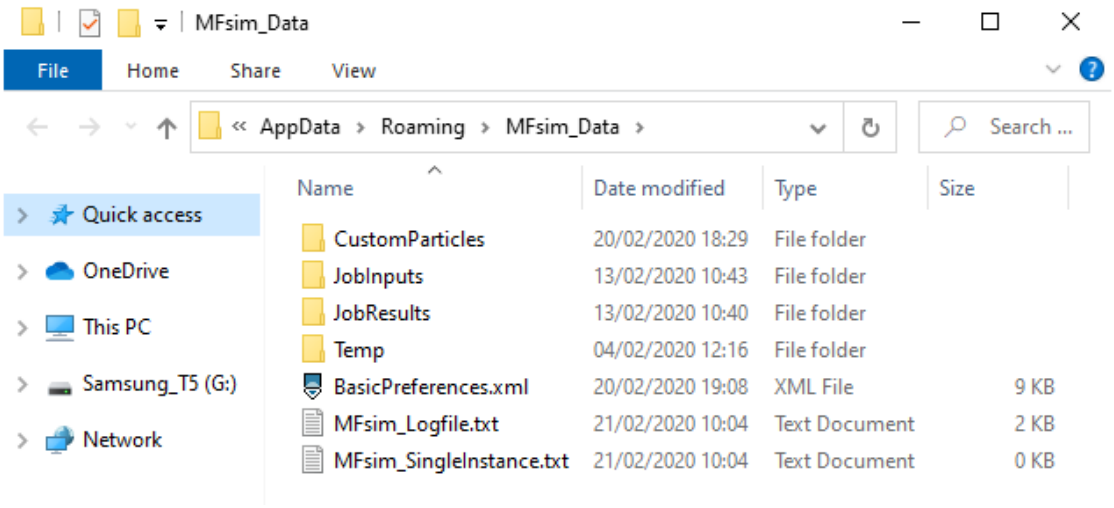


Figure 33. Placing the desired particle set text file into the correct subfolder.

Now the previously downloaded particle set has to be chosen. Select the menu item ***Particles / Choose particle set*** and search for the file *ParticleSet_AA_V03_pm.txt*. Afterwards confirm with ***Apply*** (figure 32).

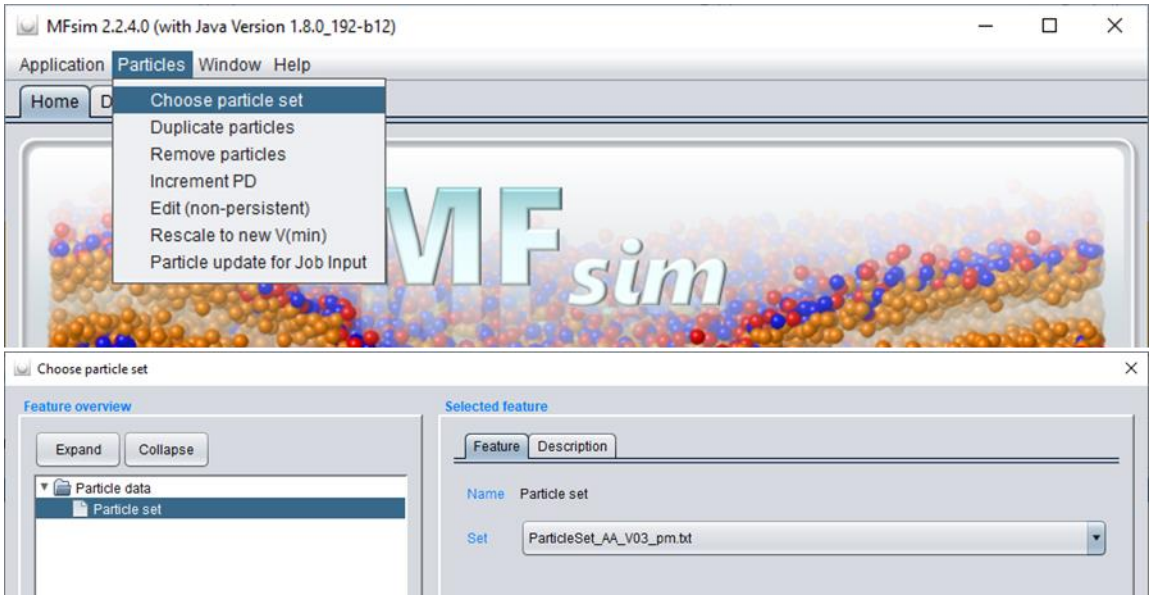


Figure 34. Selecting the appropriate particle set for the planned simulation job.

In order to create a new simulation, activate the ***Design*** tab and create a new job input with button ***New*** (compare to figure 33).

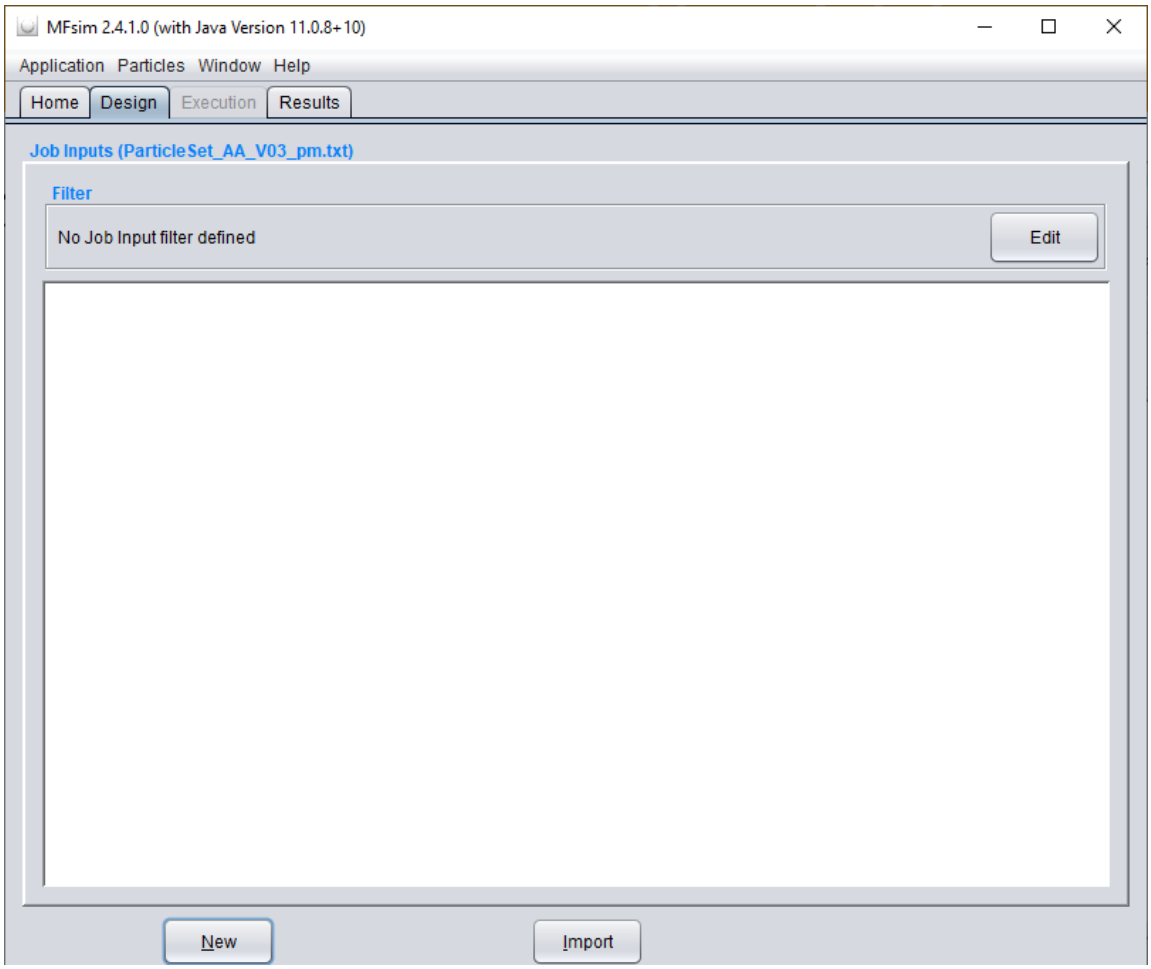


Figure 35. Creating a new job input via the ***Design*** tab.

At first, the job needs an appropriate description. In this tutorial *2 PM (50% chol) ApL 40, 1000 KB1-Y19* is chosen. For this purpose, select ***Job Input / General job description / Description***.

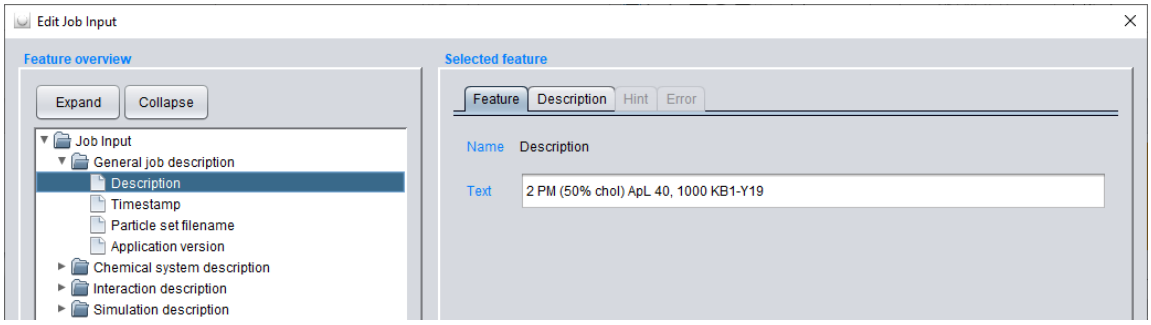


Figure 36. General job description.

The following step comprises the molecule definition (compare to figure 35). Select ***Job Input / Chemical system description / Molecule definition*** and rename the default water molecule to *WaterIn1* (the water molecules for the upper "inside" water compartment, compare to figure 22). ***Insert*** a new row and rename to *WaterOut* (the water molecules for the enclosed "outside" cyclotide/water compartment, compare to figure 22). ***Insert*** a second new row and rename to *WaterIn2* (the water molecules for the lower "inside" water compartment, compare to figure 22).

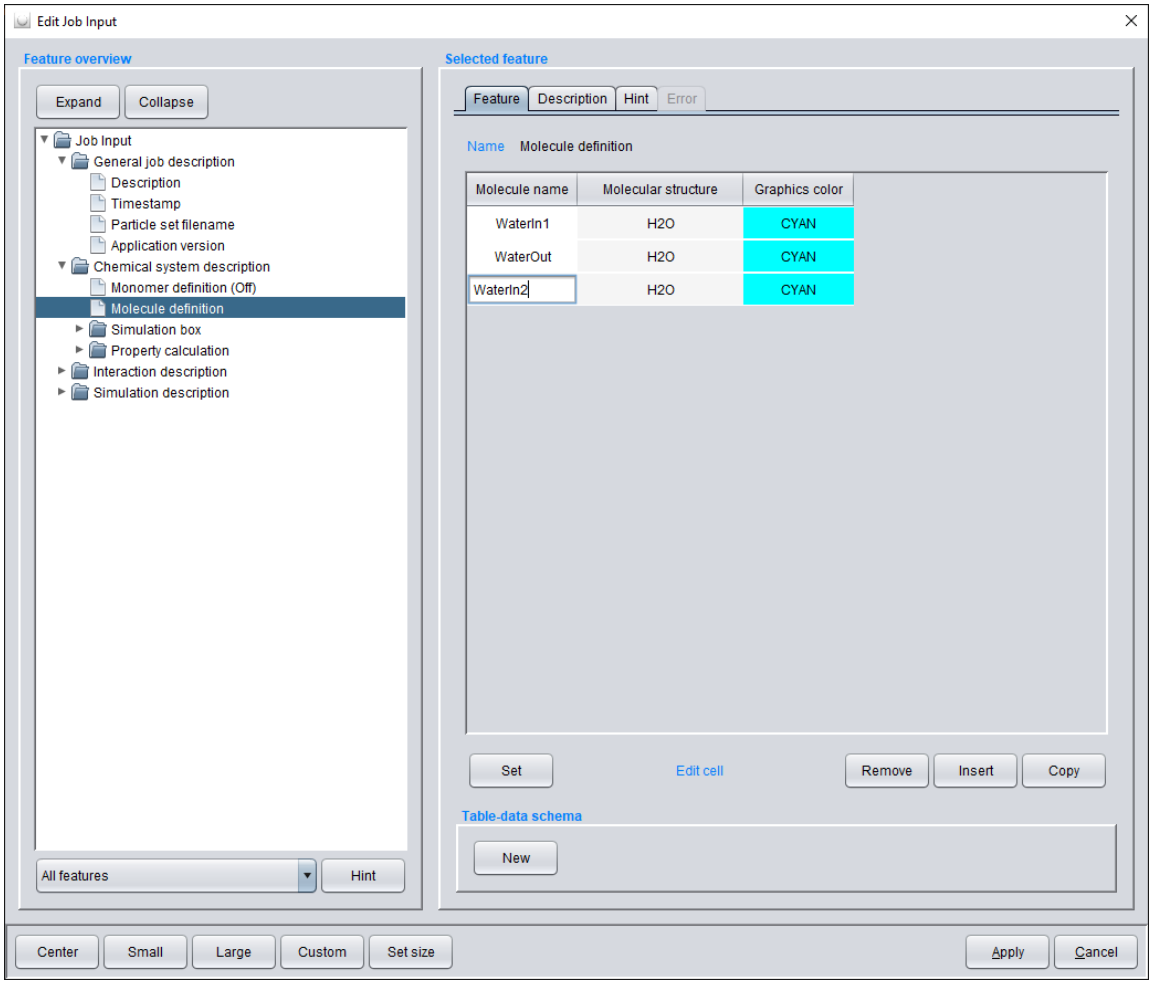


Figure 37. Defining the desired molecules for the simulation job.

For the insertion of more complex molecules than water, press ***Insert*** to create a new row and rename it into *CHOLin* (the cholesterol molecules of the inner membrane leaflets). Change the ***Graphics color*** to *OLIVE* (compare to figure 36). Open the molecular structure editor dialog by double-click on the ***Molecular structure*** column entry.

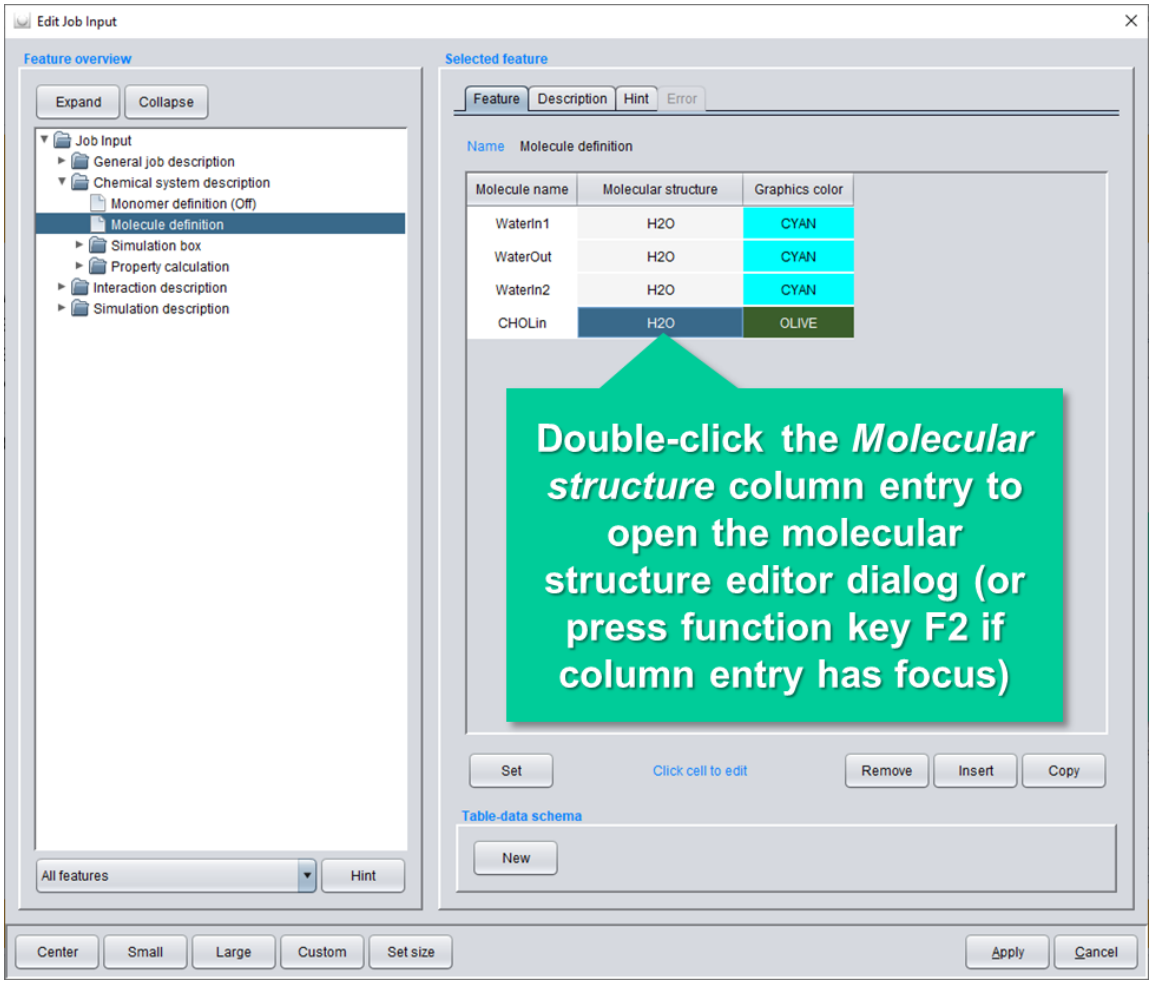


Figure 38. Adding more complex molecules to the simulation job.

Once the molecular structure editor is open, insert the following SPICES string in the ***Structure*** box to create a cholesterol molecule (compare to figure 37):

MeOHchol'1'[START](Et[2]-Et[3])-Et-Et[2](Me)(Et[1])-Et[3][4]-Et'2'[1](Me)(Et[4])-Et(Me)-Et-Et-Me[END]

A topological graph display of the ***Valid molecule*** definition is directly created below in the ***Graphics*** panel. The [START]/[END] tags allow for later orientation of the cholesterol molecules in the simulation box. The backbone tags '1' and '2' support the definition of molecular backbone forces (harmonic springs) between the tagged particles to control molecular stiffness. Particle MeOHchol is a duplicated physical MeOH particle to be exclusively used in the cholesterol molecule (e.g. for later specific display or analysis purposes). Hit ***Apply*** to confirm the entry.

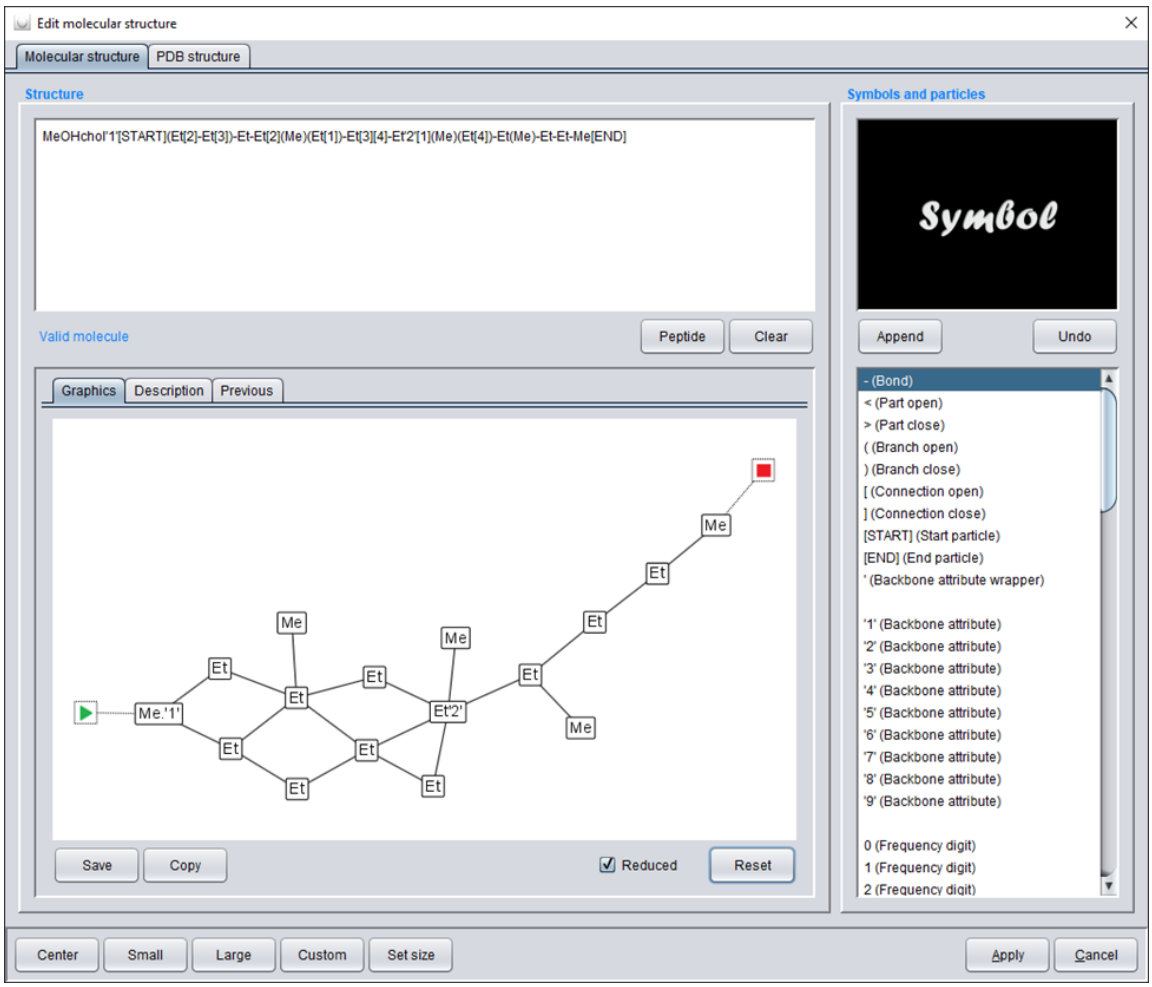


Figure 39. The molecular structure editor. Inserting cholesterol via a SPICES string.

The SPICES definition is shown in the ***Molecule structure*** field. Define the following molecules in the same manner for a complete membrane compound description (compare to figure 38):

- Cholesterol of the outer membrane leaflets:
  - ***Molecule name***: *CHOLout*
  - SPICES ***Molecular structure***: *MeOHchol'1'[START](Et[2]-Et[3])-Et-Et[2](Me)(Et[1])-Et[3][4]-Et'2'[1](Me)(Et[4])-Et(Me)-Et-Et-Me[END]*
  - ***Graphics color***: *OLIVE*
- DMPC of the inner membrane leaflets:
  - ***Molecule name***: *DMPCin*
  - SPICES ***Molecular structure***: *TriMeNPpl[START]-DMPN(MeAc-6Et[END])(MeAc-6Et)*
  - ***Graphics color***: *GREEN*
- DMPC of the outer membrane leaflets:
  - ***Molecule name***: *DMPCout*
  - SPICES ***Molecular structure***: *TriMeNPpl[START]-DMPN(MeAc-6Et[END])(MeAc-6Et)*
  - ***Graphics color***: *GREEN*
- DOPE of the inner membrane leaflets:
  - ***Molecule name***: *DOPEin*
  - SPICES ***Molecular structure***: *MeNH2Ppl[START]-DMPN(MeAc-8Et)(MeAc-8Et[END])*
  - ***Graphics color***: *PLUM*
- DOPE of the outer membrane leaflets:
  - ***Molecule name***: *DOPEout*

  - SPICES ***Molecular structure***: *MeNH2Ppl[START]-DMPN(MeAc-8Et)(MeAc-8Et[END])*
  - ***Graphics color***: *PLUM*

- PIP2 of the inner membrane leaflets:
  - ***Molecule name***: *PIP2in*
  - SPICES ***Molecular structure***: *DMPN[1](MeAc-8Et)(MeAc-9Et[END])-MeOH-DMPNpl-DMPNpl[START]-MeOH-MeOH[1]*
  - ***Graphics color***: *ORANGE*
- PIP2 of the outer membrane leaflets:
  - ***Molecule name***: *PIP2out*
  - SPICES ***Molecular structure***: *DMPN[1](MeAc-8Et)(MeAc-9Et[END])-MeOH-DMPNpl-DMPNpl[START]-MeOH-MeOH[1]*
  - ***Graphics color***: *ORANGE*
- PS of the inner membrane leaflets:
  - ***Molecule name***: *PSin*
  - SPICES ***Molecular structure***: *HAcNpl[START]-DMPN(MeNH2Ppl)(MeAc-8Et)(MeAc-8Et[END])*
  - ***Graphics color***: *MAGENTA*
- SM of the inner membrane leaflets:
  - ***Molecule name***: *SMin*
  - SPICES ***Molecular structure***: *TriMeNPpl[START]-DMPN(MeAcNH-7Et)(MeOH-7Et-Me[END])*
  - ***Graphics color***: *MINT*

- SM of the outer membrane leaflets:
  - ***Molecule name***: *SMout*
  - SPICES ***Molecular structure***: *TriMeNPpl[START]-DMPN(MeAcNH-7Et)(MeOH-7Et-Me[END])*
  - ***Graphics color***: *MINT*

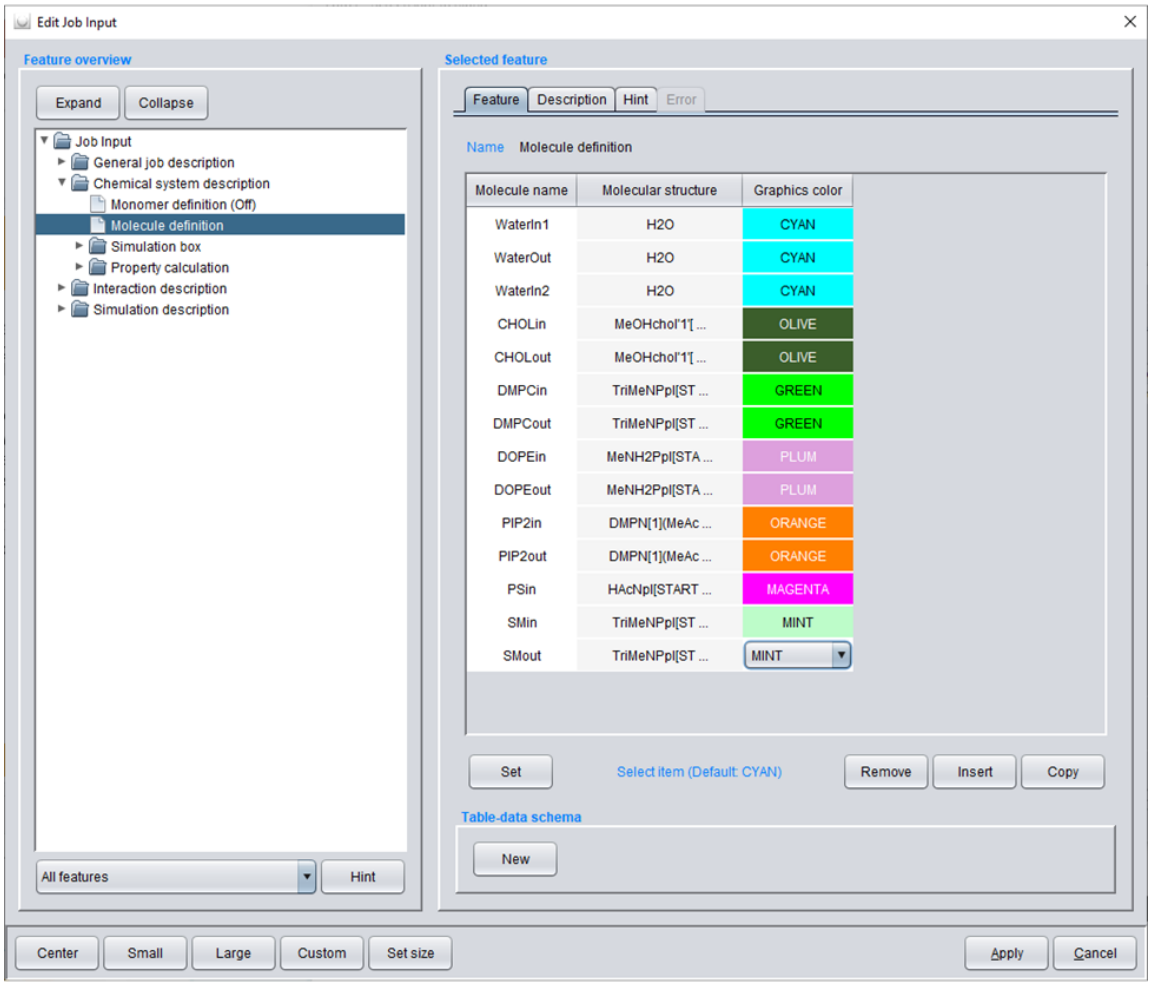


Figure 40. The growing molecule definition table, containing water and phospholipids.

