JCTC Journal of Chemical Theory and Computation

ARTICLE

pubs.acs.org/JCTC

GROMOS++ Software for the Analysis of Biomolecular Simulation Trajectories

Andreas P. Eichenberger,[†] Jane R. Allison,[†] Jožica Dolenc,[†,∥] Daan P. Geerke,[§] Bruno A. C. Horta,[†] Katharina Meier,[†] Chris Oostenbrink,[‡] Nathan Schmid,[†] Denise Steiner,[†] Dongqi Wang,[†] and Wilfred F. van Gunsteren*[,†]

[†]Laboratory of Physical Chemistry, Swiss Federal Institute of Technology, ETH, Zürich, Switzerland

[‡]Institute of Molecular Modeling and Simulation, University of Natural Resources and Life Sciences, Vienna, Austria

[§]Division of Molecular and Computational Toxicology, Free University Amsterdam, Amsterdam, The Netherlands

[∥]Faculty of Chemistry and Chemical Technology, University of Ljubljana, Ljubljana, Slovenia

ABSTRACT: GROMOS++ is a set of C++ programs for pre- and postprocessing of molecular dynamics simulation trajectories and as such is part of the **GRO**ningen **MO**lecular **S**imulation software for (bio)molecular simulation. It contains more than 70 programs that can be used to prepare data for the production of molecular simulation trajectories and to analyze these. These programs are reviewed and the various structural, dynamic, and thermodynamic quantities that can be analyzed using time series, correlation functions, and distributions are described together with technical aspects of their implementation in GROMOS. A few examples of the use of GROMOS++ for the analysis of MD trajectories are given. A full list of all GROMOS++ programs, together with an indication of their capabilities, is given in the Appendix.

1. INTRODUCTION

Over the past decades, classical simulation of the dynamics of (bio)molecules in aqueous solution, in crystalline form, or embedded in a lipid membrane has found widespread use in physicochemical, biochemical, and molecular biological research.[1−5] This has become possible through the availability of a number of general software packages for (bio)molecular simulation, such as AMBER,[6] CHARMM,[7] Desmond,[8] GROMACS,[9] GROMOS,[10] IMPACT,[11] MOLARIS,[12,13] NAMD,[14] and TINKER,[15] and the development of (bio)molecular force fields,[16] e.g., AMBER,[17] CHARMM,[18−25] ECEPP,[26] ENCAD,[27] CFF,[28] GROMOS,[29−35] and OPLS.[36,37] These simulation software packages generally contain a number of functions that can be used to analyze the atomic trajectories produced in a molecular dynamics (MD) simulation in terms of time series, correlation functions, or trajectory averages of quantities of interest. The set of different quantities that can be analyzed and the computational procedures involved differ between the various simulation packages and are generally scarcely described in the literature. This renders a great deal of research articles irreproducible because the details of the calculations cannot be recovered. Here we present the variety of functions to analyze MD (bio)molecular simulation trajectories that are part of the GROMOS software for (bio)-molecular simulation.[10,38−42]

The GROMOS software is written in C++ and comprises two major parts: (1) MD++, the programs that can be used to perform energy minimization, MD, or stochastic dynamics (SD) simulations, and (2) GROMOS++, the programs that can be used for preprocessing (bio)molecular data prior to a simulation and for postprocessing the trajectories, coordinates, velocities, energies, etc. produced by an MD simulation. The postprocessing functions, in particular, could also be used to analyze (bio)molecular trajectories generated using (bio)molecular simulation software other than GROMOS. Therefore, we only briefly review the preprocessing capabilities of GROMOS++ and describe the postprocessing analysis functions in more detail.

The architecture, implementation, and parallelization of the GROMOS software is reported elsewhere,[39] while the analysis functions that can be used for the analysis of NMR spectroscopic and X-ray and neutron diffraction data are described in ref 42 in conjunction with their use in MD simulation based on such data. Since GROMOS++ is written in an object-oriented manner, new functionality is easily implemented.

2. TRAJECTORY QUANTITIES AND BASIC TYPES OF ANALYSIS

A simulation trajectory, which allows the calculation of many different properties of the simulated system, is primary data for a computer simulation scientist. Although the simulation programs of MD++ perform the main task of generating such simulation trajectories, an appropriate setup of a simulation and analysis of the trajectories are just as important as the simulation itself.

In this section the preprocessing of data required by a molecular simulation is outlined, followed by a discussion of the different types of simulation trajectories that are used for analysis in combination with the GROMOS++ programs. A description of the different types of analysis is given, including technicalities such as the treatment of periodic boundary conditions or the atom, vector and property specifiers implemented in GROMOS++.

Received: May 31, 2011
Published: August 22, 2011

Table 1. Preparation of a Molecular System for an MD Simulation

program	action
make_top	build a molecular topology file holding all the parameters
com_top	and specifications that characterize the molecular system
pdb2g96	converts solute coordinates in Protein Data Bank (PDB) format to GROMOS format
gch	generates hydrogen coordinates based on geometric criteria
MD++	minimizes the intermolecular solute energy to remove possible strain from the solute
ran_box	generates the coordinates for a box with solvent molecules in GROMOS format
sim_box	puts the solute into a solvent box of appropriate size and removes all solvent molecules that show a given spatial overlap with solute atoms
MD++	minimizes the system energy while the solute is kept positionally restrained to remove high-energy intermolecular contacts
ion	replaces solvent molecules by ions in order to neutralize the Coulomb charge of the solute or attain a given salt content

There are three questions to be answered at the beginning of each molecular simulation: (1) What is the chemical structure of the system to be simulated, i.e. the number and types of elements, atoms, particles, and their connectivity? (2) Which force field, containing the parameters to model the interactions in the system based on physical theories and approximations, is to be used? (3) How is the interface with the surroundings of the molecular system to be modeled? The answers to the first two questions allow a system to be set up for MD simulation using the sequence of steps specified in Table 1. A short description of the functionality of each program is given in Table A1 of the Appendix. A more detailed description can be found in the GROMOS manual or the electronic code documentation of GROMOS++.

2.1. Types of Trajectory Information Handled by GROMOS++. A simulation trajectory is completely characterized by the time series of the particle coordinates and velocities, from which a variety of quantities can be calculated. However, it is efficient to also have the possibility to save forces on the particles or energetic quantities of particular sets of atoms during an MD simulation, because the calculation of these types of quantities is computationally expensive and should not have to be repeated while postprocessing MD simulation data. Depending on the input parameters, the GROMOS MD++ simulation program regularly writes specific information about the system to the corresponding trajectory file. The time interval between saved configurations should be small enough to obtain sufficient configurations for averaging, but it is generally chosen to be much larger than the MD time step in order to avoid storage of correlated data, unless, of course, the short-time correlations happen to be of interest. Depending on the desired quantity to be calculated, a postprocessing program of GROMOS++ reads one or multiple simulation trajectory files of a specific type to compute various other, more complex quantities. GROMOS++ provides more than 70 programs (see the Appendix) ready to be used for pre- and postprocessing of molecular simulations.

The following is an overview of four different types of GROMOS simulation trajectories handled by GROMOS++. The focus is on the information content of each simulation trajectory type and not on its use for analysis. The latter should be clear after reading the sections below.

Positions and Velocities. These simulation trajectory files contain—besides the time information, i.e., current integration time step and simulated time—the three-dimensional, Cartesian atom positions and atom velocities. Additionally, the box size is indicated for every configuration of the trajectory. Information such as atom names and connectivities is generally omitted but can easily be recovered with help of the corresponding molecular topology file based on the GROMOS rule that the sequence of atoms is identical in both files.

Forces. The force trajectory file is similar to the position and velocity trajectory files. It stores atomic Cartesian forces including those induced by application of constraints on the system, typically using SHAKE.[43]

Energies. Besides the time step information, energy trajectories consist of two main parts, the first being the total energy of the system and its components: kinetic and potential energies, bonded and nonbonded contributions, lattice-sum terms, self-polarization contributions when using a polarizable force field, and special energy terms, e.g., those arising from solvent accessible surface area (SASA) implicit solvent simulations,[44] from diverse restraining functions, or from enveloping distribution sampling (EDS) simulations.[41] Moreover, it contains kinetic energies for the different sets of atoms that are coupled to a thermal bath as well as the bonded, nonbonded, and special energy contributions of each so-called energy group (predefined in the MD++ input file). The second part of the energy trajectory contains information about the mass, the box dimensions, the temperature, and the pressure of the simulated system. Some properties are stored in a 3×3 tensor format, namely, the pressure, the virial, and the molecular kinetic energy. These data can also be stored as block averages.

