FlexX Commandline Documentation Version 6.0

Sascha Jung & Marcus Gastreich

May 11, 2023

Contents

1 Introduction 2

2 Technical Prerequisites 3
- 2.1 Required Software . . . 3
- 2.2 Licensing . . . 3

3 Help 5

4 Jump Start: A Classical Docking 6

5 Program Options 7

6 Docking 10
- 6.1 Standard-Docking . . . 10
- 6.2 Template-Docking . . . 10
- 6.3 Covalent-Docking . . . 10
- 6.4 Re-scoring with HYDE . . . 12

7 Algorithmic Controls: The Configuration 14
- 7.1 Allowed conformations for aliphatic six-membered rings . . . 14
- 7.2 The Number of Poses per Ligands . . . 14
- 7.3 Enumeration of Stereo Isomers . . . 15

8 General Options 15

9 Further Reading, References, How to Cite 16

1 Introduction

All links, references, table of contents lines etc. in this pdf are clickable.

FlexX is a tool for powerful protein-ligand docking at the commandline.

FlexX is also a "component" within our flagship 3D modeling platform SeeSAR that has been conceived for drug researchers of any discipline and educational level (`https://www.biosolveit.de/SeeSAR`). Within SeeSAR's Docking Mode, you can easily define pharmacophore constraints, interactively inspect and post-process the results, e.g. by scoring the docking solutions with our HYDE scoring function.

FlexX at the commandline features:

- Flexible ligand docking into target cavities
- Fuzzy template-based docking: Define a template (for example a common core), FlexX then takes your ligand input and places anything that is common onto each other and then places the rest of the ligand-to-dock into the cavity
- Covalent ligand docking, requires pre-processed ligand input (for example available from us at `https://www.biosolveit.de/SeeSAR#covlib`)
- Consideration of pharmacophore constraints *during* the docking (i.e., the pharmacophores are not acting as post-filters), thus accelarting the docking and delivering better results
- Optional enumeration of stereo isomers during the docking
- Processing of unlimited ligand numbers

FlexX does **not** support our award-winning, desolvation-aware HYDE scoring. HYDE is available either within the SeeSAR graphical suite or as a standalone commandline tool from `https://www.biosolveit.de/download`.

Please note that this package is a commandline package.

2 Technical Prerequisites

2.1 Required Software

FlexX is a commandline application. Control through a graphical user interface (GUI) is available in SeeSAR, BioSolveIT's 3D flagship modeling suite.[1]

Technically, you will need:

- The FlexX **application package** (from `https://biosolveit.de/download`). Depending on your operating system, some libraries may have to be installed. Get in touch with us if that is the case: `mailto:support@biosolveit.com`. Please mention any errors/warnings that you see in your mail.
- A **shell** (Linux/Unix) or a terminal (macOS), or a commandline environment (Windows; e.g.: cmd.exe or PowerShell)
- A valid **license** (from `mailto:license@biosolveit.de`), see below.

2.2 Licensing

FlexX needs a license to operate which is available from us. There are various sorts of licenses, but in most of the cases your early testing will employ a license file that simply needs to be put next to the executable, see below.

The license setup instructions will come with the license that we will send out — or that has already been sent out to you. In case you do not have a license yet, please get in touch with us at `mailto:contact@biosolveit.de`, and provide us with the necessary information. Please note: a SeeSAR license will be read as "valid" by FlexX.

License File Locations A "test license" that you can request online and that is sent to you instantaneously can simply be placed next to the executable (`flexx.exe`, `FlexX` or `flexx` — depending on your operating system). For macOS please read on...

[1] We highly recommend you also install SeeSAR to swiftly carry out all sorts of preparational steps for the receptor, notably any relevant pharmacophore definitions etc.; the preparation can then be written to a `*.flexx` file and used at the commandline with FlexX (this package).

macOS Specialties On MacOS, the executable will typically reside inside the *.app package:

```
/Applications/FlexX.app/Contents/MacOS/FlexX
```

To place the short term test license there, you will have to go into the *.app package using a right mouse click (or CTRL-click) on FlexX.app in the Finder, and click on "Show package contents". In there, you will see the Contents/ subfolder, in there the MacOS subfolder, and in there, the FlexX executable. If you are about to use the **test license**, place is right there, next to the executable. A longer term license will be handled separately, we will tell you how when we send that very license.