For the cyclotide definition, ***Insert*** a new row, rename to *KB1Y19* and choose *WHITE* as ***Graphics color.*** Click the ***Molecular structure*** field. Then select the ***PDB structure*** tab to import the 3D structure of Kalata B1 from its PDB file. Hit the ***Load*** button and ***Select PDB file*** *1NB1.pdb* to import and display cyclotide related PDB information (compare to figure 39). The PDB file *1NB1.pdb* can be downloaded from *https://www.rcsb.org/structure/1NB1* or from the *tutorials/Supplement* subfolder of the MFsim GitHub repository.

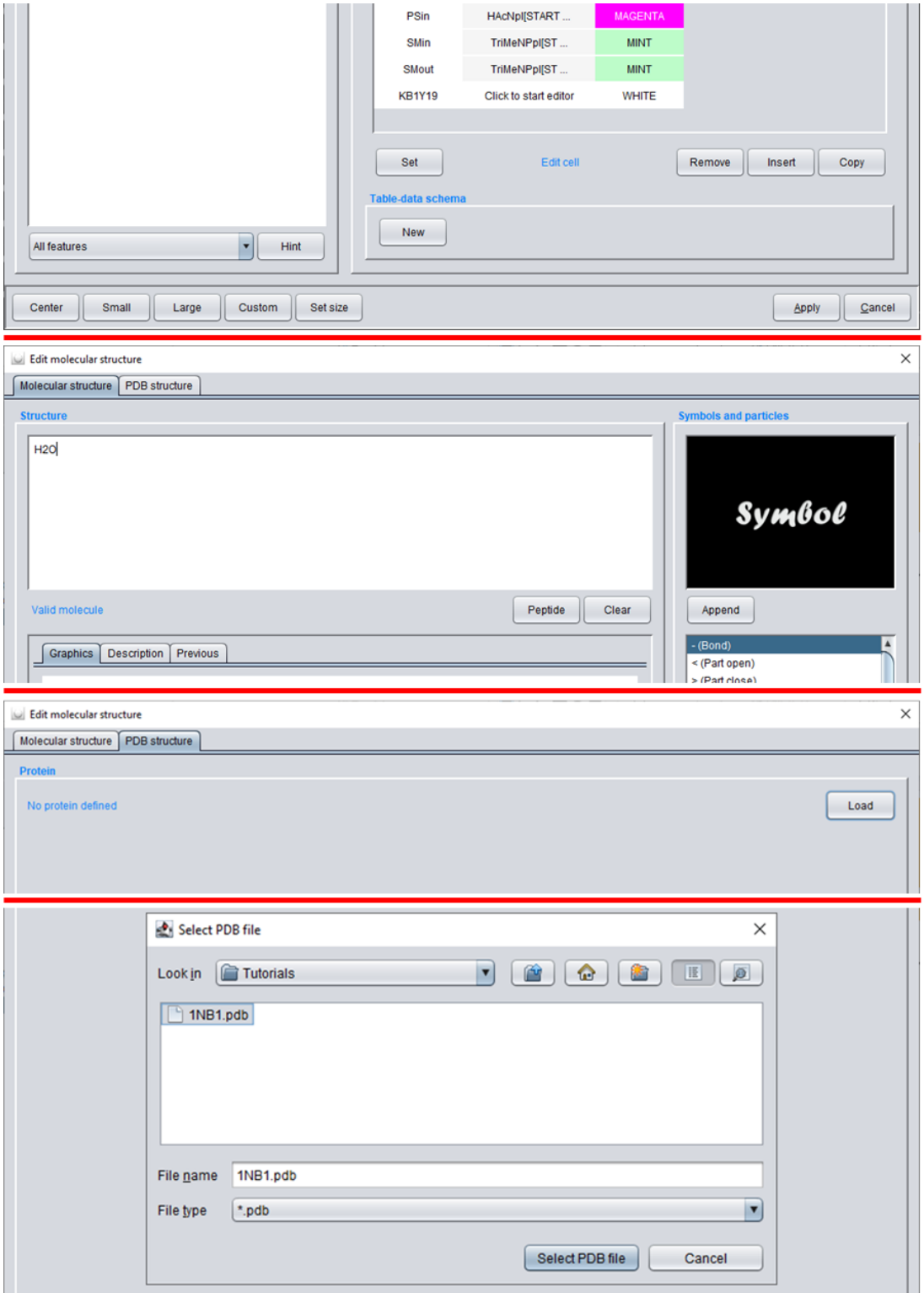


Figure 41. Partial screenshots of the workflow to load the PDB file of the cyclotide Kalata B1 into the simulator.

In order to perform the desired amino acid exchange, hit the ***Mutant*** button (compare to figure 40). Replace the amino acid via the ***Replacement*** column: Choose ***Amino acid*** *Tryptophan* with ***Index in chain*** *19* and replace by Tyrosine. Confirm the replacement with ***Apply***.

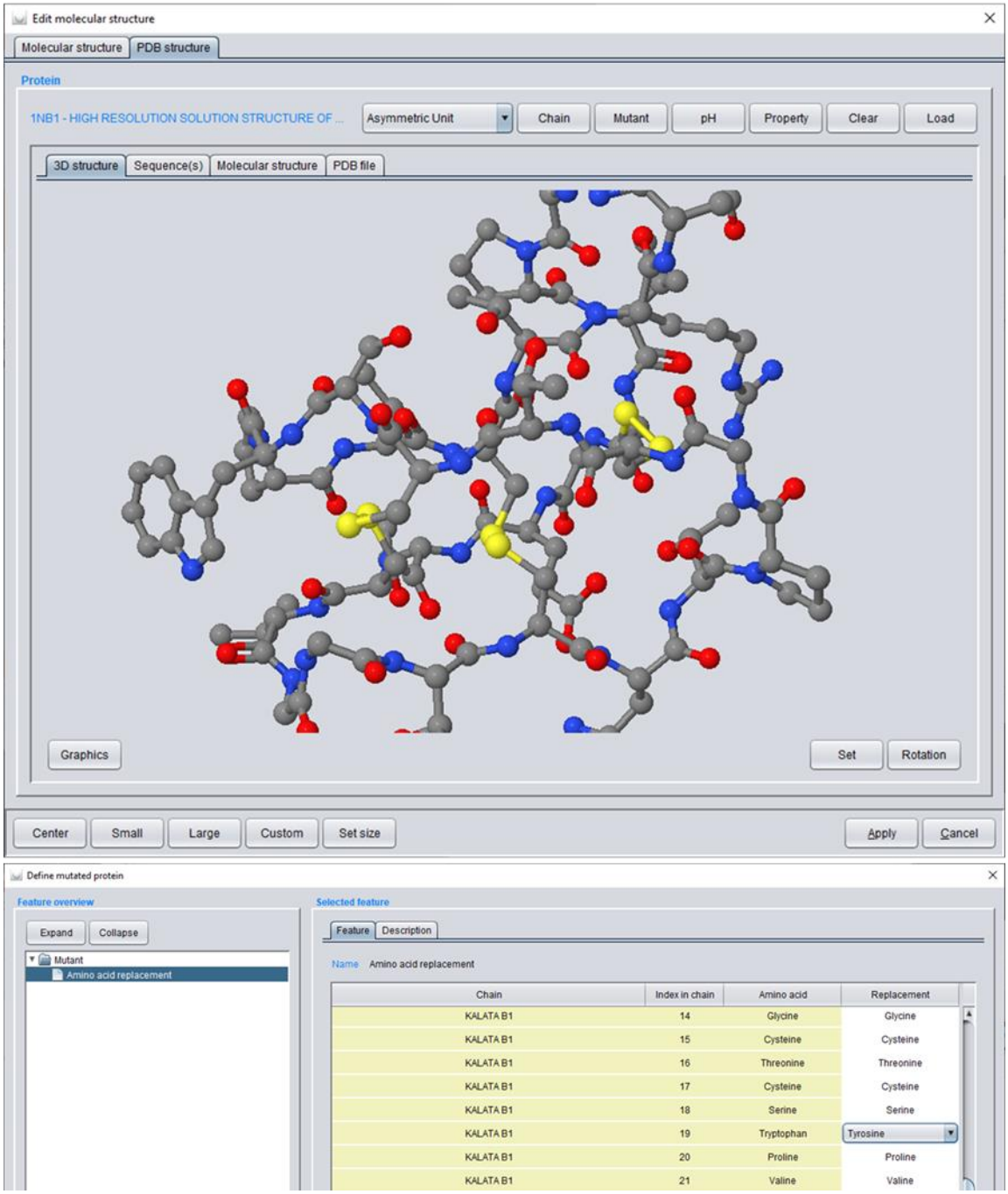


Figure 42. The procedure of mutating an amino acid within a given protein or peptide.

In the next step, the pH value has to be adjusted. Hit the ***pH*** button to set the ***pH value*** to (its physiological value of) *7.4*. Confirm with ***Apply*** (compare to figure 41).

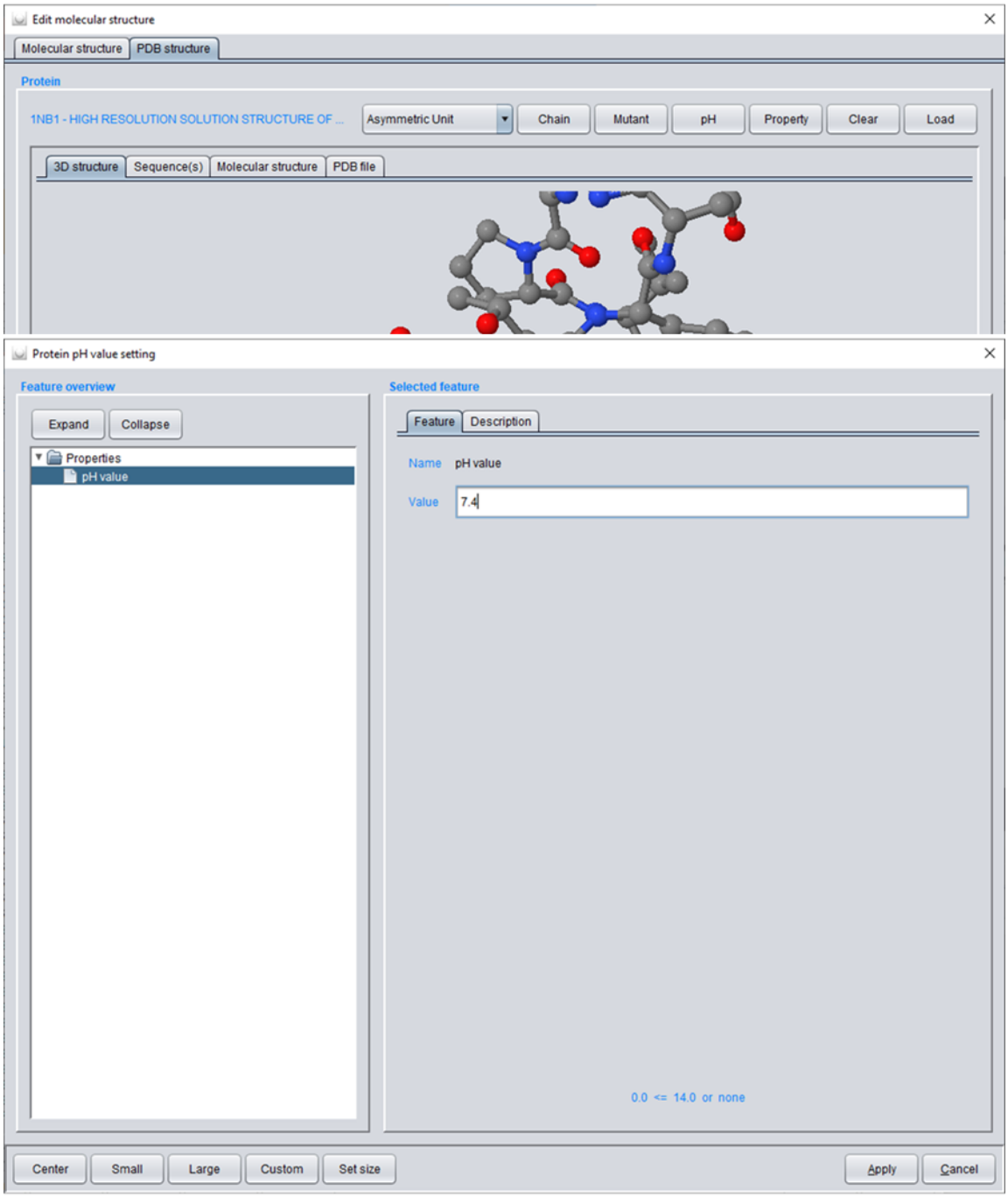


Figure 43. Applying the correct pH value to the protein or peptide of interest.

Afterwards hit the ***Property*** button and select ***Properties / Backbone probes*** (compare to figure 42.) to replace protein backbone particles by probe particles which allow for specific display later in the simulation box (without any change of "the physics").

Replace ***Backbone particle**s* of ***Amino acid**s* with ***Index*** 3, 19, 20, 21, 24, 27, 28 and 29 by the following ***Probe particle**s*

- index *3* to probe particle *MeAcNHPD2*
- index *19* to probe particle *MeAcNHPD4*
- index *20* to probe particle *AzolidPD1*
- index *21* to probe particle *MeAcNHPD1*
- index *24* to probe particle *MeAcNHPD3*
- index *27* to probe particle *MeAcNHPD1*
- index *28* to probe particle *AzolidPD1*
- index *29* to probe particle *MeAcNHPD1*

Confirm with ***Apply***.

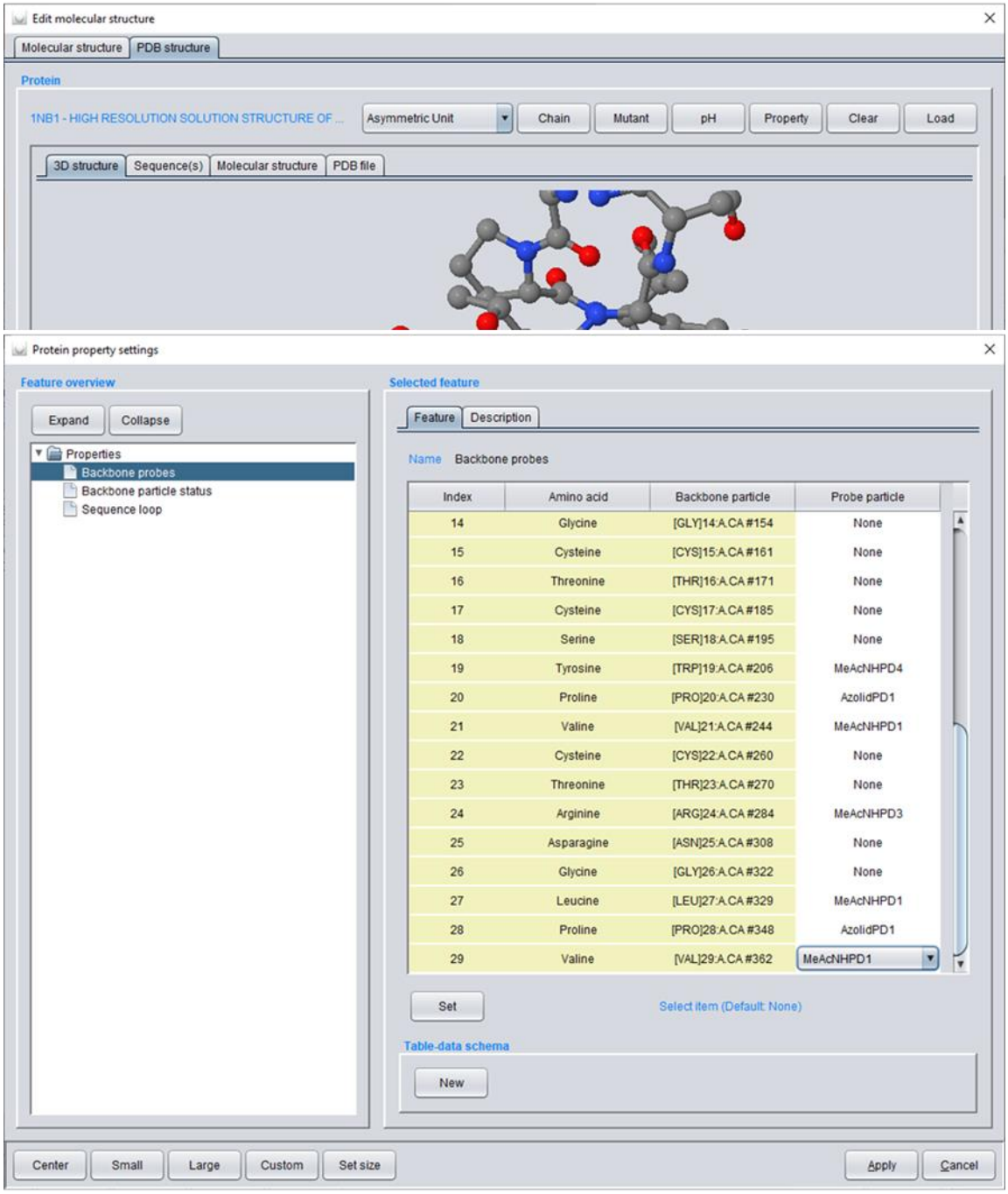


Figure 44. Defining probe particles within the backbone particles for better visualisation in the simulation box.

Since Kalata B1 is a cyclic peptide, select ***Properties / Sequence loop*** and choose *True* (compare to figure 43). Confirm with ***Apply***.

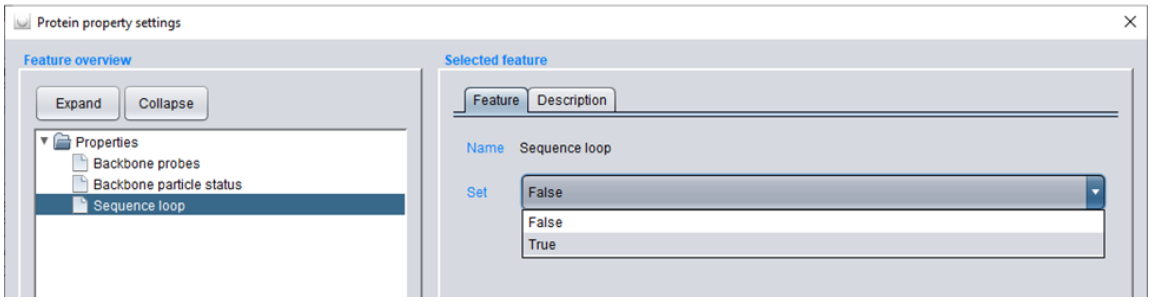


Figure 45. Selecting a sequence loop for cyclic proteins or peptides.

The applied settings can be viewed with the ***Sequence(s)*** tab and the ***Molecular structure*** tab (where each line of the SPICES string corresponds to a single amino acid representation). Confirm the entries with ***Apply*** (compare to figure 44).

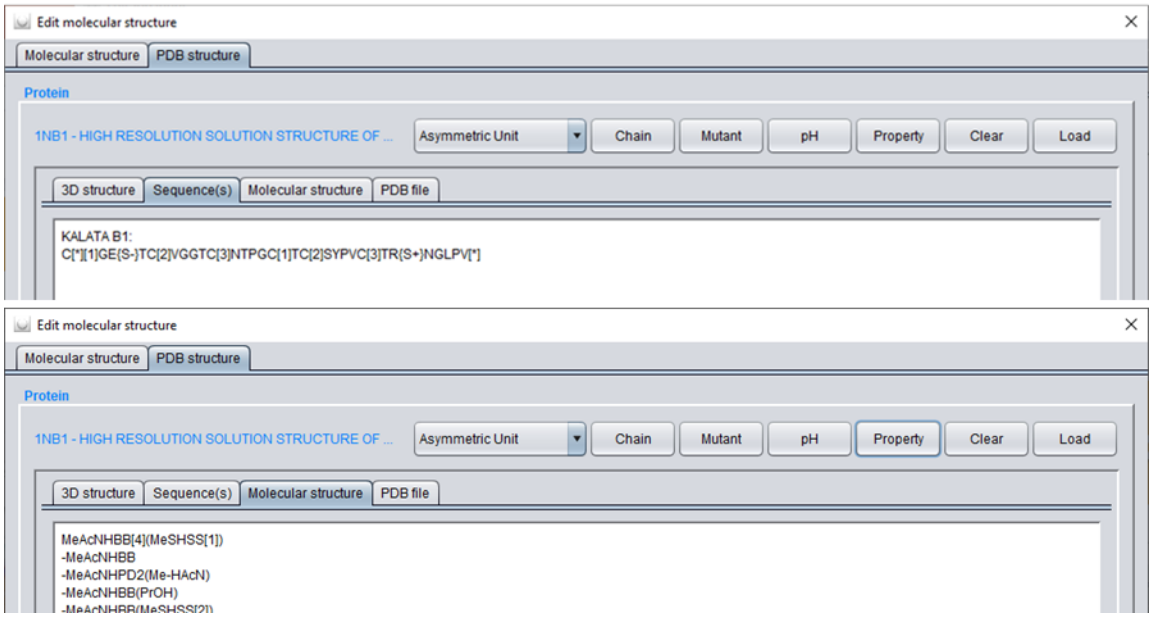


Figure 46. Review of the applied settings for the peptide Kalata B1.

Now the the completed ***Molecule definition*** should look like this:

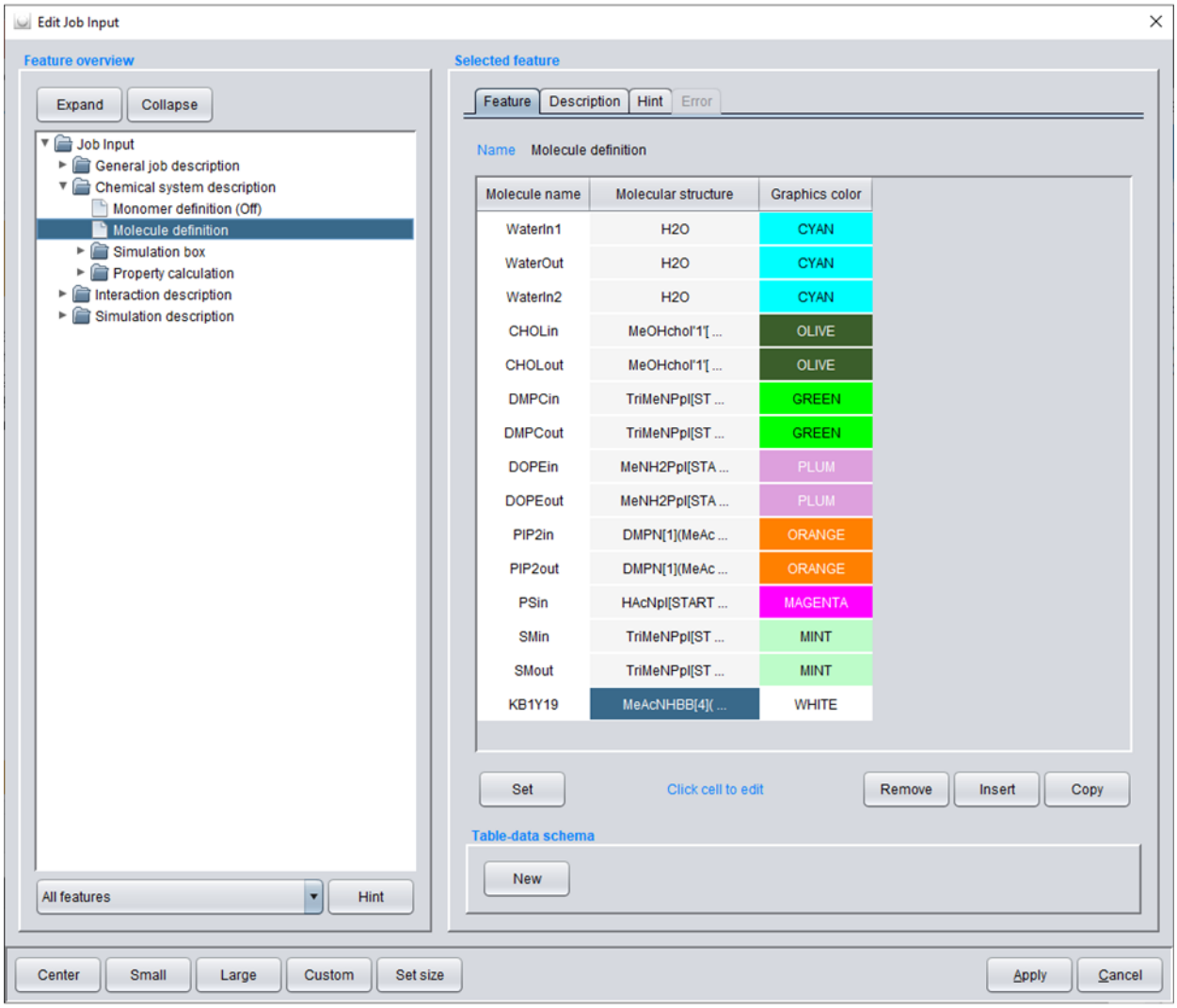


Figure 47. The entire molecule definition table for the simulation of this tutorial.

In the following step, the quantity of each molecule participating in the simulation has to be defined. Select ***Job Input / Chemical system description / Simulation box / Composition / Quantity*** and define the individual molecule quantities in the ***Value*** column of every ***Molecule name*** row as shown in figure 46.