Free Energy Data. A free energy trajectory contains the same terms of the Hamiltonian as the energy trajectory described above, but as derivatives with respect to the coupling parameter λ. Such an approach is usually credited to Kirkwood,[45] who used it to derive expressions for the chemical potential of components of mixtures. The λ-dependence is useful for the calculation of relative free energy differences, ΔF, e.g. via thermodynamic integration (TI),

$$\Delta F = F(\lambda_B) - F(\lambda_A) = \int_{\lambda_A}^{\lambda_B} \left\langle \frac{\partial H(\lambda)}{\partial \lambda} \right\rangle_\lambda d\lambda \qquad (1)$$

where the Hamiltonian H describes different systems or states of a system as a function of λ.

2.2. Types of Analysis. GROMOS++ contains a number of programs that can be used in different combinations for the calculation of a variety of quantities. The majority of these programs belong to one of the following groups, depending on the type of analysis they perform: time series, distributions, or time correlation functions.

Time series can be calculated for a variety of scalar or vector quantities $Q(t)$ or $\mathbf{Q}(t)$, respectively, which are defined in terms of Cartesian atomic coordinates or velocities. Examples of such quantities are bond angles or torsional dihedral angles or vectors defined by atoms in the molecule, inner and outer products of such vectors, etc. GROMOS++ is able to write time series for such quantities based on position or velocity trajectories. By using energy or free energy trajectories it is possible to generate

time series of temperatures, densities, or energetic properties in a straightforward manner.

Distributions of a quantity Q can be calculated using GROMOS++. Normalization allows for the computation of probabilities $P(Q)$, while the average $\langle Q \rangle_t$ and the mean-square fluctuation ΔQ^2 are accessible as the first and second moment of the distribution,

$$\Delta Q^2 = \langle (Q - \langle Q \rangle_t)^2 \rangle_t \tag{2}$$

Time correlation functions $C_Q(t)$ or $C_{\mathbf{Q}}(t)$ are defined as

$$\begin{aligned} C_Q(t) &= \langle f(Q_i(t'), Q_j(t' + t)) \rangle_{t'} \\ &= (t_{MD} - t)^{-1} \int_0^{t_{MD} - t} f(Q_i(t'), Q_j(t' + t))\, dt' \end{aligned} \tag{3}$$

The function f may be a simple multiplication of the two quantities $Q_i(t')$ and $Q_j(t' + t)$, but GROMOS++ allows a user-specified definition of this function. $C_{\mathbf{Q}}(t)$ is defined in the same way, but depends on the vector quantities $\mathbf{Q}_i(t)$ and $\mathbf{Q}_j(t' + t)$ instead of $Q_i(t')$ and $Q_j(t' + t)$. If $i = j$, the correlation function is an autocorrelation function, while for other cases it is a cross-correlation function. The calculation of Q_i and Q_j from a trajectory is usually done for N_t discrete, equally spaced time points $t_n = n\Delta t$ with $n = 0, 1, ..., N_t - 1$. The discrete equivalent of eq 3 is then

$$C_Q(n\Delta t) = (N_t - n)^{-1} \sum_{k=0}^{N_t - n - 1} f(Q_i(k\Delta t), Q_j((k + n)\Delta t)) \tag{4}$$

If $f(Q_i(t'), Q_j(t' + t)) = Q_i(t')Q_j(t' + t)$, GROMOS++ makes use of fast Fourier transforms instead of the more time-consuming summation algorithm.

2.3. Superposition of Molecular Structures. Molecules tumble in space during a molecular simulation. This is exactly what they should do when in the liquid phase. However, if this overall motion of the solute is of no interest, GROMOS++ programs can perform an alignment of the configuration of one or multiple molecules against a reference configuration or structure, e.g. the programs rmsd and rmsf; see the Appendix. This alignment is carried out by first superimposing the centers of mass of a defined set of atoms in each configuration, followed by a rotational fit with respect to this subset of molecular atom positions. This superposition of pairs of structures is done automatically when running the programs for which it is required, while the atoms and the reference structure to be used for the superpositioning can be specified by the user.

2.4. Spatial Boundary Conditions and Gathering. Cartesian position trajectories generated using (periodic) boundary conditions with the following box shapes can be handled by GROMOS++: vacuum, rectangular, truncated octahedron, and triclinic. Due to the periodicity of the nonvacuum boundary conditions, the trajectory coordinates of molecules or groups of atoms may be such that covalent bonds are broken; i.e., the nearest image of an atom may not be the one saved in the trajectory file. Before calculating interatomic quantities such as bond lengths, bond angles, torsional dihedral angles, etc., GROMOS++ offers the possibility to select the nearest-image position with respect to a reference position, i.e., to gather a trajectory before the analysis. The different gathering methods that may be chosen by the user are listed in Table 2.

Table 2. Handling of Periodic Boundary Conditions

method	action
nog	no gathering
glist	gathering with respect to a list of atom pairs
gtime	gathering with respect to the previous configuration of the trajectory
gref	gathering with respect to a reference structure or configuration of atoms
gltime	gather the first configuration of the trajectory based on a list of atoms and the following configurations with respect to the previous configuration
grtime	gather the first configuration of the trajectory based on a reference configuration and the following configurations with respect to the previous configuration
gbond	gathering based on the bond connectivities of the solute

2.5. Atom, Vector, and Property Specifiers. The more specific the analysis of a simulation, the higher the requirements with respect to the flexibility of the corresponding analysis tool. GROMOS++ makes use of three specifiers to define the quantity to be calculated as precisely and compactly as possible while a high level of flexibility is kept. All three specifiers are passed to the program as input parameters.

Atom specifiers offer a flexible way to access specific atoms of a system. It is even possible to specify atoms that are not present in the trajectory or molecular topology file, so-called virtual atoms, e.g. hydrogen atoms of a united CH_x atom, or common properties of a group of atoms, such as the center of geometry or the center of mass. Atom specifiers are of the format

<molecule>:<atoms1>[,<atoms2>, ..., <atomsN>]

where <molecule> is the molecule number and <atoms> is one or more atoms defined in one of the following ways: the atom number within the molecule <molecule>, the atom name, or the residue and atom, both specified either by its number or name. Virtual atoms are accessed via their type and the atoms needed to construct the defined virtual atom type. For example a (virtual) aromatic hydrogen atom position is generated by the three neighboring CH_1 united atom positions, based on geometric criteria.[46]

A more complicated selection of atoms is accessed using the options "?", minus(<atom specifier>), and not(<atom specifier). The wild card "?" is used to specify groups of atoms with similar atom names, e.g. all carbon atoms (CA, CB, CG,...) of a protein:

1:C?

minus(<atom specifier>) and not(<atom specifier>) both specify atoms not to be selected. The option minus(<atom specifier>) allows atoms to be added later on, while not(<atom specifier>) definitely removes the specified atoms from the selection. The following three examples all specify all CA atoms of odd numbered residues of a 10 amino acid peptide:

1:res(1,3,5,7,9:CA)

1:CA minus(1:res(2,4,6,8,10:CA))

not(1:res(2,4,6,8,10:CA)) 1:CA

Table 3. Property Specifiers

specifier	property
d	the distance between two atoms
a	the angle defined by three atoms
t	a torsional dihedral angle
hb	the presence of hydrogen bonds
st	the stacking between two groups of atoms
o	the order between two vectors
op	the order parameter
pr	a pseudorotation
pa	the pucker amplitude
expr	a quantity defined by an expression property

It is also possible to read atom specifiers from a file, e.g. written by the program atominfo, using the atom specifier file(<file name>).

Property Specifiers. Some GROMOS++ programs are able to read a property specifier, a helpful tool to describe the quantity to be calculated. The syntax for the property specifier is

<type>%<arg>

with <type> defining the property and <arg> an atom or vector specifier providing the necessary information to calculate the desired quantity. Vector specifiers are described below. The different types of property specifiers are given in Table 3. The expression type, expr, is a specifier itself with the following syntax:

expr%<f1>(<args1>)<op><f2>(<args2>)

where <op> is an arithmetic or logical operator, while <f1> and <f2> are functions with the corresponding scalar or vector arguments, <arg1> or <arg2>, depending on the nature of the function. The following functions are supported: sin, cos, tan, asin, acos, atan, abs (the norm of a scalar or vector), sqrt, abs2 (the squared norm of a vector), dot, cross, or ni (nearest image).

Vector Specifiers. Property and expression specifiers may be defined as a function of one or more vectors. In GROMOS++, a vector can be specified by its three-dimensional Cartesian coordinates

cart(<x>, <y>, <z>)

or in polar coordinates

polar(<r>, <α>, <β>)

with r being the length of the vector and α, β the two angles to orient the vector in space. In addition, the Cartesian coordinates of atoms may be used to specify a vector.

atom(<atom specifier>)

refers to the coordinates of the atom defined by an atom specifier. If the atom specifier holds two atoms, the vector is specified pointing from the first to the second atom.

