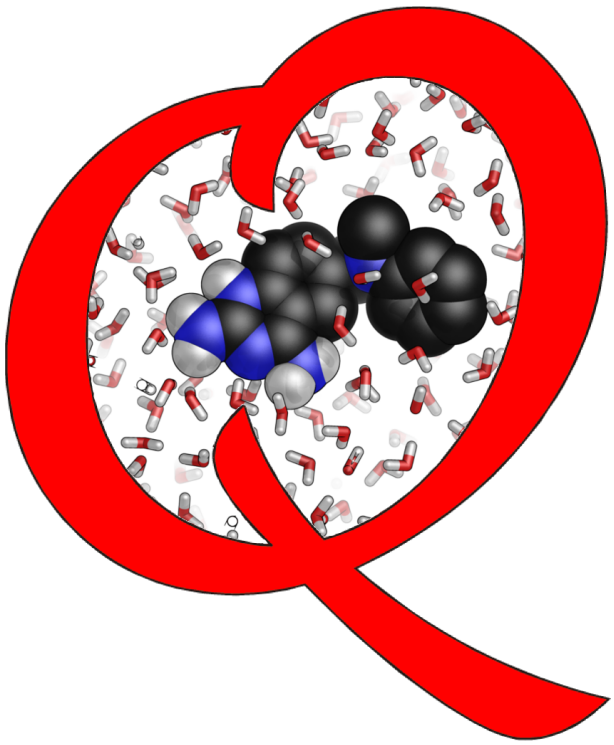

Manual for
Q6
Free Energy Calculations in (Bio)Molecular Systems

Version 6.0
August 17, 2017
http://xray.bmc.uu.se/~aqwww/q

Contents

A LICENSE STATEMENT

Q6 is a free simulation package that is distributed under the GNU General Public License, Version 2.0 (GPLv2).

Developed by: The Department of Cell and Molecular Biology Uppsala University, Uppsala, Sweden `http://xray.bmc.uu.se/~aqwww/q/` paul.bauer.q@gmail.com, qmoldyn@googlegroups.com

This license covers all modules associated with Q, including the following:

1. Qprep6
2. Qdyn6
3. Qfep6
4. Qcalc6
5. Qpi6

Therefore, the Q package in its entirety may be copied, modified or distributed, according to the terms described below. For user convenience, all parts of Q are covered by a single license file.

==========

GNU GENERAL PUBLIC LICENSE Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these

rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change. b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License. c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.) The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program

under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

B HOW TO CITE

The development of Q has been funded through grants to the Kamerlin and Åqvist groups. To allow further development, we kindly ask you to cite the relevant papers:

- Q6: A comprehensive toolkit for empirical valence bond and related free energy calculations
  P. Bauer, A. Barrozo, B. A. Amrein, M. Purg, M. Esguerra, P. B. Wilson, D. T. Major, J. Åqvist, S. C. L. Kamerlin
  To be submitted
- Q: a molecular dynamics program for free energy calculations and empirical valence bond simulations in biomolecular systems
  J. Marelius, K. Kolmodin, I. Feierberg, J. Åqvist
  Journal of Molecular Graphics and Modelling 16, 213-225, 1998

C PREFACE

Q started its development sometime in the 1990s originally by Johan Åqvist, and since then a list of collaborators have contributed to the code, among them: John Marelius, Karin Kolmodin, Isabella Feierberg, Martin Almlöf, Martin Ander, Jens Carlson, Peter Hanspers, Anders Kaplan, Kajsa Ljunjberg, Martin Nervall, Johan Sund, Paul Bauer, Alexandre Barrozo, Masoud Kazemi, Irek Szeler, and Åke Sandgren. The code is in active development and new features are being implemented mainly following the Fortran 2003 standard. Additionally the MPI implementation has been updated to enhance the speed of running parallel jobs and provide a more robust parallelization across different cluster architectures. The program code is hosted online thanks to a generous github academic account granted to the program developers at Uppsala University. Instructions on how to obtain the code can be found online at the Q-website: `http://xray.bmc.uu.se/~aqwww/q/`.

D INTRODUCTION

Molecular dynamics (MD) simulations can be used to sample the thermally accessible regions of conformational space using a microscopic model of the molecular system. From the ensemble of sampled structures and their associated potential energies (given by the force field or molecular mechanics potential energy function) it is, in principle, possible to calculate free energies. Quantities such as binding free energies, solvation free energies and activation free energies are particularly interesting to compute because they are the direct result of thermodynamic or kinetic experiments. It is thus possible both to quantitatively verify calculated results against experimental data, and to make predictions which can be tested experimentally.

Q [1] is a set of tools tailored specifically to the calculation of free energies using diverse approaches, namely (I) free energy perturbation (FEP) simulations [2, 3], (II) empirical

valence bond (EVB) calculations [4, 5] of reaction free energies, (III) linear interaction energy (LIE) calculations [6–8] of receptor-ligand binding affinities and (IV) calculation of quantum effects using the BQCP approach [9, 10].

The main features which distinguish Q from other MD packages are:

- **The spherical boundary.** Q is intended for free energy calculations in biomolecular systems solvated in a spherical droplet of explicit water molecules. Using a spherical boundary [11–13] makes it possible to limit the size of the simulated system, *i.e.* to focus the simulation on a smaller region such as a binding site, and also makes accurate treatment of long-range electrostatics rather inexpensive.
- **The flexibility in choice of force field.** The force fields are defined in parameter files, separate from the program and the choice of force field is thus simply a matter of which parameter file to use.
- **The ease of use and learning.** The simulation control input and force field definition files are organized in a flexible way and easy to understand and modify. The programs give extensive diagnostics when problems are encountered.
- **It runs on any computer and simulates any number of particles.** By using dynamic memory allocation Q can simulate biomolecules of moderate size on a personal computer, or very large molecular systems on a super-computer, without any modifications of the program.

Several features have been added since the last release of Q:

- **Quantum classical path calculations.** The bisection QCP method of Major and Gao [9, 10] allows the calculation of quantum corrections and isotope effects of EVB free energy profiles.
- **Organic solvent models.** Updates to the solvent handling routines allow in principle any organic solvent to be modelled.
- **Group contributions on–the–fly calculations.** New routines to handle the calculation of residue or atom specific energy contributions during a free energy calculation make it possible to cheaply assess the effect of removing specific inetractions in a molecular system.

E INSTALLATION AND SETUP

E.1 System requirements

Q 6.0 can be compiled and run in Windows 7, Mac OSX and Linux. The parallel **Qdyn6p** version of the main dynamics program **Qdyn6** has been greatly improved from that of version 5.0 which was using an older Message Passing Interface (MPI) standard and one-sided communication instead of the more portable point to point communication.

The memory requirements vary with the size of the simulated system but are, in general, modest. A system of 18 Å radius with a cutoff of 10 Å in non-bonded interactions uses

about 50 to 200 Mb of RAM memory depending slightly on the computer and operating system.

E.1.1 Compilers

A modern Fortran compiler is required to build the main programs (**Qdyn6**, **Qprep6**, **Qfep6**, **Qdum6** , **Q6calc**, **Qpi6**).

A Fortran compiler and an MPI library is required to build the parallel version of **Qdyn6p** and **Qpi6p**.

The program **git** is needed to access the source code repository.

E.2 Installation

The installation can be performed either through the use of executable images, available at `https://github.com/qusers/Q6/releases`, or through compilation of the source code that is available through github at `https://github.com/qusers/Q6`. The repository also contains a collection of force field parameter and fragment libraries. We recommend to build the software from source, as it allows better optimizations to be used. To do so, please follow the instructions below.

E.2.1 Installing executable images

Windows 7 Executables for Windows need to be placed into a folder used to perform the calculation, and can be executed from the "cmd" prompt. Please note that precompiled images are not fully optimized and are not recommended for use with large calculations.

Linux Copy the exectuable files from the current release version to a folder that is included in the system $PATH environment (e.g. ∼/bin). Mark the files as executable using "chmod +x" and run them as any other command.
Please note that prebuild binaries are not fully optimized, and can not be used to perform calculations using parallelization such as MPI.

Mac OSX The binaries on OSX can be used in the same way as under Linux. The user just needs to place them in a directory and mark them as executable.

E.2.2 Building the programs from source code

Windows 7 Building under Windows in Visual Study might be possible, but is not currently supported. To compile binaries from source, cross–compilation has to be performed from one of the other supported build environments mentioned below.

Linux Clone the Q6 repository from github, using
`git clone https://github.com/qusers/Q6.git`

This will create a new folder "Q6" in the current location, with the repository. Change directory to the repository and use
`git checkout master && git pull master`

to switch to the main branch and collect any recent updates of the code. Those commands also have to be used if you want to update your version to the newest one in our repository.

Afterwards, change directory to the "src" subfolder, and type "make" to have an overview over the compilation options. To compile both the serial and parallel version of Q6 with the GCC compiler, issue the command
`make all mpi COMP=gcc`

Afterwards the binary files will be located in the folder "bin".

Mac OSX Building from source is similar to Linux. The only additional prerequisite is that a OSX build environment like homebrew or fink is available, or that the nessecary compilers have been installed previously. All other instructions are the same as for Linux.

F USER GUIDE

The structure of Q resembles that of many other MD simulation programs. The main program is **Qdyn6** which carries out the actual trajectory calculations. Besides normal input control data, it needs the type of data usually referred to as a molecular topology. This is a file containing information about how atoms are bonded to each other etc. together with all the parameters of the force field (FF) to be used. This topology file is created with the preparation program **Qprep6** which is an interactive program that uses pdb (Protein Data Bank) coordinate files together with FF specific files to generate the topology. **Qprep6** can also be used for various other data transformation purposes as described below. Input data for FEP, EVB or receptor-ligand complex simulations is given in a separate file, referred to below as the fep file. The fep file lists atoms to be transformed, called Q-atoms and force field parameters for the different states in perturbation simulations. **Qpi6** is used to perform the calculation of quantum effects following the BQCP approach as a postprocessing option. For analysis of the computations the main tool is **Qfep6** which is a program that carries out FEP and EVB calculations of free energies using energy data produced by **Qdyn6** and **Qpi6**. A number of utility programs (trajectory and energy averaging, radial distribution functions, ...) are also provided. Additonally, the graphical user interface QGui (`https://github.com/qusers/qgui`) [14] can be used, or the command line utilities QTools (`https://github.com/mpurg/qtools`) [15]. The general outline of Q is shown in figure 1.

In the following sections we will go through the normal sequence of steps for preparing a topology file with **Qprep6** and then describe in detail the input data required for running MD simulations with **Qdyn6**.

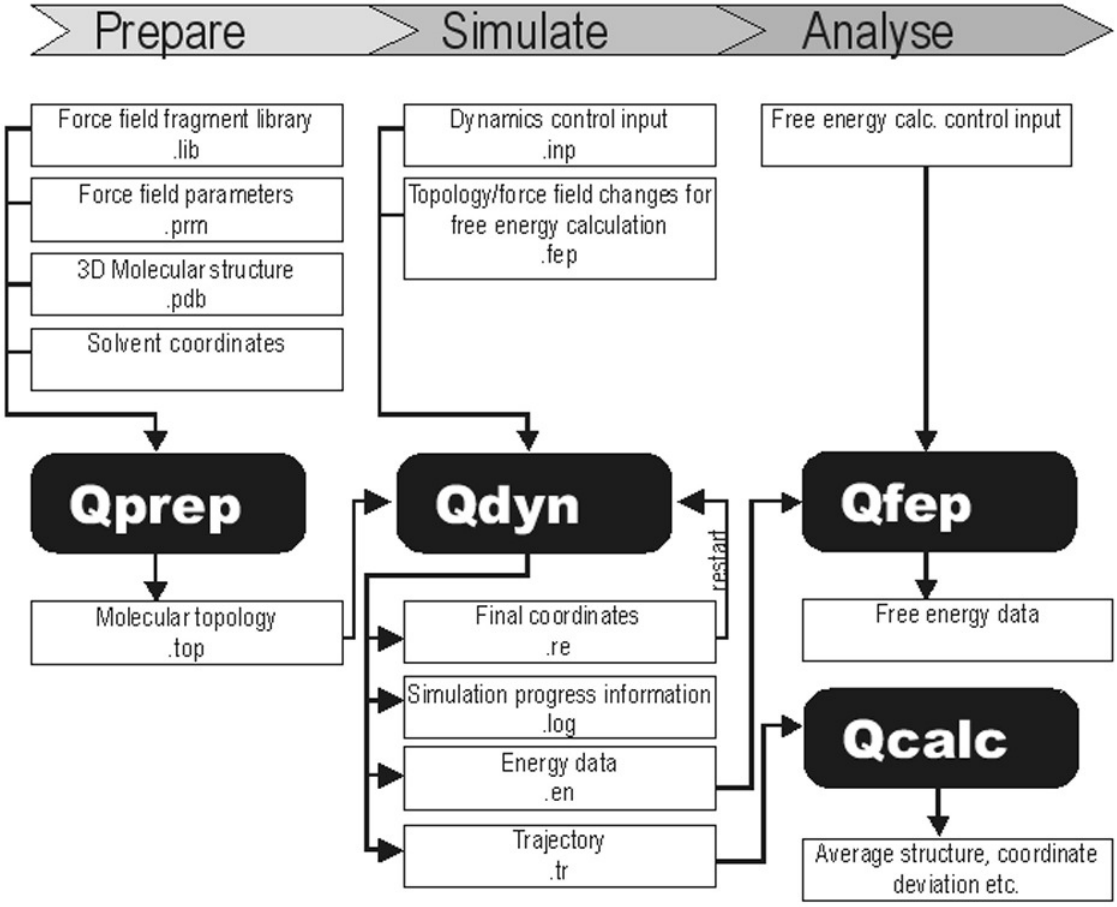

Figure 1: Overview of the procedure for free energy calculation with Q6. The white boxes represent files and also show typical file name extensions. The black boxes are programs.

F.1 Preparing a molecular topology

The topology file prepared with the program **Qprep6** contains all the information about the molecular system needed for a simulation with **Qdyn6**. To make a topology you need

- A molecular fragment library file describing the atoms and connectivity of each protein residue, ligand etc., collectively referred to as library entries.
- A force-field parameter file.
- Coordinates for all (non-solvent) atoms in the form of a PDB file.

F.1.1 Qprep6 - Preparing coordinates

The PDB file format [16] used for co-ordinate input to **Qprep6** is a fixed format, *i.e.*, the number of spaces between columns of data is significant and tab characters are not permitted. Case is significant - lower-case letters should not be used. Only ATOM and HETATM records are read by **Qprep6**, all other records are ignored.

Hydrogen atoms are not required - their coordinates will be generated by **Qprep6** in such a way that the bond and angles to the hydrogen atom are at an energy minimum. To further control the placement of hydrogens, a special torsion potential defined in the fragment library entry may be included. See [**build_rules**] on page 57. If hydrogen atom coordinates are given in the PDB file, they will not be altered.

When a structure containing more than one molecule is loaded into **Qprep6**, the program will identify the boundaries between molecules either by the type of fragment or using 'gap' marker lines. Fragments which are monomers in a polymer chain, like amino acids, have designated 'head' and 'tail' atoms in their library entries, defining how they should be connected to the neighbouring residues. Library entries describing separate molecules such as solvent or ligands do not contain this linkage information. To distinguish two peptide chains from each other it is necessary to introduce a gap marker line in the PDB file after the last atom of the first molecule. The gap marker consists of the word GAP in capital letters on a separate line.

The numbering of atoms in the PDB is not significant. Note that **Qprep6** will renumber all atoms in a single sequence starting at one. This is necessary in order to incorporate hydrogen atoms in the sequence. The result is thus that *atom numbers in the PDB file read by **Qprep6** and PDB files generated by **Qprep6** will be different.* Residue numbers are used merely to distinguish one residue from the next and the residues will also be renumbered by **Qprep6**. Residue numbers must be numeric, alphanumeric identifiers such as 60A sometimes encountered in the protein data bank are not permitted.

Protein structures determined by X-ray crystallography are normally refined with a large number of water molecules. Some of these well-ordered water molecules may be important for the structure and function of the protein and should be included in the simulation. However, most of the crystallographic waters surrounding the protein can be removed. In fact, the presence of a large number of water molecules around the protein surface will disturb the solvation algorithm leading to inhomogenous water density. **Qprep6** assumes that solvent molecules appear after solute molecules in the PDB file and keeps track of the last solute atom. Solvent molecules are identified by their residue names. The list of solvent residue names is a user-setable preference in **Qprep6** (see Setting preference on page 17) with the default value WAT, HOH, H2O, SPC, TIP3. Solvent molecules added by the solvation algorithm will appear at the end of the atom sequence. Solvent molecules outside the simulation sphere (more than water radius + 2 Å from solvent centre) will be excluded like solute atoms outside the solute sphere.

F.1.2 Selecting a force field

Q is designed to run with a wide selection of force fields. Fragment libraries and force field parameters for AMBER95, AMBER14, AMBER/OPLS, OPLS-AA, CHARMM v.22, GROMOS87 and GROMOS96 are supplied with the program. The corresponding library and FF parameter files are listed in table 8 on page 51. In addition to all the interaction parameters, a Q force field parameter file also include overall properties of the force field, such as the van der Waals parameter combination rule. Selecting a force field is thus equivalent to loading the appropriate fragment library and parameter file into **Qprep6**.

F.1.3 Adding force field library entries

Your simulations will probably include molecular fragments other than the amino acid residues and solvent molecules that are included in the standard library files and you therefore need to write a library entry for each 'new' residue, ligand, co-factor or other molecule. We suggest you keep your library entries in a separate file rather than adding to the standard library file. Load first the standard library and then your own into **Qprep6**. If you need to modify an entry in the standard library, add the modified entry (keeping the same name) to your own library file - it will replace the old entry when loaded.

The best starting point for generating a new entry is to copy an old one, this ensures the correct syntax. For details, see the section Fragment library file format on page 55.

F.1.4 Running Qprep6

Qprep6 is a command-oriented program designed to be run interactively. The first thing you need to learn about **Qprep6** is that there is a help command which gives a list of available commands. The program's input parser will prompt for the parameters required for each command but will also accept complex command lines including the parameters. Thus, you may either enter the command **readlib** and await the prompt 'Name of molecular library' or you may type **readlib my_ligands.lib** on one line.

A good deal of effort has been spent to make **Qprep6** handle errors gracefully. When a problem is encountered in a fragment library, parameter file or PDB file, you will be notified but the processing of the file will proceed, thus exposing also errors later in the file. After correcting the problem you simply read the file again without the need to restart the program.

The different steps to generate a topology with **Qprep6** are described below, in the sequence they are normally executed. More detailed information is available in the section Topology preparation reference on page 52.

Loading fragment libraries

Use the **readlib** command to load fragment libraries. Many libraries may be loaded by repeating the **readlib** command for each of them. Note that if one entry name occurs more than once, the latest definition takes precedence. A warning message is issued when a library entry is overloaded.

Errors encountered while reading a library will be displayed and after they have been corrected, the library may be loaded again. To remove all library entries from memory, use the **clearlib** command.

Loading force field parameters

Use the **readprm** command to load force field parameters. As opposed to the case with fragment libraries, only a single parameter file can be loaded. If you need to modify the parameter file, edit and save it and load it again. It is not necessary to reload fragment libraries or the structure.

Loading coordinates

Use the **readpdb** command to load the molecular structure file. Before loading the structure, the program will verify that all the required library entries are loaded and that the number of heavy atoms of each fragment is correct. Only one file can be loaded - reading another file clears the previously loaded structure. This is useful as you reload the same file after correcting a problem.

Choosing boundary condition

Use the **boundary** command to set the boundary condition; sphere or box. When a boundary has been chosen the centre and radius in the spherical case, and boxlengths in the periodical case are specified. It is important that the boundary condition defined in the topology file is consistent with the boundary condition used when running dynamics with **Qdyn6**.

Adding solvent

Use the **solvate** command to add solvent molecules to the loaded structure. Using spherical boundary solvent can be generated by any of the following methods:

- Using randomly oriented molecules on a grid. This option does not depend on any solvent coordinate file but only asks which library entry should be used. The density is specified in the library entry.
- By reading solvent coordinates from a solvent file. The solvent file is similar to a PDB file, see Solvent file format on page 60. The solvent file contains coordinates for a sphere or box of solvent. In the case of a box, it will be replicated in all directions as needed and can thus be used to solvate a system of any size.
- By reading solvent coordinates from a restart file from a previous simulation of the same molecular system.

When using one of the two first methods, **Qprep6** first fills a sphere completely with solvent and then deletes molecules where heavy atoms are closer than a threshold distance to any heavy atom in the loaded structure. The threshold distance is controlled by the **Qprep6** preference value **solvent_pack**. Crystallographic solvent molecules (in the PDB file loaded by readpdb) more than 2 Å outside the solvent sphere will be excluded from the simulation to avoid excessive radial restraining forces.

If periodic boundary is used it is not possible to use a solvent file with a sphere of solvent. Moreover the threshold distance is also applied to solvent molecules near the boundary, to avoid crashes between solvent molecules in neighbouring boxes.

Currently, Q6 is able to simulate both tri-atomic molecules like the SPC [17] and TIP3P [18] water models, as well as more complex organic solvents such as chloroform, dichloromethane, methanol and ethanol in all-atom representation.

Note: Make sure that crystallographic water molecules and waters added by the solvate command have *the same residue name*! Otherwise **Qprep6** will mark the topology as mixed-solvent, which is not implemented in **Qdyn6** at present.