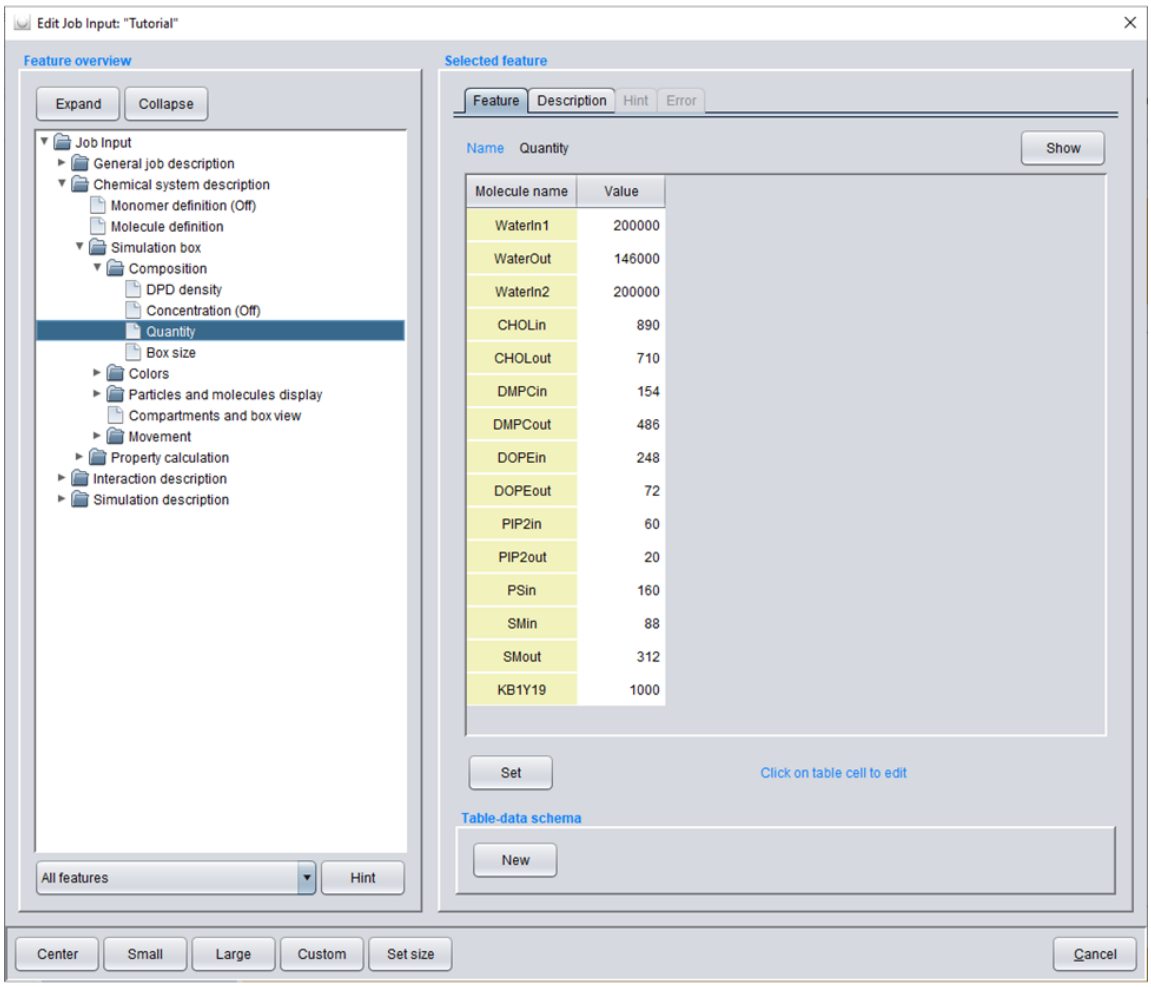


Figure 48. Setting the quantities for each participating molecule.

Now the size of the simulation box has to be adjusted (compare to figure 47). Therefore choose ***Job Input / Chemical system description / Simulation box / Composition / Box size*** select *fixed* for ***State X*** and change the fixed simulation box length ***x [A]*** to *185.1* Ångstrom. Repeat these operations for a fixed simulation box y length of *185.1* Ångstrom. The value of the *flexible* z length is automatically adjusted accordingly (depending on the number and volume of the particles in the box and the DPD density).

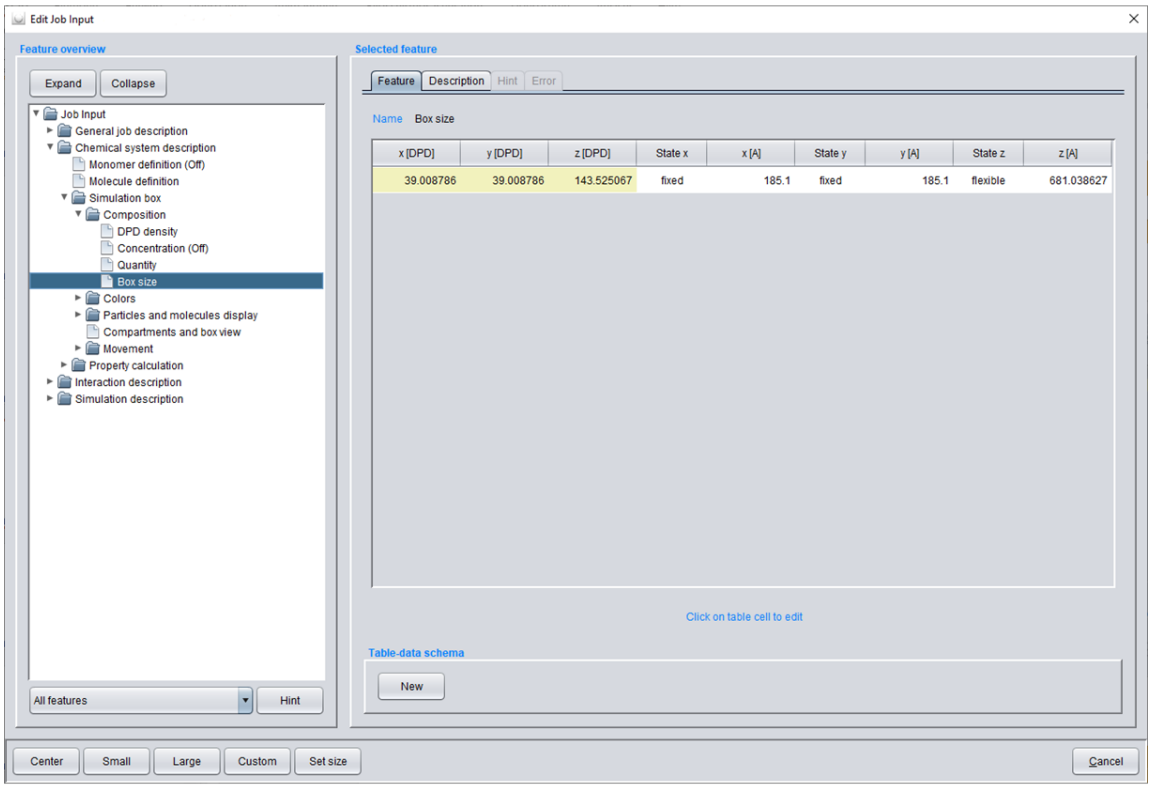


Figure 49. Box size adjustments for the x- and y-direction.

Subsequently the particle display can be chosen for better visualization of the simulation results. Select ***Job Input / Chemical system description / Simulation box / Colors / Mode*** and choose *Molecule-particle color*.

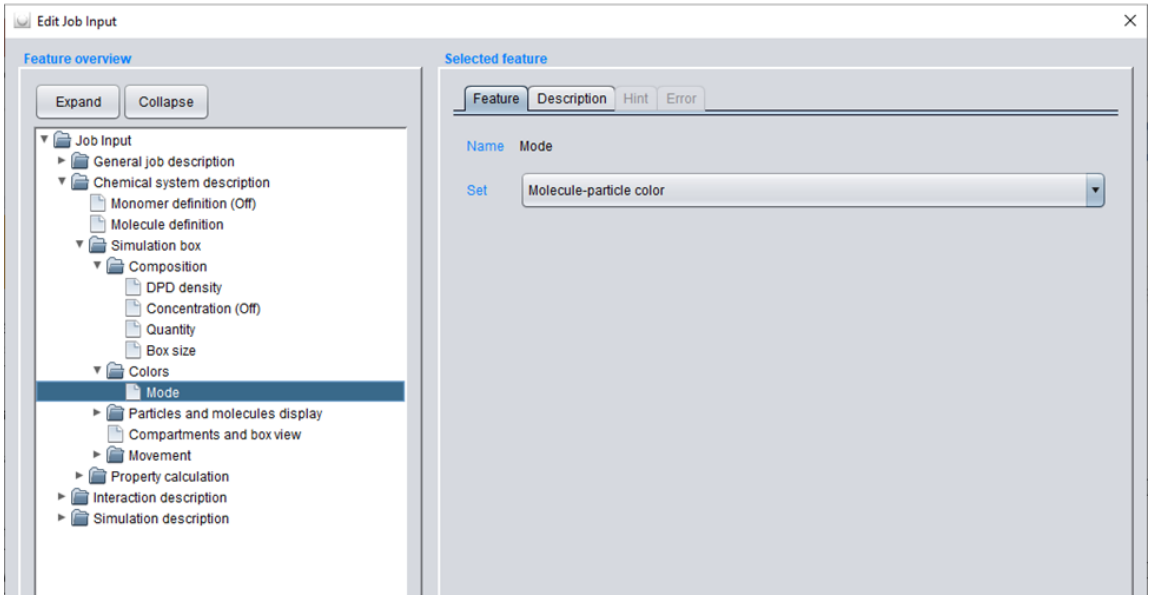


Figure 50. Changing the display of particles.

Select ***Job Input / Chemical system description / Simulation box / Particles and molecules display / Molecule particles*** to define ***Display*** and ***Graphics color*** of all particles within their related molecules (compare figure 49, top). To turn off the display of all H2O particles hit the ***Set*** button for (filtered) bulk settings. Select the (by default disabled) ***Display*** setting and enable this setting by ticking the ***On*** checkbox (compare figure 49, middle). The bulk setting for ***Display*** is set to *Off* (i.e. no display) by default. Change particle ***Filter 2*** to ***On*** and select the target *H2O* particle (to which the bulk setting will be applied excusively) under ***2 - Particle***. Confirm the entries with ***Apply***. The ***Display*** of all H2O particles in the simulation box is turned *Off*. Define all other entries by individual or filtered bulk settings (compare to figure 49, bottom), where only backbone particles of KB1Y19 are to be displayed, i.e. *AzolidBB*, *AzolidPD1*, *MeAcNHBB*, *MeAcNHPD1*, *MeAcNHPD2*, *MeAcNHPD3* and *MeAcNHPD4* to complete the particle display definitions, which can be saved to a ***Table-data schema*** by hitting the ***New*** button for comfortable re-use (not covered by this tutorial).

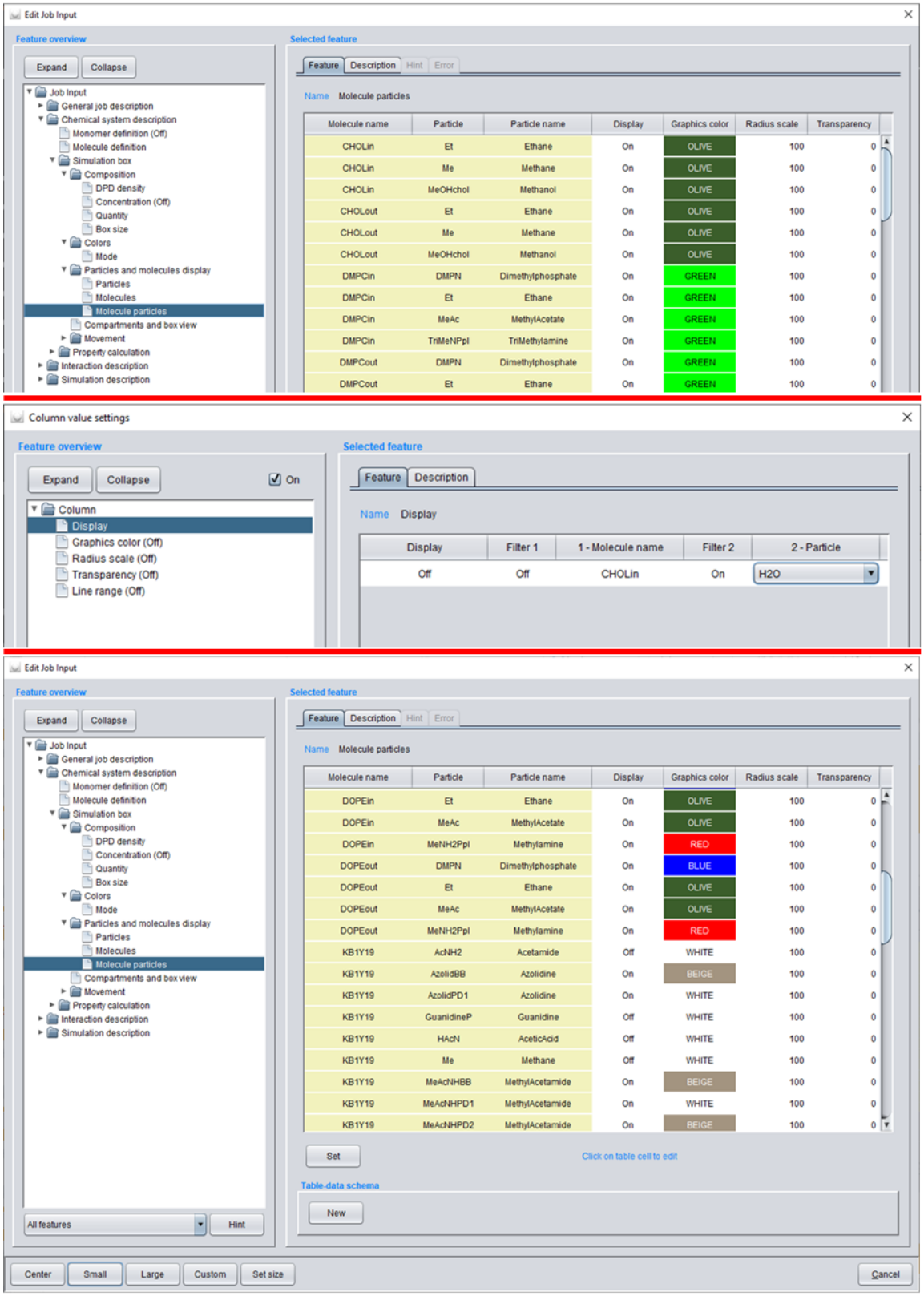


Figure 51. Defining display and graphics color of all particles within their related molecules.

Following the display adjustments, the arrangement of all participating particles has to be specified via the compartment definition. Select ***Job Input / Chemical system description / Simulation box / Compartments and box view*** and hit the ***Edit*** compartments button to open the ***Edit compartments of simulation box*** dialog with all molecules being initially located in the simulation box ***Bulk*** volume.

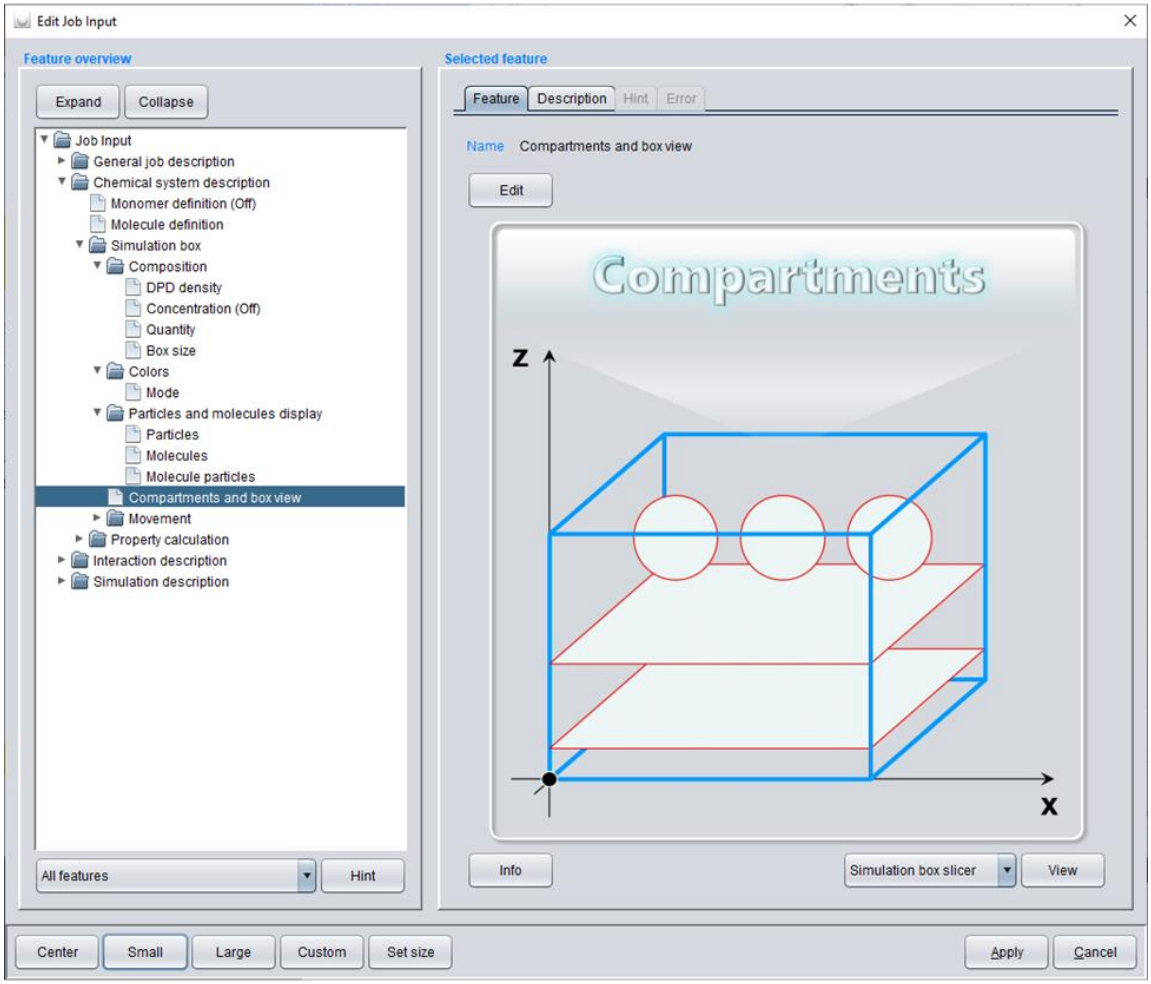


Figure 52. Opening the compartment editor.

In order to create the first compartment, hit the ***Name*** button (compare to figure 51, top) and input *Water in 1* and hit ***Apply*** (compare to figure 51, bottom).

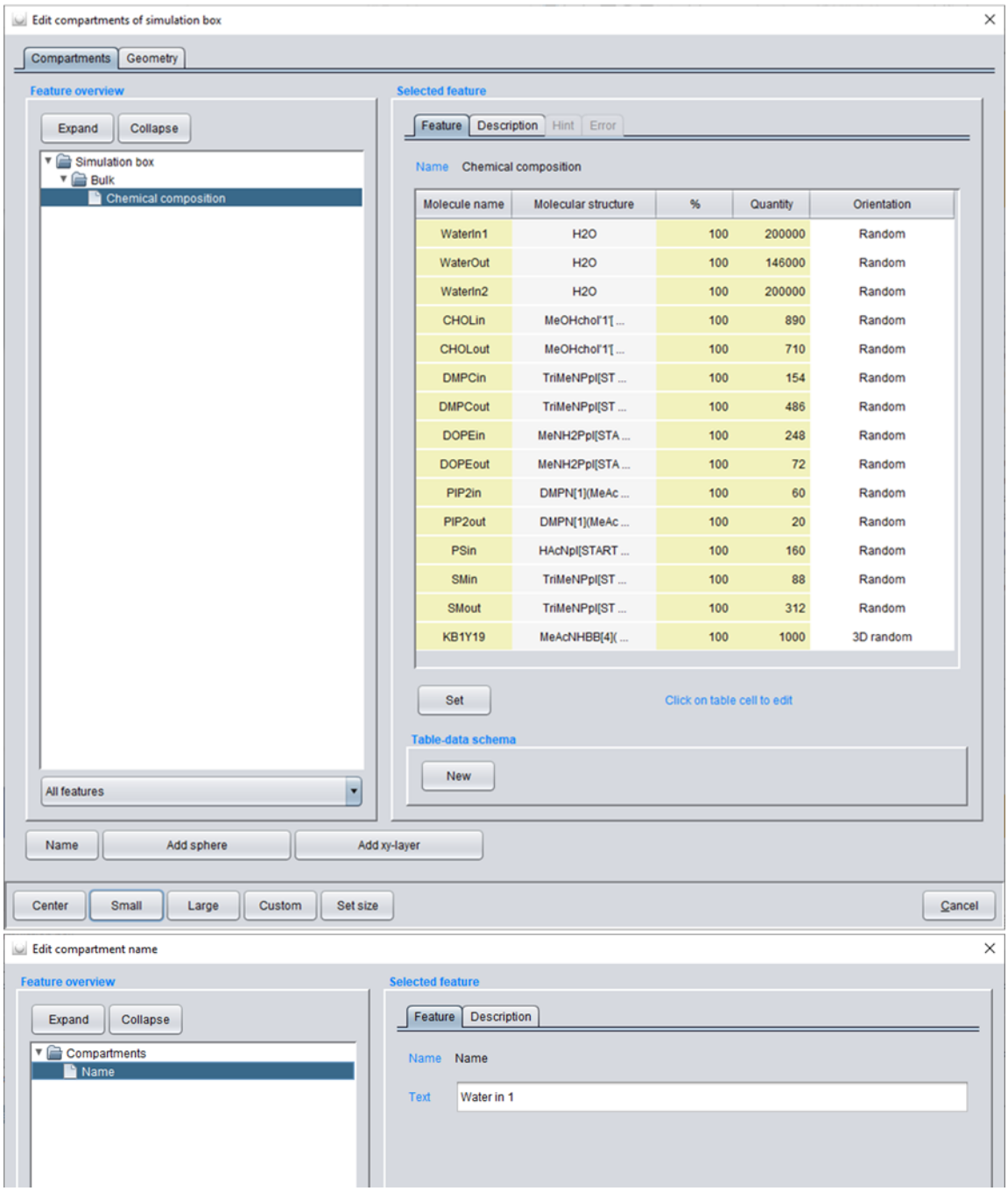


Figure 53. Designing the first xy-layer compartment.

Hit button ***Add xy-layer: Water in 1*** to create the *Water in 1* layer for the water molecules at the top of the simulation box. Specify *100 %* of *WaterIn1* to be located within this compartment.

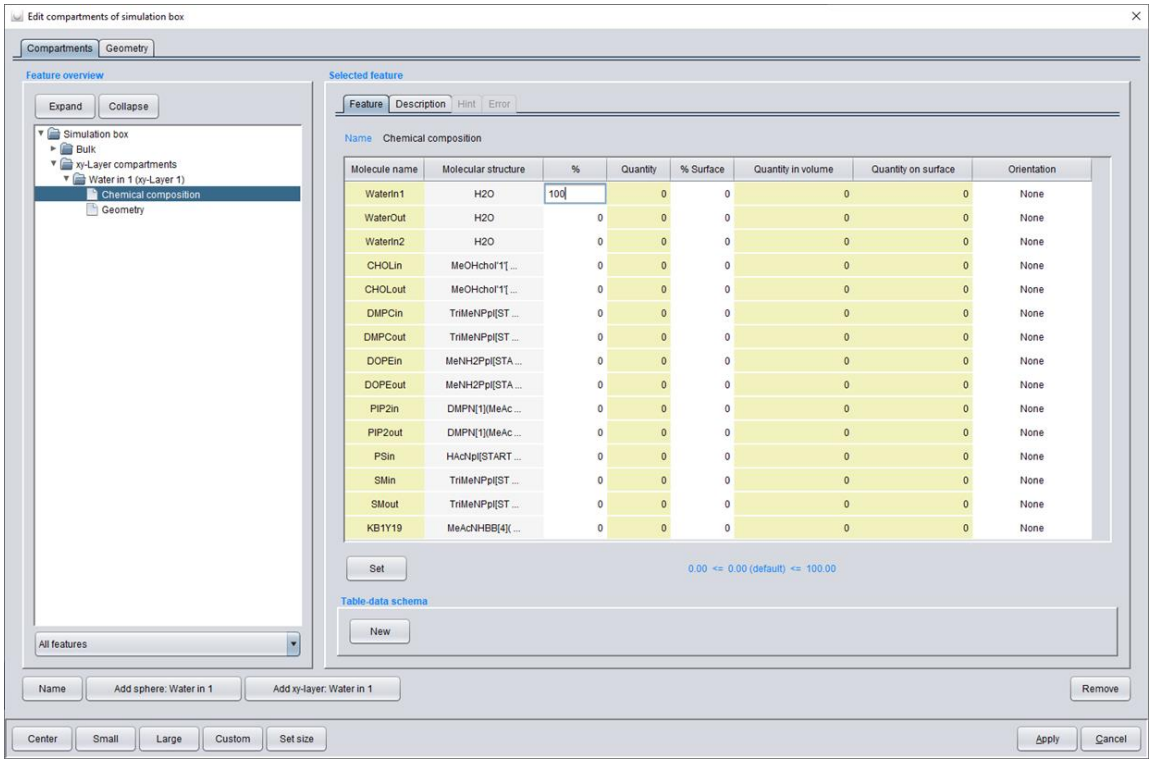


Figure 54. Assign 100 % of the “WaterIn1”-particles to the xy-layer named “Water in 1”.

Select the ***Geometry*** tab (compare to figure 52) where the new layer compartment is placed in the middle of the simulation box by default (compare to figure 53, top). Click (the compartment color is changed to white) & drag the *Water in 1* layer to the top of the simulation box (compare to figure 53, bottom). Switch back to the ***Compartments*** tab.

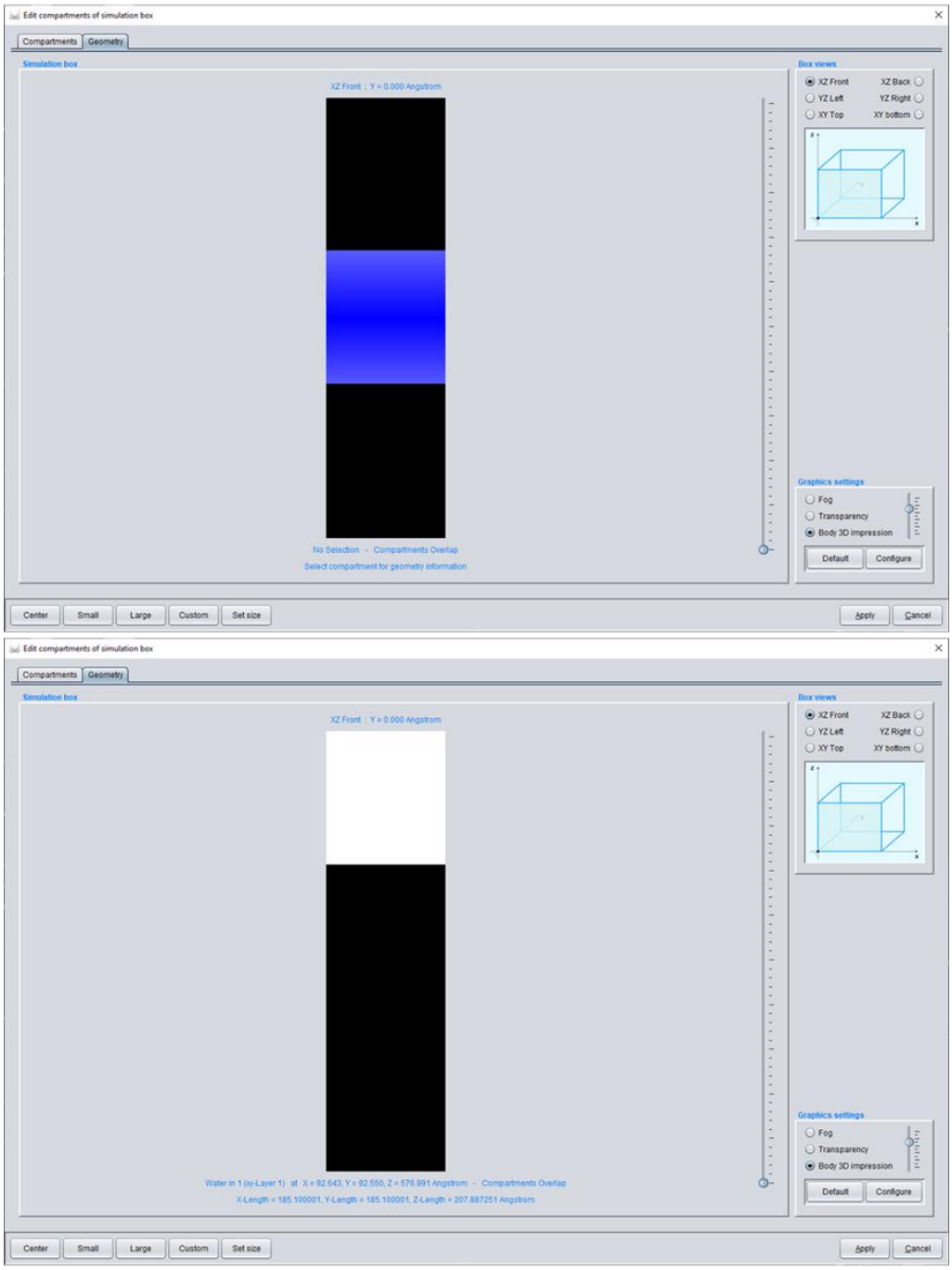


Figure 55. Replacing the new xy-layer compartment to the top of the simulation box by clicking and dragging.