2.6. Input/Output Formats. All GROMOS++ programs are called from the command line, taking the necessary additional information as command line arguments. A usual program call looks like

program@<flag1><arg1>[@<flag2><arg2>
...@<flagN> <argN>]

The different arguments may also be collected in a single input file. The program call then looks like

program @f file

Since GROMOS++ is used for pre- and postprocessing of molecular simulations, its input and output functionality is complex. Input flags and arguments may point to files of various formats (e.g., topology files, simulation trajectories, NOE bound specifications) that are described in more detail in the GROMOS manual. However, GROMOS++ usually writes its output to the standard output, which may be directed into a text file.

The reading and writing of coordinate files (single configuration or trajectories) is often done and is therefore optimized: compressed trajectory files can be read directly and are decompressed while reading. Additionally, the writing of coordinate files may be done in one of the following user specified formats: standard or reduced GROMOS format, PDB format, and AMBER format.

Further, the two argument flags @inG96 and @outG96 ensure the compatibility with earlier GROMOS versions that read or write previous file formats.

2.7. Physical Units. Physical constants define the units used in GROMOS and are therefore not hard coded in GROMOS++. If a topology file is given, the physical constants and their units are derived from the following four physical constants, defined in the molecular topology file: Boltzmann's constant, Planck's constant, the electric permittivity of vacuum, and the speed of light. Analyses carried out without the information of a molecular topology file use physical constants that are initialized to defaults while printing a warning.

The GROMOS example files use the following basic units of the standard international system:

length, nm = 10^{-9}m
mass, u = atomic mass unit
time, ps = 10^{-12} s
temperature, K
charge, e = electronic charge = 1.602 177 3 $\times$ 10^{-19} C

3. OBSERVABLE AND NONOBSERVABLE QUANTITIES

GROMOS++ consists of more than 70 individual programs for many different tasks. To describe them all in detail is beyond the scope of this work. However, we report the calculation of some structural, dynamic, and thermodynamic quantities, covering the practically most relevant aspects of the postprocessing functionalities of GROMOS++. The calculation of experimentally observable and derived quantities is briefly mentioned but described in more detail elsewhere.[42] A complete list of all current GROMOS++ programs with a short description of each is found in the Appendix (Tables A1 and A2).

3.1. Structural Quantities. *rmsd, rmsf, rgyr.* These programs are mainly developed for the analysis of structural properties of a peptide or protein, but may also work for other molecules. The atom positional root-mean-square deviation (RMSD) between two molecular structures is calculated according to

$$\mathrm{RMSD}(\mathbf{r}^{N_a}, \mathbf{r}_{\mathrm{ref}}^{N_a}) = \sqrt{\frac{1}{N_a}\sum_{i=1}^{N_a}(\mathbf{r}_i - \mathbf{r}_{i,\mathrm{ref}})^2} \quad (5)$$

where $\mathbf{r}^{N_a} = (\mathbf{r}_1,\mathbf{r}_2,...,\mathbf{r}_{N_a})$ indicates the positions of the atoms, N_a the number of atoms considered, $\mathbf{r}_i$ the position of atom i in the first structure, and $\mathbf{r}_{i,\mathrm{ref}}$ the position of atom i in the second, reference structure.

The atom positional root-mean-square fluctuation (RMSF) of an atom i is computed according to

$$\mathrm{RMSF}_i = \sqrt{\frac{1}{N_T}\sum_{t=1}^{N_T}(\mathbf{r}_i(t) - \langle \mathbf{r}_i \rangle)^2} \quad (6)$$

where $\langle \mathbf{r}_i \rangle$ is the time-averaged position of atom i and N_T the number of configurations or time frames in the simulation trajectory.

In contrast to the RMSD and RMSF functions, which describe the positional change and mobility of atoms of a molecule, the radius of gyration (R_{gyr}) is a measure of the compactness of the structure. It can be related to light-scattering intensities and is calculated using the definition

$$R_{gyr} = \sqrt{\frac{1}{N_a}\sum_{i=1}^{N_a}(\mathbf{r}_i - \mathbf{R}_{cm})^2} \quad (7)$$

with

$$\mathbf{R}_{cm} = \frac{1}{M}\sum_{i=1}^{N_a} m_i\mathbf{r}_i \quad (8)$$

and

$$M = \sum_{i=1}^{N_a} m_i \quad (9)$$

in which $\mathbf{r}_i$ denotes the Cartesian position of atom i, m_i its mass, and N_a the number of considered atoms.

sasa. The solvent-accessible surface area can be calculated as described by Lee and Richerds.[47] A spherical probe of given radius r is rolled over the surface of the solute. The path traced out by its center is proportional to the solvent-accessible surface area.

In GROMOS, the radii of the heavy atoms are obtained by calculating the minimum energy distance of the interaction between the solute atom and the solvent. This value is reduced by the specified probe radius r to account for the radius of the solvent atom.

Alternatively, program sasa_hasel, which contains an implementation of the algorithm described by Hasel et al.,[48] can be used.

dssp. The detection of secondary structure in a protein is implemented according the rules of Kabsch and Sander.[49] An overview over the different residues with the percentage assigned to each secondary structure element over the trajectory is printed, while time series for each secondary structure element are saved in separate files. It may occur that one residue is assigned to be part of two secondary structure elements at the same time. In order to avoid ambiguous assignments, the following priority rules are applied: β-sheet/bridge > α-helix > π-helix > 3_{10}-helix > hydrogen-bonded turn > bend.

hbond. The occurrence of two-center and three-center hydrogen bonds defined using geometric criteria can be monitored over a trajectory.[50] A two-center hydrogen bond is considered to be present if (1) the position of a hydrogen atom H connected to a donor atom D is within $d_{HA} = 0.25$ nm from that of an acceptor atom A and (2) the D−H−A angle is larger than $A_{DHA} = 135°$. Occurrences of three-center hydrogen bonds are defined for a donor atom D, hydrogen atom H, and two acceptor atoms A_1 and A_2, if

(i) the distances H−A_1 and H−A_2 are smaller than 0.27 nm,
(ii) the angles D−H−A_1 and D−H−A_2 are larger than 90°,
(iii) the sum of the angles D−H−A_1, D−H−A_2, and A_1−H−A_2 is larger than 340°, and
(iv) the dihedral angle defined by the planes through the atoms D−A_1−A_2 and H−A_1−A_2 is smaller than 15°.

All distance and angle bounds used in the criteria above may be changed by the user. If a reference structure is given, only the hydrogen bonds present in the reference structure are monitored. Otherwise, all intramolecular and/or intermolecular hydrogen bonds between user-specified subsets of atoms in the system are followed. Averages for the monitored distances, angles, and occurrences are printed to the standard output while the corresponding time series are written to a file.

tser and tcf. Structural quantities defined by a property specifier, e.g., distances, angles, torsional angles, and more complex, user-specified structural properties to be calculated from the atomic coordinates, can be computed as a time series and/or a (normalized) distribution. The output of the program tser may be used for further treatment with the program tcf, which computes time correlation functions according to eq 3.

dipole. The molecular electric dipole moment $\mathbf{p}$ can be calculated according to

$$\mathbf{p}^{N_a}(\mathbf{r}^{N_a}) = \sum_{i=1}^{N_a} q_i\mathbf{r}_i \quad (10)$$

where $\mathbf{r}^{N_a} = (\mathbf{r}_1,\mathbf{r}_2,...,\mathbf{r}_{N_a})$ indicates the atom positions, N_a is the number of atoms considered, q_i the charge of atom i, and $\mathbf{r}_i$ its Cartesian position. In the case of a total net charge Q of the N_a atoms, $Q = q_1 + q_2 + ... + q_{N_a} \neq 0$, the dipole moment is dependent on the origin of the coordinate system. Either a particular origin, i.e., the center of geometry of the N_a atoms,

$$\mathbf{p}^{N_a}(\mathbf{r}^{N_a}) = \sum_{i=1}^{N_a} q_i(\mathbf{r}_i - \mathbf{R}_{cog}) \quad (11)$$

with

$$\mathbf{R}_{cog} = \frac{1}{N_a}\sum_{i=1}^{N_a}\mathbf{r}_i \quad (12)$$

can be chosen or a uniform background charge of $-Q/N_a$ is added to each atomic charge to make $\mathbf{p}^{N_a}$ origin independent.

cluster. This program performs a conformational clustering based on a similarity matrix as calculated by the program rmsdmat. The clustering algorithm is described by Daura et al.[51] Structures with RMSD values smaller than a user specified cutoff are considered to be structural neighbors. The structure with the highest number of neighbors is considered to be the central member of the cluster of similar structures forming a conformation. After removing all structures belonging to this first cluster, the procedure is repeated to find the second, third, etc. most populated clusters.

A specific structure can be forced to be the central member structure of the first cluster, which can also be the reference structure. The clustering can be performed on a subset of the matrix by specifying the maximum number of structures to consider. This allows for an assessment of the development of the number of clusters over time.