When you call FlexX for the first time, go to the Finder, and navigate to the Applications folder. Do a right(!) click on FlexX.app, and — if applicable — confirm that you want to open the program. It will flash up once, and you are good to go at the terminal prompt from there on.

Obtaining a License File Using `--license-info` you can obtain information about the specification of your license server machine, the searched directories, and the validity of the currently used license files. This may also be useful when FlexX is not starting up as you would expect it to.

Call FlexX with the `--license-info` option, to see an output like this:

```
./FlexX --license-info
================================================================================
 License Information:
================================================================================
Host-ID: "28f0763fa408 38c98636b9cb"
BIOSOLVE_LICENSE_FILE: /Applications/BioSolveIT/License
LM_LICENSE_FILE:

Currently used key/path after environment variables:
/Applications/FlexX.app/Contents/MacOS

FlexX:
 >> Valid key, expires on Thursday, 30 June 2023

Request a license:
***
*** https://www.biosolveit.de/license/?product=flexx&operating_system=darwin...
***
```

Request an evaluation or longer-term license using the link that is provided at the very bottom of the output. Also, this output may help us to find out if there are any problems with your license or its setup.

3 Help

An overview of all commmandline options is available by calling FlexX with `--help`; alternatively get help for a command with a tailing `-h` or `--help`. Default vaues are bracketed:

```
./flexx  -h

Program options:
 -i [ --input ] arg              Library input molecule file. Supported file types are *.mol, *.mol2 and
                                 *.sdf.
                                 Note: 3D coordinates must be provided, otherwise molecule is skipped.
 -o [ --output ] arg             Output file, only '.sdf' is supported.
 -p [ --protein ] arg            Protein file. Only '.pdb' is supported.
                                 Note: Can't be used together with '--docking-definition'.
 -r [ --refligand ] arg          Reference ligand file. Supported file types are *.mol, *.mol2 and *.sdf.
                                 Note: The reference ligand must have 3D coordinates and lie in the pocket
                                 in the protein as it is used to determine the binding site for docking.
                                 Note: Can't be used together with '--docking-definition'.
 --docking-definition arg        Run docking on the basis of this docking definition file (an .flexx file
                                 that can be exported from SeeSAR's Docking mode) - it contains the protein
                                 with predefined binding site and pharmacophore constraints.
                                 Note: Can't be used together with '--protein' or '--refligand'.
 -d [ --docking-type ] arg (=0)  Determines the algorithm which is used for docking:
                                      0 [Standard-Docking]
                                      1 [Template-Docking] The algorithm for template docking mandates
                                        the simultaneous use of '--template'.
                                      2 [Covalent-Docking] The algorithm for covalent docking mandates
                                        the simultaneous use of '--docking-definition'. The docking
                                        definition file has to be prepared for covalent docking and
                                        the library molecules must contain exactly one linker atom.
 -t [ --template ] arg           Template ligand file. Supported file types are *.mol, *.mol2 and *.sdf.
                                 Note: The template ligand must fulfill the same requirements as the
                                 reference ligand. Also, if a template ligand is provided, the library
                                 molecules must share a sizeable common substructure with the template
                                 otherwise they will be skipped.
 --thread-count arg              Maximum number of threads used for calculations. The default is to use all
                                 available threads.

Configuration:
 --allowed-6ring-confs arg (=0)  Conformations allowed for aliphatic six-membered rings:
                                      0 [Only chair conformations are allowed.]
                                      1 [Chair and twist-boat conformations are allowed.]
                                      2 [Chair, twist-boat and boat conformations are allowed.]
 --max-nof-conf arg (=10)        Maximum number of top-ranking result conformations for each docked
                                 molecule. Default value is 10.
                                 Note: If a docking definition file is used, the value specified in the file
                                 is used as default.
 --stereo-mode arg (=0)          Automatically flip acyclic stereo centers during docking:
                                      0 [Do not flip stereo centers]
                                      1 [Flip R/S stereo centers]
                                      2 [Flip E/Z stereo centers]
                                      3 [Flip R/S and E/Z stereo centers]

General options:
 -h [ --help ]                   Prints this help message
 --license-info                  License info
 --thread-count arg              Maximum number of threads used for calculations. The default is to
                                 use all available cores.
 --version                       Prints version info
 -v [ --verbosity ] arg (=2)     Set verbosity level
                                      0 [silent]
                                      1 [error]
                                      2 [warning] 3 [workflow] 4 [steps]
```

Please note that the abbreviated, one-letter options are preceded with one dash **-** whereas the longer, named options are preceded with two dashes: `--`.