The software now has added support to more complex solvents than the previously sup-

ported three–atom solvent models. Those solvents are added in the same way to the final topology as previous solvents. A set of currently supported solvents is included in the force field directory of the program distribution. Other solvents can also be included, as long as the simulation parameters are known.

Adding cross-link bonds

Cross-link bonds such as disulphide bridges in proteins are not generated automatically they are not defined in the fragment library. Extra bonds can be added automatically by searching the solute molecules for close but not bonded atom pairs, or manually by specifying atom numbers. To generate cross-link bonds automatically, use the **xlink** command. For each close atom pair found, you need to confirm or reject the making of a bond. To add bonds manually, use the **addbond** and specify the atoms to be bonded, either by atom numbers or in the form residue_number:atom_name.

Generating the topology

The **maketop** command is used to create the topology in memory. The only input for this command is a name for the topology. The process involves the following steps which are all carried out by the maketop routine without further user input:

- Setting atom types and partial charges.
- Generating the lists of bonds, bond angles, torsion angles and improper torsions.
- Generating the neighbour exclusion and 1-4 neighbour lists.
- Generating coordinates for hydrogen atoms not included in the loaded structure.
- Marking atoms outside the simulation sphere as excluded
- Calculation of the 'effective solvent radius' used in the simulation to ensure correct density of the solvent. This radius which is based on the number and spatial distribution of solute and solvent atoms in the sphere, typically differs somewhat from the radius used in the solvation step.

Verifying the topology

The successful generation of a topology in the step above is no guarantee that it is correct. Fortunately **Qprep6** offers a number of commands to check the topology:

- **Checkbonds** is used to list bonds with a potential energy exceeding a specified threshold. This helps to identify errors in the connectivity.
- **Checkangs** lists angles with energies over a threshold.
- **Checktors** lists torsion angles with energies over a threshold.
- **Checkimps** lists high energy improper torsion angles. This is a very important step for users of force fields with harmonic improper potentials (GROMOS, charmm) since with a non-periodic potential it makes a huge difference if the angle is *e.g.* -179^o instead of +179^o if the (single) energy minimum is at +180^o although the structural difference is small. *Impropers with the wrong sign give rise to high energies and strong forces which will distort the molecule during the simulation and must be corrected.* (This is not a problem with periodic improper torsion potentials used in the other

force fields.) The sign of the angle depends on the order of the bonds in the fragment library entry, each permutation of the sequence of the bonds involved will change the sign of the angle. Instead of modifying library entries, reloading the libraries and remaking the topology, the **changeimp** command can be used to flip the sign of selected impropers or of all impropers with energy exceeding a threshold value.

Writing topology and coordinate files

The final step in making a topology is saving it to a file by using the **writetop** command.

You will also need to make a PDB (or mol2, see below) file from the topology to see the new numbers of the atoms. These are the atom numbers you need to refer to when setting up restraints and topology modifications for perturbation simulations. Use the **writepdb** command to write a PDB file containing all the atoms of the topology.

Setting preferences

A number of parameters that affect the operation of **Qprep6**, *e.g.* during solvation, but which are not normally changed are not required as input to the commands. These parameters may be changed by experience users by the preference mechanism in **Qprep6**.

The **prefs** command is used to list the values of user-setable parameters and the **set** command to change a value.

The preference parameters and their default values are listed in the table on page 55.

F.2 Running dynamics with Qdyn6

Once a molecular topology file has been generated with **Qprep6**, you can carry out MD, FEP and EVB simulations with **Qdyn6**. The simulation can have either spherical or periodic boundary condition, and can be executed either sequentially or in parallel.

In this section we will describe the basic functions of **Qdyn6** and go through the different options that are available for control of the dynamics runs. There are two main input files that are used to set up the dynamics specifications:

1. The **Qdyn6** input file that controls things like time-step, temperature, cut-offs, restraints etc.
2. The FEP file which is an auxiliary file whose function is to redefine the topology information for certain atoms. This enables the explicit control over selected force field parameters that is necessary for FEP and EVB calculations.

F.2.1 Simulation procedure

The MD simulation required for a free energy calculation often proceeds in multiple stages. Normally, the initial stage is run at a very low temperature with strong coupling to the temperature bath (similar to energy minimisation) to relax strain in the initial structure. Then may follow stepwise heating of the simulated system and equilibration for some time at the target temperature. After this comes the main simulation during which energy and

structure data is collected. For perturbation simulations, this phase is composed of a series of simulations using intermediate potentials defined by different sets of weight coefficients for the FEP states.

A separate **Qdyn6** input file is used for each sub-simulation. It is therefore practical to prepare a command file (shell script or batch file) which executes all the sub-simulations sequentially. The name of the input file is passed to **Qdyn6** as the first (and only) argument on the command line. (It is not possible to use redirection of the standard input stream by the < operator.) A simple example of such a file where **Qdyn6** is invoked once for each input file and the output redirected to a log file follows:

Table 1: Multi-stage simulation command file

Qdyn6	relax.inp	>	relax.log
Qdyn6	eq1.inp	>	eq1.log
Qdyn6	eq2.inp	>	eq2.log
Qdyn6	eq3.inp	>	eq3.log
Qdyn6	data1.inp	>	data1.log
Qdyn6	data2.inp	>	data2.log

F.2.2 Output generated by Qdyn6

The different data files generated by **Qdyn6** are (shown in the overview in Figure 1):

- General information about the progress of the simulation including energy summaries and temperature is written to the standard output device and normally redirected to a log file.
- Final coordinates and velocities are written to a 'restart' file to be used to start the next sub-simulation and, after conversion to a structure file (see Analyzing structures from the simulation on page 34), for viewing the final structure. This file is also updated during the simulation and if the forces and velocities become too large and the simulation is terminated prematurely. The file thus also serves a diagnostic purpose.
- Energy data for Q-atoms in each FEP state is written to an energy file every few time steps (determined in the input file).
- Coordinates for all or a subset of atoms are written to a trajectory file at regular intervals (determined in the input file).

The restart and energy files are Fortran binary files. The trajectory file follows the DCD format also used in other MD programs (Charmm, X-plor) and can be read by many visualization and trajectory animation programs. Please note that the trajctory reading routines in Q6 are not able to read files from other programs.

F.2.3 Preparing Qdyn6 input files

In this overview the various aspects of defining the simulation set-up are introduced. For more complete information, see **Qdyn6** input file format on page 67.

The **Qdyn6** input file contains the specification of the dynamics simulation. It is a text file divided into sections, starting with a section heading and containing information on the different aspects of the simulation. The sections are of two kinds:

- Sections where each line consists of a keyword and a values, with different keywords on each line.
- Sections where all the lines have the same formatting and together constitute a data set. No keywords are used here.

Only a few sections are mandatory, most are optional and they may appear in any order. The formatting within a section is flexible in that blank lines are permitted as well as comments (starting with !, # or *) at the end of lines or on separate lines. The format of the data in each record within a section is free (white space is not significant), but all data in the record must be on the same line.The units used are based on Å, K and kcal/mol (see Units on page 64).

Dynamics control information

The section [**MD**] is normally the first in the input file the most apparently required section, since it defines the core parameters of the simulation like the number of time steps, their size and the temperature. Below is an example of a basic MD section in an input file:

```
[MD]
steps           10000
stepsize        2.0
temperature     300
shake_solvent   on
shake_solute    off
lrf             on
```

Periodic boundary conditions

The [**PBC**] section contains options and settings for simulations with periodic boundary. The mere existence of the section header [**PBC**] is enough to indicate the boundary condition. Additional options is added as exemplified below:

```
[PBC]
rigid_box_centre    on
constant_pressure   on
max_volume_displ    65
pressure            1.5
```

The rigid_box_centre option gives a periodic box with fixed coordinates instead of centering the box around the solute. In the above example the Monte-Carlo constant pressure algorithm, is performed with target pressure 1.5 bar and maximum volume displacement 65 $Å^3$.

Non-bonded interactions

Cut-off radii for the non-bonded interactions for different categories of atoms are given in the section [**cut-offs**], as exemplified below:

```
[cut-offs]
solute_solute     10
solvent_solvent   10
solute_solvent    10
q_atom            10
lrf               10
```

The q_atom entry defines the cut-off for interactions between Q-atoms and non-Q-atoms. The lrf entry defines the distance for the Local Reaction Field long-range electrostatics cut-off. No cut-off is used for interactions among Q-atoms. When using periodic boundary conditions, make certain all cut-off radii are less than half the shortest boxlength. Another option is to set the lrf and q_atom distances to -1, indicating that the atoms should interact with all atoms within the simulation box.

Sphere

The **sphere** section defines parameters concerning the spherical boundary. The most frequently used parameter is the shell_radius that allow the user to restrain solute atom in a shell to their original coordinates as defined in the topology.

```
[sphere]
shell_radius   18   !Restrain solvent in inner shell
shell_force    10   !Restraining force constant
```

Solvent

The [**solvent**] section controls the solvent boundary restraints when simulating with a spherical boundary. This section is thus omitted when periodic boundary is used. It is possible to fine-tune the restrains, but the default values used if no data is given are adequate for most simulations. The contents may often be as simple as:

```
[solvent]
polarisation   on   !Enable solvent polarisation restraints
```

For more complex organc solvents, the polarisation restraints have to be turned off at the moment.

Update and data collection intervals

The frequencies of regular events in the simulation are defined in the section [**intervals**]. These events are the regeneration of the non-bonded pair lists and the writing of energies or coordinates to the energy, trajectory and output files. Example:

```
[intervals]
non_bond     25
output       5
energy       0     !No energy file
trajectory   100
```

This example specifies that the non-bonded pair lists should be regenerated every 25 time steps, energy summaries written to the terminal or log file every 5 steps, no energy file is to be written and coordinates written to the trajectory every 100 steps.

Trajectory

If the coordinates of only a subset of the atoms are to be stored in a trajectory file, the selection of atoms is done in the section [**trajectory_atoms**], which could look as follows:

```
[trajectory_atoms]
heavy not excluded residue  1    104
residue                     105  106
residue                     109
```

In this atom mask the heavy atoms of residues 1 to 104 which are inside the simulation sphere and all atoms of residues 105-106 and 109 are selected. For further information see Atom masks on page 52.

Files

The names of files to be read and written are grouped together in this section. A topology file and a name for the final coordinates file must always be specified here. A restart file may be specified to start the simulation using the final coordinates and velocities from a previous simulation of the same system. For perturbation simulations the name of an FEP file is needed. If trajectory and energy files should be generated they need to be named here.

```
[files]
topology    molecule.top
final       data_01.re
trajectory  data_01.dcd
energy      data_01.en
fep         molecule.fep
```

FEP state weight coefficients λ

For multi-state perturbation simulations the mapping vector λ whose components are the weight coefficients for the FEP states is given on a single line under the section heading [**lambdas**]. For a simple two-state mapping potential with 70% of state 1 and 30% of state 2 it would look like this:

```
[lambdas]
0.70 0.30
```

Restraints

Several types of geometrical restraints can be applied to the simulated system to eliminate large movements, maintain interatomic distances or to stop the diffusion of a solute towards the sphere boundary. The most straight-forward type of restraints are harmonic potentials applied to restrain a sequence of atoms to their initial coordinates (in the topology file). This type of restraining is specified in the [**sequence_restraints**] section and requires only the number of the first and last atom of the sequence and a force constant. The restraints may be applied to heavy atoms only or to all atoms in the sequence. Instead of restraining

each atom individually to its initial position, the set of atoms can be restrained as a whole to its initial geometrical centre. In this case identical forces are applied to all the atoms. This alternative, used with a low force constant, is useful *e.g.* to keep a small solute molecule at the centre of the simulation sphere without hindering its tumbling motion (rotation). Both variants are exemplified below:

```
[sequence_restraints]
21   40   5.0   0
65   72   2.0   1   1
```

Here, atoms 21 to 40 are restrained to their initial positions by 5.0 kcal·mol^{-1}·Å^{-2} but hydrogens are excepted (0). Atoms 65 to 72 including hydrogens (1) are restrained as a group to their initial geometrical centre (1).

Restraints on individual atoms are not restricted to use the initial position as a reference since the "target" position is specified in the input. The restraint may be applied only in a single FEP state or in all states. In the first case the force is scaled by the weight coefficient λ for that state. Different force constants may also be used for the x, y and z axes. By setting one or two force constants to zero, the atom will be restrained to a line or a plane, respectively. An example of an [**atom_restraints**] specification follows:

```
[atom_restraints]
8   82.5   28.32   72.6   5.   5.   5.   0
```

In this case atom 8 is restrained to the point (x, y, z) = (82.5, 28.32, 72.6) with 5.0 kcal·mol^{-1}·Å^{-2} along all axes in all FEP states (0).

The distance between two atoms may be restrained using either a standard harmonic potential or a flat-bottomed harmonic well potential, by adding an entry under the heading [**distance_restraints**] as follows:

```
[distance_restraints]
13   20   4.5   5.0   10.0   1
```

Atoms 13 and 20 are here held together by a flat-bottomed harmonic well potential which is zero between 4.5 and 5.0 Å and has a force constant of 10.0 kcal·mol^{-1}·Å^{-2} for other distances. It is active in FEP state 1 only.

Another means of restricting the overall motion of a molecule (when using spherical boundary) is to apply a soft-wall or half-harmonic restraint outside a given radius from the (solvent) sphere centre. This is done in the section [**wall_restraints**] *e.g.*:

```
[wall_restraints]
80    99    14.0   5.0   0   0   0
102   102   14.0   5.0   0   0   0
```

In this example atoms 80 to 99 and 102 will experience an inward harmonic force if they are beyond 14 Å from the sphere centre. The force constant is 5.0 kcal·mol^{-1}·Å^{-2} but force will not be applied to hydrogen atoms (last 0). D_e is the depth of the Morse potential and a is the exponential coefficient of the Morse term. For obvious reasons the [**wall_restraints**] section is not used in combination with periodic boundary conditions. One can choose between harmonic or Morse potential.

As one last possibility of restraints in Q6, the [**angle_restraints**] are a specific type of force, based on the harmonic angle forces present in any force field. It can be useful, for instance, if you want to avoid an aspartate bound to a metal center to not be bidentally coordinated. In order to define an angle restraint, one has to define three atoms, as follows:

```
[angle_restraints]
133   132   3400   180.0   3.0   2
```

In this case, a harmonic angle force is applied between the atoms 133, 132 and 3400, in order to enforce a 180°, with 3.0 kcal·mol^{-1}·degrees^{-2} at the state 2.

If the interactions between residues and the reactive regions should be excluded for the energy calculation, this can be specified using the [**group_contribution**] keyword. After this, a list of residues or atoms can be specified as follows below:

```
[group_contribution]
residue   full   123
atom      elec   359 360 361
residue   vdw    14 15
```

Here, three groups of residues or atoms will be removed in additional energy calculations to assess their influence on the overall reaction, with the total, electrostatic or van–der Waals interactions being affected, respectively.

The newest addition to Q6 is performing BQCP [9, 10] calculations. This is possible both during the normal dynamics and as post–processing of trajectory files. In either case, the section [**QCP**] is added to the input file as shown below:

```
[QCP]
qcp_size     default
selection    hydrogen
qcp_pdb      qcp.pdb
sampling     20
```

This calculation would generate the energies for the system with quantum corrections for the hydrogen atoms, with 20 sampling steps for the classical coordinate and a ring–polymer size of 32 beads, writing the bead coordinates to the file qcp.pdb.

F.2.4 Qdyn6 input file examples

We give two annotated examples below. The first is the simplest possible input file, using default values for all optional parameters. The second is a bit more elaborate and exemplifies the use of many extra options such as special restraints. Detailed information about the data in each section is found in the section **Qdyn6** input file format on page 67.

Table 2: Minimal **qdyn** input file

Data		**Description**
[MD]		Basic data for the simulation
steps	2000	Number of steps
stepsize	1.0	Step size (fs)

Table 2: Minimal **qdyn** input file

Data		**Description**
temperature	1	Temperature (K)
initial_temperature	1	Temperature (K) for Maxwell-distributed initial velocities
[files]		File names for input and output
topology	molecule.top	Topology file
final	molecule.re	Restart to write at end of simulation

Table 3: Advanced **Qdyn6** input file

Data		**Description**
[MD]		Basic data for the simulation
steps	10000	Number of steps
stepsize	2.0	Step size (fs)
temperature	300	Temperature (K)
thermostat	berendsen	Thermostat used (see pg. X for more info)
bath_coupling	10	Temperature bath relaxation time (fs)
random_seed	57643	Seed for random number generator (only for initial vel.)
initial_temperature	300	Temperature (K) for Maxwell-distributed initial velocities
shake_solvent	on	Shake bonds & angles of water
shake_hydrogens	on	Shake bonds to hydrogen in solute & solvent
lrf	on	Use lrf for electrostatics beyond cut-off
[cut-offs]		Cut-off radii for different groups of atoms
solute_solute	10	Solute-solute cut-off (Å)
solvent_solvent	10	Water-water cut-off (Å)
solute_solvent	10	Solute-water cut-off (Å)
q_atom	10	Q-atom non-q-atom cut-off (Å)
[sphere]		Definition of the simulation sphere
shell_radius	18	Definition of the inner restrained shell (Å).
shell_force	10.0	Restraining force constant in shell (kcal·mol^{-1}·Å^{-2})
[solvent]		Solvent boundary settings
radial_force	60.0	Force constant for radial restraining (kcal·mol^{-1}·Å^{-2})
polarisation	on	Use polarisation restraints (this is the default)
polarisation_force	20.0	Force constant for polarisation restraining (kcal·mol^{-1}·rad^{-2})
[intervals]		Intervals for saving data
non_bond	25	Interval for generation of non-bond lists (steps)
output	5	Interval for energy summary in output
energy	10	Interval for energies to energy file
trajectory	100	Interval for coordinates to trajectory file

Table 3: Advanced **Qdyn6** input file

Data	**Description**
[trajectory_atoms]	Select atoms to be included in the trajectory file
heavy not excluded 1 104 residue	Select heavy atoms of residues 1 to 104 which are not excluded
residue 105 106	Select all atoms of residue 105 to 106
residue 109	Select all atoms of residue 109
[files]	File names for input and output
topology molecule.top	Topology file
final data_01.re	Restart to write at end of simulation
trajectory data_01.dcd	Trajectory file to write
energy data_01.en	Energy file to write to
fep molecule.fep	FEP file
[lambdas]	Weights for the FEP states
0.70 0.30	Lambda value for each state
[sequence_restraints]	Restrain contiguous sequences of atoms to initial coordinates
21 40 5.0 0	First & last atom, force const. (kcal·mol^{-1}·Å^{-2}), H-flag
65 72 2.0 1 1	First & last atom, force const., H-flag, restrain-to-centre-flag
[atom_restraints]	Individual atom positional restraints
8 2.5 8.3 7.6 5. 5. 5. 0	atom, x0,y0,z0, fcx, fcy, fcz, FEP state (0=all)
[distance_restraints]	Atom-atom distance restraints
13 20 4.5 5.0 10.0 1	Atom i, atom j, lower r, upper r, fc, FEP state (0=all)
[wall_restraints]	Half-harmonic (elastic wall) sequence restraints
80 99 14.0 5.0 0 0 0	First & last atom, r0 (from water centre), fc, D_e (kcal· mol^{-1}), a (Å^{-1}), H-flag
102 102 14.0 5.0 0 0 0	First & last atom, r0 (from water centre), fc, D_e, a, H-flag

F.2.5 FEP file

The purpose of the FEP file is to define a set of atoms as Q-atoms and to redefine their interaction parameters. All kinds of force-field parameters for these atoms can be controlled and several different "states" can be defined. The parameters for the different states may differ very little, *e.g.*, in the van der Waals parameters of a single atom, or the states can represent different valence bond structures. A typical application of the latter case would be to model reactants and products of a chemical reaction to be investigated by EVB simulation as two different states or, for a multi-step reaction, one state for the products of each elementary reaction step. In such a model of a reaction bonds, angles, torsions, partial charges, vdW parameters etc. may change for many atoms.

The idea behind this definition of different states is that **Qdyn6**, for each configuration

of the system's particles, will keep track of the energies of each state and write these to the energy file. The mapping potential or sampling potential used to generate the forces controlling the dynamics is a mixture of the FEP/EVB states, determined by the mapping parameter or weight coefficient λ given to each (pure) state in the **Qdyn6** input file. The free energy differences between FEP/EVB states can then easily be calculated by **Qfep6** using the standard FEP formula or the potential of mean force (umbrella sampling) approach to obtain the EVB ground state reaction free energy profiles.

The FEP file has the same overall structure as the **Qdyn6** input file (see page 19) with various kinds of data grouped into sections, the majority of which are optional. We will describe FEP files for a couple of prototype cases, starting with the simpler ones. For a complete description of the file format, see FEP file format on page 74.

Example: Charging a benzene molecule

The FEP file shown below may be used to calculate the electrostatic contribution to the free energy of solvation for a benzene molecule. The atoms and bonds of the molecule are defined in a topology file (not shown). In our topology the carbon atoms have odd numbers and hydrogens have even numbers.