3.2. Dynamic Quantities. *diffus.* This program calculates the diffusion constant D of an atom or of the center-of-geometry of a specified set of atoms. First, the mean square-displacement, $\Delta(t)$, is obtained by averaging over all considered atoms and over multiple time windows,

$$\Delta(t) = \frac{1}{N_a}\sum_{i=1}^{N_a}\langle(\mathbf{r}_i(t+\tau)-\mathbf{r}_i(\tau))^2\rangle_{\tau \le t_{av}-t} \quad (13)$$

where $\mathbf{r}_i$ is the position of atom i, N_a the number of atoms considered, and t_{av} the averaging time. According to the Einstein relation, the diffusion constant can be estimated from the slope of $\Delta(t)$

$$D = \lim_{t\to\infty}\frac{\Delta(t)}{2N_d t} \quad (14)$$

where N_d is the number of dimensions considered.

rot_rel. The rotational relaxation time of a molecule can be estimated from the autocorrelation function of the Legendre polynomials of a molecular vector $\mathbf{v}$ of unit length,

$$C_1(t) = \langle\mathbf{v}(\tau)\cdot\mathbf{v}(\tau+t)\rangle_\tau \quad (15)$$

$$C_2(t) = \frac{1}{2}(3\langle\mathbf{v}(\tau)\cdot\mathbf{v}(\tau+t)\rangle_\tau^2 - 1) \quad (16)$$

The program rot_rel calculates the first- and second-order Legendre polynomials and the time correlation functions. The output of this program can also be produced by a combination of the programs tser and tcf.

visco. The bulk and shear viscosities can be calculated using the Einstein relation from the elements of the pressure tensor that are written to an energy trajectory. Let $P_{\alpha\beta}$ be the element of the pressure tensor, and $G_{\alpha\beta}(t)$ the time integral of $P_{\alpha\beta}$,

$$G_{\alpha\beta}(t) = \int_0^t P_{\alpha\beta}(t')\,dt' \quad (17)$$

The viscosity tensor element $\eta_{\alpha\beta}$, calculated in terms of the integral (eq 17) of the pressure component $P_{\alpha\beta}$, is proportional to the mean-square change of $G_{\alpha\beta}(t)$ in the limit of infinite time,

$$\eta_{\alpha\beta} = \frac{V}{2k_B T}\lim_{t\to\infty}\frac{d}{dt}\langle(G_{\alpha\beta}(t+\tau)-G_{\alpha\beta}(\tau))^2\rangle_{\tau \le t_{av}-t} \quad (18)$$

where V denotes the volume of the (periodic) box, k_B the Boltzmann constant, and T the temperature of the system. For isotropic systems, an estimate of the bulk viscosity can be obtained from the average of the three diagonal components of the pressure tensor,

$$\eta_{bulk} = \frac{1}{3}(\eta_{xx}+\eta_{yy}+\eta_{zz}) \quad (19)$$

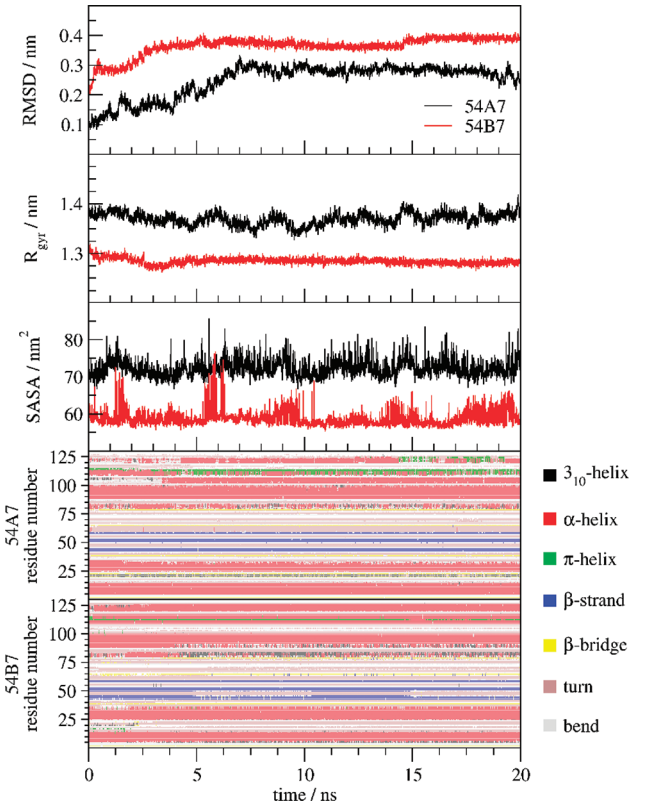

Figure 1. Atom-positional root-mean-square deviation (RMSD) from the initial X-ray structure, radius of gyration (R_{gyr}), solvent accessible surface area (SASA), and secondary structure analysis of two HEWL simulation trajectories using the GROMOS force fields 54A7 (protein in water) and 54B7 (protein in vacuum).[35]

while the shear viscosity is obtained by averaging the off-diagonal elements,

$$\eta_{shear} = \frac{1}{3}(\eta_{xy}+\eta_{xz}+\eta_{yz}) \quad (20)$$

eps_field. The static dielectric permittivity, $\epsilon(0)$, of a liquid can be obtained by applying an external electric field during a simulation and measuring the polarization response.[52] For an external field of strength E_z^{ext}, the permittivity is given by

$$\epsilon(0) = 1 + 4\pi\frac{\langle P_z\rangle_t}{E_z^{ext}} \quad (21)$$

where P_z is the average polarization of the system in the z-direction and $\mathbf{P}$ is defined as

$$\mathbf{P}(t) = \frac{\mathbf{M}(t)}{V(t)} \quad (22)$$

where $\mathbf{M}$ denotes the total dielectric dipole moment of the system and V the volume of the simulation box.

The external electric field size should be chosen to be small enough to avoid saturation and large enough to induce a significant $\langle P_z\rangle_t$.

3.3. Thermodynamic Quantities. *ene_ana.* This program can calculate time series, time averages, root-mean-square fluctuations, and statistical error estimates for properties contained in an energy or free energy trajectory file, e.g.