In addition, we are available to support your endeavors with FlexX by email (`mailto:support@biosolveit.de`). We try to answer within a day, during business hours.

4 Jump Start: A Classical Docking

To run a docking at the commandline with all defaults, you will need at least:

- An input file containing the compounds-to-dock (`.sdf`, `.mol` or `.mol2` format, with 3D coordinates for every compound)

(a) Protein and reference ligand

- A protein input file (`.pdb`) **and**
- An input ligand file (i.e., the so-called reference ligand) with 3D coordinates in a binding cavity of the input protein (`.sdf`, `.mol` or `.mol2` format). The reference ligand helps to define the binding site.

(b) A docking definition file

- A docking definition file (`.flexx`) that you have exported from within SeeSAR's Docking Mode.

Now run this — with the file names replaced by your own names of course:
In case (a), that is, if you provided a protein file and a reference ligand file:

```
./FlexX -i My3DLibrary.sdf -p MyProtein.pdb -r My3DReferenceLigand.sdf
        -o MyDockingOutput.sdf
```

In case (b), that is, if you have prepared and exported a docking definition file from SeeSAR:

```
./FlexX -i My3DLibrary.sdf --docking-definition MyDefinitionFile.flexx
        -o MyDockingOutput.sdf
```

Depending on the operating system you use, please certainly also adapt the commandline usage, e.g., use a slash or a backslash etc.

The above calls run the docking and create the output file `MyDockingOutput.sdf` that, by default, contains 10 docking solutions (poses) for every ligand — given that the docking was successful. The SD output file will also contain the respective docking score for every pose in a separate SD property field named `<BIOSOLVEIT.DOCKING_SCORE>` for post-processing purposes.

Please note:
We *highly* recommend that you re-score the docking poses with our award-winning, desolvation-aware HYDE scoring to improve the ranking. Information is here: `https://www.biosolveit.de/products/#HYDE`. Alternatively, you can load your docking results into SeeSAR for HYDE post-scoring, and take a look at the colored spheres to rationalize the findings — and make informed decisions on further steps you may want to take.

5 Program Options

`-i` / `--input` Specify the ligand input file in `.sdf`, `.mol` and `.mol2` format. The input file should contain all the compounds which you want to dock. Please make sure that all molecules have 3D coordinates, otherwise they are skipped during the docking run. Generation of 3D coordinates is automatically carried out in SeeSAR. You can load your 2D molecule file into SeeSAR's Molecule Editor Mode and export the automatically generated 3D molecules as SD file. Alternatively, you can use our 3D Coordinates Generator available as KNIME node (especially recommended for libraries > 10000 ligands). More information on how to install and use our KNIME nodes is available from our website (`https://www.biosolveit.de/wp-content/uploads/2020/10/First_Steps_in_KNIME.pdf`).

`-o` / `--output` Specify a name for the output file (`.sdf` format) containing the docking solutions (poses). The output file will contain all successfully docked molecules from the input file with a maximum number of top-ranked poses as defined with the `--max-nof-conf` argument (see page 14).

`-p` / `--protein` Specify a file containing the protein in `.pdb` format.[2] If you do so, please make sure you also specify a reference ligand with the `-r` option

[2] In addition to protein structures, FlexX can also process DNA and RNA targets.