Create a new *M1 in* layer for the inner leaflet of the upper plasma bilayer membrane. *50 %* of the inner membrane molecules are to be located here with *100 %* at the ***Surface*** with a *Random xy top* ***Orientation*** so that the particles labeled with a [START] tag are positioned at the top of this compartment (compare to figure 54).

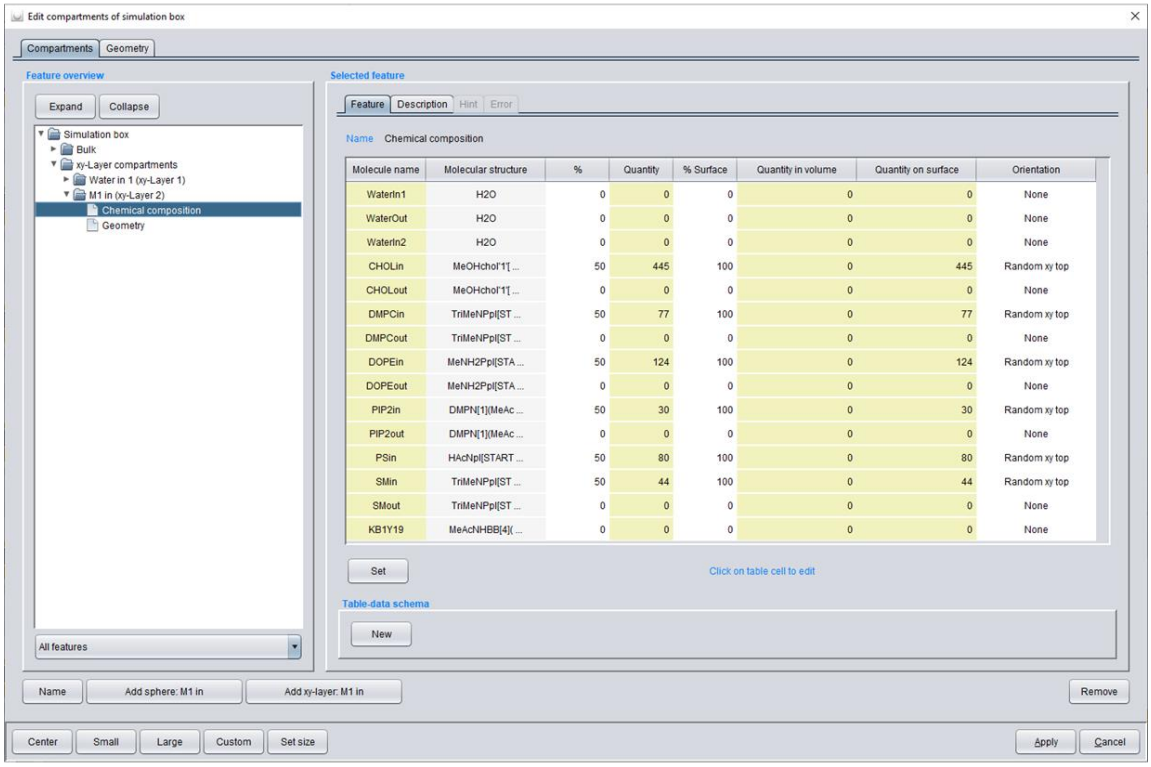


Figure 56. Creating the first membrane leaflet of the upper plasma membrane.

Select the ***Geometry*** tab (compare to figure 54) and click and drag the *M1 in*-layer directly below the *Water in 1*-layer (compare to figure 55). Switch back to the ***Compartments*** tab.

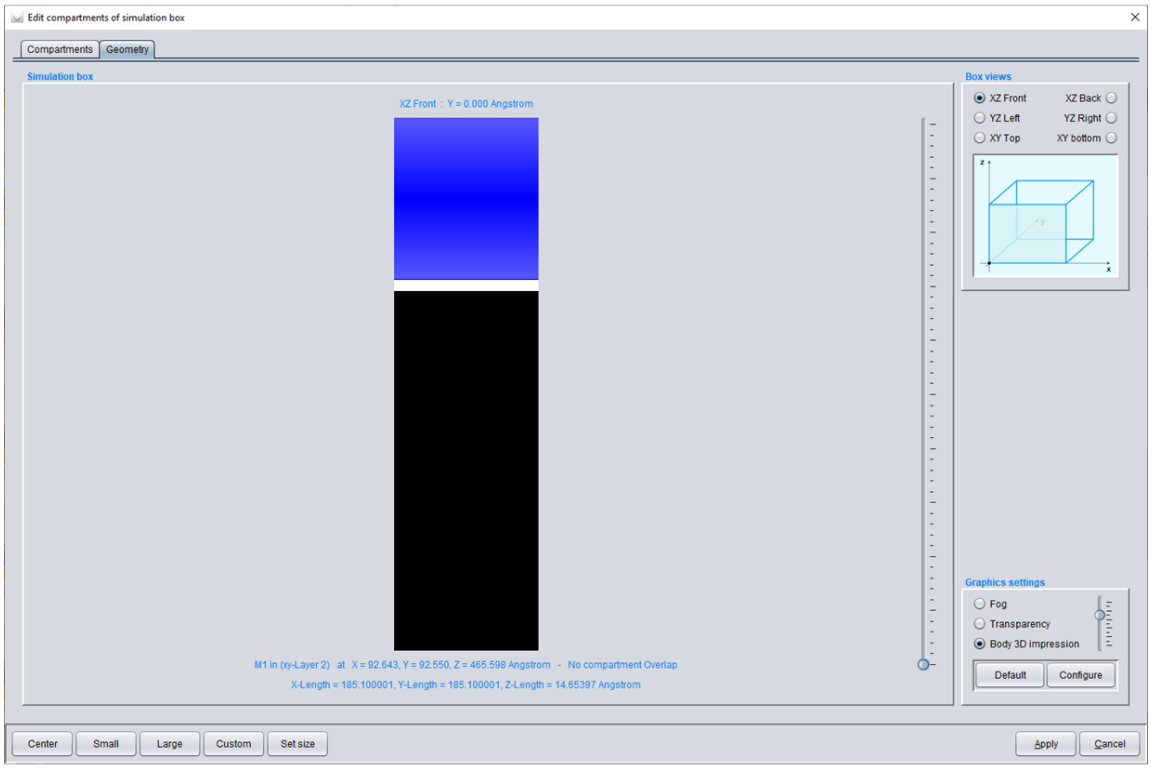


Figure 57. Positioning of the newly created membrane leaflet directly below the *Water in 1*-layer.

Create a new *M1 out* layer for the outer leaflet of the upper plasma bilayer membrane (compare to figure 56). *50 %* of the outer membrane molecules are to be located here with *100 %* at the ***Surface*** with a *Random xy bottom* ***Orientation*** so that the particles labeled with a [START] tag are positioned at the bottom of this compartment.

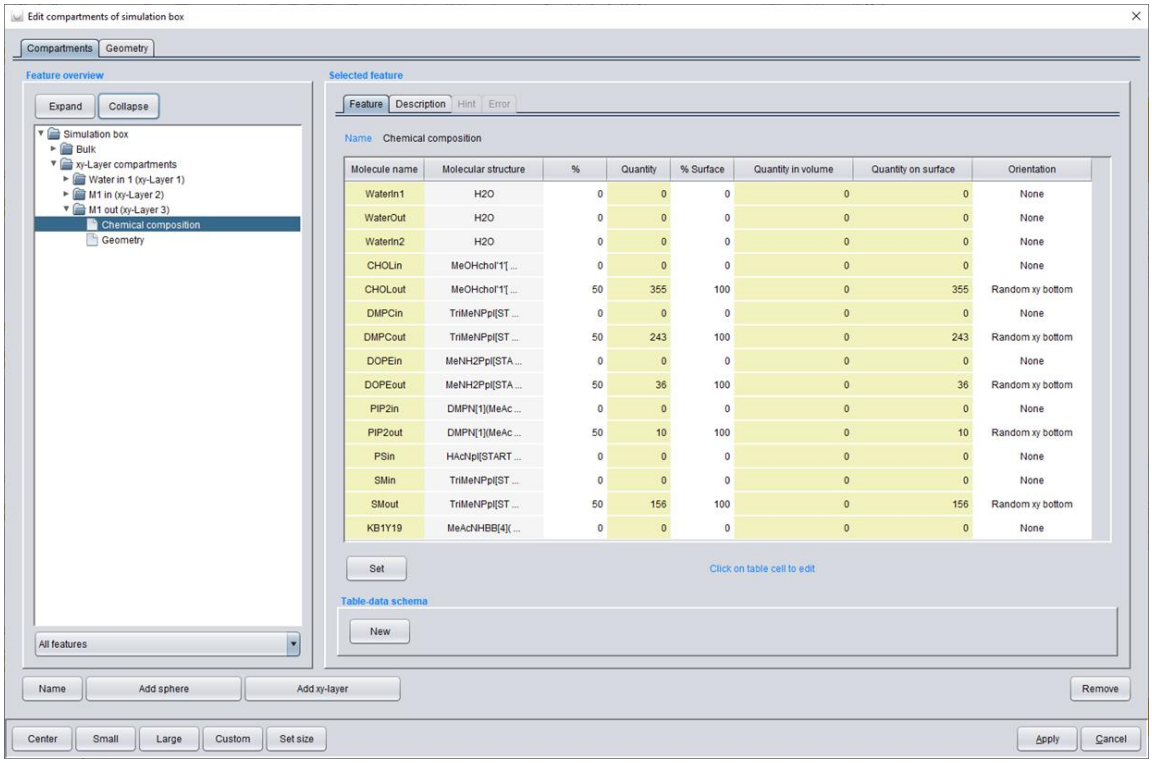


Figure 58. Creating the second membrane leaflet of the upper plasma membrane.

Select the ***Geometry*** tab (compare to figure 56) and click and drag the *M1 out*-layer directly below the *M1 in*-layer (compare to figure 57). Switch back to the ***Compartments*** tab.

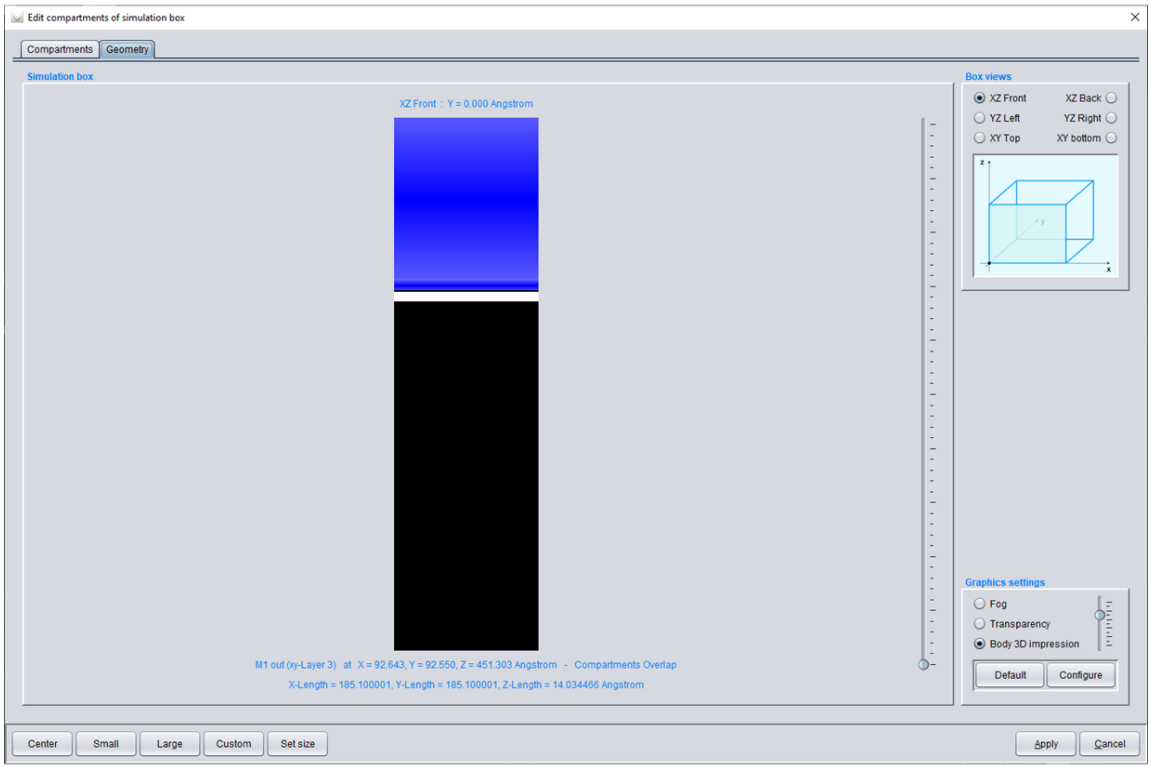


Figure 59. Placing of the newly created membrane leaflet directly below the *M1 in*-layer.

Create a new *Water out Buffer 1*-layer for the upper water buffer with *3.5* ***%*** of *WaterOut* molecules (analogous to the procedure described before). Select the ***Geometry*** tab and click & drag this layer directly below the *M1 out*-layer (compare to figure 58). Switch back to the ***Compartments*** tab.

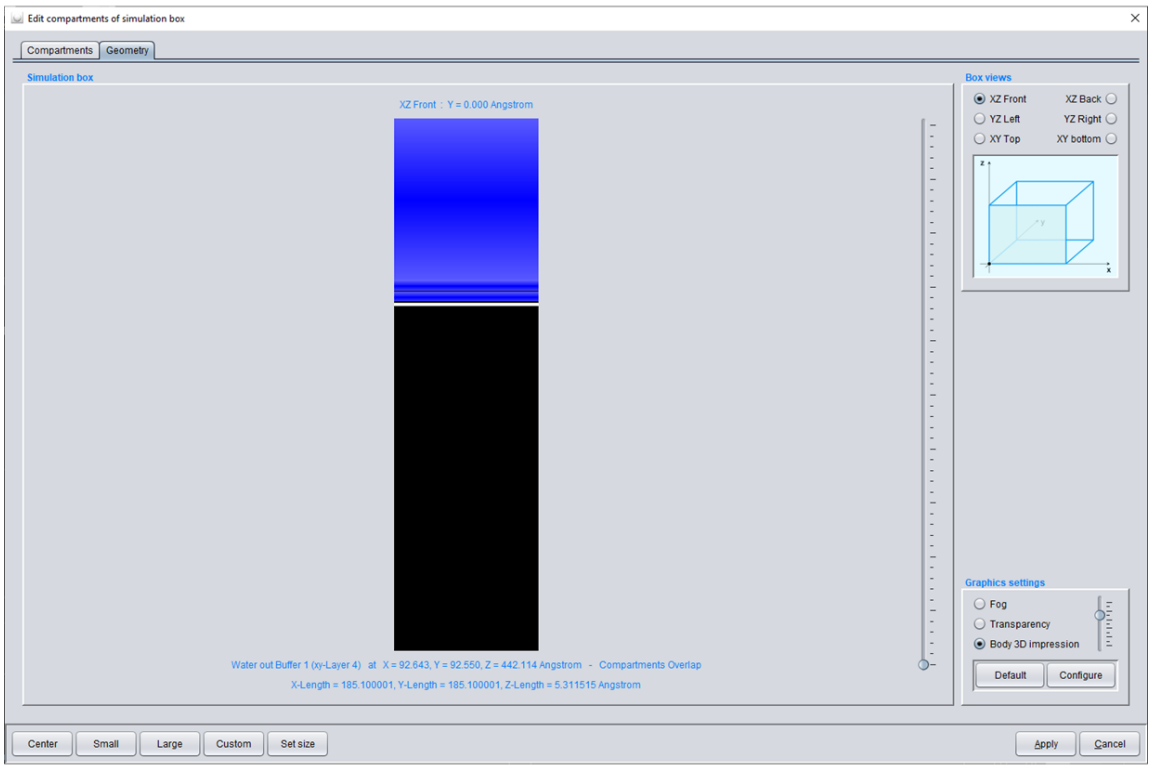


Figure 60. The *Water out Buffer 1*-layer has to be placed directly under the completed upper plasma membrane.

In order to further complete the start geometry of the cyclotide sandwich interaction model, create and position a new *Water in 2*-layer at the bottom of the simulation box, a new *M2 in*-layer (above), a new *M2 out*-layer (above) and a new *Water out Buffer 2*-layer (above) accordingly.

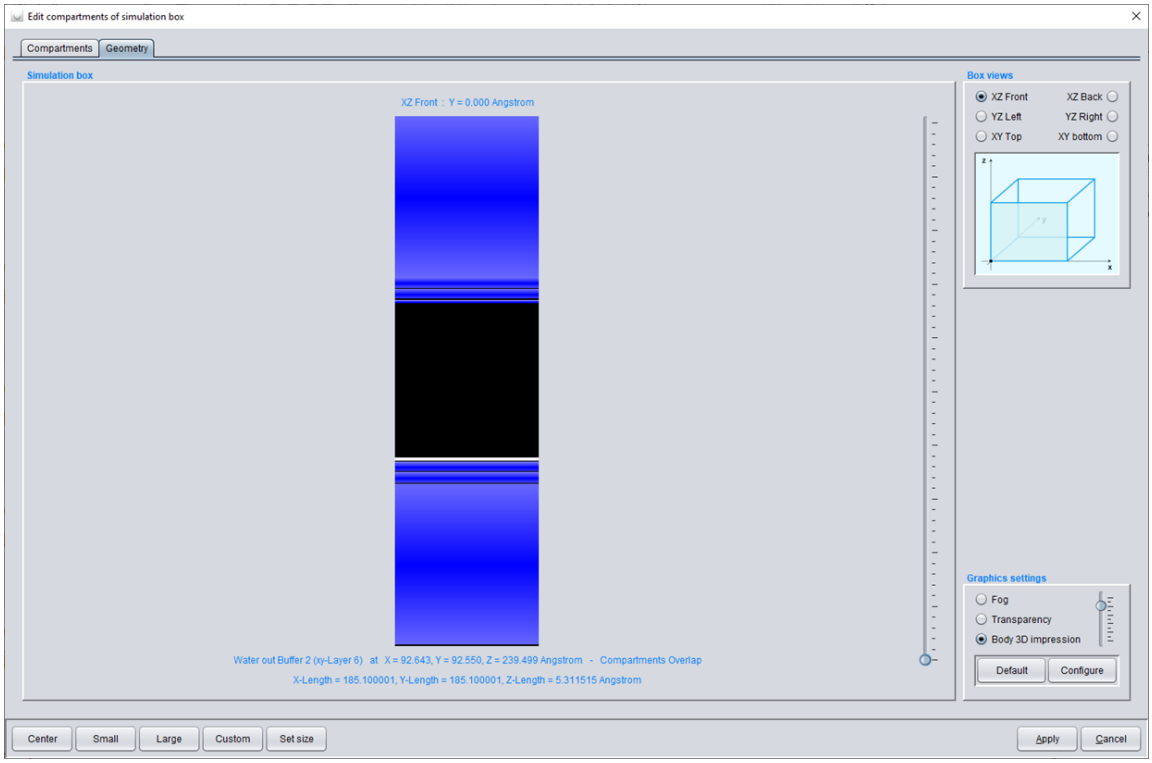


Figure 61. Completing the lower part of the simulation box by mirroring the upper half.

Finally create a new *Water out + Cyclotides*-layer with *93 %* of *WaterOut* molecules and *100 %* of KB1Y19 cyclotides, which are positioned *3D random* by default, i.e. the cyclotides are located randomly rotated at random positions within the layer without overlap. Select the ***Geometry*** tab to view the new layer in the (default and intended) middle of the simulation box (compare to figure 60).

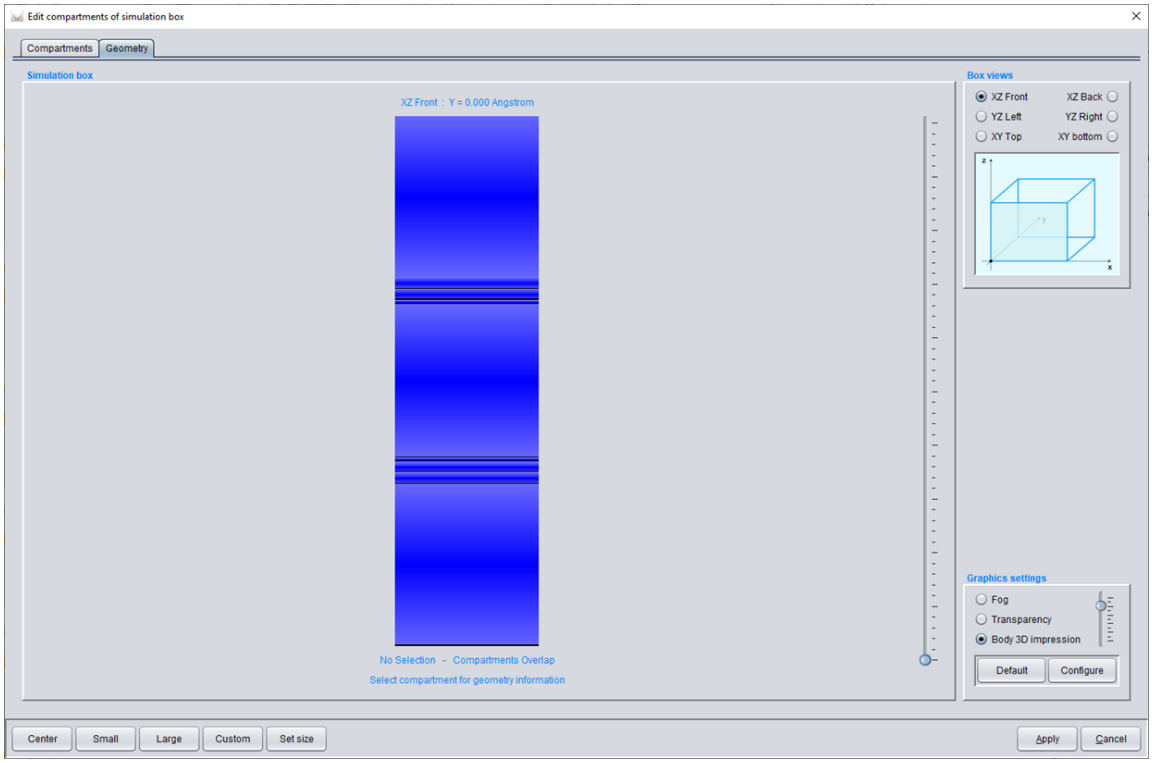


Figure 62. The complete start geometry with all prepared layers.

Switch back to the ***Compartments*** tab to control that all molecules have been assigned to compartments so that the ***Bulk*** volume is empty (compare to figure 61). ***Apply*** the compartment settings.

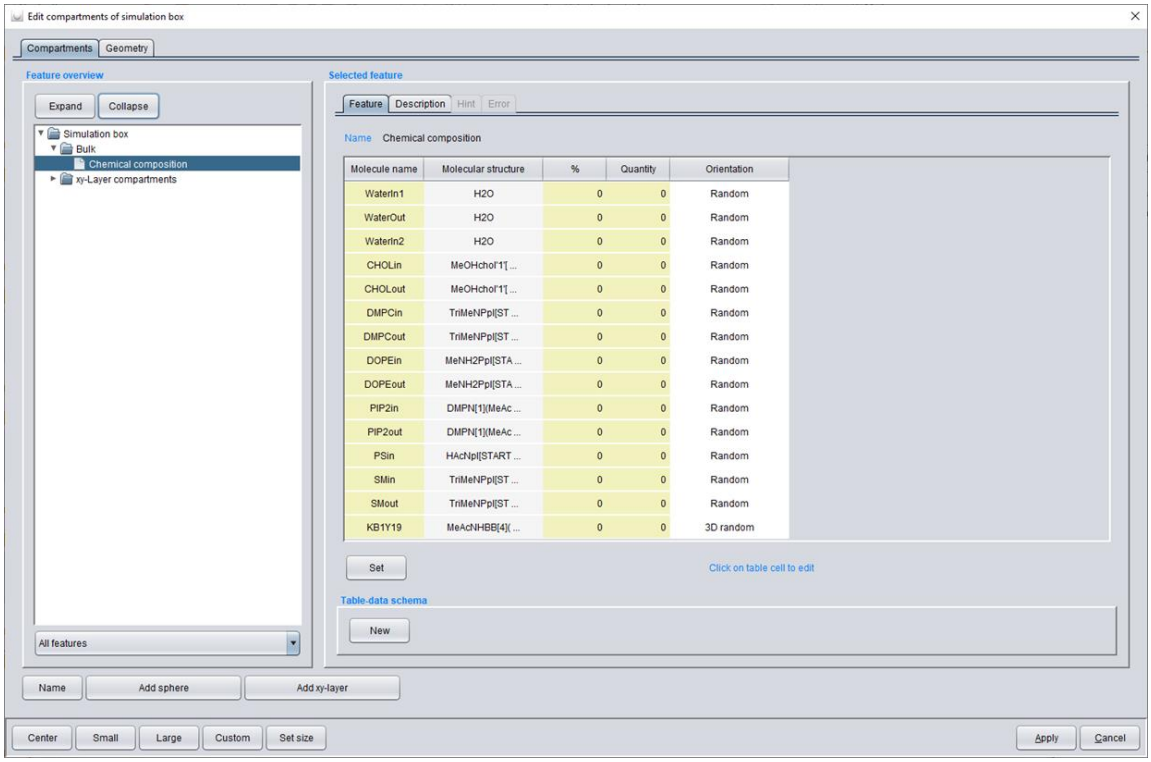


Figure 63. Controlling if the bulk volume is empty, meaning all molecules have been assigned to a compartment.

In order to see the prepared simulation box, press ***View*** to open the simulation box view dialog (compare to figure 62, top), which allows for an inspection of the (correct) compartment assignment (note that all water molecules and side chain particles of the cyclotides are excluded from display due to their particle display setting – the mutated amino acid of the cyclotide is shown in *PLUM*, the other "hydrophobic patch" amino acids in *WHITE*). ***Cancel*** the dialog (compare to figure 62, bottom).