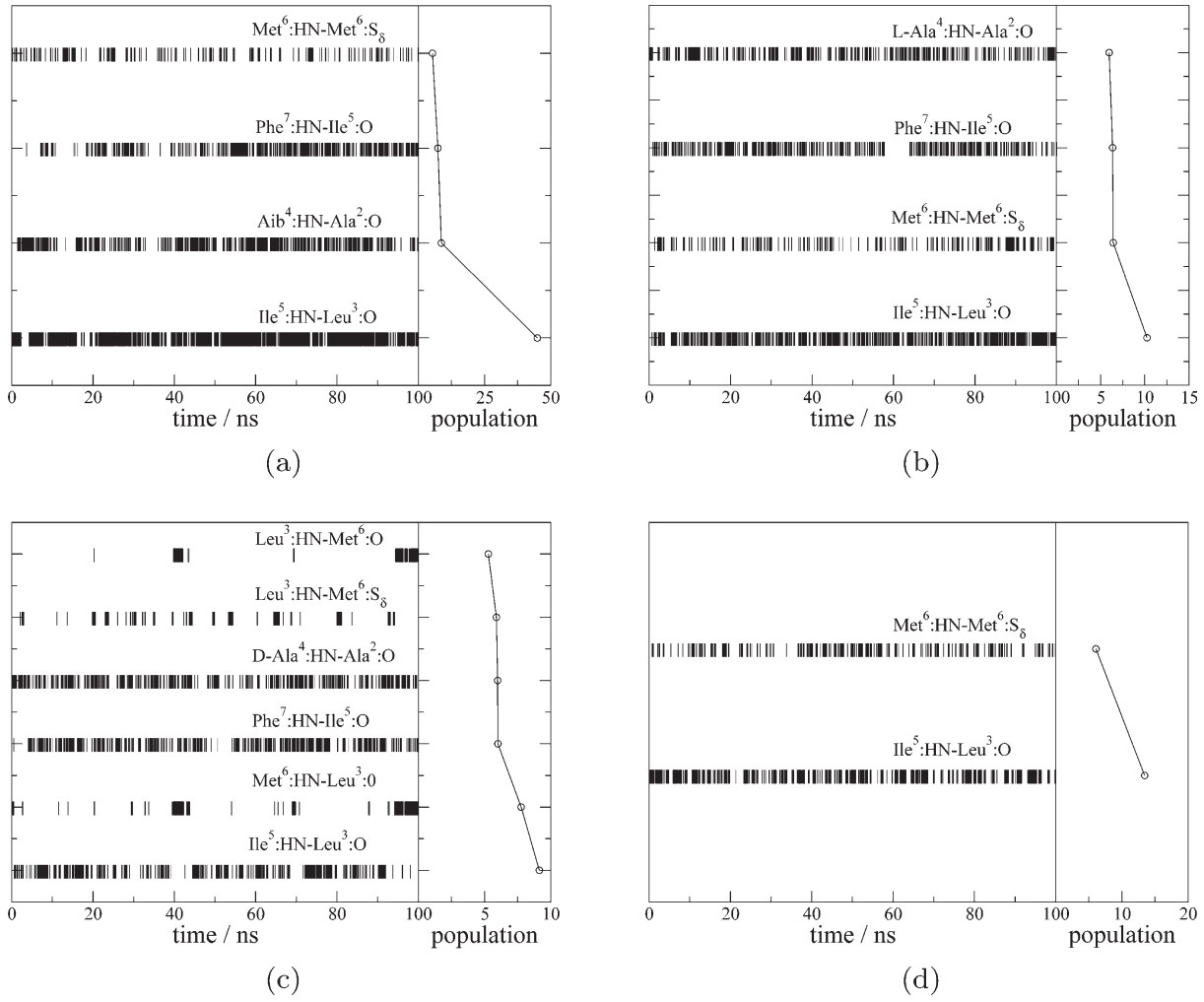

Figure 2. Occurrence of intrasolute hydrogen bonds with a population larger than 5% in the four simulations of the four heptapeptides Val-Ala-Leu-X-Ile-Met-Phe in water: (a) X = Aib, (b) X = L-Ala, (c) X = D-Ala, and (d) X = Gly.[58]

volume, temperature, pressure, different energy terms, entropy, etc. The error estimates are calculated from block averages of different sizes. In combination with a library file, ene_ana allows the analysis of quantities that are a function of one or more properties provided by the energy and/or free energy trajectory files. Note that in principle one can teach ene_ana to read any trajectory, i.e., also force- or block-averaged energies or positions, by specifying the corresponding block names and block formats in its library file.

Free Energy Differences. The calculation of relative free energies of ligand-protein binding, of solvation for different compounds, and of different conformational states of a (bio)molecule is of considerable interest with regard to an understanding of these processes and to the design or selection of potential inhibitors of enzymes. Since such processes in aqueous solution generally comprise energetic and entropic contributions from many molecular configurations, adequate sampling of the relevant parts of configurational space when calculating ensemble averages is required and can be reached through MD simulations. Most methods to obtain relative free energies require a particular modification of the Hamiltonian in an MD simulation, which leads to artificial forces on the atoms that enhance the sampling. The implementation in the GROMOS software of the most popular or promising of such methods, i.e. thermodynamic integration, umbrella sampling, local-elevation umbrella sampling, and enveloping distribution sampling, are described elsewhere.[41] Multiple GROMOS++ programs are available for the calculation of free energy differences from simulations with modified Hamiltonians.

Some methods to compute relative free energies only require postprocessing of trajectory data from a standard MD simulation and are therefore mentioned here.

In the Widom particle-insertion method,[53] a test or virtual atom is inserted randomly N_{try} times in each configuration of the molecular system and its free energy of "solvation" is then calculated as

$$\Delta F_{solv} = -k_B T \ln \left[\left\langle \frac{1}{N_{try}} \sum_{i=1}^{N_{try}} \exp\left(-\frac{V(\mathbf{r}^N, \mathbf{r}_{test})}{k_B T} \right) \right\rangle_{\mathbf{r}^N} \right] \quad (23)$$

where $V(\mathbf{r}^N,\mathbf{r}_{test})$ is the potential energy of the test particle with respect to all atoms in the system and the average is over

all configurations of the N atoms of the system. The program m_widom calculates this free energy.

In the one-step perturbation method,[54,55] the free energy change due to a change of the reference Hamiltonian H_R into a perturbed Hamiltonian H_A is calculated as

$$\Delta F_{AR} = F_A - F_R = -k_B T \ln\left[\left\langle \exp\left(-\frac{(H_A - H_R)}{k_B T}\right)\right\rangle_R\right] \quad (24)$$

where the ensemble average is over the simulation based on H_R. Such an ensemble average can be calculated using the program dg_ener.

When using a biasing potential energy term $V_{US}(\mathbf{r}^N)$ in the Hamiltonian $H_{bias}(\mathbf{r}^N)$ of the MD simulation,[56] its influence has to be removed from the ensemble averages. This is done by so-called reweighting of the configurations in the averaging:

$$\langle Q\rangle = \frac{\left\langle Q \exp\left(\frac{V_{US}}{k_B T}\right)\right\rangle_{bias}}{\left\langle \exp\left(\frac{V_{US}}{k_B T}\right)\right\rangle_{bias}} \quad (25)$$

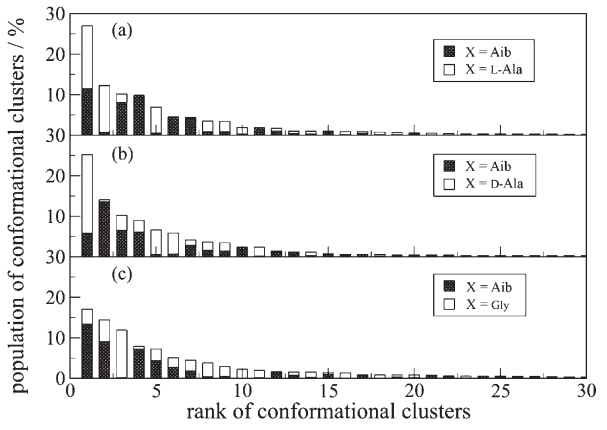

Figure 3. Conformational clustering analysis over the 100 ns trajectories of the four heptapeptides [atom-positional RMSD within 0.08 nm for backbone N, C(β), C(α), and C-atoms of residues 2−6].[58] In each panel the population of clusters observed in two joint 100 ns trajectories are shown, with the number of configurations originating from the X = Aib peptide indicated in gray.

The ensemble average $\langle ...\rangle$ for an unbiased Hamiltonian H is expressed in terms of two ensemble averages for the biased Hamiltonian $H_{bias} = H + V_{US}$. Reweighting can be performed using the program reweight.

The configurational entropy of a solute molecule can be estimated using Schlitter's heuristic formula,[57] which gives an upper bound for the true entropy

$$S_{true} \le S = \frac{1}{2} k_B \ln\left[\det\left(1 + k_B T\left(\frac{e}{\hbar}\right)^2 \mathbf{M}\boldsymbol{\sigma}\right)\right] \quad (26)$$

where $\hbar$ is Planck's constant divided by 2π, e is Euler's number, $\mathbf{M}$ is the diagonal mass matrix holding on the diagonal the masses belonging to the $3N_a$ Cartesian degrees of freedom, and $\boldsymbol{\sigma}$ is the covariance matrix of the positional fluctuations of these degrees of freedom. The elements of $\boldsymbol{\sigma}$ are

$$\sigma_{ij} = \langle (x_i - \langle x_i\rangle)(x_j - \langle x_j\rangle)\rangle \quad (27)$$

with x_i being the Cartesian coordinates of the N_a atoms considered for the entropy calculation after least-squares fitting of the position of a given subset of atoms of the coordinate trajectory. The program solute_entropy calculates Schlitter entropy and the quasi-harmonic entropy from molecular coordinate trajectories.

3.4. Comparison to Experimentally Observable and Derived Quantities. GROMOS++ is able to calculate a variety of experimentally observable quantities as well as quantities that are derived from experimental data, i.e., NOE intensities or atom−atom distance bounds, ^3J-coupling constants, residual dipolar couplings (RDCs), order parameters derived from NMR experiments, small- and wide-angle X-ray scattering (SAXS/WAXS) intensities, and total neutron scattering intensities for liquids. A detailed description of the implementation of such quantities in the GROMOS software and their use in protein refinement, including examples, is described by Schmid et al.[42]

4. EXAMPLES

Here we present a few examples of the use of GROMOS++ programs to analyze MD simulation trajectories, taken from previous work.