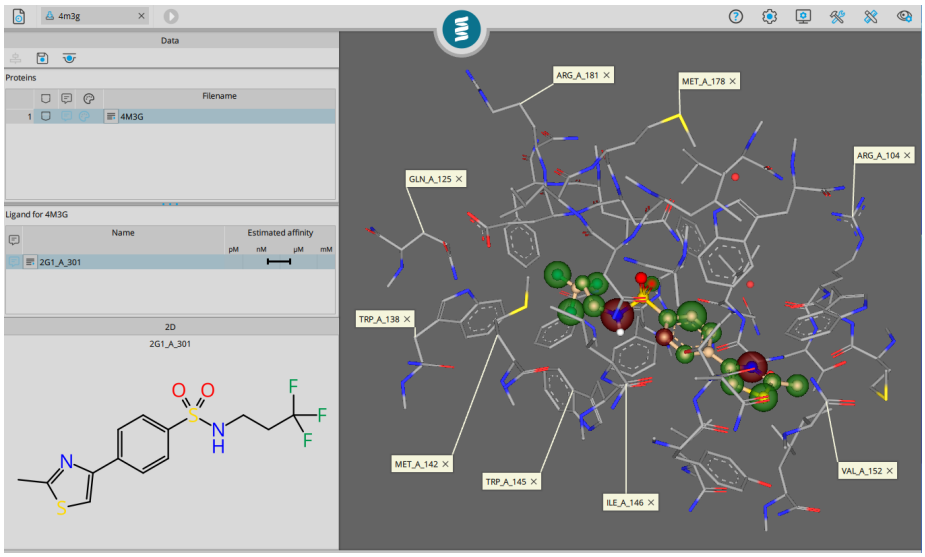

Figure 1: *Example for the automated binding site definition on the basis of a reference ligand (PDB 4M3G). All protein residues which have at least one atom lying in the 6.5 Å sphere defined around every ligand heavy atom are included.*

(see below). This is obsolete if you have prepared a docking definition file (see `--docking-definition` option below).

`-r` / `--refligand` Provide a file containing the reference ligand in `.sdf`, `.mol` and `.mol2` format. The reference ligand will be used to determine the binding site for docking. Therefore, it is important you make sure the reference ligand has 3D coordinates and occupies a pocket of the protein which you specified via the `-p` option. To define the binding pocket, FlexX automatically detects all protein residues which have at least one atom lying in a 6.5 Å sphere around the reference ligand atoms (see Figure 1 for an example). This is very reasonable in most cases. However, if your reference ligand is very small (e.g. a fragment-like binder) the defined binding site may be to small to dock the larger compounds of your library. In this case you can manually extend the binding pocket by enlarging the fragment-like binder, e.g. by adding an aliphatic chain in SeeSAR's Molecule Editor Mode. Please note that the argument cannot be used together with the `--docking-definition` option.

`--docking-definition` As an alternative way to using a separate protein and reference ligand file (see above) you can run a docking on the basis of a docking definition file (`.flexx` file). The `.flexx` file must have been exported from SeeSAR's Docking Mode and contains the protein with a predefined binding site, optionally with additional pharmacophore constraints. These pharma-

cophores are used to guide the docking process and do not act as simple post filters.[2] Therefore, the docking calculations are accelerated and you generate only relevant docking solutions according to your constraints. Please note that the `--docking-definition` option cannot be used together with `-p` and `-r` options.

`-t` / `--template` Specify a template ligand file (supported file types are `.sdf`, `.mol` and `.mol2`). FlexX can use a ligand with a known binding mode as template. Template-based docking is performed by determining the Maximum Common Substructure (MCS) between the template and the compound-to-dock. Based on the MCS multiple overlays are generated to preserve the binding mode of the common core. You can use this for fast docking of congeneric compound series or for growing-like docking from an "anchor", e.g. a fragment-like ligand. The template ligand file must fulfill the same requirements as the reference ligand, e.g. it must have 3D coordinates and lie in a pocket of the protein. If you provide a template ligand, the input molecules must share a sizeable common substructure (5 heavy atoms or more) with the template otherwise they are skipped during the docking run. Please note: To run a template docking, the `-d` option must also be set to 1 (see below). See Section 6.2 for more information on template-based docking.

`-d` / `--docking-type (0)` Specify the algorithm which is used for docking by giving an integer as argument:

0 **Standard Docking.** Flexible placement and incremental construction of the ligand in the binding site, the default. See Section 6.1 for more information.

1 **Template Docking.** Requires `--template` option to be set (see above). See Section 6.2 for more information.