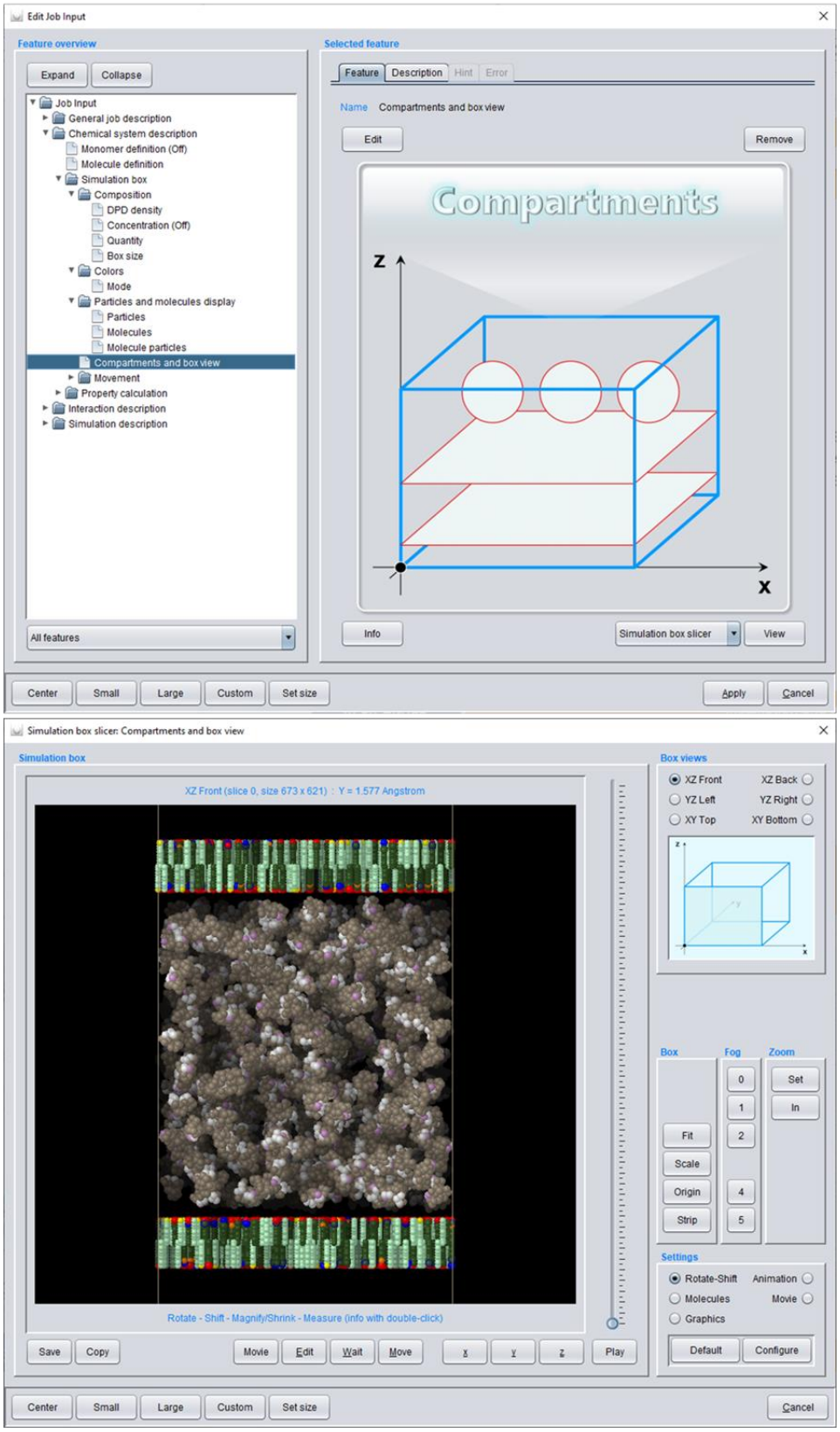


Figure 64. Viewing the simulation box with the completed starting geometry.

The next step is the application of molecule backbone forces for the cholesterol molecules. Select ***Job Input / Interaction Description / Molecule backbone forces*** to define backbone forces (harmonic springs) between particles with corresponding backbone tags in the SPICES string for the stiffness of a molecule's backbone (compare to figure 63).

For the default *CHOLin molecule* change the ***Distance [A]*** to *8.73* Angstrom and force constant ***k(BB)*** to *4*. Hit ***Insert*** to add a new line and change the ***Molecule name*** to *CHOLout*. Again change the ***Distance [A]*** to *8.73* Angstrom and force constant ***k(BB)*** to *4*.

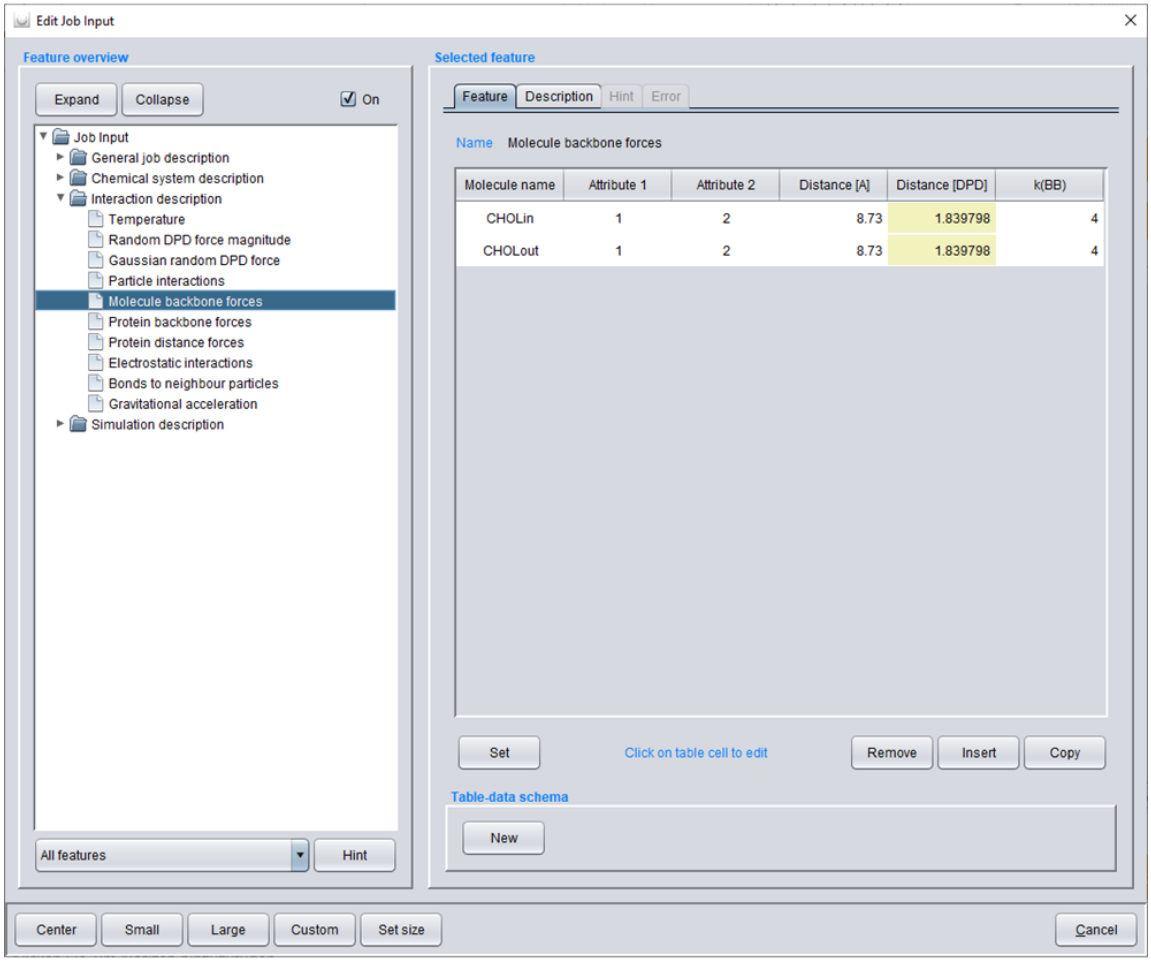


Figure 65. The application of molecule backbone forces for the cholesterol molecules.

Afterwards the protein distance forces have to be adapted. Therefore select ***Job Input / Interaction Description / Protein distance forces*** to set the backbone distance force constants k(BB) for all possible harmonic backbone springs to control the stiffness of the protein backbone. Hit the ***Set*** button for bulk settings (compare to figure 64, top). Select the (by default disabled) ***KB1Y19 – k(BB)*** setting and enable this it by ticking the On checkbox. Change the ***KB1Y19 – k(BB)*** value to *4* (compare to figure 64, middle). Confirm the entry with ***Apply*** to set all ***KB1Y19 - k(BB)*** values to 4 (compare to figure 64, bottom).

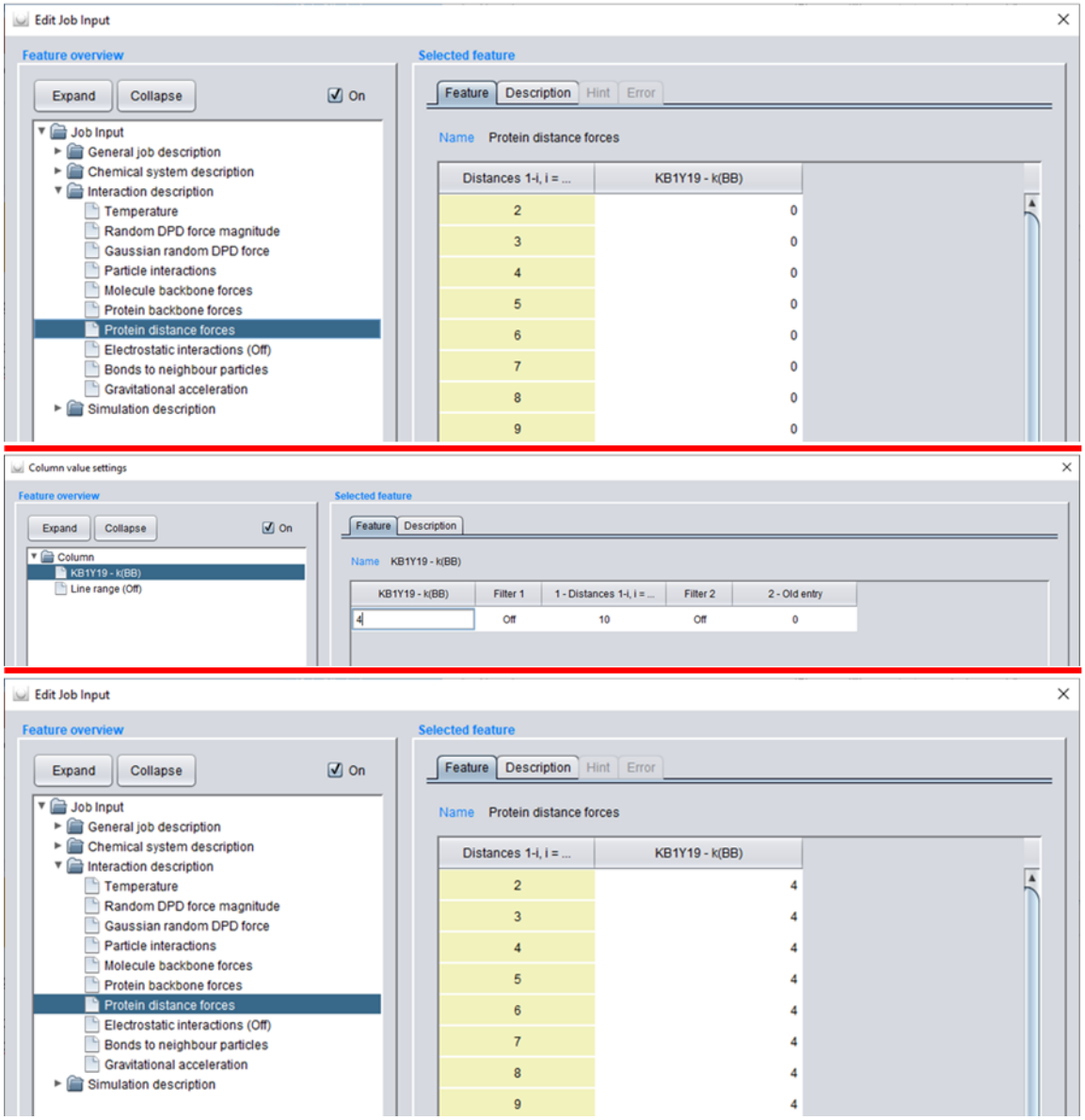


Figure 66. Adjustment of the protein distance forces for Kalata B1 molecules.

In order to enable electrostatic interactions, select ***Job Input / Interaction Description / Electrostatic interactions*** (compare to figure 65, top) and enable by ticking the On checkbox (compare to figure 65, bottom). Do not change the default settings.

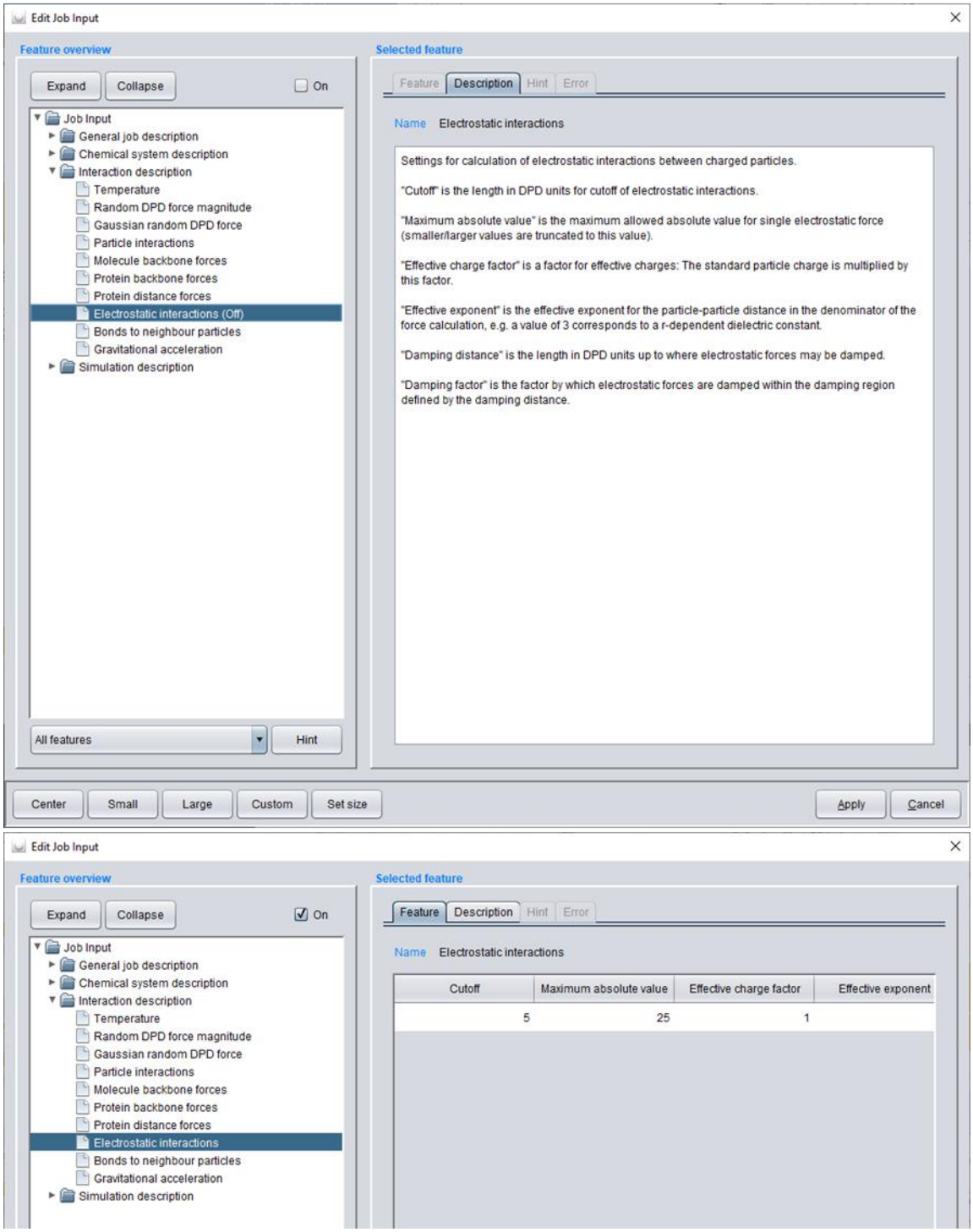


Figure 67. Activating the electrostatic interactions.

Afterwards the duration of the simulation should be specified. For this purpose, select ***Job Input / Simulation description / Simulation steps*** and change the ***Number of steps*** to *100000* to simulate about 5.8 µs (compare to figure 66).

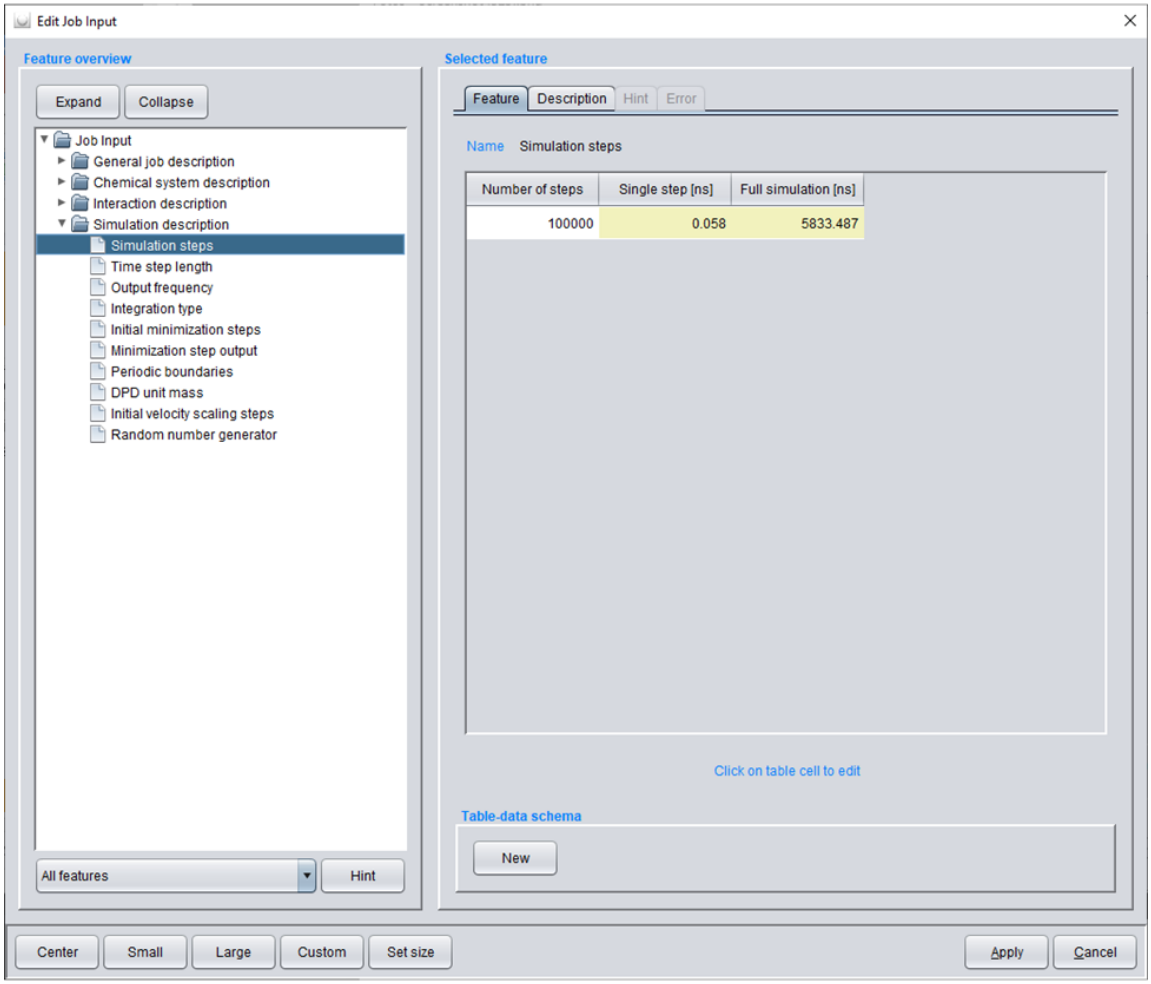


Figure 68. Adjustment of the desired simulation time.

Now the integration type has to be chosen (compare to figure 67). Select ***Job Input / Simulation description / Integration type*** and change ***Type*** to *S1MVV* (with ***Parameter 1*** being *true*).

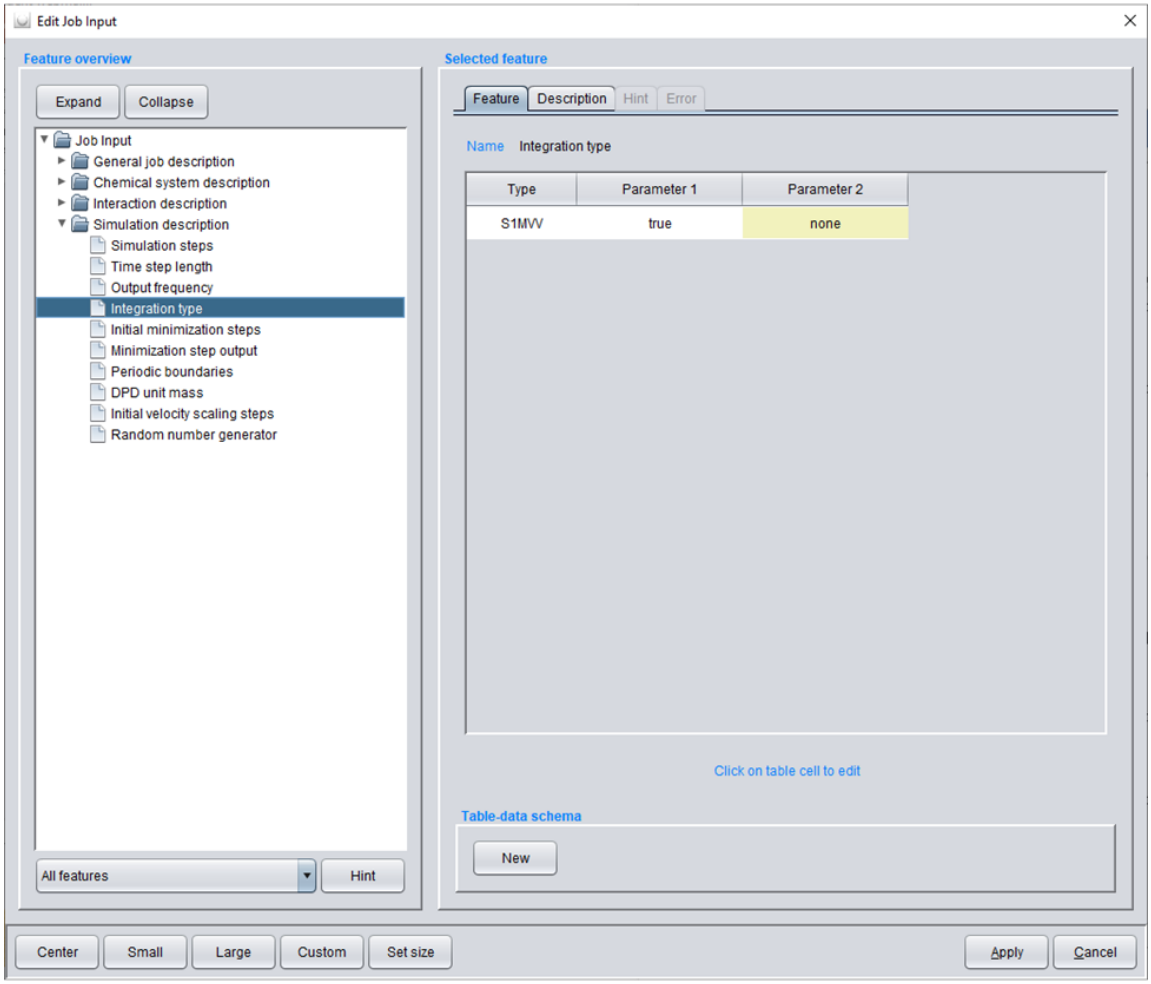


Figure 69. Specifying the integration type for the simulation.

If the initial minimization process should be observable after the simulation, select ***Job Input / Simulation description / Minimization step output*** and change to *true* (compare to figure 68).

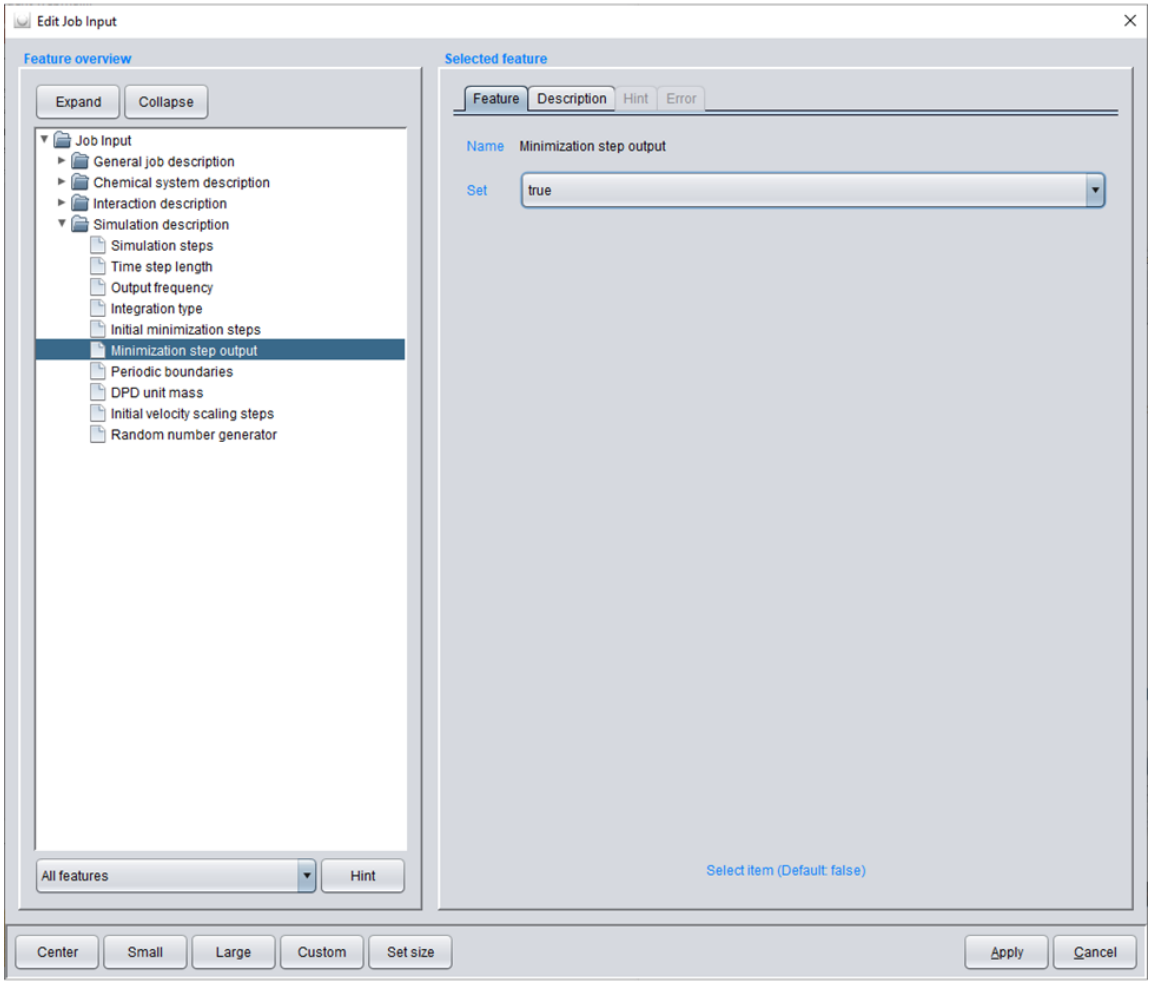


Figure 70. Making the minimization process visible after the whole simulation job has run.