Data			**Description**
[FEP]			Free energy perturbation
states	2		No. of states
[atoms]			Designate atoms in topology as q-atoms
1	1		
2	2		
3	3		
4	4		
5	5		
6	6		
7	7		
8	8		
9	9		
10	10		
11	11		
12	12		
[change_charges]			Assign new charges for each state
1	- 0.15	0.0	Q-atom no., charges in state 1 & 2
2	+0.15	0.0	
3	- 0.15	0.0	
4	+0.15	0.0	
5	- 0.15	0.0	
6	+0.15	0.0	
7	- 0.15	0.0	
8	+0.15	0.0	
9	- 0.15	0.0	
10	+0.15	0.0	
11	- 0.15	0.0	

Data			Description
12	+0.15	0.0	

The value following the keyword states in the section [**FEP**] is the number of FEP/EVB states. In the [**atoms**] sections atoms from the topology are designated as q-atoms. The first column of the data records in this section is the q-atom number given to the atom (used later to refer to it) and the second column is the number of the atom in the topology. The data in the section [**change_charges**] defines the charge of q-atoms in each state. Here we are changing the charges of all atoms, but in general only the charges which change need to be listed.

In the case above we have made no changes to the bonded or vdW parameters of the benzene molecule and the FEP file is simply used to define two "charge" states, one with the CH dipolar charges being ±0.15 e and one state with zero partial charges.

Example: Changing van der Waals parameters

In this example we will take a look at how to redefine van der Waals (Lennard-Jones) interaction parameters. The FEP file shown below may be used if we want to calculate the difference in hydration free energy between two ions, in this case Na^+ and K^+. Since the ions have the same charge the only change that needs to be made in a perturbation calculation between the two ions is to define two sets of Lennard-Jones interaction parameters. Please note that changes to the mass of a particle have no effect during the calculation, except when quantum corrections are calculated.

Table 4: FEP file for perturbation of Na^+ to K^+.

Data					Description		
[FEP]							
states	2				No. of states		
[atoms]					Designate atoms in topology as q-atoms		
1	1				Q-atom no., topology atom no.		
[atom_types]					Define new atom types (LJ parameters,)		
!Name	Ai	Bi	Ci	ai	Ai(1-4)	Bi(1-4)	Mass
Na	143.70	3.89	0.0	0.0	0.0	0.0	22.99
K	522.70	4.35	0.0	0.0	0.0	0.0	39.10
[change_atoms]					Assign new atom types to Q-atoms		
1	Na	K			Q-atom no., q-atom type in states 1 and 2		

Here we again define two states, but now only for one Q-atom that has number 1 in our simple topology file which only contains the single ion. No charges need to be changed since both ions are monovalent cations, and the section [**change_charges**] is therefore omitted. The only specific definitions needed here are the following. In the section [**atom_types**] the parameters for the atoms involved in the perturbation are given. Whether (A_i, B_i) or (R^*, ε) LJ parameters are used depends on the combination rule specified in the FF parameter file used to generate the topology. The first column is the name of the Q-atom type, then follows the Lennard-Jones A_i (or R^*) and B_i (or ε) parameters. Columns four to seven are not used in this case (two parameters for the exponential repulsion function and two LJ parameters for 1-4 interactions). The last column is the atomic mass. The

[**change_atoms**] section states that q-atom number one is of type Na in state 1 and type K in state 2. The user has to be aware at this point the the atom mass specified in this section is only evaluated for the calculation of quantum effects. All other calculations are performed with the atom masses defined in the force field used for the simulation.

So, in this example all we have done is to define the relevant LJ parameters for Na^+ and K^+ (Q-atom types for Na and K) as the two different states for our single ion.

Example: Valence bond (EVB) states for a proton transfer reaction

This is an example from the reaction of a protein tyrosine phosphatase where proton transfer from a Cys residue of the enzyme to the doubly negatively charged phosphate group of the substrate (phenylphosphate) is considered. The states representing different bonding arrangements we want to define are schematically drawn in figure 2, where also the topology number of the relevant atoms are given.

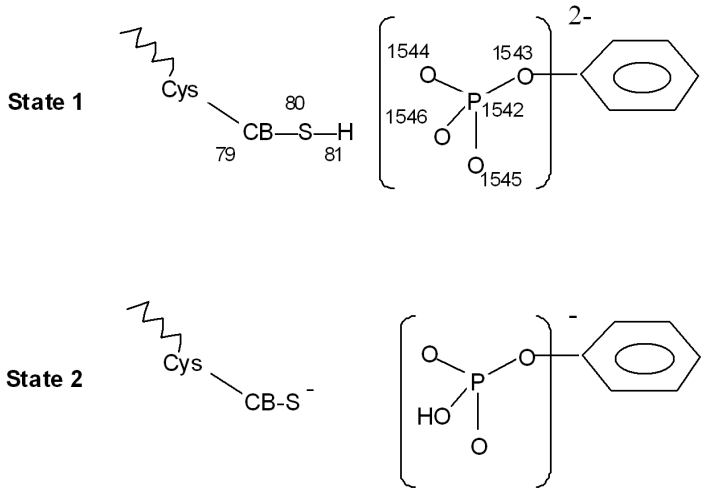

Figure 2: EVB states for a proton transfer reaction.

The FEP file below describes the two EVB states used for calculating the free energy profile of proton transfer in a particular enzyme [19]. It is beyond the scope here to describe the EVB method in detail, but reviews on this topic are available [5].

Here we want to define the first state with the proton (H) attached to the sulphur atom of the cysteine and the phosphate group doubly charged. In the second state the proton is on a phosphate oxygen and one negative charge is now on the sulphur atom. Here there are changes in both partial atomic charges, vdW parameters, bonds, angles etc. between the two states.

Table 5: FEP file for proton transfer reaction.

Data	**Description**
[FEP]	
states 2	no. of states
[atoms]	Designate atoms in topology as Q-atoms

Table 5: FEP file for proton transfer reaction.

Data							**Description**
1	79						Q-atom no., topology atom no.
2	80						
3	81						
4	1542						
5	1543						
6	1544						
7	1545						
8	1546						
[change_charges]							Assign new charges for each state
1	0.180	0.000					Q-atom no., charges in state 1 & 2
2	-0.450	-1.000					
3	0.270	0.398					
4	0.540	1.230					
5	-0.360	-0.360					
6	-0.860	-0.860					
7	-0.860	-0.860					
8	-0.860	-0.548					
[atom_types]							Define new atom types (LJ parameters, . . .)
!Type	Ai	Bi	Ci	ai	Ai(1-4)	Bi(1-4)	Mass
P	2303.00	59.35	0.0	1.581	2303.00	59.35	30.97
OE	600.00	23.25	70.0	1.581	600.00	23.25	16.00
OD	956.00	23.01	70.0	1.581	956.00	23.01	16.00
H	0.00	0.00	6.5	1.581	0.00	0.00	1.00
C2	2500.00	46.06	0.0	1.581	2500.00	46.06	14.00
SH	2001.57	44.74	165.0	1.581	2001.57	44.74	32.06
S-	2720.00	136.00	165.0	1.581	7200.00	136.00	32.06
[change_atoms]							Assign new atom types to Q-atoms
1	C2	C2					Q-atom no., Q-atom name in states 1 & 2
2	SH	S-					
3	H	H					
4	P	P					
5	OE	OE					
6	OD	OD					
7	OD	OD					
8	OD	OE					
[soft_pairs]							Atom pairs which have C*e$^{(-ar)}$ repulsion
2	3						Q-atom i, j
3	8						
[excluded_pairs]							Atom pairs to exclude from non-bonded interactions

Table 5: FEP file for proton transfer reaction.

Data						**Description**
81	1544	0	1			Atom i, j, exclusion flag for states 1 & 2
81	1545	0	1			
[bond_types]						Define Morse bond types
1	85.0	2.00	1.61			No., D_e, α, b_0
2	120.0	2.00	1.49			
3	84.0	2.00	1.43			
4	110.0	2.00	1.00			
5	94.0	2.00	1.33			
6	112.5	2.00	1.80			
7	100.0	2.00	1.53			
[change_bonds]						Redefine bonds
1542	1546	2	1			Atom i, j, type in state 1 & 2
80	81	5	0			type 0 means no bond
1546	81	0	4			
[angle_types]						Define new angle types
1	95.0	109.6				
2	140.0	120.0				
3	115.0	120.0				
4	110.0	109.6				
5	0.0	0.0				
6	110.0	113.0				
7	95.0	96.0				
[change_angles]						Redefine angles
1544	1542	1546	2	1		Atom i, j, k, type in state 1 & 2
1545	1542	1546	2	1		
1542	1546	81	0	4		Type 0 means no angle
79	80	81	7	0		
[torsion_types]						Define new torsion types
1	0.75	3.0	0.00			Number, force const., mult, delta
2	0.70	3.0	0.00			
[change_torsions]						Redefine torsions
1543	1542	1546	81	0	1	Atom i, j, k, l, type in state 1 & 2
1544	1542	1546	81	0	1	Type 0 means no torsion
1545	1542	1546	81	0	1	
78	79	80	81	2	0	
[angle_couplings]						Define angles to be coupled with Morse bonds
3	3					Q-angle no., Q-bond no.
4	2					

Table 5: FEP file for proton transfer reaction.

Data						**Description**
[torsion_couplings]						Define torsions to be coupled with Morse bonds
1	3					Q-torsion no., Q-bond no.
2	3					
3	3					
4	2					
[off_diagonals]						Define off-diagonal (H_{ij}) functions
1	2	2	8	1.0	0.45	State i, state j, Q-atom 1, Q-atom 2, $A_{i,j}$, $\mu_{i,j}$

In this example we define eight atoms as Q-atoms whose charges, vdW parameters and bonding arrangement will change between the two states (reactant and product state) that we describe by the FEP file. The sections [**FEP**], [**atoms**], [**change_charges**], [**atom_types**] and [**change_atoms**] are used as above, that is, we redefine the charges and vdW parameters of the eight Q-atoms. *e.g.*, atom no. 2, the sulphur, will change its charge from -0.45 to -1.00 and its vdW parameters are changed from Q-atom type SH to S-. In this model of the reaction we will also make use of a non-Lennard-Jones non-bonded potential for certain pairs of atom that make and break bonds as listed in the section [**soft_pairs**]. For these atoms, it is more physical to use an exponential function for the repulsion than the normal $1/r^{12}$ form which causes a too strong repulsion at very short distances. The vdW interaction between these pairs of atoms is given by:

$$V_{soft} = C_i \cdot C_j \cdot e^{(-a_i \cdot a_j \cdot r_{i,j})}$$

where $r_{i,j}$ is the distance between the specific atom pair subjected to this potential. The C's and a's are atom-type specific parameters and the combination rule is geometric as can be seen from the formula. Note the absence of the attractive $1/r^6$ term.

Morse potential parameters for bonds that are broken or formed are given in the [**bond_types**] section. The section [**change_bonds**] lists the bonds, identified by pairs of atoms and the bond parameters to use in each state. If a bond is already defined in the topology then the normal, harmonic potential will be turned off. The absence of a bond is specified by setting the bond type to 0. Bond angles are redefined in an analogous way, but the functional form of the Q-atom angles is harmonic, like the normal angles. Parameters for the new angle types are given under [**angle_types**] and the angles for which the new types should be used are listed in the [**change_angles**] section. Redefining torsions is done in the same way (sections [**torsion_types**] and [**change_torsions**]). No improper torsions are changed in this example.

Angles, torsions and impropers depend on the existence of bonds connecting the atoms defining the angle. Angles of all kinds can therefore be coupled to bonds, in which case the angle energy will be scaled by the ratio of the actual value of the Morse bond energy to the dissociation energy [20]. In the example angle 6 (POH) is coupled to bond 3 (OH) and

angle 7 (CBSH) to bond 2 (SH), according to the [**angle_couplings**] section. Coupling torsions and impropers (not in the example) work the same way.

Off-diagonal elements of the Hamiltonian are defined in the section [**off_diagonals**]. They are represented by $H_{i,j}=A_{i,j} \cdot e^{(-\mu_{i,j} \cdot r_{k,l})}$ where i and j are the two states involved and $r_{k,l}$ is the distance between a specific pair of atoms k and l. The single record in this example defines mixing of states 1 and 2 ($H_{1,2}$) for q-atoms 2 and 8 with A=1.0 and μ= 0.45.

F.2.6 Additional information for BQCP calculations

For the use of the BQCP implementation, specialized atom selections can be provided in the file under the header [**qcp_atoms**]. If this section is found, atoms will be assigned to be treated using the BQCP approach in the order giving. The atom numbers used below are taken from the proton transfer example above.

[qcp_atoms]	
1 2	! S donor
2 3	! transferred hydrogen
3 8	! O acceptor

In this case, the donor, acceptor and transferred atoms are designated to be treated this way. More general selections can also be done using atom selections in the input file for **Qdyn6**.
If the calculation of kinetic isotope effects is desired, the user needs to provide the masses for the atoms under the header [**qcp_mass**]. If this section is not found, or if atoms are missing from it, and the user requested the calculation for different isotopes in the **Qdyn6** input file, masses will be automatically read from the atom type definitions.

[qcp_mass]	
2 2.014	! Calculate for Deuterium in additon to protium

The example above instructs the program to also calculate the BQCP energies for the exhange of the transferred hydrogen to deuterium, while keeping the other atoms unchanged.

F.2.7 Monitoring non-bonded interactions

In analysing the details of *e.g.* receptor-ligand interactions, it is useful to define some groups of atoms and calculate the non-bonded interactions between pairs of atom groups. The example FEP file below describes how to use this feature to get the non-bonded energies between the pterinine ring of a dihydrofolate reductase inhibitor and some amino acid side chains and an amide group of a co-factor.

[monitor_groups]							
266	267	268					!GLU 30 COO-
317	318	319	320	321	322		!PHE 34 side chain
1897	1898	1899	1900	1901	1902	1903	!part of MTX pteridine ring 1
1908	1909	1910	1911	1912	1913	1914	!ring 2 of pterindine
1880	1881	1882	1883	1884	1885		!amide of NADPH
[monitor_group_pairs]							
1	3						
2	4						
2	5						

Five groups of atoms are defined, and the interactions between groups 13, 24 and 25 should be calculated. The energies are evaluated separately for different FEP states and presented in the energy summaries in the **Qdyn6** output. In this example only a single state is defined so the λ-weighted averages are identical to the energies in state 1. The program will check on its own in how far the specified groups are able to actually interact, and will only calculate nonbonded energies if the atoms are not bonded to each other.

========== Monitoring selected groups of nonbonded interactions ==========

pair	Vwsum	Vwel	Vwvdw	1:Vel	1:Vvdw
1	-58.48	-65.65	7.18	-65.65	7.18
2	-1.72	0.00	-1.72	0.00	-1.72
3	-0.08	0.00	-0.08	0.00	-0.08

where the columns are: atom group pair number, total energy for all states weighted by λ, weighted sum of electrostatic energies, weighted sum of Lennard-Jones energies, electrostatic energy in state 1, Lennard-Jones energy of state 1.

There is a similar feature in **Qcalc6** (see page 80) where one can analyze non-bonded interactions from saved trajectory files.

F.2.8 Obtaining residue non-bonded contributions over complete free energy calculations

The [**group_contribution**] routine mentioned above can be used to obtain the residue contributions over the full reaction, with additional sections added to the energy files for analysis in **Qfep6**. For the syntax used please refer to section H.7.1. The routine will add one addtional calculation of the residue contributions towards the ttoal energy of the system at every point that the energies are saved, to allow the substraction from the total energy in equation 1.

$$\begin{aligned} E_{total} =& E_{qq,el} + E_{qq,vdw} + E_{qp,el} + E_{qp,vdw} \\ &+ E_{pp,el} + E_{pp,vdw} + E_{bond} + E_{angle} + E_{torsion} \end{aligned} \tag{1}$$

The reduced energies are then calcualted by simply substraction the individual residue contributions from the total values in equation 2.

$$E_{exclude} = E_{Total} - E_{qq,el}^{q-exc} - E_{qq,vdw}^{q-exc} - E_{qp,el}^{q-exc} - E_{qp,vdw}^{q-exc} \tag{2}$$

F.3 Parallel version of Q6

Even though computers become faster every year the work that a single computer can do is limited. A single execution of Q will take hours or even days. If a several computers were able to work in parallel with the same job the execution time could be reduced substantially.

The most common way to run a parallel job is to use a computer cluster in which every node has a separate processor and hard drive. The nodes communicate through a fast network switch providing an environment suitable for running parallel program. Q has been parallelised to fit these type of machines.

The parts of Q6 that can be run in parallel are **Qdyn6** which contain the time demanding conformational sampling and **Qpi6**, which performs the equally time demanding quantum correction calculations. The parallel version is suitable to run on 2 - 12 processor codes depending on the size of the problem.

F.3.1 Running Q6 on a cluster

Q version 5.0 and higher can be run in parallel on clusters. The parallel version is easy to use. The only requirement is a computer cluster with high bandwidth and a version of Message Passing Interface (MPI) installed. MPI is a standard interface for communication between nodes in a cluster. To run Q6 on the cluster you need the parallel version of Q6, *i.e.* the one that has been compiled with the MPI-flag activated. If you compile the program yourself define the variable USE_MPI to the preprocessor. To check that you have the right version execute the program and confirm that the suffix "_parallel" is added to the version info in the log file. Look at the top of the file where it should read "**Qdyn** version 6.X.X_parallel initialising....". The parallel version can be executed on a single node but still requires MPI installed.

F.4 Analysis of results

F.4.1 Analyzing structures from the simulation

Qprep6 is also used after the simulation to convert the binary trajectory and restart coordinate files generated by **Qdyn6** to PDB or mol2 files suitable for viewing in a molecular graphics program.

Making PDB or mol2 files from restart or trajectory files

Use the following steps to generate viewable files from individual "snapshot" structures:

1. Load the topology file using the **readtop** command. The fragment library files used to generate the topology will be loaded automatically, if available. Otherwise load the libraries using **readlib**.

2a. Load the binary coordinate file using **readx**.

2b. Open a trajectory file with the **trajectory** command. You will be prompted if you want to use the atom mask from the trajectory so that only atoms in the trajectory will appear in structure files written. Read coordinates from the trajectory file with **readframe**

3. If you want to select specific atoms to include in the structure files to be written, use the **mask** command . First enter mask none to clear the current mask, then add atoms to the mask using the syntax described in the section Atom mask on page 52.

4a. Write a PDB file using the **writepdb** command. It may be written with or without gap markers.

4b. Write a mol2 file using the **writemol2** command.

5. Repeat steps 2 and 4 to convert more files. To read the next frame from a trajectory use the **readnext** command and then go to step 4. Note that CONECT records in PDB files (defining atomic connectivity) will only be generated for fragments whose library entries include PDBtype HETATM in their info section.

Trajectory animation

There are a number of programs for visualising and analysing MD trajectories, *e.g.* Visual Molecular Dynamics, VMD [21, 22] and gOpenMol [23, 24] which can read the DCD format trajectories generated by **Qdyn6**. To create a PDB file with the same set of atoms as in the trajectory, as required by the visualization programs, execute the steps below in **Qprep6**:

1. Load the topology file using the **readtop** command. The fragment library files used to generate the topology will be loaded automatically, if available. Otherwise load the libraries using **readlib**.
2. Open a trajectory file with the **trajectory** command. You want to use the atom mask from the trajectory (answer y for yes at the prompt).
3. If you want to use another coordinate set than that of the topology for your PDB file, use **readframe** or **readx** as described above.
4. Write a PDB file using the **writepdb** command. Don't include gap markers.

F.4.2 Free energy calculation using Qfep6

Performing free energy perturbation (FEP) calculations with Q involves running a set of consecutive input files which have the mapping parameter vector λ ranging in a desired way (usually [1, 0] to [0, 1] for two states). **Qfep6** is a program which reads the energy files generated by **Qdyn6** and calculates the total change in free energy for the complete perturbation from state A (ε_1) to state B (ε_2). The difference in free energy between the two states is calculated by Zwanzig's formula:

$$\Delta G = \sum \Delta g = \sum -R \cdot T \cdot \ln \left\langle e^{-\left(\frac{\Delta V_{eff}}{R \cdot T}\right)} \right\rangle_A \tag{3}$$

Table 6: **Qfep6** input file for FEP/EVB evaluation

Line	**Data**	**Description**
1	11	Number of energy files
2	2 0	Number of states, number of predefined off-diagonal elements (from .fep-file, 0 means redefine)
3	0.596 100	kT, number of points to skip in each energy file
4	40	Number of gap bins
5	20	Minimum number of points per bin
6	12.3	Energy shift $\Delta\alpha_{ij}$ (for states$\neq$1)
7	1	Number of off-diagonal elements $\neq$0
7.1	1 2 18.1 0.32 0.0 2.0	i, j, A_{ij}, μ_{ij}, η_{ij}, r_{0ij}
8	1 -1	Linear combination of states defining the reaction coordinate ($\varepsilon_1 - \varepsilon_2$).
9	md_00.ene	List of energy files
10...	md_01.ene	
...19	md_10.ene	

where $V_{eff} = \lambda_1 \cdot \varepsilon_1 + \lambda_2 \cdot \epsilon_2$, $\Sigma\lambda_n = 1$. ΔV_{eff} is the difference in V_{eff} between two adjacent perturbation steps.

The program returns a list containing average energies and lambda values for each file. After that, the free energy change between each perturbation step (file) is summarised. The change is calculated relative to both the previous and the following perturbation step (dGf and dGr for forward and reverse way respectively). The accumulated sum of the energy changes between ε_1 to ε_2 is also given (sum(dGf) and sum(dGr)), as well as the average accumulated change calculated from the forward and reverse way $\langle$dG$\rangle$.