2 **Covalent Docking.** This requires an appropriate `.flexx` file given via the `--docking-definition` option. Additionally, the input ligands must be preprocessed. See Section 6.3 for more information.

6 Docking

6.1 Standard-Docking

FlexX is a super-fast anchor-and-grow docker. It operates based on an incremental construction algorithm to place a ligand into a binding site.[3] First, the ligand is split into fragments and an initial fragment or combinations of fragments are placed into different positions of the pocket. These initial placements are then scored with a very fast pre-scoring scheme. From these initial anchors the ligand is fully constructed by sequentially adding the remaining fragments and by scoring the intermediate solutions against each other. The top scored final solutions are retained and given to you as output.

6.2 Template-Docking

To perform a template-based docking, an appropriate template ligand must be given via the `-t` option **and** the `-d` option must be set to 1 (see Section 5). FlexX can superimpose a fragment of the ligand-to-dock onto a known conformation of a template ligand. Typically, the template ligand is the bioactive conformation of a bound ligand from a crystal structure. Template-based docking is performed by an adaptive matching procedure. First, the Maximum Common Substructure (MCS) between the template ligand and the compound-to-dock is determined. Based on the MCS multiple superpositions are generated to accurately preserve the binding mode of the common core and then the final solutions are obtained via the FlexX incremental construction procedure. You can use the template approach, e.g. for high-speed docking of congeneric compound series where the binding mode is known; for very fast mining of template-like structures from databases (non-suited molecules are skipped); or for pseudo-growing of fragment-like binders into pockets.

6.3 Covalent-Docking

Covalent docking is performed for ligands which contain a so-called "warhead". Typically, the ligand warhead is an electrophilic group which can react with a suitable nucleophilic residue in the binding site of a protein (typically a cysteine or serine residue). A reaction between a protein and a ligand normally consists of two steps: the first step involves the formation of a non-covalent complex between the ligand and the protein in which the electrophilic warhead comes close to the nucleophilic residue. Then, in the second step, the covalent bond

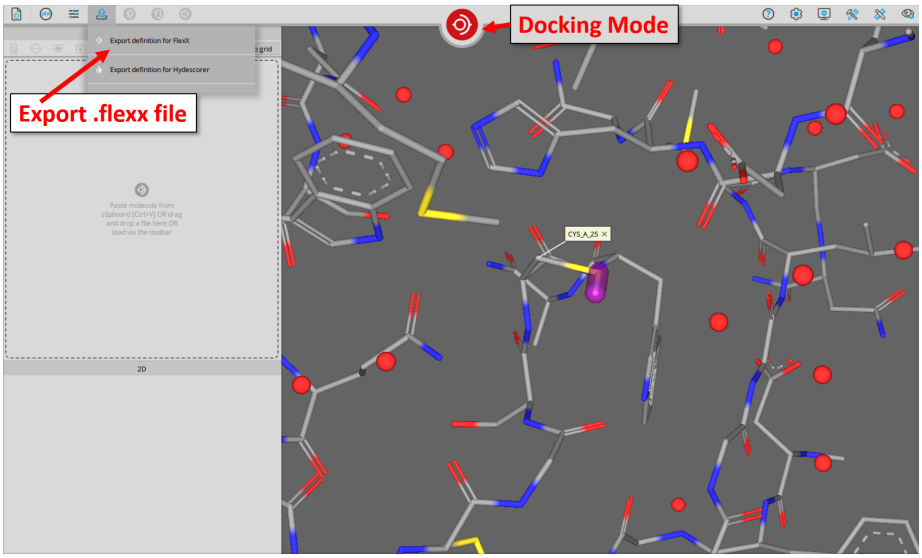

Figure 2: *Export a docking definition file for covalent docking from within SeeSAR's Docking Mode. If the reaction center is defined correctly (or automatically detected from a covalently bound ligand), a pink cylinder is displayed around the nucleophilic atom. In this example, PDB 2P7U was loaded into SeeSAR and the nucleophilic residue was automatically detected through the covalently bound ligand.*