In the last step before starting the simulation job, the periodic boundaries setting has to be adjusted for the z-axis. Select ***Job Input / Simulation description / Periodic boundaries*** .and change ***Periodic boundaries*** for the ***z***-axis to *false* (compare to figure 69).

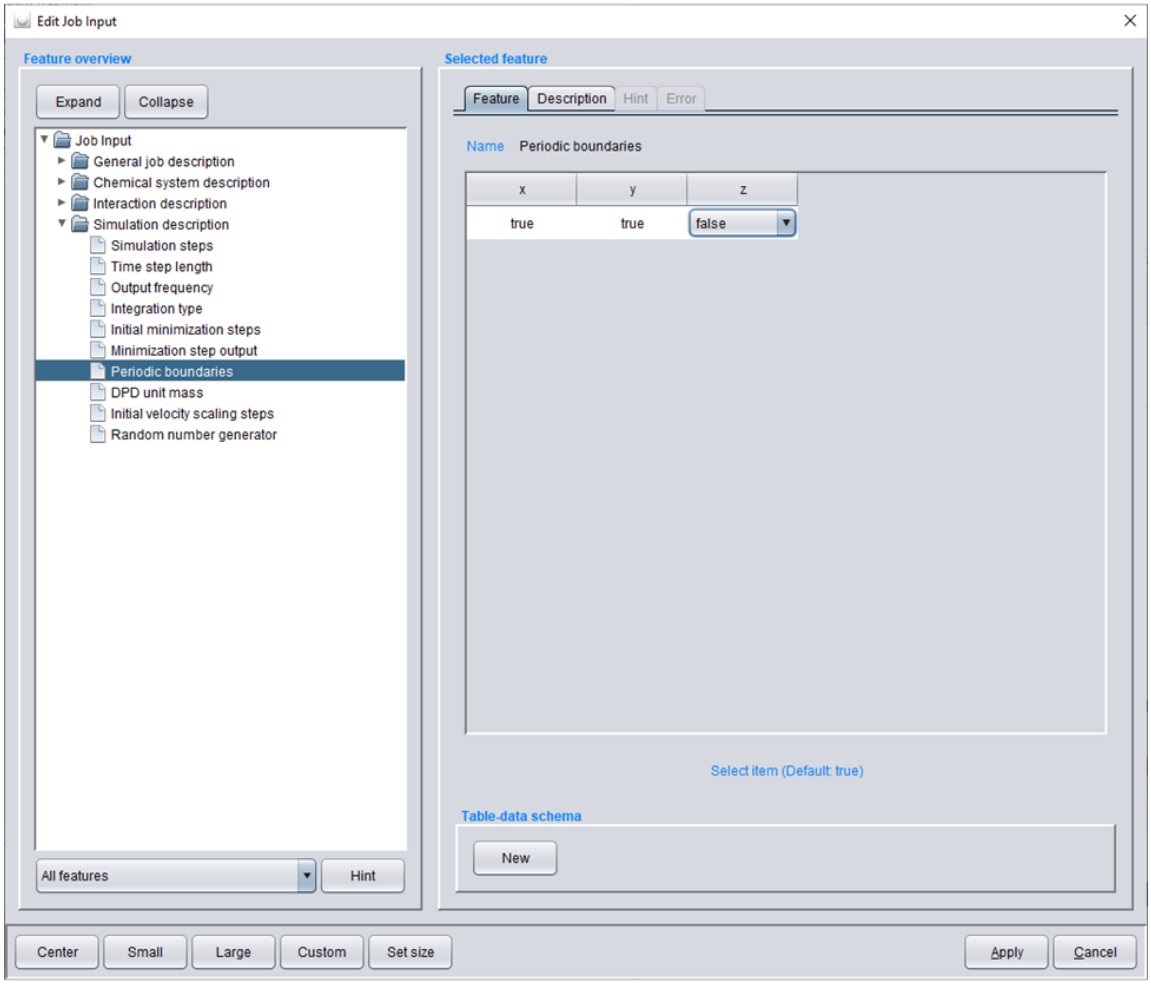


Figure 71. Adjustment of the periodic boundaries conditions.

Finally confirm all job input settings with ***Apply*** so that the new Job Input appears in the ***Job Inputs*** list (compare to figure 70, top). Change to the ***Excution*** tab (compare to figure 70, bottom) in order to start the new job (details are not covered by this tutorial).

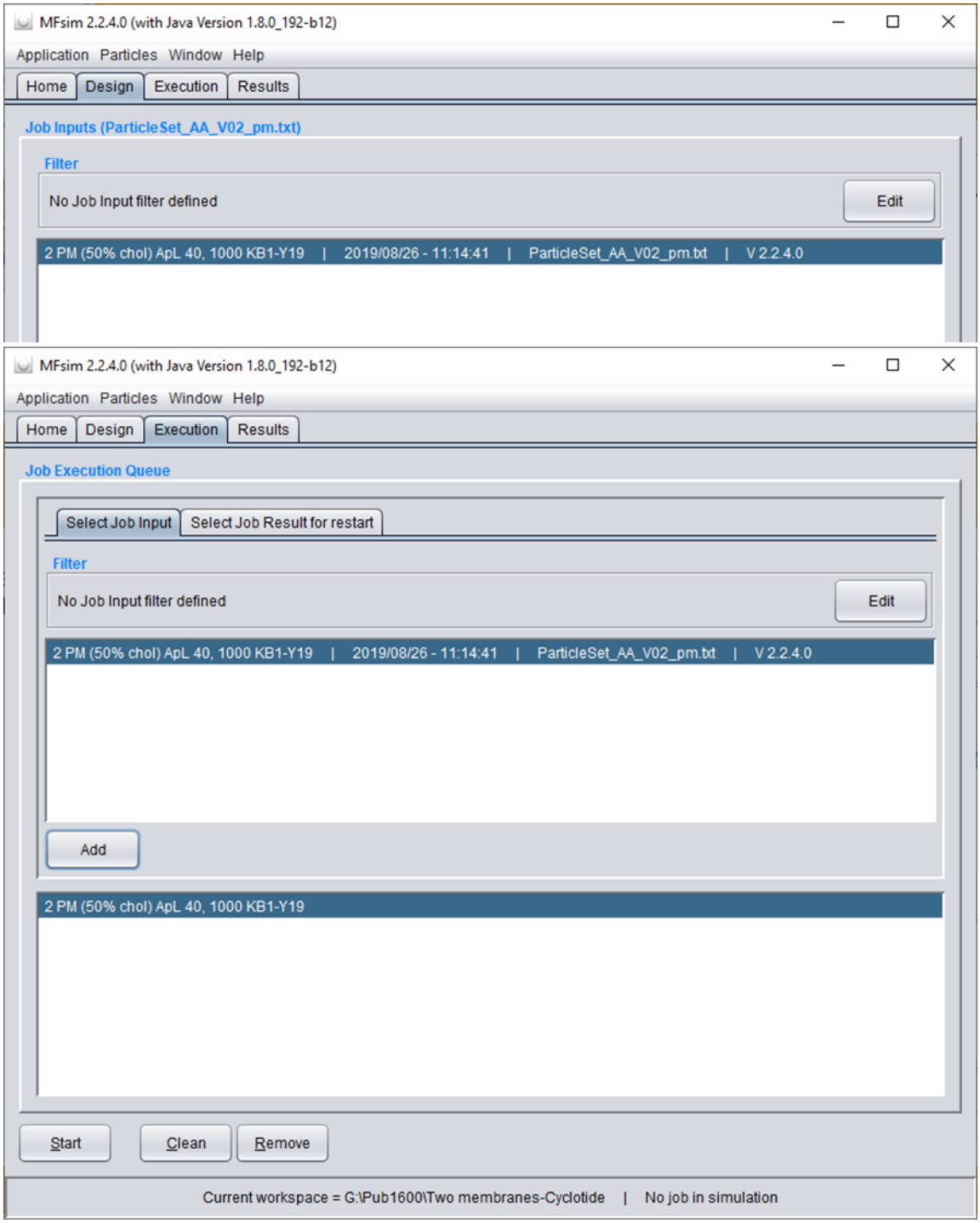


Figure 72. Starting the newly created simulation job.

After the simulation is completed, the successfully executed job generates a job result on the ***Results*** tab (compare to figure 71, top). Hit the ***View*** button to inspect the simulation results (compare to figure 71, bottom).

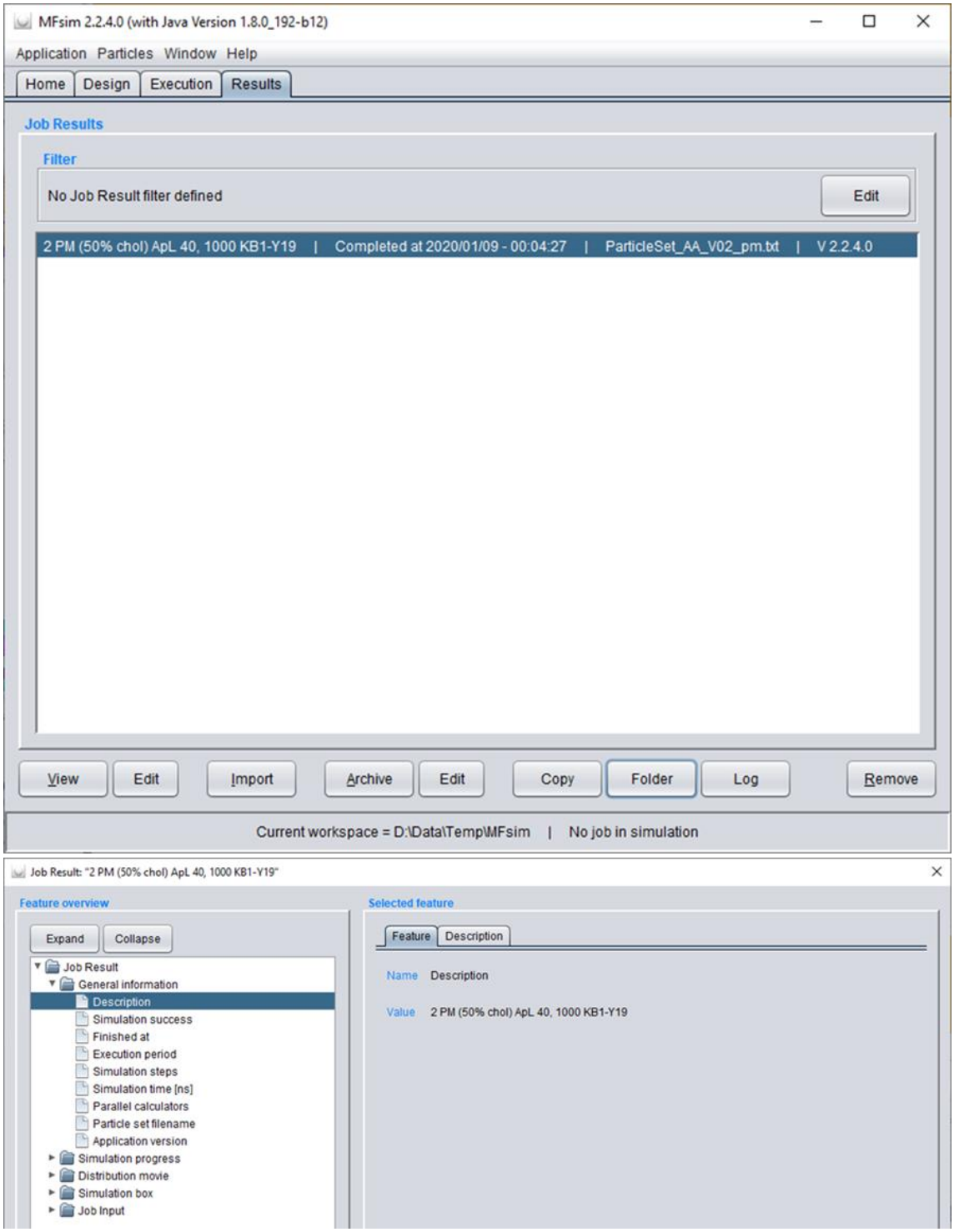


Figure 73. Evaluating the simulation results.

Now select ***Job Result / Simulation box / After simulation*** and hit the ***View*** button to open the simulation box view dialog for the last simulation step (compare to figure 72, top). Hit the ***Set*** button of the ***Zoom*** quick settings panel (compare to figure 72, bottom).

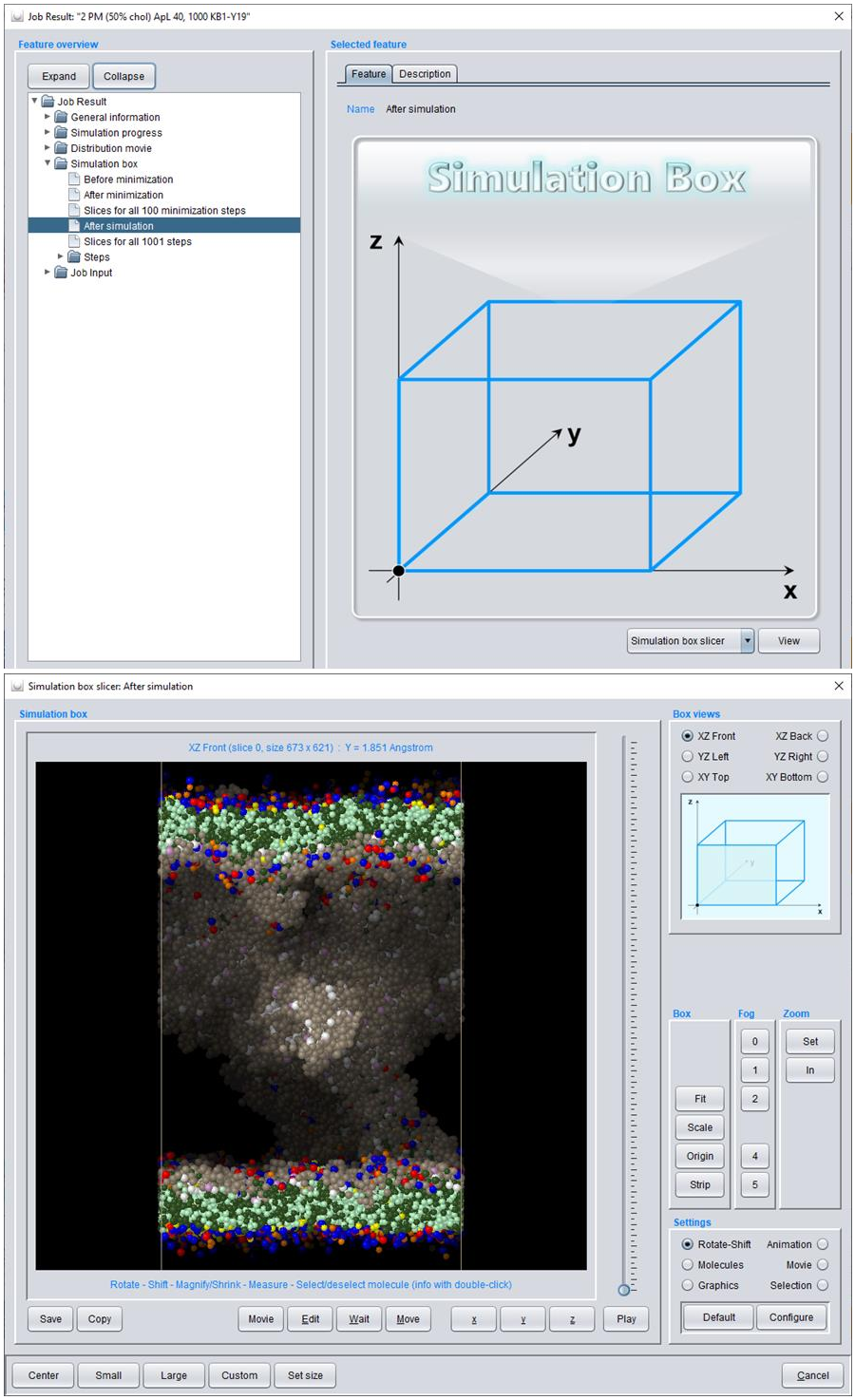


Figure 74. Analysing the simulation box.

In order to zoom into the cyclotide/water compartment change the ***z [Å]*** from *242* to *439* Ångstrom (these boundaries are to be derived from the corresponding compartment settings). Confirm with ***Apply*** (compare to figure 73).

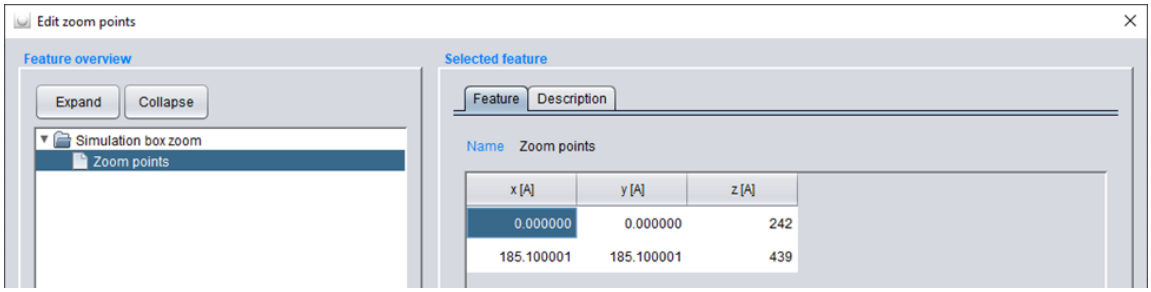


Figure 75. Zooming into the simulation box,

Hit the ***Bins*** button of the ***Zoom*** quick settings panel (compare to figure 74).

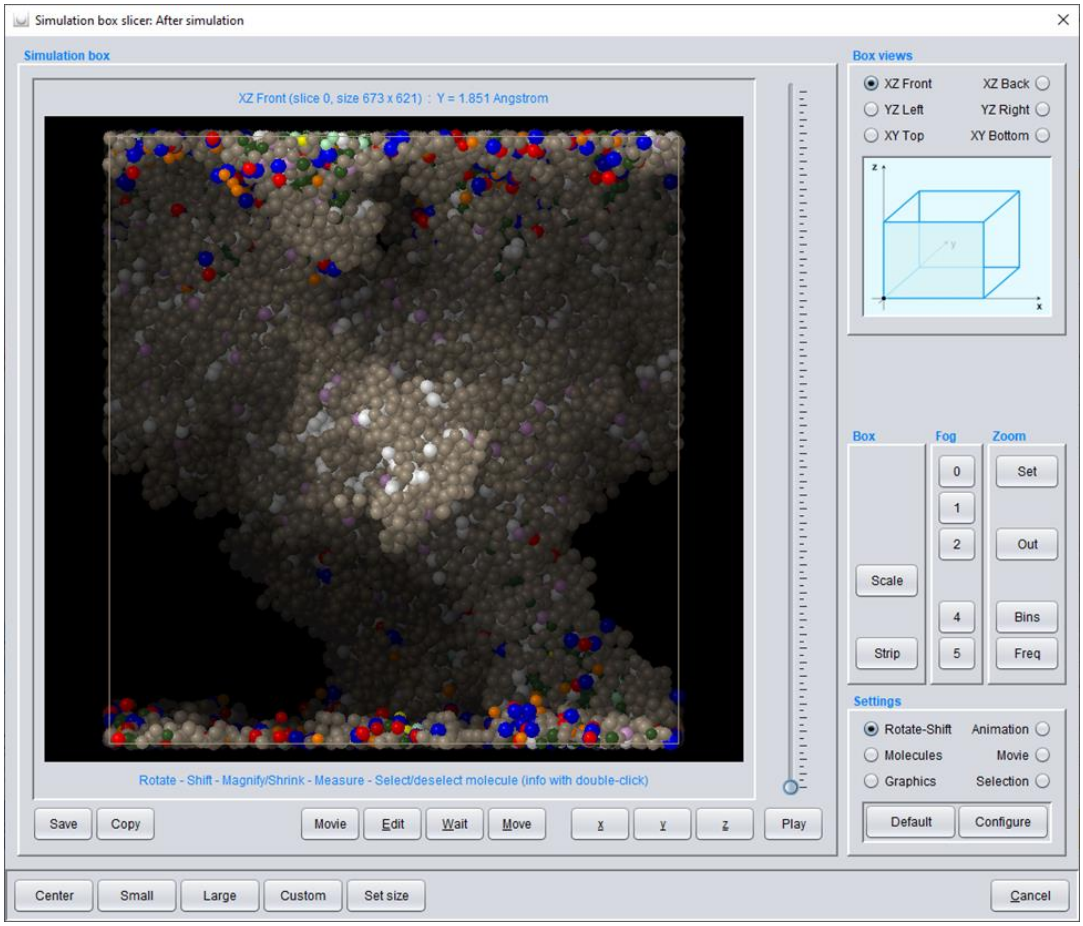


Figure 76. Viewing the zoomed-in section of the simulation box.

Now change the ***Number of volume bins*** to *10*. Confirm with ***Apply*** (compare to figure 75).

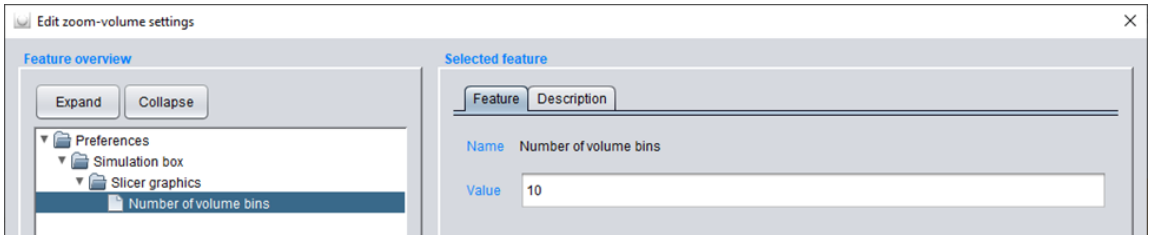


Figure 77. Setting the number of volume bins to 10.

In order to analyze the frequency distributions of molecules or particles, hit the ***Freq*** button of the ***Zoom*** quick settings panel (compare to figure 76).

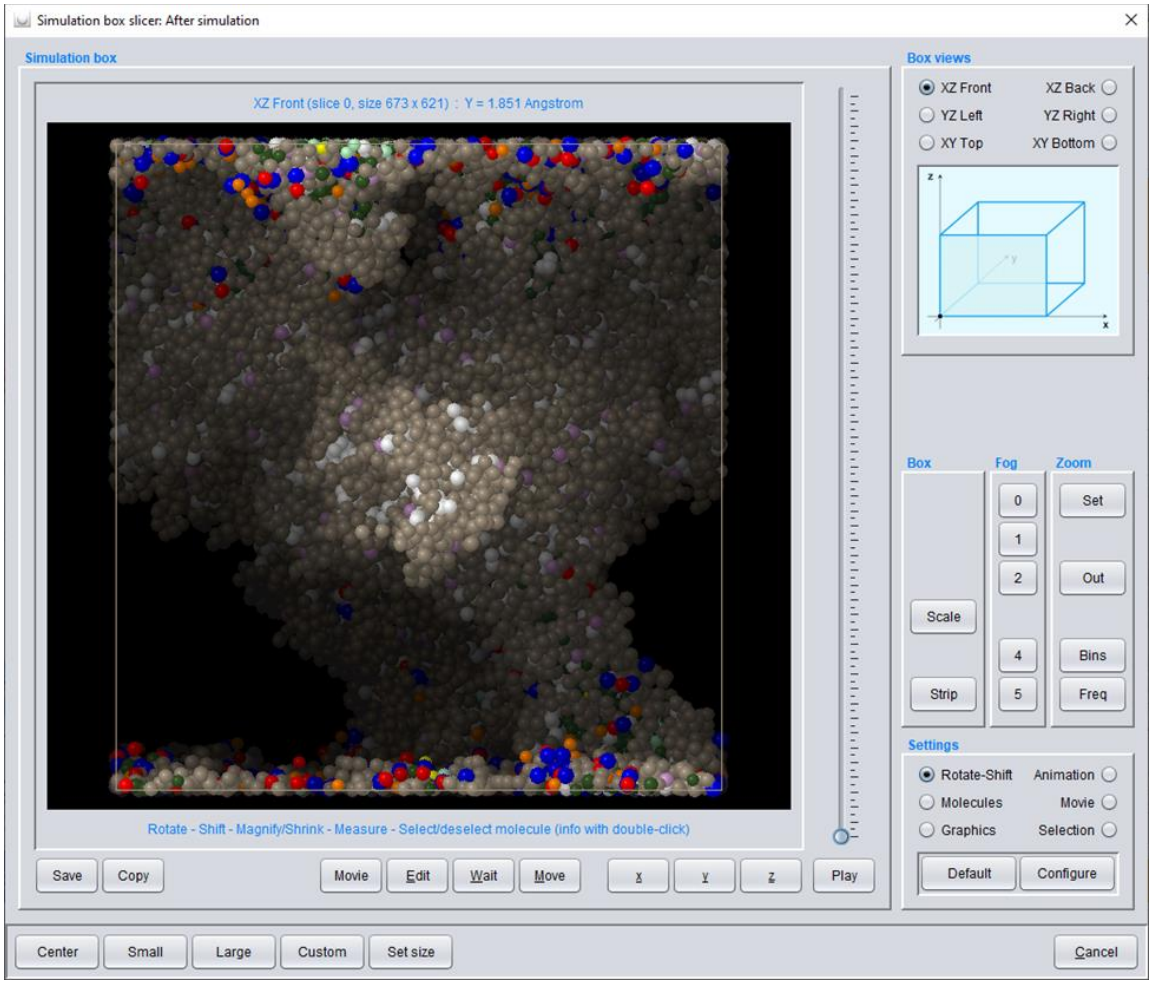


Figure 78. Opening the frequency distributions of molecules or particles.