4.1. Global Structural Properties as Function of Time. In Figure 1, the backbone atom-positional RMSD from the initial X-ray-derived structure for two MD simulations of hen egg

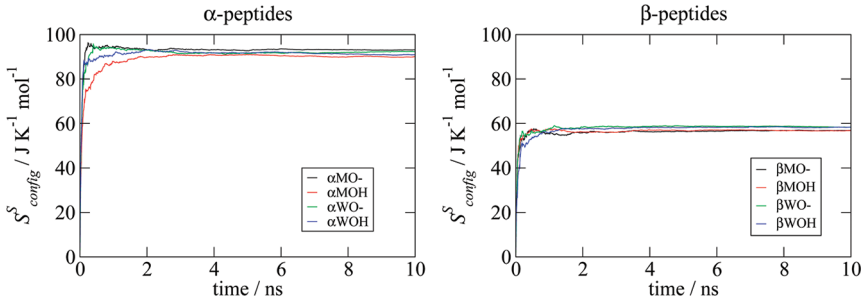

Figure 4. Buildup of the Schlitter entropy S^S_{config}, eq 26, as function of time. The Schlitter entropy was calculated for the C_α atoms during 10 ns MD simulations of various 24 backbone atom α- and β-peptides in solution as labeled (αMO- = α-peptide, methanol, O^- terminal group).[59] In each case, the structures were first aligned according to the positions of their C_α atoms.

white lysozyme (HEWL) is given as a function of time together with the radius of gyration, the solvent accessible surface area, and the occurrence of major secondary structure elements. The trajectory of the simulation in vacuo clearly deviates more from the X-ray structure than that of the simulation in water, illustrating the effect of the solvent on protein structure. The setup of these simulations is described elsewhere.[35]

4.2. Hydrogen Bond Analysis as Function of Time. In Figure 2 the intrapeptide hydrogen bonding of four seven-residue peptides in methanol as observed in 100 ns MD simulations of their folding equilibria is shown. The four peptides with sequence Val-Ala-Leu-X-Ile-Met-Phe differ in the central residue X, which is Aib, L-Ala, D-Ala, or Gly.[58] The presence of a central Aib residue clearly enhances hydrogen bonding, while a central Gly residue shows the least hydrogen bonding, as expected.

4.3. Conformational Cluster Analysis to Detect Structural Differences. The four conformational ensembles for the four seven-residue peptides were compared by performing a conformational cluster analysis for three pairs of trajectories; see Figure 3. It shows that the four ensembles are quite different. The Aib peptide shows a stronger propensity towards bent structures, while the L-Ala peptide shows a tendency to adopt more extended conformations, and the Gly peptide shows preference for a β-turn.[58]

4.4. Configurational Entropy as Function of Time. In Figure 4 the Schlitter entropy is shown as function of time for two differently protonated α- and β-peptides of similar lengths consisting purely of Ala amino acid residues and solvated in water and in methanol. On a time scale of 10 ns, the Schlitter entropy of the α-helical conformation of the eight-residue α-peptide and of the 3_{14}-helical conformation of the six-residue β-peptide is well-converged. Although all peptides contain the same number of 24 backbone atoms, the α-peptides show a significantly higher configurational entropy per atom than the β-peptides, irrespective of their protonation state or solvent.[59]

5. CONCLUSION

An overview over the different types of analysis implemented in the GROMOS++ software has been given. Three types of analysis programs were distinguished: programs that calculate structural, dynamic, or thermodynamic quantities from configurational trajectories. Additional programs are available that compute ensemble averages of experimentally observable quantities or quantities derived from experimental data, e.g. NMR NOE intensities or atom−atom distance bounds, ^3J-coupling constants, residual dipolar couplings (RDCs) and order parameters, small- and wide-angle X-ray scattering (SAXS/WAXS) intensities, and neutron scattering intensities for liquids.[42]

Compared to self-written, individual scripts computing such quantities, the use of GROMOS++ offers the advantage that most of the building blocks required for a particular type of analysis are available and tested by a wide group of users. Composing new analysis programs is relatively straightforward because of the implementation of atom, vector, and property specifiers, which allow for a very flexible description of individual quantities to be calculated. Each GROMOS++ analysis program is described in the GROMOS manual and in digital, in-code documentation. Changing the code to add new functionality does not need much effort, since GROMOS++ is written in C++ to support maximal reusability of source code.[39]

In summary, GROMOS++ is a flexible and rich collection of analysis tools ready to be used for a variety of types of analysis regarding molecular simulation trajectories.

APPENDIX: THE GROMOS++ PROGRAMS

Table A1. List of GROMOS++ Programs for Preprocessing of a Molecular Simulation

name	description
bin_box	creates a configuration of a condensed phase system consisting of two components
build_box	generates a configuration of a condensed phase system on a grid (only one component)
check_box	checks the box dimensions of a trajectory file
check_top	checks a molecular topology for (consistency) errors
com_top	combines (multiple) molecular topology files into one
con_top	converts a molecular topology to one based on a different force-field version
copy_box	repeats/extends a simulation box along a given Cartesian axis
cry	performs (crystallographic) symmetry operations on configurations of molecules
explode	places molecules of a given box on a grid thereby expanding intermolecular distances to satisfy a specific minimum intermolecular distance
gca	generates Cartesian coordinates for atoms from specified distances and/or (dihedral) angles for the atoms
gch	generates Cartesian coordinates for hydrogen atoms based on the coordinates of covalently bound neighbor atoms
ion	replaces solvent molecules by ions based on the local electrostatic potential or by random selection
make_pt_top	takes two or more molecular topologies and writes the differences in the perturbation topology format
make_sasa_top	adds the SASA block to a molecular topology file
make_top	creates a molecular topology file
mk_script	generates the scripts and input files to run a molecular simulation
pdb2g96	converts coordinate files from pdb to GROMOS format
pert_top	creates a perturbation topology to uniformly set interactions to given values for specified atoms
prep_eds	generates dual molecular and perturbation topologies for an EDS simulation
pt_top	combines molecular topologies and perturbation topologies to write new (perturbation) topologies
ran_box	creates a configuration for a condensed phase system of any composition with random molecule placements
ran_solvation	solvates a solute by randomly placing solvent molecules around it
red_top	reduces a molecular topology to one for a subset of atoms
sim_box	solvates a solute in a solvent box removing solvent molecules that are too close to solute atoms

Table A2. List of GROMOS++ Programs for Postprocessing of a Molecular Simulation

name	description
bilayer_dist	computes the atom distribution along a bilayer normal to characterize a membrane system
bilayer_oparam	calculates order parameters for bilayer systems (membranes) with respect to a fixed orientation (usually the bilayer normal)
cluster	performs a conformational clustering based on a RMSD matrix, e.g., calculated by the program rmsdmat
cog	calculates the center of geometry or center of mass position of all solute atoms of a simulation trajectory
dfmult	calculates free energy differences between multiple states A from a simulation at a reference state R
dg_ener	calculates the free energy difference between two states A and B, based on the perturbation formula; reads the output of the program ener
diffus	calculates the diffusion constant for a selected set of atoms
dipole	calculates the electric dipole moment for a selected set of atoms
ditrans	monitors transitions of torsional dihedral angle rotations with respect to the potential energy
dssp	detects secondary structure elements in a protein according to the Kabsch and Sander rules[49]
eds_mult_all	calculates the parameters needed for an enveloping distribution sampling (EDS) simulation from energy time series, based on an iterative scheme
edyn	performs an essential dynamics analysis over a trajectory file; the covariance matrix is calculated and diagonalized for specified atoms
ene_ana	writes a time series for specific values from a (free) energy trajectory file; simple statistics or calculations of combined trajectory entries are possible
ener	recalculates user specified interaction energies from molecular trajectory files using the interaction parameters from the molecular topology
eps_field	calculates the relative dielectric permittivity from a trajectory of a molecular simulation in which an external electric field was applied
epsilon	calculates the relative dielectric permittivity based on a Kirkwood−Fröhlich type of equation (fluctuation formula)
filter	reduces/filters a coordinate trajectory to contain only a specified set of atoms
follow	creates a three-dimensional trace of selected atoms through time; the program takes the nearest image with respect to the previous atom position
gathtraj	gathers a trajectory using the specified gathering method
hbond	monitors the occurrence of two- and three-centered hydrogen bonds
int_ener	recalculates the nonbonded interaction energy between two nonoverlapping sets of solute atoms using the interaction parameters specified in the molecular topology
iondens	calculates the average density of ions (or other particles) in space from a molecular trajectory file
jepot	computes the ^3J-coupling local elevation (LE) potential from a LE ^3J-coupling restrained simulation
jval	generates time series of ^3J-coupling constants based on a molecular trajectory
m_widom	calculates the free energy of inserting a test particle into configurations of a molecular system
matrix_overlap	calculates the overlap of two matrices (a mathematical definition of the overlap is given in the digital in-code documentation of GROMOS++)
mdf	for a given central set of atoms, mdf calculates the distance to the nearest atom belonging to a second set of atoms
nhoparam	calculates NH-order parameters from a simulation trajectory
noe	calculates and averages atom−atom distances; the trajectories originate either from a NOE distance restrained or free molecular simulation; the analysis may need preprocessing of data using the program prep_noe
post_noe	reanalysis of data generated by the program noe, resulting in NOE-bound violations
postcluster	performs lifetime-analysis, combined clustering, and writing of coordinates of (central) members of clusters, based on the output of the program cluster
prep_noe	converts X-plor NOE data formats to the GROMOS++ format (preparation for the noe program)
rdf	calculates a radial distribution function for specified atoms
rep_ana	used for analysis of molecular replica exchange simulations
rep_rewrite	sorts replica exchanged trajectories according to the λ or temperature values and writes them to different sorted files
reweight	reweights a time series of observed values of a quantity X sampled during a simulation at state R, i.e., using the Hamiltonian $H_R(\mathbf{p},\mathbf{r})$, to another state Y (neglecting kinetic contributions for simplicity)
rgyr	calculates the radius of gyration for a specified set of atoms
rmsd	calculates the atom-positional root-mean-square deviation of a selected set of atoms
rmsdmat	calculates the positional root-mean-square deviation matrix for a set of structures; the output may be analyzed by the program cluster
rmsf	computes the positional root-mean-square fluctuations for a specified set of atoms
sasa	calculates the solvent-accessible surface area for selected atoms using the algorithm described by Lee and Richards[47]
sasa_hasel	calculates the solvent-accessible surface area using Hasel's formula[48]
solute_entropy	calculates the configurational entropy based on a coordinate trajectory
tcf	calculates distributions and time correlation functions
tser	calculates time series of quantities
tstrip	removes solvent coordinates from a simulation trajectory
visco	calculates the bulk and shear viscosities

AUTHOR INFORMATION

Corresponding Author
*E-mail: wfvgn@igc.phys.chem.ethz.ch.