formation takes place and the final protein-bound state (covalent complex) is obtained. With covalent docking you can model this final covalent complex.
To run a covalent docking, an appropriate `.flexx` file must be given via the `--docking-definition` option **and** the `-d` option must be set to 2 (see Section 5). The docking definition file must be exported from within SeeSAR's Docking Mode. You must define the protein residue with which the input molecules should react. If you load a protein into SeeSAR which already contains a covalently bound ligand (covalent complex), then the residue to which the ligand is bound will be automatically defined as reaction point (marked with a pink cylinder in SeeSAR's Docking Mode, see Figure 2). You can now directly export the `.flexx` file. If you do not have loaded a covalent complex, first, you must define the reaction point manually by adding a linker atom (R) to the reactive residue in SeeSAR's Protein Editor Mode (see Figure 3). After saving the edited protein to the table, you can transfer it back to the Protein Mode and use it to define the binding site. Now you can again export the `.flexx` file from within the Docking Mode.

The input molecules (compounds-to-dock) also have to be prepared specifically and must also contain exactly one linker atom (R). The compounds-to-dock must be in the post-reacted, protein-bound state (see Figure 4 for an ex-

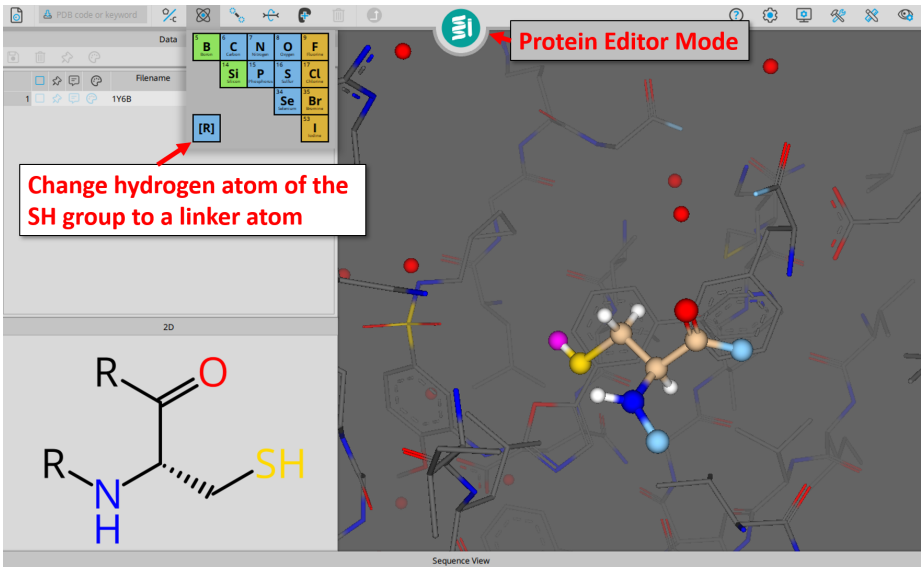

Figure 3: *Manual addition of a linker atom to a nucleophilic residue in SeeSAR's Protein Editor Mode. In this example, the pdb id 1y6b was loaded into SeeSAR and transferred to the Protein Editor Mode. After selecting the appropriate cysteine residue (CYS1043), the hydrogen atom of the SH group was replaced by a linker atom (R).*

ample of a nitrile warhead). You can transform molecules for covalent docking either manually in SeeSAR's Molecule Editor Mode (for a single or a few compounds, see Figure 5 for the above mentioned nitrile warhead) and save them as an SD file or you can use our KNIME Workflow CovXplorer (`https://www.biosolveit.de/KNIME/#CovXplorer`).[3] Further information on how to prepare compounds and protein for covalent docking and how to run a covalent docking in SeeSAR can also be obtained from our workshop on YouTube (`https://www.youtube.com/watch?v=2CSiz5guoXg`).

6.4 Re-scoring with HYDE

In any case, we recommend to re-score your obtained docking solutions with our desolvation-aware HYDE scoring function to improve the ranking. You can either load the output results file into SeeSAR or use the standalone commandline tool (`https://www.biosolveit.de/products/#HYDE`). A major advantage of using SeeSAR is the visualization of the docking results with the

[3] Additionally, you can use our pre-compiled and ready-to-use covalent docking libraries (`https://www.biosolveit.de/SeeSAR#covlib`).