Subsequently the ***Zoom-volume information*** dialog opens (compare to figure 77, top). Select ***Zoom-volume frequency distributions / Along z direction / Particles / Et along z-axis*** to view the Et-particle distribution (partitioned into 10 bins) along the z-axis in the compartment (compare to figure 77, bottom). Hit the ***Show*** button

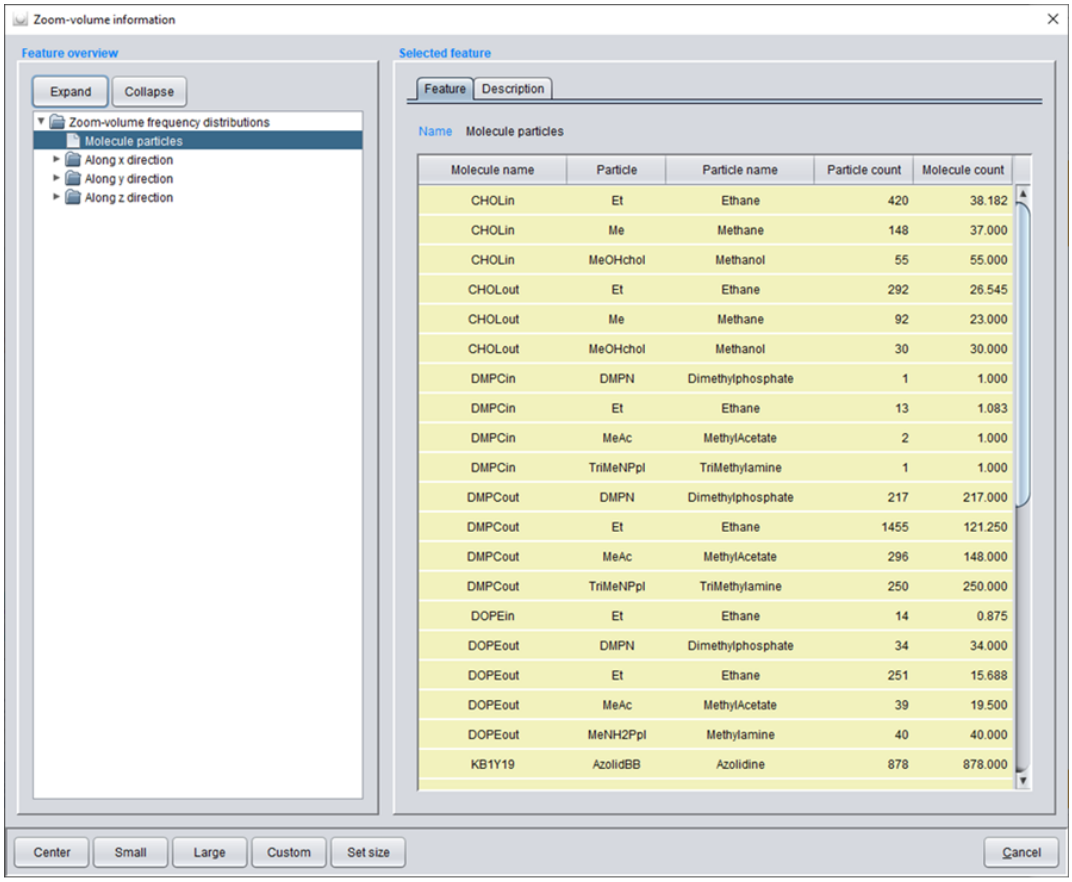


Figure 79. Investigating the zoom-volume information, particularly the Et-particle distribution in z-direction.

Now a diagram chart of the Et-distribution opens (compare to figure 78). The particle distributions may be used for further evaluation operations concerning the cyclotide/membrane interaction.

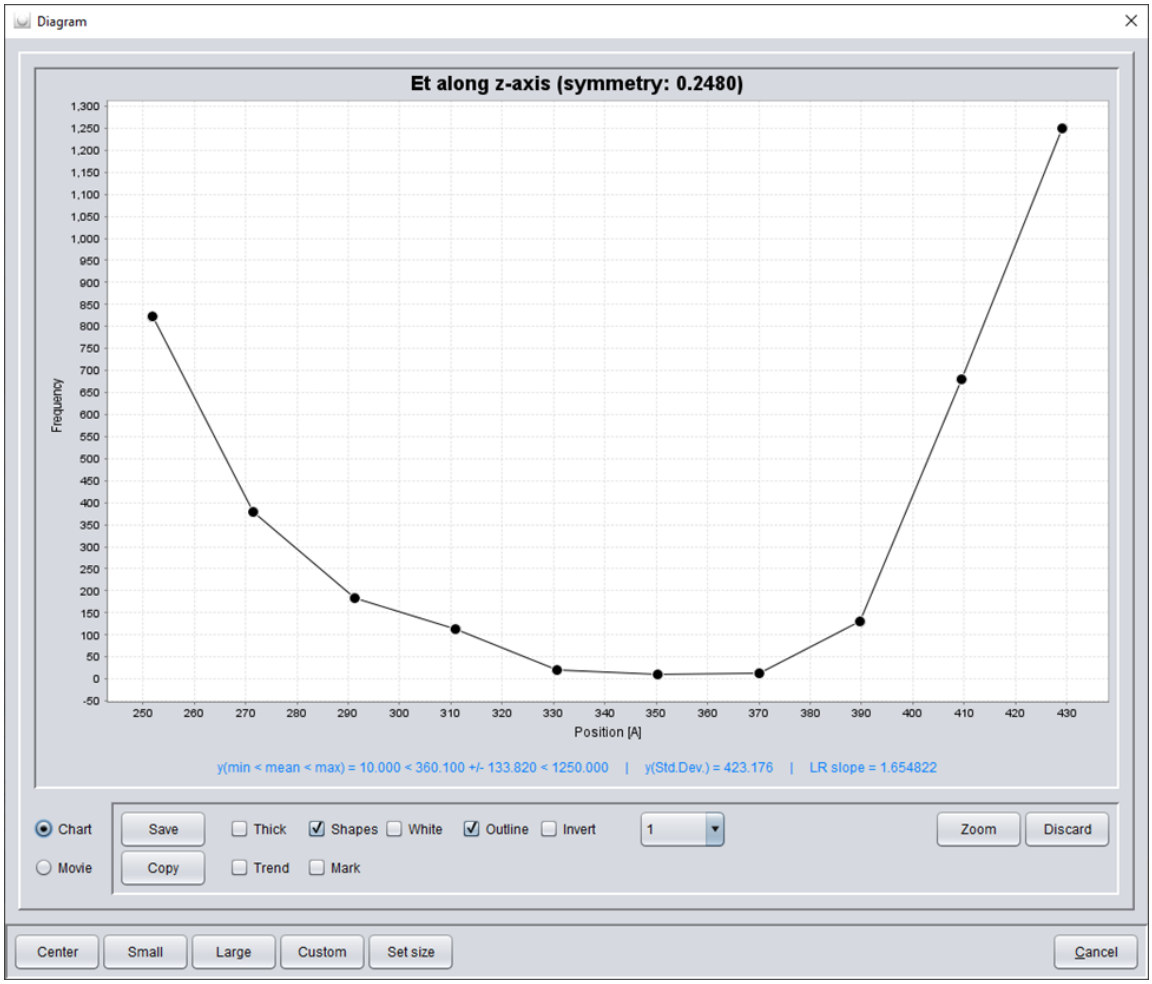


Figure 80. The diagram of the Et-particle distribution along the z-axis.

8 Tables

Table 1. Reproducibility of lipid extraction activity characterized by rate constants r_{MD} of Kalata B1 with NoC-PM and 50C-PM for simulation jobs with different random start geometries/seeds.

kB1 / NoC-PM

	Job 1	**Job 2**	**Job 3**	**Job 4**
r_{MD}	1.3	1.3	1.5	1.5

kB1 / 50C-PM

	Job 1	**Job 2**	**Job 3**	**Job 4**
r_{MD}	0.9	0.4	1.0	1.0

kB1 / 50C-PM - cholesterol

	Job 1	**Job 2**	**Job 3**	**Job 4**
r_{MD}	0.5	0.2	0.5	0.5

kB1 / 50C-PM - phospholipids

	Job 1	**Job 2**	**Job 3**	**Job 4**
r_{MD}	1.3	0.6	1.4	1.5

Table 2. Lipid extraction activity characterized by rate constants r_{MD} of Kalata B1 "hydrophobic patch" mutants with NoC-PM and 50C-PM.

NoC-PM

	kB1-W19Y	**kB1-W19Y-P20S-V21T**	**kB1-W19Y-P20S-V21T-L27T-P28S-V29T**
r_{MD}	1.0	0.4	0.0

50C-PM

	kB1-W19Y	**kB1-W19Y-P20S-V21T**	**kB1-W19Y-P20S-V21T-L27T-P28S-V29T**
r_{MD}	0.5	0.0	0.0

50C-PM - cholesterol

	kB1-W19Y	**kB1-W19Y-P20S-V21T**	**kB1-W19Y-P20S-V21T-L27T-P28S-V29T**
r_{MD}	0.2	0.0	0.0

50C-PM - phospholipids

	kB1-W19Y	**kB1-W19Y-P20S-V21T**	**kB1-W19Y-P20S-V21T-L27T-P28S-V29T**
r_{MD}	0.8	0.1	0.0

Table 3. Charged residues at pH 7.4 of Kalata B1, Cycloviolacin O2 and their mutants (see section *5.4.3* and [Truszkowski2015] for details about charging).

Cyclotide	Positively charged residues	Negatively charged residues	Total cyclotide net charge
kB1	1	1	0
kB1-W19Y	1	1	0
kB1-W19Y-P20S-V21T	1	1	0
kB1-W19Y-P20S-V21T-L27T-P28S-V29T	1	1	0
kB1-E3Q	1	0	1
kB1-R24W	0	1	-1
kB1-E3Q-R24W	0	0	0
kB1-W19K	2	1	1
kB1-T16K-N25K	3	1	2
cO2	3	1	2
cO2-W10Y	3	1	2
cO2-W10Y-I11T-P12S	3	1	2
cO2-W10Y-I11T-P12S-I14T-A17S-I18T	3	1	2
cO2-E6Q	3	0	3
cO2-K23Q-K25Q	1	1	0
cO2-K23Q-K25Q-R29W	0	1	-1
cO2-E6Q-K23Q-K25Q-R29W	0	0	0

Table 4. Lipid extraction activity characterized by rate constants r_{MD} of differently charged Kalata B1 mutants with NoC-PM and 50C-PM.

NoC-PM

	kB1-E3Q	**kB1-R24W**	**kB1-E3Q-R24W**	**kB1-W19K**	**kB1-T16K-N25K**
r_{MD}	1.8	1.8	2.0	0.9	1.6

50C-PM

	kB1-E3Q	**kB1-R24W**	**kB1-E3Q-R24W**	**kB1-W19K**	**kB1-T16K-N25K**
r_{MD}	0.4	1.0	1.0	0.3	1.0

50C-PM - cholesterol

	kB1-E3Q	**kB1-R24W**	**kB1-E3Q-R24W**	**kB1-W19K**	**kB1-T16K-N25K**
r_{MD}	0.3	0.5	0.5	0.1	0.5

50C-PM - phospholipids

	kB1-E3Q	**kB1-R24W**	**kB1-E3Q-R24W**	**kB1-W19K**	**kB1-T16K-N25K**
r_{MD}	0.5	1.4	1.5	0.4	1.4

Table 5. Lipid extraction activity characterized by rate constants r_{MD} of Cycloviolacin O2 and its "hydrophobic patch" mutants with NoC-PM and 50C-PM.

NoC-PM

	cO2	**cO2-W10Y**	**cO2-W10Y-I11T-P12S**	**cO2-W10Y-I11T-P12S-I14T-A17S-I18T**
r_{MD}	1.5	1.2	1.3	1.0

50C-PM

	cO2	**cO2-W10Y**	**cO2-W10Y-I11T-P12S**	**cO2-W10Y-I11T-P12S-I14T-A17S-I18T**
r_{MD}	1.1	0.7	0.3	0.0

50C-PM - cholesterol

	cO2	**cO2-W10Y**	**cO2-W10Y-I11T-P12S**	**cO2-W10Y-I11T-P12S-I14T-A17S-I18T**
r_{MD}	0.4	0.2	0.0	0.0

50C-PM - phospholipids

	cO2	**cO2-W10Y**	**cO2-W10Y-I11T-P12S**	**cO2-W10Y-I11T-P12S-I14T-A17S-I18T**
r_{MD}	1.7	1.2	0.4	0.0

Table 6. Lipid extraction activity characterized by rate constants r_{MD} of differently charged Cycloviolacin O2 mutants with NoC-PM and 50C-PM.

NoC-PM

	cO2-E6Q	**cO2-K23Q-K25Q**	**cO2-K23Q-K25Q-R29W**	**cO2-E6Q-K23Q-K25Q-R29W**
r_{MD}	2.2	1.0	1.4	1.2

50C-PM

	cO2-E6Q	**cO2-K23Q-K25Q**	**cO2-K23Q-K25Q-R29W**	**cO2-E6Q-K23Q-K25Q-R29W**
r_{MD}	1.1	0.5	0.9	0.5

50C-PM - cholesterol

	cO2-E6Q	**cO2-K23Q-K25Q**	**cO2-K23Q-K25Q-R29W**	**cO2-E6Q-K23Q-K25Q-R29W**
r_{MD}	0.5	0.1	0.2	0.1

50C-PM - phospholipids

	cO2-E6Q	**cO2-K23Q-K25Q**	**cO2-K23Q-K25Q-R29W**	**cO2-E6Q-K23Q-K25Q-R29W**
r_{MD}	1.6	0.9	1.4	0.9

Table 7. Lipid extraction activity characterized by rate constants r_{MD} of Kalata B1 and Cycloviolacin O2 with NoC-DMPC-M, NoC-DOPE-M, NoC-invPM, 50C-DMPC-M, 50C-DOPE-M and 50C-invPM.

	kB1	**cO2**	**kB1**	**cO2**	**kB1**	**cO2**
	NoC-DMPC-M		**NoC-DOPE-M**		**NoC-invPM**	
r_{MD}	1.3	1.7	2.3	2.5	2.5	2.4

	kB1	**cO2**	**kB1**	**cO2**	**kB1**	**cO2**
	50C-DMPC-M		**50C-DOPE-M**		**50C-invPM**	
r_{MD}	0.6	1.1	0.8	1.2	0.9	1.0

cholesterol

	kB1	**cO2**	**kB1**	**cO2**	**kB1**	**cO2**
r_{MD}	0.3	0.2	0.3	0.3	0.5	0.2

phospholipids

	kB1	**cO2**	**kB1**	**cO2**	**kB1**	**cO2**
r_{MD}	0.9	1.9	1.2	1.9	1.2	1.6

9 References

[3DVecMath2020] 3D Vector Math Package. https://mvnrepository.com/artifact/javax.vecmath/vecmath. Accessed 31 January 2020.

[Allen1987] Allen MP, Tildesley DJ. Computer Simulation of Liquids. Oxford: Oxford University Press; 1987.

[ApacheIO2020] Apache Commons IO. http://commons.apache.org/proper/commons-io. Accessed 31 January 2020.

[ApacheLang2020] Apache Commons Lang. https://commons.apache.org/proper/commons-lang. Accessed 31 January 2020.

[ApacheRNG2020] Apache Commons RNG – Random Numbers Generators. http://commons.apache.org/proper/commons-rng. Accessed 31 January 2020.

[Bar1966] Bar RS, Deamer DW, Cornwell DG (1966) Surface Area of Human Erythrocyte Lipids: Reinvestigation of Experiments on Plasma Membrane. Science 153(3739):1010-1012

[Barry2003] Barry DG, Daly NL, Clark RJ, Sando L, Craik DJ (2003) Linearization of a Naturally Occurring Circular Protein Maintains Structure but Eliminates Hemolytic Activity. Biochemistry 42:6688−6695

[Bermúdez2004] Bermúdez H, Hammer DA, Discher DE (2004) Effect of Bilayer Thickness on Membrane Bending Rigidity. Langmuir 20:540−543

[Besold2000] Besold G, Vattulainen I, Karttunen M, Polson JM. Towards Better Integrators for Dissipative Particle Dynamics Simulations. Physical Review E 2000;62(6): 7611-7614.

[BioJava2020] BioJava. https://biojava.org. Accessed 31 January 2020.

[BIOVIA2020] BIOVIA Materials Studio. https://www.3dsbiovia.com/products/collaborative-science/biovia-materials-studio. Accessed 31 January 2020.

[Burman2011] Burman R, Strömstedt AA, Malmsten M, Göransson U (2011), Cyclotide–Membrane Interactions: Defining Factors of Membrane Binding, Depletion and Disruption. Biochimica et Biophysica Acta 1808:2665–2673

[Burman2011b] Burman R, Herrmann A, Tran R, Kivelä JE, Lomize A, Gullbo J, Göransson U (2011) Cytotoxic Potency of Small Macrocyclic Knot Proteins:

Structure–Activity and Mechanistic Studies of Native and Chemically Modified Cyclotides. Organic and Biomolecular Chemistry 9:4306-4314

[Burman2014] Burman R, Gunasekera S, Strömstedt A, Göransson U (2014) Chemistry and Biology of Cyclotides: Circular Plant Peptides Outside the Box. Journal of Natural Products 77:724−736

[Castillo2013] Castillo N, Monticelli L, Barnoud J, Tieleman DP (2013) Free Energy of WALP23 Dimer Association in DMPC, DPPC, and DOPC Bilayers. Chemistry and Physics of Lipids 169:95-105

[Craik1999] Craik DJ, Daly NL, Bond T, Waine C (1999) Plant Cyclotides: A Unique Family of Cyclic and Knotted Proteins that Defines the Cyclic Cystine Knot Structural Motif. Journal of Molecular Biology 294:1327-1336

[Cranfield2017] Cranfield CG, Henriques ST, Martinac B, Duckworth P, Craik DJ, Cornell B (2017) Kalata B1 and Kalata B2 Have a Surfactant-Like Activity in Phosphatidylethanolomine-Containing Lipid Membranes. Langmuir 33:6630−6637

[CULGI2020] CULGI. https://www.culgi.com. Accessed 31 January 2020.

[Daleke2008] Daleke DL (2008) Regulation of Phospholipid Asymmetry in the Erythrocyte Membrane. Current Opinion in Hematology 15:191–195

[deVeer2019] de Veer J, Kan MW, Craik DJ (2019) Cyclotides: From Structure to Function. Chemical Reviews 119:12375−12421

[DL_MESO2020] DL_MESO. http://www.cse.clrc.ac.uk/ccg/software/DL_MESO/. Accessed 31 January 2020.

[DPDmacs2020] DPDmacs. https://www.softsimu.net/softsimu-wiki/doku.php?id=softsimu:tutorials:dpdmacs. Accessed 31 January 2020.

[Drefahl2011] Drefahl A (2011) CurlySMILES: A Chemical Language to Customize and Annotate Encodings of Molecular and Nanodevice Structures. Journal of Cheminformatics 3:1.

[Dufresne2015] Dufresne Y, Noé L, Leclère V, Pupin M (2015) Smiles2Monomers: A Link Between Chemical and Biological Structures for Polymers. Journal of Cheminformatics 7:62.

[Engel2018] Engel T (Ed.), Gasteiger J (Ed.) (2018) Chemoinformatics: Basic Concepts and Methods. Weinheim: WILEY-VCH.

[Engel2018b] Engel T (Ed.), Gasteiger J (Ed.) (2018) Applied Chemoinformatics: Achievements and Future Opportunities. Weinheim: WILEY-VCH.

[Engelman1969] Engelman DM (1969) Surface Area per Lipid Molecule in the Intact Membrane of the Human Red Cell. Nature 223:1279-1280

[Espanol1995] Espanol P, Warren P (1995) Statistical Mechanics of Dissipative Particle Dynamics. Europhysics Letters 30(4):191-196

[Espanol1995b] Espanol P (1995) Hydrodynamics from Dissipative Particle Dynamics. Physical Review E 52(2):1734-1742.

[Espanol2017] Espanol P, Warren PB (2017) Perspective: Dissipative Particle Dynamics. Journal of Chemical Physics 146:150901. doi: 10.1063/1.4979514

[ESPResSo2020] ESPResSo. http://espressomd.org/wordpress. Accessed 31 January 2020.

[FFmpeg2020] FFmpeg – A Complete, Cross-Platform Solution to Record, Convert and Stream Audio and Video. https://ffmpeg.org. Accessed 31 January 2020.

[Flory1953] Flory PJ. Principles of Polymer Chemistry. Ithaca, New York: Cornell University Press; 1953.

[Frenkel2002] Frenkel D, Smit B. Understanding Molecular Simulation. From Algorithms to Applications. 2nd ed. London: Academic Press; 2002.

[Gao2007] Gao L, Shillcock J, Lipowsky R (2007) Improved Dissipative Particle Dynamics Simulations of Lipid Bilayers. The Journal of Chemical Physics 126:015101

[Gerlach2010] Gerlach SL, Rathinakumar R, Chakravarty G, Göransson U, Wimley WC, Darwin SP, Mondal D (2010) Anticancer and Chemosensitizing Abilities of Cycloviolacin O2 from Viola Odorata and Psyle Cyclotides from Psychotria leptothyrsa. Peptide Science 94(5):617-625

[Ghani2017] Ghani HA, Henriques ST, Huang YH, Swedberg JE, Schroeder CI, Craik DJ (2017) Structural and Functional Characterization of Chimeric Cyclotides from the Möbius and Trypsin Inhibitor Subfamilies. Peptide Science 108:e22927

[GNU-GPL2020] GNU General Public License. http://www.gnu.org/licenses. Accessed 31 January 2020.

[Göransson2009] Göransson U, Herrmann A, Burman R, Haugaard-Jönsson LM, Rosengren KJ (2009) The Conserved Glu in the Cyclotide Cycloviolacin O2 Has a Key Structural Role. ChemBioChem 10:2354–2360

[Göransson2012] Göransson U, Burman R, Gunasekera S, Strömstedt AA, Rosengren KJ (2012) Circular Proteins from Plants and Fungi. Journal of Biological Chemistry 287(32):27001–27006

[Grage2017] Grage SL, Sani MA, Cheneval O, Henriques ST, Schalck C, Heinzmann R, Mylne JS, Mykhailiuk PK, Afonin S, Komarov IV, Separovic F, Craik DJ, Ulrich1 AS (2017) Orientation and Location of the Cyclotide Kalata B1 in Lipid Bilayers Revealed by Solid-State NMR. Biophysical Journal 112:630–642

[GraphStream2020] GraphStream – A Dynamic Graph Library. http://graphstream-project.org. Accessed 31 January 2020.

[Gromacs2020] Gromacs. http://www.gromacs.org. Accessed 31 January 2020.

[Groot1997] Groot RD, Warren P (1997) Dissipative Particle Dynamics: Bridging the Gap Between Atomistic and Mesoscopic Simulation. Journal of Chemical Physics 107(11):4423-4435

[Groot1998] Groot RD, Madden TJ (1998) Dynamic Simulation of Diblock Copolymer Microphase Separation. Journal of Chemical Physics 105(20):8713-8724

[Groot2003] Groot RD (2003) Electrostatic Interactions in Dissipative Particle Dynamics – Simulation of Polyelectrolytes and Anionic Surfactants. Journal of Chemical Physics 118(24):11265-11277.

[González-Melchor2006] González-Melchor M, Mayoral E, Velázquez ME, Alejandre J. Electrostatic Interactions in Dissipative Particle Dynamics Using the Ewald Sums. Journal of Chemical Physics 2006;125:224107/1-8.

[Hamilton2003] Hamilton JA (2003) Fast Flip-Flop of Cholesterol and Fatty Acids in Membranes: Implications for Membrane Transport Proteins. Current Opinion in Lipidology 14:263–271

[Henriques2011] Henriques ST, Huang YH, Rosengren KJ, Franquelim HG, Carvalho FA, Johnson A, Sonza S, TachedjianG, Castanho MARB, Daly NL, Craik DJ (2011) Decoding the Membrane Activity of the Cyclotide Kalata B1. Journal of Biological Chemistry 286(27):24231–24241

[Henriques2012] Henriques ST, Huang YH, Castanho MARB, Bagatolli LA, Sonza S, Tachedjian G, Daly NL, Craik DJ (2012) Phosphatidylethanolamine Binding is a Conserved Feature of Cyclotide-Membrane Interactions. Journal of Biological Chemistry 287:33629−33643

[Henriques2012b] Henriques ST, Craik DJ (2012) Importance of the Cell Membrane on the Mechanism of Action of Cyclotides. Chemical Biology 7:626−636

[Henriques2014] Henriques ST, Huang YH, Chaousis S, Wang CK, Craik DJ (2014) Anticancer and Toxic Properties of Cyclotides are Dependent on Phosphatidylethanolamine Phospholipid Targeting. ChemBioChem 15:1956–1965

[Henriques2015] Henriques ST, Huang YH, Chaousis S, Sani MA, Poth AG, Separovic F, Craik DJ (2015) The Prototypic Cyclotide Kalata B1 Has a Unique Mechanism of Entering Cells. Chemistry & Biology 22:1087–1097

[Henriques2017] Henriques ST and Craik DJ (2017) Cyclotide Structure and Function: The Role of Membrane Binding and Permeation. Biochemistry 56:669−682

[Henriques2019] Henriques ST, Peacock H, Benfield AH, Wang CK, Craik DJ (2019) Is the Mirror Image a True Reflection? Intrinsic Membrane Chirality Modulates Peptide Binding. Journal of the American Chemical Society 141:20460−20469

[Herrmann2006] Herrmann A, Svangård E, Claeson P, Gullbo J, Bohlin L, Göransson U (2006) Key Role of Glutamic Acid for the Cytotoxic Activity of the Cyclotide Cycloviolacin O2. Cellular and Molecular Life Sciences 63:235–245

[Hoogerbrugge1992] Hoogerbrugge PJ, Koelman JMVA (1992) Simulating Microscopic Hydrodynamic Phenomena with Dissipative Particle Dynamics. Europhysics Letters 19(3):155-160

[Huang2009] Huang YH, Colgrave ML, Daly NL, Keleshian A, Martinac B, Craik DJ (2009) The Biological Activity of the Prototypic Cyclotide Kalata B1 Is Modulated by the Formation of Multimeric Pores. Journal of Biological Chemistry 284(31):20699–20707

[Huang2010] Huang YH, Colgrave ML, Clark RJ, Kotze AC, Craik DJ (2010) Lysine-Scanning Mutagenesis Reveals an Amendable Face of the Cyclotide Kalata B1 for the Optimization of Nematocidal Activity. Journal of Biological Chemistry 285:10797–10805

[Ibergay2009] Ibergay C, Malfreyt P, Tildesley DJ. Electrostatic Interactions in Dissipative Particle Dynamics: Toward a Mesoscale Modeling of the Polyelectrolyte Brushes. Journal of Chemical Theory and Computation 2009;5(12):3245–3259.

[Ingólfsson2014] Ingólfsson HI, Melo MN, van Eerden FJ, Arnarez C, Lopez CA, Wassenaar TA, Periole X, de Vries AH, Peter Tieleman D, Marrink SJ (2014) Lipid Organization of the Plasma Membrane. Journal of the American Chemical Society 136:14554−14559

[Ishizaki2015] Ishizaki K, Hayashi A, Koblents G, Sarkar V. Compiling and Optimizing Java 8 Programs for GPU Execution. Proceedings of the 24th International Conference on Parallel Architectures and Compilation Techniques (PACT 2015,

October 18-21, 2015, IEEE Computer Society Washington, DC, USA). 2015:419-431.

[JAMA2020] JAMA: A Java Matrix Package. http://math.nist.gov/javanumerics/jama. Accessed 31 January 2020.

[Java2020] Java platform. http://openjdk.java.net. Accessed 31 January 2020.

[Java2D2020] Java2D. https://docs.oracle.com/javase/6/docs/technotes/guides/2d. Accessed 31 January 2020.

[Javadoc2020] Javadoc documentation. http://www.oracle.com/technetwork/java/javase/documentation. Accessed 31 January 2020.

[JCommon2020] JCommon. http://www.jfree.org/jcommon. Accessed 31 January 2020.

[JDOM2020] JDOM. http://www.jdom.org. Accessed 31 January 2020.

[JdpdRepository2020] Jdpd - An Open Java Simulation Kernel for Molecular Fragment Dissipative Particle Dynamics. Project at GitHub. https://github.com/zielesny/Jdpd. Accessed 31 January 2020.

[JFreeChart2020] JFreeChart. http://www.jfree.org/jfreechart. Accessed 31 January 2020.

[Jmol2020] Jmol: An Open-Source Browser-Based HTML5 Viewer and Stand-Alone Java Viewer for Chemical Structures in 3D. http://jmol.sourceforge.net. Accessed 31 January 2020.