ACKNOWLEDGMENT

Ulf Börjesson, Roland Bürgi, Markus Christen, Alice Glättli, Mika Kastenholz, Christine Peter, Heiko Schäfer, Daniel Trzesniak and Haibo Yu are kindly acknowledged for their previous work on GROMOS++, which would not be at its current state without their efforts. This work was financially supported by the National Center of Competence in Research (NCCR) in Structural Biology, by grant number 200020-121913 of the Swiss National Science Foundation, and by grant number 228076 of the European Research Council, which are gratefully acknowledged.

REFERENCES

(1) van Gunsteren, W. F.; Bakowies, D.; Baron, R.; Chandrasekhar, I.; Christen, M.; Daura, X.; Gee, P.; Geerke, D. P.; Glättli, A.; Hünenberger, P. H.; Kastenholz, M. A.; Oostenbrink, C.; Schenk, M.; Trzesniak, D.; van der Vegt, N. F. A.; Yu, H. B. Biomolecular modeling: Goals, problems, perspectives. *Angew. Chem., Int. Ed.* **2006**, *45*, 4064.

(2) Scheraga, H. A.; Khalili, M.; Liwo, A. Protein-folding dynamics: Overview of molecular simulation techniques. *Annu. Rev. Phys. Chem.* **2007**, *58*, 57.

(3) van Gunsteren, W. F.; Dolenc, J. Biomolecular simulation: Historical picture and future perspectives. *Biochem. Soc. Trans.* **2008**, *36*, 11.

(4) Lindahl, E.; Sansom, M. S. P. Membrane proteins: Molecular dynamics simulations. *Curr. Opin. Struct. Biol.* **2008**, *18*, 425.

(5) Khalili-Araghi, F.; Gumbart, J.; Wen, P.-C.; Sotomayor, M.; Tajkhorshid, E.; Schulten, K. Molecular dynamics simulations of membrane channels and transporters. *Curr. Opin. Struct. Biol.* **2009**, *19*, 128.

(6) Case, D. A.; Cheatham, T. E.; Darden, T.; Gohlke, H.; Luo, R.; Merz, K. M.; Onufriev, A.; Simmerling, C.; Wang, B.; Woods, R. J. The AMBER biomolecular simulation programs. *J. Comput. Chem.* **2005**, *26*, 1668.

(7) Brooks, B. R.; Brooks, C. L., III; MacKerell, A. D., Jr.; Nilsson, L.; Petrella, R. J.; Roux, B.; Won, Y.; Archontis, G.; Bartels, C.; Boresch, S.; Caflisch, A.; Caves, L.; Cui, Q.; Dinner, A. R.; Feig, M.; Fischer, S.; Gao, J.; Hodoscek, M.; Im, W.; Kuczera, K.; Lazaridis, T.; Ma, J.; Ovchinnikov, V.; Paci, E.; Pastor, R. W.; Post, C. B.; Pu, J. Z.; Schaefer, M.; Tidor, B.; Venable, R. M.; Woodcock, H. L.; Wu, X.; Yang, W.; York, D. M.; Karplus, M. CHARMM: The biomolecular simulation program. *J. Comput. Chem.* **2009**, *30*, 1545.

(8) Bowers, K. J.; Chow, E.; Huageng Xu; Dror, R. O.; Eastwood, M. P.; Gregersen, B. A.; Klepeis, J. L.; Kolossvary, I.; Moraes, M. A.; Sacerdoti, F. D.; Salmon, J. K.; Shan, Y.; Shaw, D. E. Scalable algorithms for molecular dynamics simulations on commodity clusters. Proceedings of the ACM/IEEE Conference on Supercomputing (SC06), Tampa, FL, 2006.

(9) Hess, B.; Kutzner, C.; van der Spoel, D.; Lindahl, E. Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation. *J. Chem. Theory Comput.* **2008**, *4*, 435.

(10) Christen, M.; Hünenberger, P. H.; Bakowies, D.; Baron, R.; Bürgi, R.; Geerke, D. P.; Heinz, T. N.; Kastenholz, M. A.; Kräutler, V.; Oostenbrink, C.; Peter, C.; Trzesniak, D.; van Gunsteren, W. F. The GROMOS software for biomolecular simulation: GROMOS05. *J. Comput. Chem.* **2005**, *26*, 1719.

(11) Banks, J. L.; Beard, H. S.; Cao, Y. X.; Cho, A. E.; Damm, W.; Farid, R.; Felts, A. K.; Halgren, T. A.; Mainz, D. T.; Maple, J. R.; Murphy, R.; Philipp, D. M.; Repasky, M. P.; Zhang, L. Y.; Berne, B. J.; Friesner, R. A.; Gallicchio, E.; Levy, R. M. Integrated modeling program, applied chemical theory (IMPACT). *J. Comput. Chem.* **2005**, *26*, 1752.

(12) Lee, F. S.; Chu, Z. T.; Warshel, A. Microscopic and semimicroscopic calculations of electrostatic energies in proteins by the POLARIS and ENZYME programs. *J. Comput. Chem.* **1993**, *14*, 161.

(13) Chu, Z. T.; Villa, J.; Strajbl, M.; Schutz, C. N.; Shurki, A.; Washel, A. MOLARIS v. beta9.05; University of Southern California, 2002.

(14) Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kale, L.; Schulten, K. Scalable molecular dynamics with NAMD. *J. Comput. Chem.* **2005**, *26*, 1781.

(15) TINKER—Software tools for molecular design, http://dasher.wustl.edu/tinker (visited on August 4, 2011).

(16) Ponder, J. W.; Case, D. A. Force fields for protein simulations. In *Protein Simulations*; Academic Press Inc.: San Diego, CA, 2003; pp 27.

(17) Cheatham, T. E.; Young, M. A. Molecular dynamics simulation of nucleic acids: Successes, limitations, and promise. *Biopolymers* **2000**, *56*, 232.

(18) MacKerell, A. D.; Bashford, D.; Bellott, M.; Dunbrack, R. L.; Evanseck, J. D.; Field, M. J.; Fischer, S.; Gao, J.; Guo, H.; Ha, S.; Joseph-McCarthy, D.; Kuchnir, L.; Kuczera, K.; Lau, F. T. K.; Mattos, C.; Michnick, S.; Ngo, T.; Nguyen, D. T.; Prodhom, B.; Reiher, W. E.; Roux, B.; Schlenkrich, M.; Smith, J. C.; Stote, R.; Straub, J.; Watanabe, M.; Wiorkiewicz-Kuczera, J.; Yin, D.; Karplus, M. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J. Phys. Chem. B* **1998**, *102*, 3586.

(19) MacKerell, A. D.; Feig, M.; Brooks, C. L. Extending the treatment of backbone energetics in protein force fields: Limitations of gas-phase quantum mechanics in reproducing protein conformational distributions in molecular dynamics simulations. *J. Comput. Chem.* **2004**, *25*, 1400.

(20) Chen, J. H.; Im, W. P.; Brooks, C. L. Balancing solvation and intramolecular interactions: Toward a consistent generalized born force field. *J. Am. Chem. Soc.* **2006**, *128*, 3728.

(21) MacKerell, A. D.; Banavali, N.; Foloppe, N. Development and current status of the CHARMM force field for nucleic acids. *Biopolymers* **2000**, *56*, 257.

(22) Vanommeslaeghe, K.; Hatcher, E.; Acharya, C.; Kundu, S.; Zhong, S.; Shim, J.; Darian, E.; Guvench, O.; Lopes, P.; Vorobyov, I.; MacKerell, A. D., Jr. CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields. *J. Comput. Chem.* **2010**, *31*, 671.