Qfep6 also calculates free energy functions, or potentials of mean force, by the perturbation formula:

$$\Delta G(X_m) = \Delta G(\lambda_i) - R \cdot T \cdot \ln \left\langle e^{-\left(\frac{E_g(X_m) - V_i(X_m)}{R \cdot T}\right)} \right\rangle_i \tag{4}$$

The reaction coordinate X is defined as the energy gap between the states $X = \Delta V = \varepsilon_1 - \varepsilon_2$ and is divided into intervals X_m (bins). The first term in the above equation represents the free energy difference between the initial state ε_1 and the mapping potential V_i:

$$\Delta G\left(\lambda_i\right) = -R \cdot T \cdot \ln \sum_{n=0}^{i-1} \left\langle e^{-\left(\frac{V_{n+1} - V_n}{R \cdot T}\right)} \right\rangle_n \tag{5}$$

The second term represents the free energy difference between the mapping potential V_i and the ground state potential E_g. The MD average in this term is only taken over those configurations where X belongs to X_m. E_g is the solution to the secular determinant:

$$\begin{vmatrix} \varepsilon_1 - E_g & \cdots & H_{1n} \\ \vdots & \ddots & \vdots \\ H_{n1} & \cdots & \varepsilon_n - E_g \end{vmatrix} = 0 \tag{6}$$

For a two-state representation the solution becomes:

$$E_g = \tfrac{1}{2} \cdot (\varepsilon_1 + \varepsilon_2) - \tfrac{1}{2} \cdot \sqrt{(\varepsilon_1 + \varepsilon_2)^2 + 4 \cdot H_{12}^2} \tag{7}$$

H_{ij} is the off-diagonal matrix element representing the quantum mechanical coupling of the states. This coupling is zero for normal FEP calculations. $H_{ij} \neq 0$ results in mixing of states i and j, which is desired when calculating reaction free energy profiles for reactions represented by the empirical valence bond (EVB) model. In **Qfep6** the off-diagonal element is a function of the form:

$$H_{ij} = A_{ij} \cdot (e^{-(\mu(r_{ij}-r_0)+\eta(r_{ij}-r_0)^2)}) \tag{8}$$

where r_{ij} is the measured distance between the reacting atoms. By choosing μ and η differently, H_{ij} can either be a constant, an exponential function or a gaussian function.

The EVB method allows calibration of simulated reference reactions to experimental data obtained from gas-phase or solution experiments. The two EVB parameters H_{ij} (mostly A_{ij}) and $\Delta\alpha_{ij}$ are varied until the calculated profile and the experimental data coincide. $\Delta\alpha_{ij}$ is a constant energy shift between the states that represents their difference in heat of formation, which is not included in the force field. Generalized, the $\Delta\alpha_{ij}$ parameter determines the ΔG° level and H_{ij} regulates the degree of mixing of the states at the transition state *i.e.* the $\Delta G^\ddagger$ level.

The energies describing the FEP functions and the reaction free energy profile are summarized in the last table generated by **Qfep6**. Note that each bin has contributions from several different values of λ. Likewise, each value of λ contributes to the sampling of several different bins.

It is possible to handle more than two valence bond states in **Qdyn6** and **Qfep6**, however sampling and calibration may be a difficult task for more than two states.

Figure 3: Example **Qfep6** output file

```
...file opened, and lambdas: md_00.ene   0.00 1.00
 Number of points:         1000
state   EQtot   EQbond  EQang   EQtor   EQimp   EQel    EQvdW
-------------------------------------------------------------
   1 -1087.71 -578.19    7.68    0.63    0.00 -541.46   22.70
   2 -1145.21 -586.25   31.48    1.16    0.00 -603.21   10.68

state   Eel_qq  EvdW_qq Eel_qp  EvdW_qp Eel_qw  EvdW_qw Eqrstr
-------------------------------------------------------------
   1   112.02    0.57  -18.78   -1.97 -634.70   24.10    0.00
   2     0.00   -0.30   -5.32   -2.51 -597.89   13.50    0.00
etc...
lambda(1) dGf   sum(dGf)    dGr   sum(dGr)    <dG>
-----------------------------------------------------
  0.00     0.00     0.00    -5.71  -284.20     0.00
  0.01     5.64     5.64    -5.54  -278.49     5.67
  0.02     5.63    11.27    -5.45  -272.95    11.26
  0.03     5.48    16.74    -5.28  -267.50    16.72
  0.04     5.38    22.12   -10.41  -262.22    22.05
  0.06    10.37    32.49   -10.07  -251.80    32.44
etc...
Min energy-gap is:   -292.238384870069
Max energy-gap is:    330.166319351831

Lambda1 Ibin Energy gap  dGa    dGb      dGg    # pts
-----------------------------------------------------
 0.00    1  -284.46   -31.28   247.20   -47.72     65
 0.00    2  -268.90   -31.28   231.53   -48.50    200
 0.00    3  -253.34   -31.28   216.31   -49.35    231
 0.00    4  -237.78   -31.28   201.25   -50.31     88
 0.01    2  -268.90   -31.40   231.23   -48.73     79
 0.01    3  -253.34   -31.26   216.16   -49.42    303
 0.01    4  -237.78   -31.13   201.59   -50.14    181
 0.01    5  -222.22   -30.99   189.55   -51.00     25
etc...
```

F.5 BQCP post-processing

The calculation of quantum corrections can be performed as a post-processing step for already completed simulations. For this, the program **Qpi6** can be used, together with the simulation topology, FEP file containing the information outlined above, a properly formatted input file and the simulation trajectory corresponding to different states of a simulation.

Input file The input file should be formatted in the same way as the file used to perform the initial simulation in **Qdyn6**, with a small number of changes. The [**MD**] section has to be renamed to [**GENERAL**], with the remaining lines left unchanged to ensure that the calculation will be performed on the same potential as during the classical simualtion. If the line **temperature** is not specified, **Qpi6** will require a restart file to calculate system temperatures from the velocities. An additional change has to be performed to the [**files**] section, where the keyword **trajectory** has to be replaced with **traj_input**. In general it is recommended to use the files from the classical simualtion and perform those edits on them, instead of preparing new files, to avoid differences in the energy calculation between classical simulation and BQCP post-processing. The files also need to include the [**QCP**] section, or the program will terminate due to invalid input. This section has to be formatted the same was as specified above. The only difference to calcualtions during dynamics is that the user needs to provide the file name for the write out of bead coordinates using the **qcp_pdb** keyword, or has to explicitly turn of the write out using **qcp_write off**. An example file is provided here, with the minimal input requirement.

BQCP Theory In the Quantum Classical Path description [25], the quantum energy of a system is calculated according to the path integral formulation of quantum mechanics as the average over the energy of the free particle distribution, represented as a ring polymer of n beads, over the average of the classical distribution. Each of the beads on the ring polymer represents one imaginary time slice in this case. The individual bead coordinates are sampled according to the bisection algorithm first developped by Ceperley [26], with the modifications later proposed by Gao *et al.* [10]. At each classical coordinate chosen for the BQCP calculation of energies, ring polymers will be generated at the position of the classical coordinates. For the complete derivation on how free particle positions are chosen, and the concepts behind QCP the reader is asked to consult the original research papers by Hwang and Warshel [25], as well as the work by Major and Gao [9, 10].

Table 7: Example **Qpi6** input file

[general]		! needed section to run calculation
lrf	on	! should be same as during simulation
temperature	300	! can be chosen freely, if missing calculated from restart file
[cut-offs]		! optional, should be identical to classical simulation
[files]		! required section
topology	sys.top	! same topology as during classical simulation
energy	sys.en	! file to write out energies to
fep	sys.fep	! same as during classical simulation
traj_input	sys.dcd	! file with coordinates to calculate on, from classical simulation
[QCP]		! required section
selection	all	! what subset of Q atoms should be treated as ring polymers
equilibration	10	! how many MC bisection steps should be taken before calculating energies
sampling	10	! how many ring polymer configurations hould be sampled
qcp_seed	-1	! random number seed for algorithm, -1 means take number from system time
qcp_kie	on	! calculate energies for default and isotope masses
qcp_write	on	! write bead coordinates to file
qcp_pdb	qcp.pdb	! file name for bead coordinate file
qcp_show	off	! only print minimal information about calculation
qcp_debug	off	! do not print debug information
qcp_size	default	! use default size of ring polymer, 32 beads
[lambdas]		! needed for FEP states
1.0	0.0	! Lambda values corresponding to classical simulation

F.6 Scoring

Three scoring functions are implemented in **Qcalc6**: X-Score [27], ChemScore [28] and PMF-Score [29]. X-Score and ChemScore are empiricial whilst PMF-Score is knowledge based.

All scoring functions require the topology to be loaded and the correct mask be specified. The initial topology (with coordinates from the .top-file) can be scored to verify atom typing.

Both trajectory and restart files can be scored. The following options are available in **Qcalc6** when specifying trajectory or restart files:

- adding *,frames=every n* means calculations will only be performed on every n:th frame.
- adding *,frames=n-m* means calculations will only be performed on frames n to m.
- specifying *mean* instead of a file name calculates the mean of all frames processed since start or since the last time "mean" was specified.

The input requested is similar for all three functions. To avoid confusion, examples of typical inputs will be given.

F.6.1 X-Score

Input Example input is presented below.

Prompt	Input
Topology file:	c:/peter/data/P450/adm/adm.top
Qcalc6>	xscore
Mask:	residue 1 407
Mask	.
Score initial topology? (yes/no)	yes
Q-atom (FEP) file:	c:/peter/data/P450/adm/lig.fep
Cofactor (. or EOL terminates):	restype=HEM
Cofactor (. or EOL terminates):	.
FF translation key:	qoplsaa
Scoring parameters:	xscore_default.input
Qcalc6>	go
Enter names of coordinate or restart files	c:/peter/data/.../md.dcd,frames=every 5 mean

Cofactors X-Score uses different typing schemes to assign atoms types to protein and ligands atoms. If needed, atoms in parts of the protein can be typed using the ligand atom-typing procedure by defining a cofactor. This is useful if the protein has some special residue, like HEM in P450, that is not defined in the X-Score residue library (file RESIDUE_DEF_XTOOL.dat). Ligand atoms are typed on the individual atom level, in

contrast to the residue level for protein atoms, using data in file ATOM_DEF_XTOOL.dat. Cofactor definitions are made on separate lines and has the form: restype=RES, where RES is the residue name, e.g. HEM. All atoms, regardless of their proximity to each other, in residues named RES will be included in the cofactor RES and typed as if they were ligand atoms (though in every other respect they are considered as part of the protein). Any number of cofactors can be defined.

Force field A force field translation key has to be given to allow for the translation of atom types according to the Q convention to types according to the Sybyl convention. The translation tables are in the file ATOM_TYPE_CONVERSIONS.dat (shared with PMF-Score). A different translation file can be specified in the input file (see below).

Parameters Scoring parameters, output specifications and data files are specified in an input file. Default parameters can be used by specifying *default* when asked for scoring parameters. In that case the following parameters and filenames are used:

SHOW_ABS	'NO'	! Show binding score for each atom?
SHOW_TOTAL	'YES'	! Show total binding score?
SHOW_LIGAND	'YES'	! Show ligand atoms?
SHOW_PROTEIN	'NO'	! Show protein atoms?
SHOW_COFACTOR	'YES'	! Show cofactor atoms?
APPLY_HPSCORE	'YES'	! Use hydrophobic contact algorithm?
APPLY_HMSCORE	'YES'	! Use hydrophobic matching algorithm?
APPLY_HSSCORE	'YES'	! Use hydrophobic surface algorithm?

RESIDUE_DEFINITIONS	residue_def_xtool.dat
ATOM_DEFINITIONS	atom_def_xtool.dat
LOGP_DEFINITIONS	atom_def_xlogp.dat
SURFACE_DEFINITIONS	surface_def_xtool.dat
ATOM_TRANSLATIONS	atom_type_conversions.dat

Applying more than one hydrophobic algorithm results in a consensus score. Default scoring coefficients are as reported in [27].

Output When SHOW_LIGAND and/or SHOW_PROTEIN is specified, a list of ligand and/or protein atoms is displayed, showing the atom type, residue, atomic properties, neighbouring atoms and bonds for each atom. In addition, a list of bonds and aromatic rings detected are output.

When scoring, the contribution from each term (van der Waals (VDW), H-bonding (HB), hydrophobic contact (HP), matching (HM) and surface (HS) and rotational(RT)) is displayed along with the total score. If SHOW_ABS was specified the contributions for every ligand atom is displayed. Atomic binding score is always displayed when scoring the initial configuration.

X-Score results are in pK_d units.

Data files The format of the data files is further explained in the respective files.

F.6.2 ChemScore

Input Example input is presented below.

Prompt	Input
Topology file:	c:/peter/data/P450/adm/adm.top
Qcalc6>	chemscore
Mask:	residue 1 407
Mask	.
Score initial topology? (yes/no)	yes
Q-atom (FEP) file:	c:/peter/data/P450/adm/lig.fep
Parameter file:	c:/peter/data/ff/chemscore_oplsaa.prm
Qcalc6>	go
Enter names of coordinate or restart files	c:/peter/data/.../md.dcd,frames=every 5 mean

Parameter file ChemScore reads all atom parameters from a single parameter file, though there are different files for different force fields. The parameter file defines the atomic properties of all atom types.

Output Prior to scoring, ChemScore outputs atom type and bond information for all ligand atoms, as well as information about rings detected. For every frame, the contribution from each term (H-bonds, metal contacts, lipophilic contacts and frozen rotatable bonds) is displayed along with the total score.

ChemScore results are in kJ/mol. A negative score means negative energy.

F.6.3 PMF-Score

Input Example input is presented below.

Prompt	Input
Topology file:	c:/peter/data/P450/adm/adm.top
Qcalc6>	chemscore
Mask:	residue 1 407
Mask	.
Score initial topology? (yes/no)	yes
Q-atom (FEP) file:	c:/peter/data/P450/adm/lig.fep
Force field translation key:	qoplsaa
Scoring parameters:	pmfscore_default.input
Qcalc6>	go
Enter names of coordinate or restart files	c:/peter/data/.../md.dcd,frames=every 5 mean

Presently, all atoms defined as solvent atoms are ignored. Critical water molecules should be defined as part of the protein or they will be excluded.

Force field As for X-Score, a force field translation key has to be given to allow for the translation of Q atom types to Sybyl atom types. The Sybyl type derived is only used to determine the hybridization of carbon and nitrogen atoms.

Scoring parameters Output options, data files and the maximum ring size considered are defined in an input file. The output options are similar to those in X-Score. The maximum ring size parameter determines the number of steps the ring finding algorithm will take in every search direction. Too low a setting will prevent the algorithm from finding all rings. Too high a setting will increase the time required for the ring search.

Output When SHOW_LIGAND and/or SHOW_PROTEIN is specified, a list of ligand and/or protein atoms is displayed, showing the atom type, residue, atomic properties, neighbouring atoms and bonds for each atom. In addition, a list of rings detected are output. If specified, bonds are also output. Atomic binding score is displayed only when scoring the initial configuration (topology).

It is safe to consider PMF-Score results as rankings where a more negative score means better binding. For details about converting PMF-Score to free energy of binding, see [29].

F.7 Useful tips

- To run FEP simulations of a ligand in water and bound to a protein using the same FEP file, use the **offset_name** keyword in the [**FEP**] section of the FEP file to instead of renumbering all the atoms!
- Make a separate library file for your new molecules and leave the amino acid library unchanged. Load both library files in **Qprep6**!

- It is possible to add parameters to the parameter file without restarting **Qprep6**. Just type maketop and the updated file will be used!
- For FEP simulations involving dummy atoms, the daring user might consider ignoring some **Qprep6** warnings about missing parameters all of those interactions are to be redefined in a FEP file. It is possible, but in general not advisable, to write a topology file with missing parameters and to use it in **Qdyn6**. Ignoring warnings means that no help has to be exoected in the case of resulting errors later in the simulation!
- Use build rules in your fragment library entries to control the positioning of hydrogens.
- Improve scoring accuracy by averaging over e.g. every 10:th frame of a short equilibration trajectory file!

G TUTORIALS

G.1 Binding affinity from LIE simulations

The example here is the binding of stearate to a muscular fatty-acid binding protein. We have used the Q version of the GROMOS87 forcefield for the simulations.

G.1.1 Editing the PDB file

The structure of the M-FABP complex with stearate, PDB-idcode 1HMT was downloaded. The PDB-file needs some editing before use, first you have to remove some of the crystal waters, if any. In the 1HMT-file, 17 waters were saved, having an important role in the binding with the ligand or in other interactions. To decide which waters to save, pick an atom in the ligand to be the centre of your system and choose how large simulation sphere you are going to use. Here, a sphere of 18.0 Å radius has been used. Then keep the waters inside the sphere that seem to bee involved in any interactions and that lie inside the protein. We have deleted the waters by hand in a molecular viewer program and then saving only the lines holding the coordinates. The part left from the original pdb-file is the coordinates of the protein, ligand and some waters. But it takes some more editing. All lines that are blank or say TER also have to be removed. There has to be a line saying GAP between the different molecules. All the cysteines that are connected through sulphur bridges should be renamed CSS.

Note also that hydrogen atoms need not to be present in the PDB file, they will be added by the **Qprep6** program.

G.1.2 Modeling ionic groups of the protein

Be aware that the default model of the charged amino acid residues (ASP, GLU, ARG, LYS) in the Q-GROMOS87 fragment library have the protonation state of the ionic form, but the net charge replaced by a dipole similar to that of neutral form. The corresponding

charged side chains are described by library entries AS-, GL-, AR+ and LY+, respectively. Below, we refer to the process of renaming *e.g.* an ASP residue to AS- as "turning on" the charge of that side chain.

Now it is time to decide which of the charged amino acids that should be "turned on". You can use the same rules here as in choosing which waters to keep. Amino acids near the ligand, creating a salt bridge or interacting in any other way in the function of the protein and which lie inside the 18.0 Å sphere should be charged. Residues closer than about 3-5 Å from the boundary should be neutral unless they form an ion pair with a more central group. In a case like this, when the ligand has an ionic group, is it important to make the protein net neutral. In this example, amino acids 17, 72, 76, 77, 78, 106 and 126 were charged.

G.1.3 Writing the library file

The next step is to write a library file for the ligand. This is easiest done by editing an old library file. You can also get a lot of help from looking at the amino acid library file (*e.g.* Qgrm87.lib). In the lib file, all the atoms of the ligand, the bonds and the charge groups are listed. For each atom, you need to specify name, type and charge. Make sure the charges in the complex file add up to the charge of the ligand file. All the different types of atoms are listed in the parameter file (Qgrm87j.prm). For the charges, one can sometimes compare with an amino acid from the amino acid library file. For stearate, the lib file is stearate.lib. Also make sure you name the atoms the same way as in the pdb file. Since the ligand will not be connected to any other fragment, the head and tail connections can be omitted.

You also need a pdb file for the ligand, copy the relevant lines from the PDB file of the complex to a separate file.

G.1.4 Qprep6

Now it is time to make the topology files. They contain all the information about the system and are used in the **Qdyn6** simulation. This is done in **Qprep6**, where the sulphur bridges also are created. You have to make two topology files, one for the ligand in protein simulation and one for the ligand without the protein simulation. In **Qprep6**, you start by reading the library files and the pdb files (*readlib* filename.lib, *readpdb* filename.pdb). In the ligand-protein topology, both the amino acid library and the ligand library files has to be read, and the sulphur bridges created. This is done by the command addbond. (*addbond* atomnumber atomnumber). For the atom numbers of the sulphurs, you can list the atoms in the cysteins (*listres* residuenumber). Note that you cannot use the numbers from the PDB file since atoms will be renumbered as hydrogens are added. After this you select boundary and solvate the system. Then it is time to create the topology, *maketop*. If there are any parameters missing, **Qprep6** will tell you here. To create new parameters, edit the parameter file in a way that you can see the changes made. You can always write comments in the file after a "!". After saving the modified parameter file, all you need to do is *maketop* again. To write the topology file, use the command *writetop.*

There are many other useful applications to **Qprep6**, among other things you can list the high energy bonds, angles, torsions and impropers by the commands *checkbond* energylevel, *checkangs* energylevel and so on. If you get a too high bond, angle or torsion energy, perhaps you have connected the sulphur bridges wrongly or forgotten a GAP between two molecules. If an improper has a very high energy, it might have the wrong sign (e.q. 180 instead of -180 degrees), use the command changeimp to redefine them automatically (*changeimp* 2 energylevel). After using changeimp, you need to write the topology again. You may also want to make a new PDB file, use *writepdb,* with atoms renumbered and hydrogens added. By typing *help*, **Qprep6** lists all the commands with a small explanation.

G.1.5 FEP files

In this example the only thing specified in the FEP files are the Q-atoms, that is, the ligand. In the simulation without the protein, this is simply all the atoms, but in the protein-ligand simulation, you have to find the atom numbers of the ligand in the new pdb file. There are a lot of other things that can be specified in the FEP file, but none of those functions are used in this example.

G.1.6 Creating input files

The input file controls the simulation in **Qdyn6**. It contains information on how many steps, how long steps, what temperature, which topology to use and a lot of other things. In this example, the data collection phase was split into five identical, consecutive steps to make it easier to restart after an interruption. This gives five input files to each of the two simulations and another 6 for the equilibration of the ligand-protein complex. An input file is easiest created by editing an old input file. The things that need to be specified to a specific simulation are the centre of the simulation sphere and water sphere, the topology- and FEP files, and restraints. The coordinates of the water- and simulation sphere should be the same, coming from the atom in the ligand that you picked earlier.