[KalataB12020] High Resolution Solution Structure of Kalata B1. https://www.rcsb.org/structure/1NB1. Accessed 31 January 2020.

[Kiselev2006] Kiselev MA, Zemlyanaya EV, Aswal VK, Neubert RHH (2006) What Can We Learn About the Lipid Vesicle Structure from the Small-Angle Neutron Scattering Experiment? European Biophysics Journal 35:477–493

[Kiselev2013] Kiselev MA, Zemlyanaya EV, Ryabova NY, Hauss T, Almasy L, Funari SS, Zbytovska J, Lombardo D (2013) Influence of Ceramide on the Internal Structure and Hydration of the Phospholipid Bilayer Studied by Neutron and X-Ray Scattering. Applied Physics A 116(1):319-325

[Koelman1993] Koelman JMVA, Hoogerbrugge PJ (1993) Dynamic Simulations of Hard-Sphere Suspensions Under Steady Shear. Europhysics Letters 21(3):363-368

[Kučerka2009] Kučerka N, Gallová J, Uhríková D, Balgavý P, Bulacu M, Marrink SJ, Katsaras J (2009) Areas of Monounsaturated Diacylphosphatidylcholines. Biophysical Journal 97:1926–1932

[Kučerka2011] Kučerka N, Nieh MP, Katsaras J (2011) Fluid Phase Lipid Areas and Bilayer Thicknesses of Commonly Used Phosphatidylcholines as a Function of Temperature. Biochimica et Biophysica Acta 1808:2761–2771

[Kutzner2019] Kutzner C, Páll S, Fechner M, Esztermann A, de Groot BL, Grubmüller H (2019) More Bang for Your Buck: Improved Use of GPU Nodes for GROMACS 2018. arXiv: 1903.05918

[LAMMPS2020] LAMMPS. https://lammps.sandia.gov. Accessed 31 January 2020.

[LAMMPS-GPU2020] GPU Package in LAMMPS. https://lammps.sandia.gov/doc/Speed_gpu.html. Accessed 31 January 2020.

[Lange1981] Lange Y, Dolde J, Steck TL (1981) The Rate of Transmembrane Movement of Cholesterol in the Human Erythrocyte. Journal of Biological Chemistry 256(11):5321-5323

[Leimkuhler2015] Leimkuhler B, Shang X. On the Numerical Treatment of Dissipative Particle Dynamics and Related Systems. Journal of Computational Physics. 2015;280:72–95.

[Lewis1983] Lewis B, Engelman DM (1983) Lipid Bilayer Thickness Varies Linearly with Acyl Chain Length in Fluid Phosphatidylcholine Vesicles. Journal of Molecular Biology 166:211-217

[Lindholm2002] Lindholm P, Göransson U, Johansson S, Claeson P, Gullbo J, Larsson R, Bohlin L, Backlund A (2002) Cyclotides: A Novel Type of Cytotoxic Agents. Molecular Cancer Therapeutics 1:365–369

[Marrink2007] Marrink SJ, Risselada HJ, Yefimov S, Tieleman DP, de Vries AH (2007) The MARTINI Force Field: Coarse Grained Model for Biomolecular Simulations. The Journal of Physical Chemistry B 111:7812-7824

[Marsh2002] Marsh D (2002) Membrane Water-Penetration Profiles from Spin Labels. European Biophysics Journal 31:559–562

[MFsimClip01] kB1 and cO2 Backbone Flexibility. MP4 Clip. https://w-hs.sciebo.de/s/OXJdy4we5qgufr7. Accessed 31 January 2020.

[MFsimClip02] Plasma Membrane Cholesterol Mobility. MP4 Clip. https://w-hs.sciebo.de/s/BphC3r3fhcXc3EK. Accessed 31 January 2020.

[MFsimClip03] Cyclotide-Membrane Pore Formation. MP4 Clip. https://w-hs.sciebo.de/s/6TeHpVWMAgjDag6. Accessed 31 January 2020.

[MFsimClip04] MFsim - Cyclotide-Membrane Sandwich Interaction Model. MP4 Clip. https://w-hs.sciebo.de/s/XDgcNDwM6KtYxrh. Accessed 31 January 2020.

[MFsimClip05] MFsim - Installation and Initial Test (Windows OS). MP4 Clip. https://w-hs.sciebo.de/s/Ln0Q6OIQhWxUC8i. Accessed 31 January 2020.

[MFsimClip06] kB1 Hydrophobic Patch Mutants Membrane Interaction. MP4 Clip. https://w-hs.sciebo.de/s/JyQNKNdtVqbYKyi. Accessed 31 January 2020.

[MFsimClip07] kB1 Membrane Interaction. MP4 Clip. https://w-hs.sciebo.de/s/mPl2E2PEtkqtWv6. Accessed 31 January 2020.

[MFsimClip08] Quantitative Cyclotide-Membrane Interaction Analysis. MP4 Clip. https://w-hs.sciebo.de/s/VGwXDawV4a05JUC. Accessed 31 January 2020.

[MFsimClip09] MFsim - Simulation of a DMPC Bilayer Membrane Model. MP4 Clip. https://w-hs.sciebo.de/s/wzfNGCrXSGeqEna. Accessed 31 January 2020.

[MFsimClip10] MFsim - Cyclotide-Membrane Sandwich Interaction Model. MP4 Clip. https://w-hs.sciebo.de/s/yYAyLdRxyLrUoWx. Accessed 31 January 2020.

[MFsimStudy2020] Subfolder *2020 Cyclotide-Membrane Interaction Study* of MFsim Repository at https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimParticleSet2020] MFsim Particle Set Text File *ParticleSet_AA_V02.txt* in MFsim_Source/Particles Subfolder of MFsim Repository. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimPDF01] MFsim - Cyclotide-Membrane Sandwich Interaction Model. PDF Document in Tutorials Section. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimPDF02] MFsim - Installation and Initial Test (Windows OS). PDF Document in Tutorials Section. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimPDF03] MFsim - Cyclotide-Membrane Sandwich Interaction Model. PDF Document in Tutorials Section. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimPDF04] MFsim - Simulation of a DMPC Bilayer Membrane Model. PDF Document in Tutorials Section. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[MFsimRepository2020] MFsim - An Open Java All-In-One Rich-Client Simulation Environment for Mesoscopic Simulation. Project at GitHub. https://github.com/zielesny/MFsim. Accessed 31 January 2020.

[Mitra2004] Mitra K, Ubarretxena-Belandia I, Taguchi T, Warren G, Engelman DM (2004) Modulation of the Bilayer Thickness of Exocytic Pathway Membranes by

Membrane Proteins Rather Than Cholesterol. Proceedings of the National Academy of Sciences 101(12):4083–4088

[Moeendarbary2009] Moeendarbary E, Ng TY, Zangeneh M (2009) Dissipative Particle Dynamics: Introduction, Methodology and Complex Fluid Applications - a Review. International Journal of Applied Mechanics 1(4):737-763

[Nawae2014] Nawae W, Hannongbua S, Ruengjitchatchawalya M (2014) Defining the Membrane Disruption Mechanism of Kalata B1 via Coarse-Grained Molecular Dynamics Simulations. Scientific Reports 4:3933

[Nawae2014b] Nawae W, Hannongbua S, Ruengjitchatchawalya M (2014) Dynamic Scenario of Membrane Binding Process of Kalata B1. PLoS ONE 9(12):e114473

[Nawae2017] Nawae W, Hannongbua S, Ruengjitchatchawalya M (2017) Molecular Dynamics Exploration of Poration and Leaking Caused by Kalata B1 in HIV-Infected Cell Membrane Compared to Host and HIV Membranes. Scientific Reports 7:3638

[Netbeans2020] NetBeans IDE Version 8.2. https://netbeans.org. Successor: https://netbeans.apache.org. Accessed 31 January 2020.

[Nikunen2003] Nikunen P, Karttunen M, Vattulainen I (2003) How Would You Integrate the Equations of Motion in Dissipative Particle Dynamics Simulations? Computer Physics Communications 153(3):407–423

[Nishimura2018] Native Seed Generation According to Improved Initialization 2002/1/26 Coded by Takuji Nishimura and Makoto Matsumoto: http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/MT2002/CODES/mt19937ar.c. Accessed 31 January 2020.

[Nourse2004] Nourse A, Trabi M, Daly NL, Craik DJ (2004) A Comparison of the Self-Association Behavior of the Plant Cyclotides Kalata B1 and Kalata B2 via Analytical Ultracentrifugation. Journal of Biological Chemistry 279(1):562–570

[Pagonabarraga1998] Pagonabarraga I, Hagen MHJ, Frenkel D. Self-Consistent Dissipative Particle Dynamics Algorithm. Europhysics Letters 1998;42(4):377-382.

[PCG2018] Minimal C Implementation of PCG Random Number Generators: http://www.pcg-random.org/. Accessed 31 January 2020.

[PCG-RNG2020] Minimal C Implementation of PCG Random Number Generators. http://www.pcg-random.org. Accessed 31 January 2020.

[PCG-Java2020] PCG-Java – PCG Pseudorandom Generator Implementation for Java. https://github.com/alexeyr/pcg-java. Accessed 31 January 2020.

[Piana2015] Piana S, Donchev AG, Robustelli P, Shaw DE (2015) Water Dispersion Interactions Strongly Influence Simulated Structural Properties of Disordered Protein States. The Journal of Physical Chemistry B 119:5113−5123

[PositionBonds2018] Text File ”PositionsBonds1.txt”. https://github.com/zielesny/Jdpd/tree/master/src/ de/gnwi/jdpd/tests/test_DMPC. Accessed 31 January 2020.

[Pränting2010] Pränting M, Lööv C, Burman R, Göransson U, Andersson DI (2010) The Cyclotide Cycloviolacin O2 from Viola Odorata Has Potent Bactericidal Activity Against Gram-Negative Bacteria. Journal of Antimicrobial Chemotherapy 65:1964–1971

[Prlic2012] Prlic A, Yates A, Bliven SE, Rose PW, Jacobsen J, Troshin PV, Chapman M, Gao J, Koh CH, Foisy S, Holland R, Rimsa G, Heuer ML, Brandstatter-Muller H, Bourne PE, Willis S (2012) BioJava: An Open-Source Framework for Bioinformatics. Bioinformatics 28(20):2693-2695.

[ProteinDataBank2020] Protein Data Bank. https://www.rcsb.org. Accessed 31 January 2020.

[Reenskaug2020] Reenskaug TMH, MVC, Xerox PARC 1978-79. http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. Accessed 31 January 2020.

[Rigby2004] Rigby D (2004) Fluid Density Predictions Using the COMPASS Force Field. Fluid Phase Equilibria 217:77–87

[Rosengren2013] Rosengren KJ, Daly NL, Harvey PJ, Craik DJ (2013) The Self-Association of the Cyclotide Kalata B2 in Solution is Guided by Hydrophobic Interactions. Peptide Science 100(5):453-460

[Ryjkina2002] Ryjkina E, Kuhn H, Rehage H, Müller F, Peggau J (2002) Molecular Dynamic Computer Simulations of Phase Behavior of Non-Ionic Surfactants. Angewandte Chemie International Edition 41(6):983-986

[Schulz2004] Schulz SG, Kuhn H, Schmid G, Mund C, Venzmer J (2004) Phase Behavior of Amphiphilic Polymers: A Dissipative Particles Dynamics Study. Colloid and Polymer Science 283:284–290

[Shardlow2003] Shardlow T (2003) Splitting for Dissipative Particle Dynamics. SIAM Journal on Scientific Computing 24(4):1267–1282

[Shenkarev2006] Shenkarev ZO, Nadezhdin KD, Sobol VA, Sobol AG, Skjeldal L, Arseniev AS (2006) Conformation and Mode of Membrane Interaction in Cyclotides. FEBS Journal 273:2658–2672

[Siani1994] Siani MA, Weininger D, Blaney JM (1994) CHUCKLES: A Method for Representing and Searching Peptide and Peptoid Sequences on Both Monomer and Atomic Levels. Journal of Chemical Information and Computer Scientists 34(3): 588–593.

[Siani1995] Siani MA, Weininger D, James CA, Blaney JM (1995) CHORTLES: A Method for Representing Oligomeric and Template-Based Mixtures. Journal of Chemical Information and Computer Scientits 35(6): 1026–1033.

[Simonsen2008] Simonsen SM, Sando L, Rosengren KJ, Wang CK, Colgrave ML, Daly NL, Craik DJ (2008) Alanine Scanning Mutagenesis of the Prototypic Cyclotide Reveals a Cluster of Residues Essential for Bioactivity. Journal of Biological Chemistry 283:9805–9813

[SPICESRepository2020] SPICES – A Particle-Based Molecular Structure Line Notation and Support Library for Mesoscopic Simulation. Project at GitHub. https://github.com/zielesny/SPICES. Accessed 31 January 2020.

[Steinbeck2003] Steinbeck C, Han Y, Kuhn S, Horlacher O, Luttmann E, Willighagen EL (2003) The Chemistry Development Kit (CDK): An Open-Source Java Library for Chemo- and Bioinformatics. Journal of Chemical Information and Computer Scientists 43(2):493-500.

[Steinbeck2006] Steinbeck C, Hoppe C, Kuhn S, Floris M, Guha R, Willighagen EL (2006) Recent Developments of the Chemistry Development Kit (CDK) - An Open-Source Java Library for Chemo- and Bioinformatics. Current Pharmaceutical Design 12(17):2111-2120.

[Sumatra2018]] OpenJDK Project Sumatra. https://wiki.openjdk.java.net/display/Sumatra/Main. Accessed 31 January 2020.

[Sun1998] Sun H (1998) COMPASS: An Ab Initio Force-Field Optimized for Condensed-Phase Applications - Overview with Details on Alkane and Benzene Compounds. The Journal of Physical Chemistry B 102:7338-7364

[Sutton2005] Sutton R, Sposito G, Diallo MS, Schulten HR (2005) Molecular Simulation of a Model of Dissolved Organic Matter. Environmental Toxicology and Chemistry 24(8):1902–1911

[Svangard2007] Svangård E, Burman R, Gunasekera S, Lövborg H, Gullbo J, Göransson U (2007) Mechanism of Action of Cytotoxic Cyclotides: Cycloviolacin O2 Disrupts Lipid Membranes. Journal of Natural Products 70(4):643−647

[Swing2020] Swing GUI Toolkit. http://openjdk.java.net/groups/swing. Accessed 31 January 2020.

[SYMPLER2020] SYMPLER. http://sympler.org. Accessed 31 January 2020.

[Truszkowski2013] Truszkowski A, Epple M, Fiethen A, Zielesny A, Kuhn H (2013) Molecular Fragment Dynamics Study on the Water–Air Interface Behavior of Non-Ionic Polyoxyethylene Alkyl Ether Surfactants. Journal of Colloid and Interface Science 410:140–145

[Truszkowski2014] Truszkowski A, Daniel M, Kuhn H, Neumann S, Steinbeck C, Zielesny A, Epple M (2014) A Molecular Fragment Cheminformatics Roadmap for Mesoscopic Simulation. Journal of Cheminformatics 6:45

[Truszkowski2015] Truszkowski A, van den Broek K, Kuhn H, Zielesny A, Epple M (2015) Mesoscopic Simulation of Phospholipid Membranes, Peptides, and Proteins with Molecular Fragment Dynamics. Journal of Chemical Information and Modeling 55:983−997

[USER-MESO2020] USER-MESO. http://www.cfm.brown.edu/repo/release/USER-MESO. Accessed 31 January 2020.

[vandenBroek2018] van den Broek K, Daniel M, Epple M, Kuhn H, Schaub J and Zielesny A (2018) SPICES: A Particle-Based Molecular Structure Line Notation and Support Library for Mesoscopic Simulation. Journal of Cheminformatics 10:35.

[vandenBroek2018b] van den Broek K, Kuhn H, Zielesny A (2018) Jdpd - An Open Java Simulation Kernel for Molecular Fragment Dissipative Particle Dynamics. Journal of Cheminformatics 10:25

[vandenBroek2020] van den Broek K, Daniel M, Epple M, Hein JM, Kuhn H, Neumann S, Truszkowski A, Zielesny A (2020) MFsim - An Open Java All-In-One Rich-Client Simulation Environment for Mesoscopic Simulation. Journal of Cheminformatics 12:29

[vandenBroek2021] van den Broek K, Epple M, Kersten LS, Kuhn H, Zielesny A (2021) Quantitative Estimation of Cyclotide-Induced Bilayer Membrane Disruption by Lipid Extraction with Mesoscopic Simulation. Journal of Chemical Information and Modeling. Epub ahead of print. PMID: 34008405

[vanMeer2008] van Meer G, Voelker DR, Feigenson GW (2008) Membrane Lipids: Where They Are and How They Behave. Nature Reviews Molecular Cell Biology 9:112-124

[Vattulainen2002] Vattulainen I, Karttunen M, Besold G, Polson JM. Integration Schemes for Dissipative Particle Dynamics Simulations: From Softly Interacting Systems Towards Hybrid Models. Journal of Chemical Physics 2002;116(10):3967-3979.

[Vishnyakov2013] Vishnyakov A, Lee M-T, Neimark AV (2013) Prediction of the Critical Micelle Concentration of Nonionic Surfactants by Dissipative Particle Dynamics Simulations. Journal of Physical Chemistry Letters 4:797−802

[Walton1983] Walton JPRB, Tildesley DJ, Rowlinson JS, Henderson JR. The Pressure Tensor at the Planar Surface of a Liquid. Molecular Physics 1983;48(6):1357-1368.

[Wang2009] Wang CK, Hu SH, Martin JL, Sjogren T, Hajdu J, Bohlin L, Claeson P, Göransson U, Rosengren KJ, Tang J, Tan NH, Craik DJ (2009) Combined X-Ray and NMR Analysis of the Stability of the Cyclotide Cystine Knot Fold That Underpins Its Insecticidal Activity and Potential Use as a Drug Scaffold. Journal of Biological Chemistry 284(16):10672–10683

[Wang2009b] Wang CK, Colgrave ML, Ireland DC, Kaas Q, Craik DJ (2009) Despite a Conserved Cystine Knot Motif, Different Cyclotides Have Different Membrane Binding Modes. Biophysical Journal 97:1471–1481

[Wang2012] Wang CK, Wacklin HP, Craik DJ (2012) Cyclotides Insert into Lipid Bilayers to Form Membrane Pores and Destabilize the Membrane through Hydrophobic and Phosphoethanolamine-specific Interactions. Journal of Biological Chemistry 287(52):43884–43898

[Wayne2011] Wayne R, Sedgewick K (2011) Algorithms. Chapter 4: Graphs. 4th ed. Boston: Addison-Wesley.

[Weidmann2016] Weidmann J, Craik DJ (2016) Discovery, Structure, Function, and Applications of Cyclotides: Circular Proteins from Plants. Journal of Experimental Botany 67(16):4801–4812

[Weininger1988] Weininger D (1988) SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. Journal of Chemical Information and Computational Scientists 28:31−36.

[Weininger1989] Weininger D, Weininger A, Weininger JL (1989) SMILES. 2. Algorithm for Generation of Unique SMILES Notation. Journal of Chemical Information and Computational Scientists 29(2):97−101.

[Weininger1990] Weininger D (1990) Smiles. 3. Depict. Graphical Depiction of Chemical Structures. Journal of Chemical Information and Computational Scientists 30(3):237−243.

[Willighagen2017] Willighagen EL, Mayfield JW, Alvarsson J, Berg A, Carlsson L, Jeliazkova N, Kuhn S, Pluska T, Rojas-Chertó M, Spjuth O, Torrance G, Evelo CT, Guha R, Steinbeck C (2017) The Chemistry Development Kit (CDK) v2.0: Atom Typing, Depiction, Molecular Formulas, and Substructure Searching. Journal of Cheminformatics 9:33

[Wimley2010] Wimley WC (2010) Describing the Mechanism of Antimicrobial Peptide Action with the Interfacial Activity Model. Chemical Biology 5(10):905–917

[Wimley2011] Wimley WC, Hristova K (2011) Antimicrobial Peptides: Successes, Challenges and Unanswered Questions. Journal of Membrane Biology 239:27–34

[XeonE5-2697-2018] Product Specification Intel Xeon Processor E5 2697 v2. https://ark.intel.com/products/75283/Intel-Xeon-Processor-E5-2697-v2-30M-Cache-2_70-GHz. Accessed 31 January 2020.

[Yamamoto1963] Yamamoto T (1963) On the Thickness of the Unit Membrane. Journal of Cell Biology 17(2):413–421

[Zaccai1975] Zaccai G, Blasie JK, Schoenborn BP (1975) Neutron Diffraction Studies on the Location of Water in Lecithin Bilayer Model Membranes. Proceedings of the National Academy of Sciences 72(1):376-380

[Zaccai1982] Hoekstra D (1982) Role of Lipid Phase Separations and Membrane Hydration in Phospholipid Vesicle Fusion. Biochemistry 21:2833-2840

[Zhang2012] Zhang T, Li H, Xi H, Stanton RV, Rotstein SH (2012) HELM: A Hierarchical Notation Language for Complex Biomolecule Structure Representation. Journal of Chemical Information and Modeling 52(10): 2796–2806.

10 Attachment

10.1 List of publications

Kraft A, Razum M, Potthoff J, Porzel A, Engel T; Lange F, **van den Broek K**, Furtado F (2016) The RADAR Project – A Service for Research Data Archival and Publication. International Journal of Geo-Information 5:28

Kraft A, Razum M, Potthoff J, Porzel A, Engel T; Lange F, **van den Broek K** (2016) Archivierung und Publikation von Forschungsdaten: Die Rolle von digitalen Repositorien am Beispiel des RADAR-Projekts. De Gruyter 7:50

Kraft A, Engel T, **van den Broek K** (2017) Wohin mit den Forschungsdaten? Nachrichten aus der Chemie – Wiley Online Library 5:65

van den Broek K, Daniel M, Epple M, Kuhn H, Schaub J and Zielesny A (2018) SPICES: A Particle-Based Molecular Structure Line Notation and Support Library for Mesoscopic Simulation. Journal of Cheminformatics 10:35.

van den Broek K, Kuhn H, Zielesny A (2018) Jdpd - An Open Java Simulation Kernel for Molecular Fragment Dissipative Particle Dynamics. Journal of Cheminformatics 10:25

van den Broek K, Daniel M, Epple M, Hein JM, Kuhn H, Neumann S, Truszkowski A, Zielesny A (2020) MFsim - An Open Java All-In-One Rich-Client Simulation Environment for Mesoscopic Simulation. Journal of Cheminformatics 12:29

van den Broek K, Epple M, Kersten LS, Kuhn H, Zielesny A (2021) Quantitative Estimation of Cyclotide-Induced Bilayer Membrane Disruption by Lipid Extraction with Mesoscopic Simulation. Journal of Chemical Information and Modeling Epub ahead of print. PMID: 34008405

10.2 Poster presentations

30th Molecular Modeling Workshop, Friedrich-Alexander Universität Erlangen-Nürnberg, Erlangen, Germany, April 2016, **van den Broek K**, Kuhn H, Zielesny A, Epple M, Mesoscopic Simulation of the Membrane-Disrupting Activity of the Cyclotide Kalata B1 (poster presentation)

12th German Conference on Chemoinformatics, Division Computers-in-Chemistry of the German Chemical Society (Gesellschaft Deutscher Chemiker e.V.), Fulda, Germany, November 2016, **van den Broek K**, Kuhn H, Zielesny A, Epple M, Mesoscopic Simulation of the Membrane Disrupting Activity of the Cyclotide Kalata B1 (poster presentation)

31th Molecular Modeling Workshop, Friedrich-Alexander Universität Erlangen-Nürnberg, Erlangen, Germany, March 2017, **van den Broek K**, Albrecht S, Kuhn H, Zielesny A, Epple M, Improved Plasma Membrane Models as Test Systems for the Membrane Disrupting Activity of Kalata B1 (poster presentation)

13th German Conference on Chemoinformatics, Division Computers-in-Chemistry of the German Chemical Society (Gesellschaft Deutscher Chemiker e.V.), Mainz, Germany, November 2017, **van den Broek K**, Kuhn H, Zielesny A, Epple M, Mesoscopic Simulations: New Membrane Models & Studying the Mechanism of the Cyclotide Kalata B1 and it's Mutants (poster presentation)

11th International Conference on Chemical Structures, Noordwijkerhout, The Netherlands, May 2018, **van den Broek K**, Daniel M, Schaub J, Kuhn H, Zielesny A, Epple M, PSMILES – A Particle-Based Molecular Structure Representation for Mesoscopic Simulation (poster presentation)

14th German Conference on Chemoinformatics, Division Computers-in-Chemistry of the German Chemical Society (Gesellschaft Deutscher Chemiker e.V.), Mainz, Germany, November 2018, **van den Broek K**, Kuhn H, Zielesny A, Epple M, Steps towards an Open All-In-One Rich Client Environment for Particle-Based Mesoscopic Simulation (poster presentation)

10.3 Curriculum Vitae

Der Lebenslauf ist in der Online-Version aus Gründen des Datenschutzes nicht enthalten.

Der Lebenslauf ist in der Online-Version aus Gründen des Datenschutzes nicht enthalten.

10.4 Danksagung

An erster Stelle möchte ich mich bei Herrn Prof. Matthias Epple für die tolle Möglichkeit bedanken, meine Forschung an der Anwendung und Weiterentwicklung der Molekularen Fragmentdynamik aus meiner Masterarbeit in Form einer Doktorarbeit weiterzuführen. Ich habe mich immer sehr an der tollen Zusammenarbeit und der Begeisterung für mein Projekt, die mir entgegengebracht wurde, erfreut.

Zu großem Dank bin ich auch Herrn Prof. Achim Zielesny verpflichtet, der mein Interesse an computergestützter Forschung bereits in meiner Studienzeit so sehr gefördert hat. Herzlichen Dank für die tolle Hilfe und Unterstützung, die Freude an der Arbeit, und auch ein Dank sei gesagt für all die anderen Lebensweisheiten, die Sie mit mir geteilt haben. „Per aspera ad astra“.

Für ihr Mitwirken und ihre Unterstützung bedanke ich mich recht herzlich bei Dr. Hubert Kuhn, Mirco Daniel und Stefan Neumann. Dr. Andreas Truskowski danke ich für die tolle Vorarbeit. Ich werde niemals vergessen, wie die Idee zu dieser Forschungsarbeit in einem Erlangener Hörsaal zustande kam, und in einem netten Bierkeller weiter vertieft wurde.

Jonas Schaub, Jan-Mathis Hein, Sarah Albrecht und Lisa Sophie Kersten: Ich danke Euch allen für Eure tolle Hilfe.

Nicht zuletzt möchte ich mich bei meiner Familie und meinen Freunden bedanken, für einfach Alles: Die Unterstützung, die Liebe, ja auch die Geduld und die Motivation. Danke vor allem, für das beste Kinderlachen der Welt.

“I am just a child who has never grown up. I still keep asking these 'how' and 'why' questions. Occasionally, I find an answer.” - Stephen Hawking

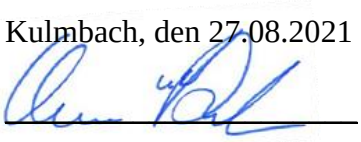
10.5 Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit mit dem Titel

“Development of an Open Biomolecular Mesoscopic Simulation Environment for the Study of Cyclotide/Membrane Interactions”

selbst verfasst und keine außer den angegebenen Hilfsmitteln und Quellen verwendet zu haben. Zudem erkläre ich, dass ich die Arbeit in dieser oder einer ähnlichen Form bei keiner anderen Fakultät eingereicht habe.

Kulmbach, den 27.08.2021



Karina van den Broek