(23) Patel, S.; MacKerell, A. D.; Brooks, C. L. CHARMM fluctuating charge force field for proteins: II protein/solvent properties from molecular dynamics simulations using a nonadditive electrostatic model. *J. Comput. Chem.* **2004**, *25*, 1504.

(24) Lamoureux, G.; Roux, B. Modeling induced polarization with classical drude oscillators: Theory and molecular dynamics simulation algorithm. *J. Chem. Phys.* **2003**, *119*, 3025.

(25) Lamoureux, G.; Harder, E.; Vorobyov, I. V.; Roux, B.; MacKerell, A. D. A polarizable model of water for molecular dynamics simulations of biomolecules. *Chem. Phys. Lett.* **2006**, *418*, 245.

(26) Arnautova, Y. A.; Jagielska, A.; Scheraga, H. A. A new force field (ECEPP-05) for peptides, proteins, and organic molecules. *J. Phys. Chem. B* **2006**, *110*, 5025.

(27) Levitt, M.; Hirshberg, M.; Sharon, R.; Daggett, V. Potential-energy function and parameters for simulations of the molecular-dynamics of proteins and nucleic-acids in solution. *Comput. Phys. Commun.* **1995**, *91*, 215.

(28) Warshel, A. *Computer Modeling of Chemical Reactions in Enzymes and Solution*; John Wiley & Sons, Inc.: New York, 1991.

(29) Schuler, L. D.; Daura, X.; van Gunsteren, W. F. An improved GROMOS96 force field for aliphatic hydrocarbons in the condensed phase. *J. Comput. Chem.* **2001**, *22*, 1205.

(30) Chandrasekhar, I.; Kastenholz, M.; Lins, R. D.; Oostenbrink, C.; Schuler, L. D.; Tieleman, D. P.; van Gunsteren, W. F. A consistent potential energy parameter set for lipids: Dipalmitoylphosphatidylcholine as a benchmark of the GROMOS96 45A3 force field. *Eur. Biophys. J. Biophys.* **2003**, *32*, 67.

(31) Soares, T. A.; Hünenberger, P. H.; Kastenholz, M. A.; Kräutler, V.; Lenz, T.; Lins, R. D.; Oostenbrink, C.; van Gunsteren, W. F. An

improved nucleic acid parameter set for the GROMOS force field. *J. Comput. Chem.* **2005**, *26*, 725.

(32) Lins, R. D.; Hünenberger, P. H. A new GROMOS force field for hexopyranose-based carbohydrates. *J. Comput. Chem.* **2005**, *26*, 1400.

(33) Oostenbrink, C.; Villa, A.; Mark, A. E.; van Gunsteren, W. F. A biomolecular force field based on the free enthalpy of hydration and solvation: The GROMOS force-field parameter sets 53A5 and 53A6. *J. Comput. Chem.* **2004**, *25*, 1656.

(34) Poger, D.; van Gunsteren, W. F.; Mark, A. E. A new force field for simulating phosphatidylcholine bilayers. *J. Comput. Chem.* **2010**, *31*, 1117.

(35) Schmid, N.; Eichenberger, A. P.; Choutko, A.; Riniker, S.; Winger, M.; Mark, A. E.; van Gunsteren, W. F. Definition and testing of the GROMOS force-field versions: 54A7 and 54B7. *Eur. Biophys. J.* **2011**, *40*, 843.

(36) Jorgensen, W. L.; Maxwell, D. S.; Tirado-Rives, J. Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. *J. Am. Chem. Soc.* **1996**, *118*, 11225.

(37) Kaminski, G. A.; Friesner, R. A.; Tirado-Rives, J.; Jorgensen, W. L. Evaluation and reparametrization of the OPLS-AA force field for proteins via comparison with accurate quantum chemical calculations on peptides. *J. Phys. Chem. B* **2001**, *105*, 6474.

(38) Scott, W. R. P.; Hünenberger, P. H.; Tironi, I. G.; Mark, A. E.; Billeter, S. R.; Fennen, J.; Torda, A. E.; Huber, T.; Krüger, P.; van Gunsteren, W. F. The GROMOS biomolecular simulation package. *J. Phys. Chem. A* **1999**, *103*, 3596.

(39) Schmid, N.; Christ, C. D.; Christen, M.; Eichenberger, A. P.; van Gunsteren, W. F. Architecture, implementation and parallelisation of the GROMOS software for biomolecular simulation. *Comput. Phys. Commun.* **2011**submitted.

(40) Kunz, A. P. E.; Allison, J. R.; Geerke, D. P.; Horta, B. A. C.; Hünenberger, P. H.; Riniker, S.; Schmid, N.; van Gunsteren, W. F. New functionalities in the GROMOS biomolecular simulation software. *J. Comput. Chem.* **2011**submitted.

(41) Riniker, S.; Christ, C. D.; Hansen, H.; Hünenberger, P. H.; Oostenbrink, C.; Steiner, D.; van Gunsteren, W. F. Computation of relative free energies for ligand protein binding, solvation and conformational transitions using the GROMOS biomolecular simulation software. *J. Phys. Chem.* **2011**, submitted.

(42) Schmid, N.; Allison, J. R.; Dolenc, J.; Eichenberger, A. P.; Kunz, A. P. E.; van Gunsteren, W. F. Biomolecular structure refinement using the GROMOS simulation software. *J. Biomol. NMR* **2011**; DOI: 10.1007/s10858-011-9534-0.

(43) Ryckaert, J. P.; Ciccotti, G.; Berendsen, H. J. C. Numerical-integration of Cartesian equations of motion of a system with constraints: Molecular dynamics of *n*-alkanes. *J. Comput. Phys.* **1977**, *23*, 327.

(44) Allison, J. R.; Boguslawski, K.; Fraternali, F.; van Gunsteren, W. F. A refined, efficient mean solvation force model that includes the interior volume contribution. *J. Phys. Chem. B* **2011**, *115*, 4547.

(45) Kirkwood, J. G. Statistical mechanics of fluid mixtures. *J. Chem. Phys.* **1935**, *3*, 300.

(46) van Gunsteren, W. F.; Billeter, S. R.; Eising, A. A.; Hünenberger, P. H.; Krüger, P.; Mark, A. E.; Scott, W. R. P.; Tironi, I. G. *Biomolecular Simulation: The GROMOS96 Manual and User Guide*; vdf Hochschulverlag AG an der ETH Zürich and BIOMOS b.v.: Zürich, Groningen, 1996.

(47) Lee, B.; Richards, F. M. Interpretation of protein structures: Estimation of static accessibility. *J. Mol. Biol.* **1971**, *55*, 379.

(48) Hasel, W.; Hendrickson, T. F.; Still, W. C. A rapid approximation to the solvent accessible surface areas of atoms. *Tetrahedron Comput. Methodol.* **1988**, *1*, 103.

(49) Kabsch, W.; Sander, C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **1983**, *22*, 2577.

(50) Koehler, J. E. H.; Saenger, W.; van Gunsteren, W. F. On the occurrence of three-center hydrogen bonds in cyclodextrins in crystalline form and in aqueous solution: Comparison of neutron diffraction and molecular dynamics results. *J. Biomol. Struct. Dyn.* **1988**, *6*, 181.

(51) Daura, X.; van Gunsteren, W. F.; Mark, A. E. Folding-unfolding thermodynamics of a β-heptapeptide from equilibrium simulations. *Proteins* **1999**, *34*, 269.

(52) Riniker, S.; Kunz, A. P. E.; Gunsteren, W. F. On the calculation of the dielectric permittivity and relaxation of molecular models in the liquid phase. *J. Chem. Theory Comput.* **2011**, *7*, 1469.

(53) Widom, B. Some topics in theory of fluids. *J. Chem. Phys.* **1963**, *39*, 2808.

(54) Liu, H. Y.; Mark, A. E.; vanGunsteren, W. F. Estimating the relative free energy of different molecular states with respect to a single reference state. *J. Phys. Chem.* **1996**, *100*, 9485.

(55) Zwanzig, R. W. High-temperature equation of state by a perturbation method. I. nonpolar gases. *J. Chem. Phys.* **1954**, *22*, 1420.

(56) Torrie, G. M.; Valleau, J. P. Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling. *J. Comput. Phys.* **1977**, *23*, 187.

(57) Schlitter, J. Estimation of absolute and relative entropies of macromolecules using the covariance-matrix. *Chem. Phys. Lett.* **1993**, *215*, 617.

(58) Wang, D.; Friedmann, M.; Gattin, Z.; Jaun, B.; van Gunsteren, W. F. The propensity of α-aminoisobutyric acid (=2-methylalanine; Aib) to induce helical secondary structure in an α-heptapeptide: A computational study. *Helv. Chim. Acta* **2010**, *93*, 1513.

(59) Allison, J. R.; Müller, M.; van Gunsteren, W. F. A comparison of the different helices adopted by α- and β-peptides suggests different reasons for their stability. *Protein Sci.* **2010**, *19*, 2186.