In this example, the equilibration warms up the system, starting with 0 degrees gradually raising the temperature to 300 degrees. All the heavy atoms, including those of the ligand, are restrained during this equilibration. When the system is equilibrated, 5x50 000 simulation steps of 1 fs are taken for both of the simulations. All the files except the initial one are restarted from the coordinates and velocities of the previous step. When a new temperature is given, you also need to give a random seed. When the temperature is the same as in the previous file, set the random seed to zero.

The coordinates of the water sphere must be specified in the first input file. In this example, a co-ordinate file with randomly oriented water molecules on a grid, was used.

G.1.7 Qdyn6

To start the simulation, write: **Qdyn6** filename.inp > filename.log, to use a specific input file and write the output to a log file. When many input files are used it is much easier to write all the commands to a command file and then run that.

G.1.8 Evaluating the simulation

By using the script lsextr (*lsextr* md0*.log > filename.txt), the van der Waals and electrostatic energies are respectively extracted from the log files. It is a good idea to plot these energies, *e.g.* by using gnuplot, to see if there are any large changes in the energies throughout the simulation, see figure 4.

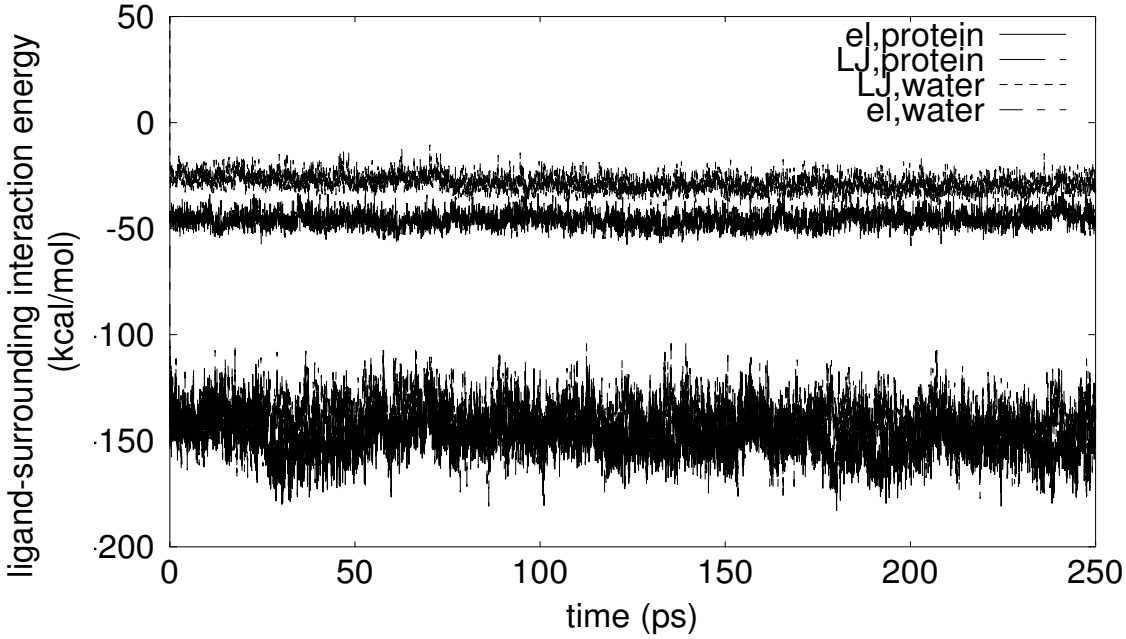

Figure 4: Energies

Viewing the structures after the different parts of the simulation is a very important part of the evaluation of the simulations. To make pdb files of the restart files use **Qprep6**. Read the topology file, then read the restart file (*readx* md0#.re) and write the new pdb file.

To get averages of the different energies, a program called tstart (*tstart* q filename.txt # # #), was used. Tstart calculates the overall average and also an average where you can split the energies in two, each giving the same average, skipping the first values. The numbers in the command are different choices you can make where the first is either 1 or 2, van der Waal or electrostatic energies. With the second you can choose where to start, zero meaning the beginning, the third how many rows to read, zero meaning all.

The next thing to do is to calculate the electrostatic free energy contribution from the ligand's interaction with ionic groups of the protein that were neglected (not "turned on") during the simulation.

Then you can calculate the binding free energy, using the LIE formula:

$$\Delta G_{bind} = \alpha \left(\langle V_{l-s}^{LJ} \rangle_{bound} - \langle V_{l-s}^{LJ} \rangle_{free} \right) + \beta \left(\left\langle V_{l-s}^{el} \right\rangle_{bound} - \left\langle V_{l-s}^{el} \right\rangle_{free} \right) + \Delta G_{onoff} \quad (9)$$

In this example, we use $\beta = 0.5$ (no deviations from electrostatic linear response for a charged ligand) and $\alpha = 0.181$ (from previous calibration using GROMOS87). This gives a ΔG_{bind} = -8.0 kcal/mol, which is close to the experimentally determined value of ΔG_{bind} = -7.9 kcal/mol.

The β-value varies with the number of OH - groups on the ligand, when using GROMOS87 for ligands with no ionic groups, β should be selected from a set of values accordingly to the composition of the ligand (number of OH-groups). [8]

H REFERENCE GUIDE

H.1 Program modules

Q is build from Fortran90 modules, which are combined in different sets in the Q programs, as shown in figure 5. This makes it easier to maintain the software. It also makes it rather straight-forward for users with experience in programming to create their own special-purpose programs by re-using *e.g.* the trajectory and topology modules.

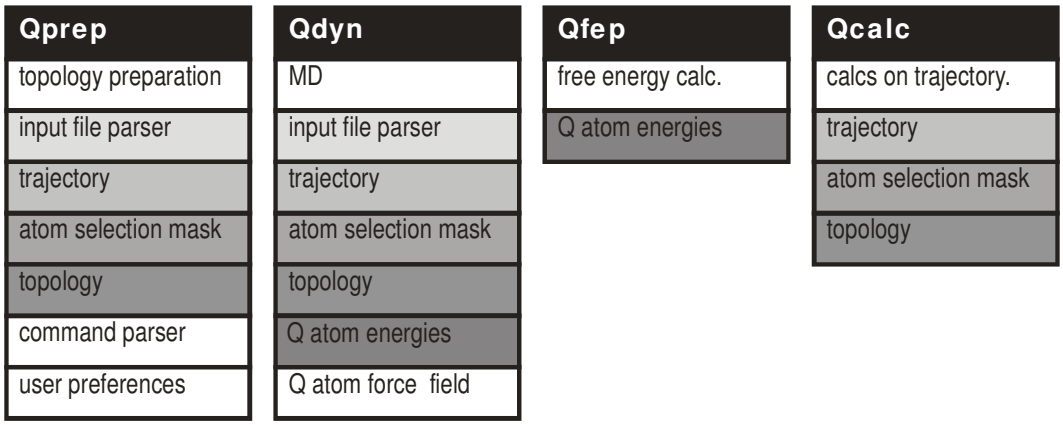

Figure 5: The modules that build up Q.

The figure does not show the dependence of some modules on others.

H.2 Force field reference information

Q6 is not associated with any particular force field, that is, it's force-field agnostic. The force fields are defined in parameter files, separate from the program and the choice of force field is thus simply a matter of which parameter file to use. Any force field could be used with the program, as long as it shares the common functional form of eq. 10.

$$V_{pot} = \sum_{bonds} \frac{1}{2} k_b \cdot (r - r_0)^2 + \sum_{angles} \frac{1}{2} k_\theta \, (\theta - \theta_0)^2 + \sum_{dihedrals} K_\varphi \cdot (1 + \cos(n \cdot \varphi - \delta)) + \sum_{\substack{improper \\ dihedrals}} \frac{1}{2} k_\xi \, (\xi - \xi_0)^2 + \sum_{\substack{atom \\ pairs\, i,j}} \frac{1}{4 \cdot \pi \cdot \varepsilon_0} \cdot q_i \cdot q_j \cdot r_{ij}^{-1} + A_{ij} \cdot r_{ij}^{-12} - B_{ij} \cdot r_{ij}^{-6} \quad (10)$$

where V_{pot} is the total potential energy, k_b is a bond stretching force constant, r is the distance between two bonded atoms, r_0 is the reference bond length, k_θ is an angle bending force constant, θ the angle between two bonds, θ_0 is the reference angle, k_φ is a force constant for rotation around a dihedral angle, n is the multiplicity (number of minima per full turn) of the dihedral angle φ, δ is the phase shift (δ/n gives the location of first maximum), k_ξ is an out-of-plane bending force constant for the improper dihedral angle ξ with reference angle ξ_0, q_i and q_j are the partial charges of atoms i and j separated by the distance r_{ij}. A_{ij} and B_{ij} are the geometric Lennard-Jones parameters for the interaction between atoms i and j. The Lennard-Jones parameters are defined per atom type as A_i and B_i and are combined using either of the two standard rules to determine the effective interaction parameters. The geometric rule is simply: $A_{ij} = A_i \cdot A_j$ and $B_{ij} = B_i \cdot B_j$ where $A_i = A_{ii}^{1/2}$ and $B_i = B_{ii}^{1/2}$. Some force fields use the form:

$$\varepsilon_{ij} \cdot \left(\left(\frac{R_{ij}^*}{r_{ij}} \right)^{12} - 2 \cdot \left(\frac{R_{ij}^*}{r_{ij}} \right)^{6} \right)$$

for the 6-12 Lennard-Jones potential. In this case the atom type-parameters ε_i, ε_j, R_i^* and R_j^* are combined using the rules: $\varepsilon_{ij} = (\varepsilon_i \cdot \varepsilon_j)^{1/2}$and $R_{ij}^* = R_i^* + R_j^*$. Several Fourier components of the dihedral terms, with different K_φ, n and δ, can be added for the same dihedral angle to allow a more accurate modelling of the barriers for rotation. An alternative form of the improper dihedral potential using trigonometric functions just as for normal dihedrals is also implemented.

The molecular fragments, *e.g.* amino acid residues, defined in the force fields are divided into charge groups which are groups of atoms whose partial charges add up to an integer. Cut-off of non-bonded interactions is then done based on these groups, *i.e.* either all pairwise interactions between the two groups are evaluated, or none. The average size of the charge groups varies between force fields, from a few atoms to entire residues. Some force fields designate a "switching centre" in each charge group and performs cut-off only based on the distance between the switching centres, while others include all interactions between two groups if any pair of atoms is within the cut-off radius.

Some properties of the force fields available for Q6 are given in table 8. Please note that these are our translations of the force field and we cannot guarantee 100% identity with the original.

Table 8: Force fields available in Q6.

Force field[1]	**CH$_n$ groups**[2]	**LJ type**[3]	**Impropers**[4]	**Charge groups**[5]	**Cut-off**[6]	**Atom types**[7]	**Ref**
Amber95	all-atom	ε_{ij},R$_{ij}$	periodic	residues	any	48	[30]
Amber14	all-atom	ε_{ij},R$_{ij}$	periodic	residues	any	65	[31]
Amber/OPLS	extended	A$_{ij}$,B$_{ij}$	periodic	$\leq$ 11 atoms	switching	39	[32]
CHARMM v.22	all-atom	ε_{ij},R$^*_{ij}$	harmonic	$\leq$ 13 atoms	switching	186	[33]
GROMOS87	extended	A$_{ij}$,B$_{ij}$	harmonic	$\leq$ 10 atoms	switching	28	[34]
GROMOS96	extended	A$_{ij}$,B$_{ij}$	harmonic	$\leq$ 10 atoms	switching	28	[35]
OPLS-AA	all-atom	A$_{ij}$,B$_{ij}$	periodic	$\leq$ 15 atoms	switching	35	[36]
OPLS-AA/2015	all-atom	A$_{ij}$,B$_{ij}$	periodic	$\leq$ 15 atoms	switching	84	[37]

The individual files for the fragment and parameter libraries are found in the folder "ff" of the source code distribution. The file names of the different fragment libraries and parameter files are given in Table 9.

Table 9: Force field files.

Force field	**Folder name**	**Library file**	**FF parameter file**
Amber95	amber95	qamber95.lib	qamber95.prm
Amber14	amber14sb	qamber14.lib	qamber14.prm
Amber/OPLS	oplsamber	qopls.lib	qopls.prm
CHARMM v.22	charmm22	qcharmm22.lib	qcharmm22.prm
GROMOS87	gromos87	qgrm87.lib	qgrm87.prm
GROMOS96	gromos96	qgrm96.lib	qgrm96.prm
OPLSAA	oplsaa	qoplsaa.lib	qoplsaa.prm
OPLSAA-2015	oplsaam2105	qoplsaa.lib	qoplsaa.prm

H.2.1 Solvent selection

Different organic solvents can be used for calculations with Q6, as long as a fragment library entry and associated parameters are available. For the format of the solvent file, please consult page 60. A set of organic solvents that have been tested for use in Q6 with the OPLS-AA force field are added in the respective folder and are listed below in table

[1] Our implementation of the named force field.

[2] Hydrogen atoms on aliphatic carbons may either be explicitly treated (all atom) or modelled as an extended atom.

[3] The Lennard-Jones potential can be written either as $\frac{A_{ij}}{r^{12}} - \frac{B_{ij}}{r^6}$ or $\varepsilon_{ij} \cdot \left(\left(\frac{R^*_{ij}}{r_{ij}} \right)^{12} - 2 \cdot \left(\frac{R^*_{ij}}{r_{ij}} \right)^6 \right)$, using the geometric or arithmetic rules, respectively, to combine parameters for pairs of atom types. Treatment of 1-4 interactions (LJ and electrostatic) is specific for each force field.

[4] Improper dihedrals can be modelled either with harmonic potentials or with a periodic function like ordinary dihedrals.

[5] Typical number of atoms in a charge group.

[6] Cut-offs are always applied to whole charge groups and are based either on the distance between designated switching atoms or on the smallest distance between any pair of atoms in two charge groups.

[7] The number of different atom types defined in the Q implementation of the force field.

10.

Table 10: Solvents for the OPLS-AA force field.

Solvent name	**File name**	**Density [1/Å^3]**
Chloroform	CLF.lib	0.00752
Dichloromethane	DCM.lib	0.00941
Methanol	MTH.lib	0.01489
Ethanol	ETH.lib	0.01031

H.3 Topology preparation reference

H.3.1 Coordinate files for input into Qprep6

Atomic coordinates are entered into **Qprep6** as PDB files. The PBD file must conform to some rules to be accepted by **Qprep6** (if not specified use the PDB standard):

- **Qprep6** will only accept and read ATOM and HETATM records. All other record types are ignored and will generate warnings.
- Residue numbers must be numeric, *i.e.* alphanumeric residue identifiers like 60B are not allowed. Use the renumber script to renumber residues. The numbering does not have to start at 1, but **Qprep6** will renumber residues starting at 1.
- Molecules must be separated with a gap marker line. This line should contain only the word GAP in capital letters, optionally preceded by blanks or tabs. There should not be a gap marker at the end of the file. Qprep6 will try to detect gaps between molecules by itself, but the user should make sure that the molecules are properly separated in the final topology.
- Atom numbering is not significant and **Qprep6** will renumber atoms starting from 1. Note that the numbering will change due to the insertion of hydrogen atoms.
- Only upper case letters may be used in PDB files.
- Residue names must match fragment library entry names. The maximal length of residue names is 3 alphanumeric characters (position 13 to 16). Most other characters like +, -, _ are also permitted.
- Atom names must match atom names within the relevant fragment library entry.
- Temperature factors and occupancy numbers are ignored by **Qprep6** and are optional.

H.3.2 Atom masks

An atom mask defines a subset of atoms and is used:

- in **Qprep6** to select which atoms are included when writing structure files (PDB and mol2)

- in **Qdyn6** to select which atoms to include in the trajectory and
- in **Qcalc6** to select atoms for superposition and coordinate deviation calculations.

The atom mask is constructed by selecting atoms within a sequence that match a set of properties. The properties that can be selected are:

- solute: all atoms except solvent atoms
- heavy: all atoms except hydrogens (atoms with mass ≥ 4 mass units to be precise)
- excluded: atoms outside the simulation sphere including excluded solvent molecules
- restrained: atoms in the restrained boundary zone and atoms outside the sphere

Each property can be negated by putting 'not' before it. Multiple properties on the same line are combined with a logical and. At the end of a line an atom sequence is defined by its first and last numbers. A sequence of residues can be selected using the word residue before the numbers. The number of the last atom or residue may be omitted to select a single entity. Multiple such lines may be used to construct the mask in an additive manner the atom sets specified by each line are combined with a logical or. If no mask is given, all atoms are selected by default.

H.3.3 Qprep6 commands

Command	Alias	Arguments (optional)	Description
addbond	ab		Add extra bonds (*e.g.* S-S).
boundary	bc	[boundary condition]	Set the boundary condition (sphere(1), box(2))
changeimp			Redefine (specified) improper torsions.
checkangs	ca	[energy threshold]	Check angle energies.
checkbonds	cb	[energy threshold]	Check bond energies.
checkimps	ci	[energy threshold]	Check improper torsion energies.
checktors	ct	[energy threshold]	Check torsion energies.
clearbond			Clear extra bonds.
clearlib	cl		Clear all loaded molecular libraries.
cleartop			Clear the current topology.
help	h, ?		Show command list.
listres	lr	[residue number]	List atoms in residue.
listseq	ls		List the residue sequence.
makeshell	ms		Fix the mask of the atoms in the restrained shell.
maketop	mt		Generate the topology.
mask	ma	[mask_def] or [none]	Add to or clear atom mask.
preferences	prefs		List atoms in residue.
quit	q		Quit the program.

Command	Alias	Arguments (optional)	Description
readframe	rf	[trajectory file [frame no.]]	Load coordinates from Q trajectory file (unformatted)
readlib	rl	[library file]	Read library file. This command may be repeated to load multiple libraries.
readnext	rn		Load next frame of coordinates from current trajectory file
readpdb	rp	[pdb file]	Read pdb file.
readprm	rprm	[parameter file]	Read force field parameter file.
readtop	rt	[topology file]	Read topology file.
readx	rx	[restart file]	Load coordinates from Q restart file (unformatted).
set	s		Set preferences.
solvate (box)	so	[grid] [solvent name] [boxcentre] [boxsize]	Add solvent molecules from a grid to a box.
		[file name] [boxcentre] [boxsize]	Add solvent from a file with a box of solvent molecules.
		[restart] [solvent name][filename][boxcentre][boxsize]	Add solvent molecules from a restart file containing the same solute with solvent.
solvate (sphere)	so	[centre][radius][grid] [solvent name]	Add solvent molecules from a grid to a sphere.
		[centre][radius][file][file name]	Add solvent from a file with a box or a sphere of solvent molecules.
		[centre][radius][restart] [file name][solvent name]	Add solvent molecules from a restart file containing the same solute with solvent.
trajectory	tr	[trajectory file]	Open trajectory file.
writemol2	wm	[mol. no. [mol2 file [hydrogen flag [water flag]]]]	Write SYBYL mol2 file for one or all molecules in topology using current coordinates. [mol. no.] is the number of the molecule in the topology (separated by a GAP). Enter 0 for all molecules.[mol2 file] is the name of the file to be created. An existing file will be overwritten. Enter auto to generate the name automatically using the template coord_file.frame_no.molecule_no.mol2 [hydrogen flag] specifies whether hydrogens should be written. Enter y for yes or n for no.[water flag] specifies whether water should be written. Enter y for yes or n for no.

COMMAND	ALIAS	ARGUMENTS (OPTIONAL)	DESCRIPTION
writepdb	wp	[pdb file [gap flag]]	Write PDB file containing the atoms specified by the current mask (default all atoms).[gap flag] specifies whether GAP lines between molecules should be written. Enter y for yes or n for no.
writetop	wt	[topology file ["title"]]	Write topology file.
xlink	xl		Search for possible cross-links such as disulphides and make bonds. For each non-bonded heavy atom pair separated by less than 2.1 Å, the program will ask whether to add a bond or not.

H.3.4 Qprep6 preferences

Use together with set, *e.g.*: set solvent_pack 2.6.

NAME	MEANING	DEFAULT VALUE
solvent_pack	minimum distance between solute and solvent heavy atoms when adding solvent	2.4 Å
solute_density	number density of solute	0.05794 $Å^{-3}$
max_xlink	upper limit of bond length when searching for possible cross-link bonds	2.1 Å
random_seed_solute	integer seed value for the random number sequence used to generate solute hydrogen atom coordinates	179857
random_seed_solvent	integer seed value for the random number sequence used to generate solvent hydrogen atom coordinates	758971

H.3.5 Fragment library file format

The fragment library file is a text file containing definitions of all the molecular building blocks, *i.e.* amino acid residues, ligands etc., and is used by **Qprep6** to generate a molecular topology from a PDB file. The format of the library file follows the same standard as the parameter file. Each fragment/residue entry starts with an entry name enclosed in curly braces, *e.g.* {ALA}, maximum 3 positions. The lists of atoms, bonds, etc. appear as sections within the entry.

The optional [**info**] section contains the keyword SYBYLtype which identifies the SYBYL substructure type (residue or group) for the entry and is used only for writing mol2 files with **Qprep6**. The [**atoms**] section defines the sequential number, name, type and charge of the

atoms in the entry. The atom name must match the name used in PDB files to be read, but the order of atoms is not important. In the following sections atoms are identified by their names. The [**bonds**] section lists all bonds within the entry. The optional [**connections**] contains the keywords head and tail which identify the atoms involved in inter-residue bonds (head is bonded to the tail of the preceding residue and tail is bonded to the head of the next residue). Charge groups are defined, one per line, in the [**charge_groups**] section as lists of atoms starting with the atom designated as switching atom. The tables below describes the format of these sections, and an example file is included as fig. 6.

[**info**]: General information about the fragment.

KEYWORD	VALUE	COMMENT
SYBYLtype	SYBYL substructure type: RESIDUE or GROUP	Optional, default none. Only used for writing Sybyl mol2 files.
PDBtype	PDB substructure type:ATOM or HETATM	Optional, default ATOM. CONECT records in PDB file will be generated for HETATM groups.
Solvent	on or off	If on, this entry is recognised as a solvent.
Density	number density ($Å^{-3}$)	Only used for solvents, to solvate using a grid and to calculate the effective solvent radius.

[**atoms**]: Define atom names, types and partial charges.

COL.	DESCRIPTION
1	sequence number (from 1 to number of atoms)
2	atom name (4 character string)
3	atom type
4	partial charge (e)

[**bonds**]: Define bonds within the fragment.

COL.	DESCRIPTION
1	name of atom 1
2	name of atom 2

[**connections**]: Define sites of inter-residue bonds.

KEYWORD	VALUE	COMMENT
head	name of head atom to which the preceding residue in the sequence is bonded.	Optional, default none. Should not be defined for entries which are entire molecules.
tail	name of tail atom to which the next residue is bonded.	Optional, default none. Should not be defined for entire molecules.

[**build_rules**]: Define rules for generating hydrogen atom coordinates.

COL.	DESCRIPTION
1	type of rule. The only type available is 'torsion'.
2	name of the hydrogen atom to be generated
3	names of atom 2 in the torsion (the atom to which the hydrogen is bonded)
4	name of atom 3 of the torsion
5	name of atom 4 of the torsion
6	target value of the torsion angle formed by the atoms named in columns 2-3-4-5.

[**impropers**]: Define improper torsion angles. This is only used for force fields where impropers are explicitly defined rather than automatically generated (see parameter file).

COL.	DESCRIPTION
1	name of atom 1. Use a + or - before to refer to atoms in previous or next residue.
2	name of atom 2, the central atom to which 1, 3 and 4 are connected
3	name of atom 3 (use +/- as for atom 1)
4	name of atom 4 (use +/- as for atom 1)

[**charge_groups**]: Define charge groups.

COL.	DESCRIPTION
1	name of switching atom
2...	names of other atoms

```
{ALA}                ! Alanine
    [info]
        SYBYLtype  RESIDUE     !SYBYL substructure type
    [atoms]
         1 N      NH1         -0.470 !At. no., name, type, charge
         2 HN     H            0.310 !At. no., name, type, charge
         3 CA     CT1          0.070 !At. no., name, type, charge
         4 HA     HB           0.090 !At. no., name, type, charge
         5 CB     CT3         -0.270 !At. no., name, type, charge
         6 HB1    HA           0.090 !At. no., name, type, charge
         7 HB2    HA           0.090 !At. no., name, type, charge
         8 HB3    HA           0.090 !At. no., name, type, charge
         9 C      C            0.510 !At. no., name, type, charge
        10 O      O           -0.510 !At. no., name, type, charge
    [bonds]
       CB   CA
       N    HN
       N    CA
       O    C
       C    CA
       CA   HA
       CB   HB1
       CB   HB2
       CB   HB3
    [connections]   !how to bond to previous and next
        head     N
        tail     C
    [impropers]
         N     -C     CA     HN
         C     CA     +N      O
    [charge_groups] !charge groups, with switch atom first
        N    HN   CA   HA
        CB   HB1  HB2  HB3
        C    O
```

Figure 6: The CHARMM library entry for alanine

H.3.6 Force field parameter file format

The force field parameter file is a text file, based on the same standard as the FEP file described in section H.7.3 on page 74. It is divided into sections which can appear in any order and which start with a section title enclosed in square brackets. The data in the file is the constants, which are defined for each multiplet of atom types, in

$$V_{pot} = \sum_{bonds} \frac{1}{2} k_b \cdot (r - r_0)^2 + \sum_{angles} \frac{1}{2} k_\theta \left(\theta - \theta_0\right)^2 + \sum_{dihedrals} K_\varphi \cdot (1 + \cos\left(n \cdot \varphi - \delta\right))$$
$$+ \sum_{\substack{improper \\ dihedrals}} \frac{1}{2} k_\xi \left(\xi - \xi_0\right)^2 + \sum_{\substack{atom \\ pairs\ i,j}} \frac{1}{4 \cdot \pi \cdot \varepsilon_0} \cdot q_i \cdot q_j \cdot r_{ij}^{-1} + A_{ij} \cdot r_{ij}^{-12} - B_{ij} \cdot r_{ij}^{-6}$$

The [**atom_types**] section defines a name, and lists Lennard-Jones parameters and mass for each atom type. There are three sets of LJ parameters: set 1 for normal non-bonded interactions, set 2 for pairs of polar atom types (listed in a the section LJ_type2_pairs) and type 3 for pairs of atoms in 1–4 position. The B parameter is the same for sets 2 and 3. Atom type names are alphanumeric. The length of the name is limited to 8 characters.

The section [**atom_aliases**] is used facilitate the transition from numeric atom type names used in earlier versions. In this section alias names may be assigned to atom types defined in the atom_types section, *e.g.* to be able to use the old numeric name as an alias for the new descriptive atom type name in library files.

The [**LJ_type2_pairs**] section lists pairs of polar atom types that should interact with LJ parameters from set 2.

The [**bonds**] section lists force constant and equilibrium distance for bonds between two atoms. It contains one line for each (applicable) unique pair of atom types. The pair 1–2 is equivalent with the pair 2–1, so only one of these should be included. The pairs may appear in any order, but for reasons of readability it is convenient to sort the lines by both atom types and always have the lower atom type number first on each line.

The [**angles**] section lists force constant and equilibrium angle for 3-atom angles. contains one line for each (applicable) unique triplet of atom types. The pair 1–2–3 is equivalent with the pair 3–2–1, so only one of these should be included. The pairs may appear in any order, but for reasons of readability it is convenient to sort the lines by the middle atom type first, then on the left and finally by the right atom type. It is also preferred to have the left atom type less than or equal to the right one (*i.e.* 1–2–3 rather than 3–2–1).

The [**torsions**] section lists parameters for torsion angles. Torsions can be defined by 4 atom types but in many cases only the two middle atoms are significant. The latter case is indicated by "0" (zero) or "?" in column one and four. A torsion defined with four atom types overrides two-atom definitions. The force constant in the cosine-shaped function for the torsion potential is equal to half the barrier height. The periodicity is the number of maxima passed in a full 360 ° rotation. The phase shift divided by the periodicity is the angle where the first maximum should be. The number of paths is the number of ways that a two-atom torsion can be defined, *i.e.* the product of the number of atoms bonded to

the two middle atoms. It is used to distribute the force over all the atoms involved. The preferred order of the lines is analogous with the bonds section: sort by the two middle atom types. Multiple torsion potential terms may be defined for the same set of atom types, to enable more complex torsion potentials. All terms are then added together.

The [**impropers**] section lists force constant and equilibrium angle for improper torsion angles, which are modelled by a harmonic potential. The impropers may be defined by two atom types, but in many cases the second type is not used and set to "0" or "?" as in the torsions section.

The [**options**] section contains three keywords. Vdw_rule selects the rule for combining LJ parameters from two atom types and takes the values "geometric" or "arithmetic". Scale_14 is the scaling factor for electrostatic interactions between atoms in 1–4 positions. Switch_atoms selects the cut-off logic for non-bonded interaction: On = use designated switch (central) atoms of charge groups. Off = include the charge groups if any pair of atoms is within the cut-off distance.

Table 9 lists the data and units for each column in the different sections, and an example is included as a file example on page 61.

Table 9: Parameter file format.

[**atom_types**]: Define atom types

col.	**description**
1	atom type name, max 8 characters
2	Lennard-Jones A parameter for type 1 pairs ($\text{kcal}^{1/2} \cdot \text{mol}^{-1/2} \cdot \text{Å}^6$) for geometric combination or R* ($\text{kcal}\cdot\text{mol}^{-1}\cdot\text{Å}^{12}$) for arithmetic combination)
3	LJ A parameter type 2 ($\text{kcal}^{1/2} \cdot \text{mol}^{-1/2} \cdot \text{Å}^6$) or R* ($\text{kcal}\cdot\text{mol}^{-1}\cdot\text{Å}^{12}$)
4	LJ B parameter type 1 ($\text{kcal}^{1/2} \cdot \text{mol}^{-1/2} \cdot \text{Å}^3$) or ε ($\text{kcal}\cdot\text{mol}^{-1}\cdot\text{Å}^6$)
5	LJ A parameter type 3 ($\text{kcal}^{1/2} \cdot \text{mol}^{-1/2} \cdot \text{Å}^6$) or R* ($\text{kcal}\cdot\text{mol}^{-1}\cdot\text{Å}^{12}$)
6	LJ B parameter type 2 and 3 ($\text{kcal}^{1/2} \cdot \text{mol}^{-1/2}\cdot\text{Å}^3$) or ε($\text{kcal}\cdot\text{mol}^{-1}\cdot\text{Å}^6$)
7	mass of (extended) atom (u)
8	SYBYL atom type (5 character string), optional

[**atom_aliases**]: Define alias names for atom types col. description

col.	**description**
1	alias name, max 8 characters
2	atom type name defined in atom_types section

[**LJ_type2_pairs**]: list pairs of atom types that use the alternate set of LJ parameters col. description

col.	**description**
1	atom type 1
2	atom type 2

[**bonds**]: Define harmonic bond parameters for pairs of atom types col. description

col.	**description**
1	atom type 1
2	atom type 2
3	force constant ($\text{kcal} \cdot \text{mol}^{-1} \cdot \text{Å}^{-2}$)
4	equilibrium length (Å)
5	SYBYL bond type (2 character string), optional

[**angles**]: Define harmonic angle parameters for triplets of atom types col. description.

col.	description
1	atom type 1
2	atom type 2
3	atom type 3
4	force constant (kcal $\cdot$ mol^{-1} $\cdot$ rad^{-2})
5	equilibrium angle (°)

[**torsions**]: Define torsion angle parameters for quadruplets or pairs of atom types col. description

col.	description
1	atom type 1 or 0 or ? to match any atom type
2	atom type 2
3	atom type 3
4	atom type 4 or 0 or ? to match any atom type
5	force constant = 1/2 $\cdot$ barrier height (kcal $\cdot$ mol^{-1})
6	periodicity (number of maxima per turn)Add a minus sign before to indicate that more components follow on subsequent lines, i. e. for a torsion potential with multiple components all but the last component should be entered with negative periodicity.
7	phase shift (δ/n define the location of first maximum) (°)
8	number of paths

[**impropers**]: Define harmonic improper torsion parameters for pairs of atom types or for single atom types col. description

col.	description
1	atom type 1
2	atom type 2 or 0 or ? to match any atom type
3	force constant (kcal $\cdot$ mol^{-1} $\cdot$ rad^{-2})
4	equilibrium angle (°)

H.3.7 Solvent file format

Qprep6 can solvate a molecular systems by filling empty space in the simulation sphere or box by solvent molecules taken from a solvent file. This file is a special PDB-like file containing a box or a sphere of solvent molecules. The residue name in the water file is used to designate the library entry to use for generating bonds etc.

When using a box for solvation, the box can be replicated in all direction so that a small box can be used to solvate a big simulation sphere or box. Spheres cannot be replicated and must be larger than the intended simulation system. The box or sphere will be translated to the solvent generation centre, so the origin used in the file is arbitrary. A sphere can not be used to solvate a system intended for simulation with periodic boundaries. The file format is described in Table 10.

For larger solvent molecules, the remaining atoms again have to follow in the same format as above.

```
* Q-FF params: GROMOS87 parameters
*-------------------------------------------------------
[options] force-field options
vdw_rule       geometric   ! vdW combination rule
scale_14       1.0 ! electrostatic 1-4 scaling factor

[atom_types] atom type definitions
*tac--- ---Avdw1---Avdw2---Bvdw1----Avdw3---Bvdw2&3---mass----SYBYL-name-old-
comment
H.np         0.00      0.00     0.00      0.00     0.00      3.000  H   ! HC   -
non-polar H
C.3        898.00      0.00    23.65    898.00    23.65     12.001  C.3 ! Csp3 -
bare sp3 C

[atom_aliases]
10     H.np
20     C.3

[LJ_type2_pairs] type-2 van der Waals interaction atom type pairs
*-------------------------------------------------------------------------
N.pep   O.SPC
N.pep   Cl-

[bonds] bond types definitions
*iaci iacj   force.c.  dist.     SYBYL
*-------------------------------------------------------------------------
H.np    C.3         700.000   1.090      1
H.np    C.2         700.000   1.090      1

[angles] angle type definitions
* iaci iacj iack  forceK angle0
*-------------------------------------------------------------------------
H.np    C.3     H.np       90.00  106.60
H.np    C.3     C.ar6H     90.00  109.50

[torsions] torsion type definitions
*iaci iacj iack iacl  forceK  #minima phase   #paths
*-------------------------------------------------------------------------
?       C.3     C.3H    ?          1.400   3.000    0.000    6
O.3s    C.3sH   C.3sH   O.3s       1.400  -3.000    0.000    1

[impropers] improper torsion type definitions
*iaci iacj forceK   imp0
*-------------------------------------------------------------------------
C.2     ?         40.000  180.000
```

Figure 7: Example of a parameter file.

H.4 Boundary conditions

H.4.1 Solute boundary restraints

Solute atoms outside the simulation sphere are excluded from non-bonded interactions and are tightly restrained to their initial coordinates by a harmonic potential with a force constant of 200 kcal·mol^{-1}·Å^{-2}. All bonded interactions are evaluated as for atoms inside the sphere. Solute atoms in the outermost shell of the simulation sphere are also restrained to their initial coordinates with a harmonic potential to avoid distortion of bonds across the sphere boundary. The radius of this shell and the force constant is given in the **Qdyn6** input file (section [**sphere**], keywords shell_radius and shell_force). The restrained shell radius is by default equal to the outer, *i.e.* no atoms will be restrained unless the restrained shell radius is redefined. The force constant has a default value of 10 kcal·mol^{-1}·Å^{-2}.

The grouping of atoms in the inside shell or excluded regions is done before the simulation starts using the initial coordinates from the topology and is not updated during simulation. The initial coordinates for grouping the shell-atoms can also be taken from or from a separate restraint coordinate file (section [**files**], keyword restraint).

Table 10: Solvent file format

line	content
1	For a box: side of the box (Å).For a sphere: radius of the sphere followed by the word **sphere**.
3 · n-1	Coordinates of the first atom of solvent molecule n in PDB format.
3 · n	Coordinates of the second atom of solvent molecule n in PDB format.
3 · n+1	Coordinates of the third atom of solvent molecule n in PDB format.

H.4.2 Solvent boundary restraints

Solvent molecules near the sphere boundary must be restrained to avoid "evaporation" and to keep the density correct and uniform in the whole sphere. A central value in the radial restraining of water is the effective solvent radius r_w, which is only almost equal to the simulation sphere radius. It is the solution to the equation $V_{sphere}(r_w) = \frac{4\pi}{3} \cdot r_w^3 = N_p(r_w) \cdot v_p + N_w \cdot v_w$ where $N_p(r_w)$ is the number of heavy solute atoms within a radius r_w from the water centre, v_p is the average volume per heavy atom in proteins (17.3 Å^3), N_w is the (fixed) number of solvent molecules and v_w is the volume of a solvent molecule (29.9 Å^3 for water).

The radial restraining potential has a half-harmonic term that pushes solvent molecules back into the sphere and a Morse-like term that pulls molecules from inside out towards the boundary [38]:

$$V_{solvent}(r) = \begin{cases} \frac{1}{2} \cdot K \cdot (r - r_0)^2 - D_e & \text{if } r > r_0 \\ D_e \cdot \left(\left(e^{(\alpha \cdot (r - r_0))}\right)^2 - 2 \cdot e^{(\alpha \cdot (r - r_0))} \right) & \text{otherwise} \end{cases} \tag{11}$$

where r is the distance from the water centre, K is the force constant of the half-harmonic potential, D_e is the depth ("dissociation energy") of the Morse potential, α the exponential coefficient of the Morse term. r_0 is the effective solvent radius minus the average deviation distance from the minimum of the half-harmonic potential at the current temperature T: $r_0 = r_w - \sqrt{\frac{k_b \cdot T}{K}}$ where r_w is the target solvent radius. The appropriate values of D_e and α depend on r_w and are calculated using empirical functions calibrated to give correct values for water spheres from 12 to 30 Å:

$$\begin{aligned} D_e(r_w) &= 0.26 \frac{kcal}{mol} \cdot e^{\left(-0.19 \frac{1}{\text{Å}} \cdot (r_w - 15\text{Å})\right)} + 0.74 \frac{kcal}{mol} \\ \alpha(r_w) &= 0.20 \frac{1}{\text{Å}} \Big/ \left(1 + e^{\left(0.4 \frac{1}{\text{Å}} \cdot (r_w - 25\text{Å})\right)}\right) + 0.30 \frac{1}{\text{Å}} \end{aligned} \tag{12}$$

K, D_e and α can be set in the input file to override the calculated values used by default (section [**solvent**], keywords radial_force and morse_depth, respectively).

Water molecules (in the topology) that are initially outside the simulation sphere are excluded from the simulation (with respect to non-bonded interactions and restraints).

Polar solvent molecules near boundary will not be randomly oriented like in bulk solvent and a restraining force is required to make the surface solvent molecules follow the probability distribution of angles between radial axis and dipole vector found in the bulk solvent.

When a net charge in the Q-atoms polarises the solvent, the distribution of solvent molecule dipole angles changes. This correction of the average polarisation given by Born's formula is taken into account unless disabled by setting charge_correction to off in the input file. The polarisation distribution restraints are applied in three thin shells to minimise the radial dependence of the polarisation which occurs in a single, thicker shell. The outermost shell is 0.5 Å thick, the second 1.0 Å, the third 1.5 Å, so the polarisation is restrained in the outermost 3 Å of the simulation sphere.

The restraining works by sorting the molecules of each shell according to the angle between dipole vector and radial axis and applying a force to each molecule i to adjust the angle towards the angle of molecule i in a sorted sequence of molecules that follow the target distribution. The potential can be written $V_{polarisation}\left(\Theta_i\right) = \frac{1}{2} \cdot K_{pol} \cdot \left(\Theta_i - \Theta_i^{target}\right)^2$ where K_{pol} is the force constant. K_{pol} can be set in the input file (keyword polarisation_force in section water), the default value is 20 kcal·mol^{-1}·rad^{-2}.

H.4.3 Periodic boundary conditions

In periodic boundary conditions (PBC) no restraints as described in the previous section is applied to the atoms and no atoms are excluded. This allows larger flexibility in the molecular system but increases runtime. This section contains some things worth to keep in mind when using Q with PBC.

The shape of the box may be cubic or rectangular. The cut-off must never be larger than one half of the shortest side of the box. This also accounts for the constant pressure algorithm. If the cut-off is too big, the program will stop.

Particles are moved across box boundaries in terms of molecules. When simulating a large protein in water solution the option rigid_box_centre should be set to off (false). This means that the box is in each time step centered around the solutes geometrical center. If there is no solute the box will be centred around the geometrical centre of the solvent.

If rigid_box_centre is on, the centre of the box will be the same as given in the topology throughout the simulation. Note that, if solute is present in a simulation like this, each solute molecule must be assigned just one charge group. This is done by changing the library file.

H.4.4 Constant pressure algorithm

The constant pressure algorithm is a combination of molecular dynamics and Monte Carlo volume sampling. A change in volume is chosen randomly $\Delta V = n_{rand} \cdot \Delta V_{max}$ where n_{rand} is a random number between -1 and 1 and ΔV_{max} is the maximum allowed volume displacement in one move. The new volume is defined as $V' = V + \Delta V$, prime indicating the new configuration. The coordinates are then changed, the system is contracted or expanded. The scaling factor for the side length of the box, $l_{x,y,z}$ is $\sqrt[3]{\frac{V'}{V}}$, thus $l_i' = l_i \cdot \sqrt[3]{\frac{V'}{V}}$.

The proportions of the box are maintained, meaning that a rectangular box stays rectangular. The coordinates, $r_{x,y,z}$ of each molecules centre of mass are scaled according to

$r_i' = (r_i - ci)\frac{l_i'}{l_i} + c_i$, where c_i is the coordinate of the centre of the box. This variable is included to handle the case when the box centre does not coincide with the origin of the coordinate system. The contraction of expansion is in terms for molecules, not atoms, which means that all intra molecular distances are kept fixed.

After a new configuration has been set, the potential is recalculated. Only the non-bonded interactions need to be taken in to account because the interior of molecules are not changed. The Metropolis sampling equation is $\Delta W = (U_{pot}' - U_{pot}) + P_0(V' - V)$, where P_0 is the target pressure. The new configuration is accepted with probability

$$P(\Delta V) = \begin{cases} e^{-\frac{\Delta W}{k_B T}} & \Delta W > 0 \\ 1 & \Delta W \leq 0 \end{cases} \tag{13}$$

If ΔW is zero or negative the move is always accepted. Otherwise a new random number, $n \in [0, 1]$, is generated and the configuration is accepted if $n \leq e^{-\frac{\Delta W}{k_B T}}$. The acceptance ratio is controlled with the variable max_volume_displacement, which corresponds to ΔV_{max}.

H.5 Units

The units used in Q6 are the basic units in table 11 and combinations thereof.

Table 11: Units in Q6

time	fs
temperature	K
length	Å
energy	kcal·mol^{-1}
charge	e

H.6 Molecular dynamics algorithms

The way the equation of motions are integrated can affect drastically the physics of the system. There are several ways of integrating a differential equation, being some methods more precise than others. Nevertheless, high precision is not necessarily the most important feature of an algorithm for molecular dynamics simulations. An equation of motion must be time-reversible in order to conserve the phase space volume if one is looking for physical conservation laws. Q6 provides two options of symplectic integrators: the Leapfrog and the velocity Verlet algorithms.

H.6.1 Leapfrog algorithm

The idea of the Leapfrog algorithm is to use simple first-order expansions of both position and velocity variables, $\mathbf{r}(t)$ and $\mathbf{v}(t)$, but calculating them at different times.

$$\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t+\Delta t/2)\Delta t \tag{14}$$

$$\mathbf{v}(t+\Delta t/2) = \mathbf{v}(t-\Delta t/2) + \frac{\mathbf{F}(t)}{m}\Delta t \tag{15}$$

The assynchronicity of the velocities with respect to the position is assigned at the moment that the velocities are randomly generated, being set at a half step back. Leapfrog algorithm is derived from another symplectic algorithm: the Verlet algorithm. Thus, it is also symplectic. This integrator is the fastest, although it has the drawback of calculating the positions and velocities at different times. This creates errors at energy measurements, although for small time steps these deviations are not significant.

H.6.2 Velocity Verlet algorithm

Unlike the Leapfrog algorithm, the velocity Verlet calculates the position and velocities of particles at the same time.

$$\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\mathbf{F}(t)}{2m}\Delta t^2 \tag{16}$$

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \frac{\mathbf{F}(t+\Delta t) + \mathbf{F}(t)}{2m}\Delta t \tag{17}$$

This is scheme is prone to less error, as it computes positions and velocities at the same time interval. However, the computational cost becomes slightly higher.

H.6.3 Constant temperature algorithms

Q6 comes with the option of choosing the thermostat which the user judges appropriate. The three thermostats present are Berendsen [39], Nosé-Hoover chains [40] and Langevin [41]. One can change the thermostat by inserting the keyword thermostat in the section [**MD**] in the input file, followed by "berendsen", "nose-hoover" or "langevin". Every thermostat has its own setup parameters, which will be described briefly below.

Berendsen thermostat: The idea of this thermostat is to rescale the velocities in order to control the kinetic energies in the system. In this algorithm, every particle is coupled to a heat bath, which will, in turn, increase or decrease the velocities of them. We set the thermostat temperature T_0, as well as the bath coupling time τ, which will give the interval of time where the rescaling of the temperature (and consequently velocities) will take place. After every bath coupling time interval, the temperature is measured and rescaled according to:

$$\frac{dT}{dt} = \frac{T_0 - T}{\tau} \tag{18}$$

The velocities are then rescaled in order to reproduce the new temperature.

Nosé-Hoover chain thermostat: The Nosé-Hoover thermostat also employs the idea of a rescaling factor. It inserts an extra particle that will interact with the whole system in a specific way in a different set of generalized coordinates from the original system. The idea is to solve this system with this extra particle as if we are performing a standard MD simulation, and the averages performed for the particles in your system will correspond to the ones from the canonical ensemble. The Hamiltonian of a N-particle system looks like this:

$$\mathcal{H}_{Nosé}(\mathbf{p},\mathbf{r},p_s,s)=\sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{2m_is^2}+V(\mathbf{r}^N)+\frac{p_s^2}{2Q}+gk_BT\ln(s), \tag{19}$$

where s and p_s refer to the new generalized coordinate and momentum for the thermostat particle, Q is the mass of this particle, and g is the total number of degrees of freedom of the system. The equations of motion will be:

$$\frac{d\mathbf{r}_i}{dt}=\frac{\mathbf{p}_i}{m_is^2} \tag{20}$$

$$\frac{d\mathbf{p}_i}{dt}=\mathbf{F}_i \tag{21}$$

$$\frac{ds}{dt}=\frac{p_s}{Q} \tag{22}$$

$$s\frac{dp_s}{dt}=\left(\sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{m_is^2}-gk_BT\right) \tag{23}$$

Now, if we call $\mathbf{p'}=\mathbf{p}/s$, and define a new Hamiltonian as

$$\mathcal{H}(\mathbf{p'},\mathbf{r})=\sum_{i=1}^{N}\frac{\mathbf{p'}_i^2}{2m_i}+V(\mathbf{r}^N), \tag{24}$$

it is possible to show that the average of any observable is (see [42] for more details):

$$\langle A(\mathbf{p}/s,\mathbf{r})\rangle_{NVE}=\langle A(\mathbf{p'},\mathbf{r})\rangle_{NVT} \tag{25}$$

Thus, by running standard MD simulations with this new particle interacting with the whole system will sample the same conformations of the system without this particle at the canonical ensemble. Nevertheless, it has been shown that a single particle Nosé-Hoover particle was not able to sample properly the canonical ensemble in some cases. The solution to the problem was to create additional particles that would interact with each other in a chain-like way [40]. If we apply the Nosé-Hoover thermostat in a chain way C times, the new equations of motion become:

$$\frac{d\mathbf{r}_i}{dt} = \frac{\mathbf{p}_i}{m_i} \tag{26}$$

$$\frac{d\mathbf{p}_i}{dt} = \mathbf{F}_i - \frac{p_{s_1}}{Q_1}\mathbf{p}_i \tag{27}$$

$$\frac{ds_j}{dt} = \frac{p_{s_j}}{Q_j} \tag{28}$$

$$s_1\frac{dp_{s_1}}{dt} = \left(\sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{m_i} - gk_BT\right) - \frac{p_{s_2}}{Q_2}p_{s_1} \tag{29}$$

$$s_j\frac{dp_{s_j}}{dt} = \left(\frac{\mathbf{p}_{s_{j-1}}^2}{m_{j-1}} - k_BT\right) - \frac{p_{s_{j+1}}}{Q_{j+1}}p_{s_j} \tag{30}$$

$$s_C\frac{dp_{s_C}}{dt} = \left(\frac{\mathbf{p}_{s_{C-1}}^2}{m_{C-1}} - k_BT\right) \tag{31}$$

In Q6, one can control the mass of the thermostat particles. The larger the mass, the larger the coupling with the thermostat, and less temperature oscilation will occur. The standard value is set to $\mathrm{k_b}T/\tau^2$.

Langevin thermostat: While both thermostats above have a deterministic behavior, the Langevin thermostat has a stochastic character. The interaction between the heat bath and the system is characterized by two forces: a drag force, which will act against the movement of the particles, in order to remove excess of kinectic energy; a random force, which will give random impulses to every particle. The equation of motion becomes

$$\frac{d\mathbf{p}_i}{dt} = \mathbf{F}_i - \gamma\mathbf{p}_i + \sqrt{\frac{2\gamma k_BT}{m_i}}\Gamma, \tag{32}$$

where γ is the friction coefficient and Γ is a white noise random force with zero mean. The idea of using a Langevin equation as a mean of maintaining constant temperature comes from the fact that such equation is related to the Fokker-Planck equation. Due to the noise, the momentum becomes a stochastic variable, and, in this very case, the probabilities of finding a particle with a given value of momentum is described by the Fokker-Planck equation. The solution for this Fokker-Planck equation formed from variables generated by the equation (27) is exactly the Maxwell-Boltzmann distribution. Thus, the Langevin equation enables the canonical ensemble sampling.

H.7 File and format descriptions

H.7.1 Qdyn6 input file format

The name of the input file is passed to **Qdyn6** as the first argument on the command line. The file is divided into sections, each starting with a section heading enclosed in square brackets. Within a section are lines with a keyword and a value, or just values in a defined

order. Comments, starting with '!', '#' or '*' may appear after the data on a line or on separate lines anywhere in the file. The order of sections and the order of data within sections is not important (but the order of table 12 is preferred). Many keywords have default values and can be omitted, to make it easier to set up a simple simulation. The use of default values will be shown in the output from **Qdyn6**. Sections with no required entries are optional.

Table 12: **Qdyn6** input file format

[**MD**]:Basic simulation data.

keyword	**value**	**comment**
steps	Number of MD steps.	Required.
stepsize	MD stepsize (Δt) (fs).	Required.
temperature	Target temperature (K).	Required.
bath_coupling	Temperature bath relaxation time T (fs) to use with berendsen thermostat [39].	Optional, default 100 fs. Must be $\geq$ stepsize.
separate_scaling	Enable (on) or disable (off) separate temperature scaling of solute and solvent.	Optional, default on.
random_seed	Integer value to seed the random number generator.	Optional. Used only for random velocities. Change number to get a different set of velocities. Negative value or omission will generate number from system time.
initial_temperature	Temperature (K) of Maxwellian distribution for random velocities.	Optional except when no restart file is used. Use *only* when velocities should be randomised!
constraint_method	Method to apply constraints to the system, can be either SHAKE or LINCS.	Optional, default SHAKE.
shake_solvent	Enable (on) or disable (off) constraining of bonds and angles of water.	Optional, default on. We recommend the use of constraints for water.
shake_all	Constraining all bonds on or off.	Optional, default off.
shake_solute	Constraining all solute bonds on or off.	Optional, default off.
shake_heavy	Constraining of bonds to heavy atoms in solute and solvent on or off.	Optional, default off.
shake_hydrogens	Constraining of bonds to hydrogen atoms in solute and solvent on or off.	Optional, default on.
lrf	LRF Taylor expansion of electrostatic field beyond cut-off radius (on/off).	Optional, default on.
verbose_temp	Turn extended output for simulation temperature (on) or (off). If on, will print list of atoms exceeding temperature maximum and will print update to current system temperature if changes exceed 2%. By default those are only printed at the desired ouptut interval.	Optional, default off.

Table 12: **Qdyn6** input file format

thermostat	Choice of thermostat used for system. Available thermostats are berendsen, langevin, nose-hoover.	Optional, default berendsen.
langevin_friction	Friction constant for langevin thermostat.	Optional, default 1/bath_coupling.
nhchains	Number of chains for Nose-Hoover thermostat	Optional, default 10.
integrator	Choice of intergrator for simulation, available options are leap-frog and velocity-verlet.	Optional, default leap-frog.

[**PBC**]: Settings for periodic boundary conditions.

rigid_box_centre	Enables the solute to move periodically between boxes.	Optional, default off.
constant_pressure	Enable (on) or disable (off) simulation in the isothermal isobaric ensemble. Set trial interval in the [intervals] section.	Optional, default off.
max_volume_displ	Maximum change in volume in one Monte Carlo step when using the isothermal isobaric ensemble.	Required when constant_pressure is on.
pressure_seed	Seed for the random number generator used to generate new volume conformations in the isothermal isobaric ensemble.	Optional. Use if simulation in isothermal isobaric ensemble is split into several separate input files. Assures good sampling.
pressure	Target pressure when using the isothermal isobaric ensemble.	Optional, default = 1.0 bar. Use only when constant_pressure is on.
put_solute_back_in_box	Enable (on) or disable (off) the putting back of solute molecules in the box. Does not affect energies.	Optional, default on. Disable if measuring diffusion coefficients etc.
put_solvent_back_in_box	Enable (on) or disable (off) the putting back of solvent molecules in the box. Does not affect energies.	Optional, default on. Disable if measuring diffusion coefficients etc.

[**cut-offs**]: Cut-off radii for non-bonded interactions.

solute_solute	Cut-off radius (Å) for solute-solute atom pairs.	Optional, default 10 Å.
solvent_solvent	Cut-off radius (Å) for solvent-solvent.	Optional, default 10 Å.
solute_solvent	Cut-off radius (Å) for solute-solvent.	Optional, default 10 Å.
q_atom	Cut-off radius (Å) for Q-atoms interaction with all atoms.	Optional, default 99 Å, *i.e.* no cut-off. If PBC used, compare to boxlength.
lrf	Cut-off radius (Å) for LRF expansion.	Optional, default 99 Å, *i.e.* no cut-off.

[**sphere**]: Not used in PBC simulations.

shell_force	Force constant (kcal·mol^{-1}·Å^{-2}) for shell restraints.	Optional, default 10 kcal·mol^{-1}·Å^{-2}.
shell_radius	Inner radius of restrained shell (Å).	Optional, default outer shell radius.

Table 12: **Qdyn6** input file format

exclude_bonded	Flag controlling whether bonded interactions between excluded atoms should be eliminated (on) or retained (off) to reproduce energies in earlier versions.	Optional, default off.

[**solvent**]: Boundary conditions of solvent sphere. Optional, can be omitted in vacuum simulations and must be omitted in PBC simulations.

radius	Target solvent radius.	Optional. Use this only to override the calculated target radius from the topology.
radial_force	Force constant (kcal·mol^{-1}·Å^{-2}) for half-harmonic radial restraint at boundary.	Optional, default 60 kcal·mol^{-1}·Å^{-2}.
polarisation	Polarisation restraints in outer solvent shell on or off.	Optional, default on.
charge_correction	Enable (on) or disable (off) correction of solvent polarisation restraints for total charge of Q-atoms by Born's formula.	Optional, default on.
polarisation_force	Force constant (kcal·mol^{-1}·rad^{-2}) for solvent polarisation restraints.	Optional, default = 20 kcal·mol^{-1}·rad^{-2}.
morse_depth	Depth (dissociation energy, kcal·mol^{-1}) of Morse-type boundary attraction potential.	Optional, default is a function of water radius.

[**intervals**]: Intervals between saving data and updating non-bond lists.

non_bond	Non-bond list (and LRF data) update interval.	Optional, default 10.
output	Interval for printing energy summaries.	Optional, default 10.
temperature	Interval for printing temperature. It will always be printed if it changed by $> 2\%$ since last printed and if verbose_temp is set. As tempearture write out now includes statistics concerning the fluctuation, longer write out intervals can be used.	Optional, default 10.
energy	Interval for writing Q-atom energies to energy file.	Optional, default 0 (disabled).
trajectory	Interval for writing coordinates to trajectory file.	Optional, default 0 (disabled).
volume_change	Interval for Monte-Carlo trial of new volume.	Optional, default 0 (disable).

[**files**]: File names for input and output.

topology	Topology file name.	Required.

Table 12: **Qdyn6** input file format

restart	Name of restart file from which initial coordinates and velocities are loaded.	Optional. If absent, initial coordinates are taken from the topology and random velocities are generated (see section MD).
final	Restart file to which the final coordinates and velocities are written.	Required.
trajectory	Trajectory file name.	Optional, except when trajectory interval > 0.
energy	Energy file name.	Optional, except when energy interval > 0.
fep	FEP file name.	Optional, except when lambda values are given.
restraint	Restart file with coordinates to be used for restraining.	Optional. When used, coordinates in the topology will be replaced. This only changes the co-ordinate set used, the restraints must still be specified in *e.g.* the section sequence_restraints.

[**trajectory_atoms**]: which atoms to include in the trajectory.

The data in this section is an atom mask specification, it follows the rules in section Atom masks. Multiple lines may be used.

[**QCP**]: BQCP calculation of quantum corrections.

qcp_seed	Random number seed for MC calculation.	Optional, can be calculated from system time.
qcp_kie	Enable (on) or disable (off) calculation of BQCP energies for different isotopes.	Optional, default off.
qcp_write	Enable (on) or disable (off) write out of bead coordinates to file. Enabled by default in post processing, disabled bey default during MD.	Optional.
qcp_pdb	Name of file for writing out bead coordinates.	Required if qcp_write is on.
qcp_show	Enable (on) or disable (off) printing of additional information concerning BQCP calculations.	Optional, default off.
qcp_debug	Enable (on) or disable (off) printing of BQCP debug information.	Optional, default off.
selection	Specification of atoms that should be converted into ring polymers for BQCP. Can be “hydrogen”, “hyd”, “h” to only include protons, “all”, “qatom”, “fep” to include all atoms in the FEP file, or “individual” to read the information from the FEP file	Optional, default “hydrogen”

Table 12: **Qdyn6** input file format

qcp_size	Specify size of the ring polymers. Can be "default" size 32, "small" size 16, "large" size 64 or user specified later with "userdefine". Please note that the bisection strategy requires ring polymers that are results of 2^n.	Optional, default size 32.
qcp_size_user	Userdefined size of ring polymer. Needs to be larger than 4, and a result of 2^n. Only evaluated if qcp_size selection is "userdefine".	Optional, default 16.
equilibration	Number of MC sampling steps before start of the energy calculation.	Optional, default 10.
sampling	Number of free particle sampling steps.	Optional, default 10.

The following sections do not contain keywords, but data in columns.

[**lambdas**]: λ weights for the FEP states.

column	**description**
1...	Weight for state 1, state 2, ... All λ_i ϵ[0,1] and $\Sigma\lambda_i = 1$.

[**sequence_restraints**]: Restrain sequences of atoms.

1	Number of first atom in sequence.
2	Number of last atom in sequence.
3	Force constant (kcal·mol^{-1}·Å^{-2}) for harmonic potential.
4	Flag for restraining also hydrogens (0=no, 1=yes).
5	Flag for restraining the sequence of atoms to its mass center (2), geometrical centre by the force proportianl to the corresponding atom mass normalized with C12 mass and acting on all atoms (1), or each atom to its initial coordinates (0 or missing).

[**atom_restraints**]: Restrain individual atom positions.

1	Atom number.
2,3,4	Target x y and z coordinates (Å).
5,6,7	Force constants (kcal·mol^{-1}·Å^{-2}) in x y and z directions. With separate force constants for x,y,z this can be used to restrain atoms to lines and planes as well.
8	FEP state where the restraint is active (energies and forces will be scaled by lambda) or 0 = active in all states.

[**distance_restraints**]: Restrain atom-atom distances.

1	Number of first atom.
2	Number of last atom.
3	Lower distance limit for unrestrained region (Å).
4	Upper distance limit for unrestrained region (Å). Set lower limit = upper limit for standard harmonic potential.
5	Force constant (kcal·mol^{-1}·Å^{-2}).
6	FEP state where the restraint is active (energies and forces will be scaled by lambda) or 0 = active in all states.

Table 12: **Qdyn6** input file format

[**wall_restraints**]: Elastic wall (half-harmonic) restraints of sequences of atoms.

1	Number of first atom in sequence.
2	Number of last atom in sequence.
3	Distance from water centre beyond which the restraining potential is applied.
4	Force constant (kcal·mol^{-1}·Å^{-2}) for harmonic potential.
5	Constant D_e in the Morse potential, depth of the potential energy minimum.
6	Constant a in the Morse potential.
7	Flag for restraining also hydrogens (0=no, 1=yes).

[**group_contribution**]: Calculate energies differences of residue or atom groups on free energy profile.

1	Selection name, either atom (for individual atoms) or residue (for complete residues).
2	Calculation type, either "full" for exclusion of both vdW and electrostatic interactions, "electro" to only exclude electrostatics or "vdw" to only exclude vdW interactions. A special selector is "all" that instructs the program to perform all three calculations for the same group.
3...	Atom or residue numbers that should be excluded.

H.7.2 Qpi6 input file format

Input files for **Qpi6** are handled the same way as for **Qdyn6**. The same restrictions concerning file format and comments apply as outlined above in section H.7.1. Files only need limited changes, and commands that are not needed in **Qpi6** compared to **Qdyn6** are ignored, so that users can reuse their files. The major changes needed are mentioned below in table 13. As above, default values are chosen for most keywords and sections if they are omitted.

Table 13: **Qpi6** input file format

[**GENERAL**]:Basic simulation data.

keyword	**value**	**comment**
temperature	Simulation temperature, needed for chosing correct bead distribution.	Optional, if not defined then calculated from restart file.
steps	Ignored in **Qpi6**.	Ignored.
stepsize	Ignored in **Qpi6**.	Ignored.
bath_coupling	Ignored in **Qpi6**.	Ignored.
thermostat	Ignored in **Qpi6**.	Ignored.
integrator	Ignored in **Qpi6**.	Ignored.

[**PBC**]:Setting for periodic boundary conditions.

All settings are ignored in **Qpi6**. Only used to detect if PBC is on or off.

[**intervals**]:Intervals between saving data and updating lists.

All settings are ignored in **Qpi6**.

Table 13: **Qpi6** input file format

[**intervals**]:Intervals between saving data and updating lists.

topology	Topology file name.	Required.
restart	Name of file containing coordinates and velocities.	Optional, only needed to calculate temperatures from velocities if not defined in section [GENERAL].
final	Ignored in **Qpi6**.	Ignored.
traj_input	File name of trajectory file containing classical coordinates from previous simulation that should be used to calculate quantum corrections.	Required.
energy	File name for energy output file.	Required.
fep	FEP file name.	Required for calculations in **Qpi6**.
restart	Restart file with new coordinates for restraining.	Optional, used to redefine positions for the calculation of restraint energies.

All remaining keywords and sections are handled identically to **Qdyn6**. The user should set the same values there as they have been chosen during the classical simulation.

H.7.3 FEP file format

The FEP file (given a .fep extension) designates some atoms from the topology as Q-atoms and redefines the topology for these atoms in a number of states between which transformations can be made. This file is a plain text file. It is divided into sections which can appear in any order and which start with a section title. Within each section the data appears either as lines with values or as keyword-value pairs.

Section titles are enclosed in square brackets and must be the first (non-white space) item on a line. They are not case-sensitive. Keyword-value pairs appear in that order, anywhere on a line but together on the same line. Keyword-value lines can appear in any order (within a section). White space is not significant in value-list lines.

Comments start with "!", "#" or "*" and may appear after values or as separate lines.

A .fep file only needs as a requirement for the **[atoms]** section (where Q-atoms are declared) to be present, all other sections are optional. They may appear in any order, but the preferred order is that seen in table 14. The section [**FEP**] contains the keyword states followed by the number of FEP states defined in the FEP file. An offset value to add to all topology atom numbers in order to avoid renumbering all atoms between *e.g.* free ligand and bound ligand simulations can also be defined in the [**FEP**] section. New atom types for Q-atoms are defined in the [**atom_types**] section. The assignment of Q-atom types to Q-atoms is done, for each state, in the [**change_atoms**] section. Pairs of atoms between which bonds are made or broken should use the exponential repulsion non-bonded potential instead of the standard Lennard-Jones by listing them under the [**soft_pairs**] heading. In some cases it is desirable to completely turn off certain non-bonded interactions. This can be done on a per-state basis in the section [**excluded_pairs**].

New bond, angle, torsion and improper types can also be defined and used for any atoms in the topology, not only between Q-atoms. Atoms are therefore referred to by their number in the topology rather than by a Q-atom number. Definitions in the topology are overridden by definitions in the FEP file. To disable an interaction in one state, a zero should be used in place of the type number for that state.

Angles, torsions and impropers which depend on the existence of a bond being formed or broken should be "coupled" to that bond by scaling the angle energy by the ratio of the actual value of the Morse bond energy to the dissociation energy. These couplings are defined in the sections [**angle_couplings**], [**torsion_couplings**] and [**improper_couplings**].

Extra shake constraints can be imposed between any pair of atoms in the topology using the heading [**shake_constraints**]. The effective constraint distance will be the sum of the distances for each state weighted by their respective λ's.

Quantum-mechanical mixing of states used in EVB calculations by introducing off-diagonal Hamiltonian matrix element functions is defined in the section [**off_diagonals**].

If vanishing or appearing atoms are part of your FEP strategy, it may be desirable to use a *softcore* potential for the Q-atoms in question. [43] Softcore potentials have been implemented in **Qdyn6** according to equation 33. Depending on the value of **softcore_use_max_potential** given in the [**FEP**] section, the α-values are either read directly from the [**softcore**] section or calculated by **Qdyn6** in a pairwise manner based upon the desired potentials at $r = 0$ given in the [**softcore**] section. Softcore potentials are only available for Q-atoms, i.e. Q-Q, Q-water and Q-solute interactions are treated with softcore. In the case of a Q-Q interaction where both Q-atoms have softcore potentials, the α which is used is that which gives rise to the lowest potential at $r = 0$.

$$V_{vdW}(r_{ij}) = \frac{A_{ij}}{(r_{ij}^6 + \alpha)^2} - \frac{B_{ij}}{r_{ij}^6 + \alpha} \qquad \text{or} \qquad V_{vdW}(r_{ij}) = \epsilon \cdot \left(\frac{{R_{ij}^*}^{12}}{(r_{ij}^6 + \alpha)^2} - 2 \cdot \frac{{R_{ij}^*}^{6}}{r_{ij}^6 + \alpha}\right) \quad (33)$$

If periodic boundary conditions are used an additional section [**PBC**] is needed. In this section one switching atom for all Q-atoms is defined. This switching atom is used when generating the Q-surrounding nonbonded pair lists.

Table 14 lists the data and units for each column in the different sections, and an example is included as file example on page 31.

Table 14: FEP file format

[**atoms**]: Define Q-atoms.

column	**description**
1	Q-atom number (counting from 1 up).
2	Topology atom number.

[**PBC**]: For periodic boundary conditions.

keyword	**value**	**comment**

Table 14: FEP file format

switching-_atom	Topology atom number.	Required with periodic boundary conditions. Defines which atom to use as switching atom when calculating nonbonded pairlist for qatom interactions

[**FEP**]: General perturbation information.

keyword	**value**	**comment**
states	Number of FEP/EVB states.	Optional, default 1.
offset	Topology atom number.	Optional, default 0. This number is added to all topology atom numbers given in the FEP file.
offset_residue	Residue/fragment number.	Optional. Set offset to the topology number of the first atom in the given residue minus one.
offset_name	Residue/fragment name.	Optional. Set offset to the topology number of the first atom in the first residue with the given name minus one.
qq_use-_library-_charges	This is a special feature for studying *e.g.* electrostatic linear response. Set to 'on' to use the library charges from the topology for intra-Q-atom interactions, i. e. change only Q-atom-surrounding electrostatic interactions.	Optional, default off.
softcore-_use_max-_potential	Set to 'on' if the values entered in the [**softcore**] section are the desired maximum potentials (kcal/mol) at $r = 0$. **Qdyn6** will then calculate pairwise α_{ij} to be used in equation 33. 'off' means the values are to be used directly in equation 33.	Optional, default off.

[**change_charges**]: Redefine charges of Q-atoms.

column	**description**
1	Q-atom number (referring to numbering in atoms section).
2...	Charge (e) in state 1, state 2, ...

[**atom_types**]: Define new atom types for Q-atoms: Standard LJ parameters and parameters for the exponential repulsion potential $V_{soft} = C_i \cdot C_j \epsilon^{-a_i \cdot a_j \cdot r_{i,j})}$.

1	Name (max 8 characters).
2	Lennard-Jones A parameter ($\text{kcal}^{\frac{1}{2}} \cdot \text{mol}^{-\frac{1}{2}} \cdot \text{Å}^6$) for geometric combination or R* ($\text{kcal} \cdot \text{mol}^{-1} \cdot \text{Å}^{12}$)for arithmetic combination rule.
3	LJ B parameter ($\text{kcal}^{\frac{1}{2}} \cdot \text{mol}^{-\frac{1}{2}} \cdot \text{Å}^3$) or ϵ ($\text{kcal} \cdot \text{mol}^{-1} \cdot \text{Å}^6$).
4	Soft repulsion force constant C_i ($\text{kcal}^{\frac{1}{2}} \cdot \text{mol}^{-\frac{1}{2}}$) in V_{soft}.
5	Soft repulsion distance dependence parameter a_i ($\text{Å}^{-\frac{1}{2}}$) in V_{soft}.
6	Lennard-Jones A parameter ($\text{kcal}^{\frac{1}{2}} \cdot \text{mol}^{-\frac{1}{2}} \cdot \text{Å}^6$) or R* ($\text{kcal} \cdot \text{mol}^{-1} \cdot \text{Å}^{12}$) for 1-4 interactions.
7	LJ B parameter ($\text{kcal}^{\frac{1}{2}} \cdot \text{mol}^{-\frac{1}{2}} \cdot \text{Å}^3$) or e ($\text{kcal} \cdot \text{mol}^{-1} \cdot \text{Å}^6$) for 1-4 interactions.
8	Atomic mass (u).

Table 14: FEP file format

[**change_atoms**]: Assign Q-atom types to Q-atoms.

1	Q-atom number.
2...	Q-atom type name in state 1, state 2, ...

[**soft_pairs**]: Define pairs which use soft repulsion.

1	Q-atom number of first atom in pair.
2	Q-atom number of second atom in pair.

[**excluded_pairs**]: Define pairs to exclude from non-bonded interactions. Note: also non-Q-atoms can be excluded.

1	Topology atom number of first atom in pair.
2	Topology atom number of second atom in pair.
3...	Exclusion effective (1) or not (0) in state 1, state 2, ...

[**el_scale**]: Define q-atom pairs for scaling of the electrostatic interaction. Can be useful e.g. when highly charged intermediates appear in FEP/EVB. The scale factor applies to all states. Note: only Q-atom pairs can be scaled.

1	q-atom number of first atom in pair
2	q-atom number of second atom in pair
3	electrostatic scale factor (0..1)

[**softcore**]: Define q-atom softcore potentials. The meaning of these entries depends on the value of softcore_use_max_potential.

1	q-atom number
2...	Desired potential at $r = 0$ for all of this q-atom's vdW interactions in state 1, state 2, ... or the actual α value used in equation 33. An α of 200 yields vdW potentials at $r = 0$ of 10-50 kcal/mol for heavy atom - heavy atom interactions. Set to 0 if softcore is not desired for this q-atom.

[**monitor_groups**]: Define atom groups whose non-bonded interactions are to be monitored (printed in the log file).

1...	Topology atom number of first and following atoms in group.

[**monitor_group_pairs**]: Define pairs of monitor_groups whose total non-bonded interactions should be calculated.

1	First monitor_group number.
2	Second monitor_group number.

[**bond_types**]: Define Q-bond types using Morse or harmonic potentials,
$E_{Morse} = D_e \left(1 - e^{-\alpha(r-r_0)}\right)^2$ $E_{Harmonic} = \frac{1}{2} k_b \left(r - r_0\right)^2$.
Morse and harmonic potentials can be mixed (but each bond type is either kind). Entries with four values are Morse potentials and entries with three values are harmonic.

	Morse potential	**Harmonic potential**
1	Q-bond type number (starting with 1).	
2	Morse potential dissociation energy, D_e (kcal·mol^{-1}).	Harmonic force constant k_b (kcal·mol^{-1}·Å^{-2}).
3	Exponential co-efficient α in Morse potential (Å^{-2}).	Equilibrium bond length r_0 in harmonic potential (Å).

Table 14: FEP file format

4	Equilibrium bond length r_0 in Morse potential (Å).	

[**change_bonds**]: Assign Q-bond types. Note: shake constraints for the redefined bonds are removed. The order in which atoms are given is not important.

1	Topology atom number of first atom in bond.
2	Topology atom number of second atom in bond.
3...	Q-bond type number (referring to numbering in bond_types section) or 0 to disable bond in state 1, state 2, ...

[**angle_types**]: Define Q-angle types.

1	Q-angle type number (starting with 1).
2	Harmonic force constant (kcal·mol^{-1}·rad^{-2}).
3	Equilibrium angle (°).

[**change_angles**]: Assign Q-angle types.

1	Topology atom number of first atom in angle.
2	Topology atom number of middle atom in angle.
3	Topology atom number of third atom in angle.
4...	Q-angle type number (referring to numbering in angle_types section) or 0 to disable angle in state 1, state 2, ...

[**torsion_types**]: Define Q-torsion types.

1	Q-torsion type number (starting with 1).
2	Force constant = $\frac{1}{2}$·barrier height (kcal·mol^{-1}).
3	Periodicity (number of maxima per turn).
4	Phase shift (°).

[**change_torsions**]: Assign Q-torsion types. Note: The order of atoms (1, 2, 3, 4 or 4, 3, 2, 1) is not important.

1	Topology atom number of first atom in torsion.
2	Topology atom number of second atom in torsion.
3	Topology atom number of third atom in torsion.
4	Topology atom number of fourth atom in torsion.
5...	Q-torsion type number (referring to numbering in torsion_types section) or 0 to disable torsion in state 1, state 2, ...

[**improper_types**]: Define Q-improper types.

1	Q-improper type number (starting with 1).
2	Harmonic force constant (kcal·mol^{-1}·rad^{-2}). N.B. new impropers defined here are always harmonic.
3	Equilibrium angle (°).

[**change_impropers**]: Assign Q-improper types. Note: The order of atoms (1, 2, 3, 4 or 4, 3, 2, 1) is not important.

1	Topology atom number of first atom in improper.
2	Topology atom number of second atom in improper.
3	Topology atom number of third atom in improper.
4	Topology atom number of fourth atom in improper.

Table 14: FEP file format

5...	Q-improper type number (referring to numbering in improper_types section) or 0 to disable improper in state 1, state 2, ...

[**angle_couplings**]: Couple Q-angles to Q-bonds, *i.e.* scale angle energy by the ratio of the actual value of the Morse bond energy to the dissociation energy.

1	Q-angle number (line number within change_angles section).
2	Q-bond number (line number within change_bonds section).

[**torsion_couplings**]: Couple Q-torsions to Q-bonds.

1	Q-torsion number (line number within change_torsions section).
2	Q-bond number (line number within change_bonds section).

[**improper_couplings**]: Couple Q-impropers to Q-bonds.

1	Q-improper number (line number within change_impropers section).
2	Q-bond number (line number within change_bonds section).

[**shake_constraints**]: Define extra shake constraints. The effective constraint distance will be the sum of the distances given for each state, weighted by their λ values. Note: constraints defined here do not override constraints imposed by setting the shake flag to *on* in the **qdyn** input file. To remove a constraint the bond must be redefined as a Q-bond. The order in which atoms are given is not important.

1	Topology atom number of first atom.
2	Topology atom number of second atom.
3...	Constraint distance (Å) in state 1, state 2, ...

[**off-diagonals**]: Define off-diagonal elements of the Hamiltonian, represented by $H_{i,j} = A_{i,j} \cdot \epsilon^{-\mu_{i,j} \cdot r_{k,l}}$where i and j are states and k and l are Q-atoms.

1	State i.
2	State j.
3	Q-atom k.
4	Q-atom l.
5	$A_{i,j}$ (kcal·mol^{-1}).
6	$\mu_{i,j}$ (Å^{-1}).

[**qcp_atoms**]: Individual definition of atoms to include for BQCP calculations. Only needed if not defined in Qdyn6 or Qpi6 input file.

1	QCP atom index.
2	Q-Atom number.

[**qcp_mass**]: Definition of atom massed for isotope calculations. All masses not defined here will be taken from the mass information in the FEP file [**atom_types**] section. Masses in amu.

1	QCP atom index number.
2	Isotope mass in amu.

H.8 Utility programs

Analysis of simulation data is possible through the use of either the graphical user interface QGui (available at `https://github.com/qusers/qgui`) or through a set of command

line tools available at `htpps://github.com/mpurg/qtools`. Those programs only rely on the programs available in Q and can facilitate both simulation set–up and analysis.

Within Q6 itself, the following two utility tools are available.

H.8.1 Qcalc6

Command	Description
xscore	Scores topology, trajectories and restart files using the X-Score algorithm
chemscore	Scores topology, trajectories and restart files using the ChemScore algorithm
PMF-Score	Scores topology, trajectories and restart files using the PMF-Score algorithm

See section F.6 on page 41 for more information about scoring.

H.8.2 Qdum6

Qdum6

function:	Test **Qdyn6** input files quickly.
input:	**Qdyn6** input file.
output:	Log file, restart file.
usage:	`Qdum6 test.inp`
notes:	**Qdum6** is a version of **Qdyn6** without the dynamics loop. It reads all input, initiates the simulation, writes a restart file and terminates. **Qdum6** is build from the same source code as **Qdyn6**.

H.8.3 Qpi6

Qpi6

function:	Perform trajectory post–processing to calculate quantum corrections using the BQCP approach.
input:	Modified **Qdyn6** input file as outlined above in section F.5. Requires Qdyn6 trajectory file, FEP file, Topology file and optionally Qdyn6 restart file
output:	Log file, energy file.
usage:	`Qpi6 test.inp`
notes:	**Qpi6** calculates path integral qunatum corrections to the classical energies using the BQCP approach. It reads the coordinates from the trajectory file and performs a number of calculations on each classical structure after converting the coordinates of the atoms in the reacting center into ring–polymers.

References

(1) Marelius, J.; Kolmodin, K.; Feierberg, I.; Åqvist, J. *Journal of Molecular Graphics and Modelling* **1998**, *16*, 213–225.
(2) Kollman, P. *Chemical Reviews* **1993**, *93*, 2395–2417.
(3) Beveridge, D. L.; DiCapua, F. M. *Annual Review of Biophysics and Biophysical Chemistry* **1989**, *18*, 431–492.
(4) Warshel, A., *Computer Modeling of Chemical Reactions in Enzymes and Solutions*, Wiley Prof; Wiley Interscience: 1997.
(5) Åqvist, J.; Warshel, A. *Chemical Reviews* **1993**, *93*, 2523–2544.
(6) Åqvist, J.; Medina, C.; Samuelsson, J. E. *Protein engineering* **1994**, *7*, 385–91.
(7) Jones-Hertzog, D. K.; Jorgensen, W. L. *Journal of Medicinal Chemistry* **1997**, *40*, 1539–1549.
(8) Hansson, T.; Marelius, J.; Åqvist, J. *Journal of computer-aided molecular design* **1998**, *12*, 27–35.
(9) Major, D. T.; Gao, J. *Journal of Chemical Theory and Computation* **2007**, *3*, 949–960.
(10) Gao, J.; Wong, K.-Y.; Major, D. T. *Journal of Computational Chemistry* **2008**, *29*, 514–522.
(11) Warshel, A. *Chemical Physics Letters* **1978**, *55*, 454–458.
(12) Berkowitz, M.; McCammon, J. A. *Chemical Physics Letters* **1982**, *90*, 215–217.
(13) Brünger, A.; Brooks, C. L.; Karplus, M. *Chemical Physics Letters* **1984**, *105*, 495–500.
(14) Isaksen, G. V.; Andberg, T. A. H.; Åqvist, J.; Brandsdal, B. O. *Journal of Molecular Graphics and Modelling* **2015**, *60*, 15–23.
(15) Purg, M.; Bauer, P. qtools v0.5.9., 2017.
(16) http://www.rcsb.org/pdb/docs/format/pdbguide2.2/guide2.2 frame.html.
(17) Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; Hermans, J., *Intermolecular Forces; Reidel: Dordrecht, The Netherlands, 1981*; Pullman, B., Ed.; Reidel: Dordrecht, The Netherlands: 1981, pp 331–342.
(18) Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L. *The Journal of Chemical Physics* **1983**, *79*, 926–935.
(19) Kolmodin, K.; Åqvist, J. *FEBS Letters* **1999**, *456*, 301–305.
(20) Åqvist, J. *The Journal of Physical Chemistry* **1991**, *95*, 4587–4590.
(21) Humphrey, W.; Dalke, A.; Schulten, K. *Journal of Molecular Graphics* **1996**, *14*, 33–38.
(22) http://www.ks.uiuc.edu/Research/vmd/.
(23) Laaksonen, L. *Journal of molecular graphics* **1992**, *10*, 33–4, 24.
(24) http://www.csc.fi/~laaksone/gopenmol/gopenmol.html.
(25) Hwang, J. K.; Warshel, A. *The Journal of Physical Chemistry* **1993**, *97*, 10053–10058.
(26) Ceperley, D. M. *Reviews of Modern Physics* **1995**, *67*, 279–355.
(27) Wang, R.; Lai, L.; Wang, S. *Journal of computer-aided molecular design* **2002**, *16*, 11–26.
(28) Eldridge, M. D.; Murray, C. W.; Auton, T. R.; Paolini, G. V.; Mee, R. P. *Journal of computer-aided molecular design* **1997**, *11*, 425–45.

(29) Muegge, I.; Martin, Y. C. *Journal of Medicinal Chemistry* **1999**, *42*, 791–804.
(30) Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W.; Kollman, P. A. *Journal of the American Chemical Society* **1995**, *117*, 5179–5197.
(31) Maier, J. A.; Martinez, C.; Kasavajhala, K.; Wickstrom, L.; Hauser, K. E.; Simmerling, C. *Journal of Chemical Theory and Computation* **2015**, *11*, 3696–3713.
(32) Jorgensen, W. L.; Tirado-Rives, J. *Journal of the American Chemical Society* **1988**, *110*, 1657–1666.
(33) Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S.; Karplus, M. *Journal of Computational Chemistry* **1983**, *4*, 187–217.
(34) Van Gunsteren, W. F.; Berendsen Groningen Molecular Simulation (GROMOS) Library Manual., Biomos, Nijenborgh 16, Groningen, NL, 1987.
(35) Van. Gunsteren, W. F.; Billeter, S.; Eising, A.; Hünenberger, P.; Krüger, P.; Mark, A.; Tironi, I., *Biomolecular simulation : the GROMOS96 manual and user guide*; Biomos ; Vdf, Hochschulverlag AG an der ETH Zürich: Zürich; Groningen; Zürich, 1996.
(36) Jorgensen, W. L.; Maxwell, D. S.; Tirado-Rives, J. *Journal of the American Chemical Society* **1996**, *118*, 11225–11236.
(37) Robertson, M. J.; Tirado-Rives, J.; Jorgensen, W. L. *Journal of Chemical Theory and Computation* **2015**, *11*, 3499–3509.
(38) Essex, J. W.; Jorgensen, W. L. *Journal of Computational Chemistry* **1995**, *16*, 951–972.
(39) Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; DiNola, A.; Haak, J. R. *The Journal of Chemical Physics* **1984**, *81*, 3684–3690.
(40) Martyna, G. J.; Klein, M. L.; Tuckerman, M. *The Journal of Chemical Physics* **1992**, *97*, 2635–2643.
(41) Schneider, T.; Stoll, E. *Physical Review B* **1978**, *17*, 1302–1322.
(42) *Understanding Molecular Simulation: From Algorithms to Applications*, 1st; Frenkel, D., Smit, B., Eds.; Academic Press, Inc.: Orlando, FL, USA, 1996.
(43) Zacharias, M.; Straatsma, T. P.; McCammon, J. A. *The Journal of Chemical Physics* **1994**, *100*, 9025–9031.