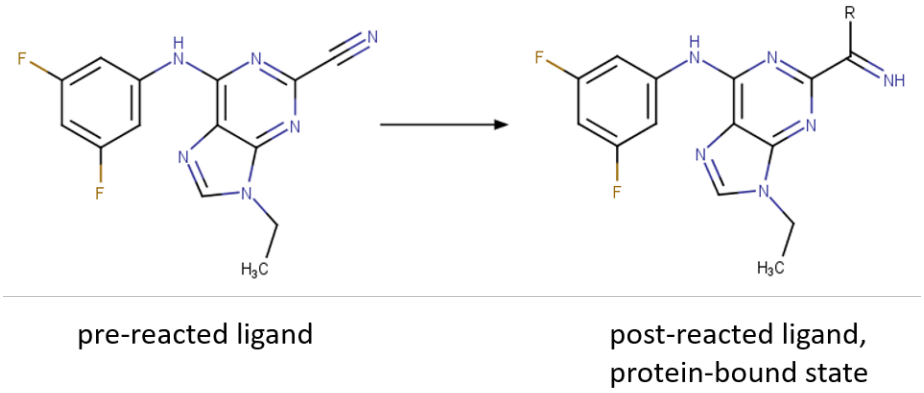

Figure 4: *Example for the transformation of a ligand with nitrile warhead into its protein-bound state.*

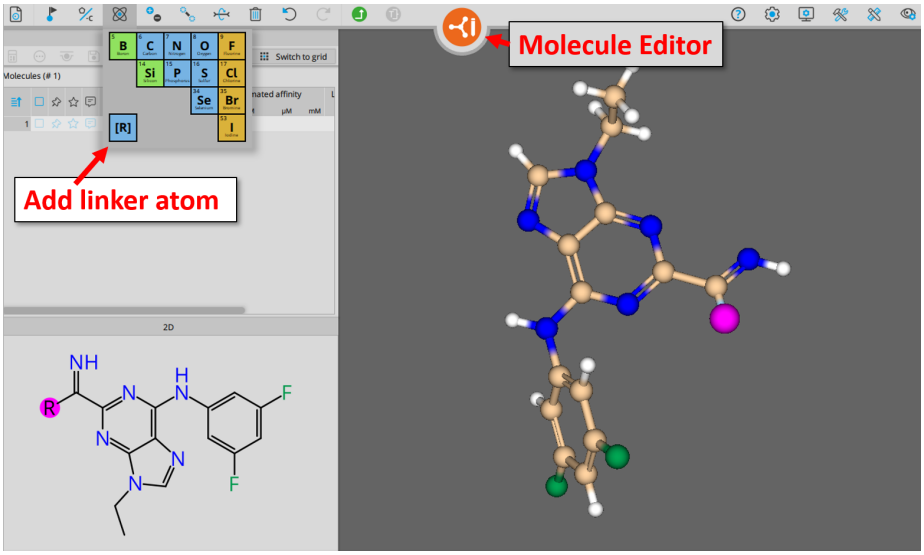

Figure 5: *Using the capabilities of SeeSAR's Molecule Editor Mode to transform a ligand with nitrile warhead to its protein-bound state.*

per-atom HYDE contributions. This enables you to easily rationalize the results and select compounds for further investigation.

7 Algorithmic Controls: The Configuration

In this section, the default values for the respective option are bracketed.

7.1 Allowed conformations for aliphatic six-membered rings

`--allowed-6ring-confs (0)` Select the conformations allowed for six-membered aliphatic rings. By default, only low-energy chair conformations are tolerated. You can gradually adjust this behavior by allowing also the energetically less favorable twist-boat and boat conformations:

- `0` Only chair conformations are allowed. The default.
- `1` Chair and twist-boat conformations are allowed.
- `2` Chair, twist-boat and boat conformations are allowed.

Please note that if you provide a docking definition file the parameter will be read from the respective `.flexx` file and used as default. However, you can overwrite it with the `--allowed-6ring-confs` option.

7.2 The Number of Poses per Ligands

`--max-nof-conf (10)` With this argument you can control the maximum number of generated docking solutions (conformations) per ligand. The default value is 10 conformations per compound. Please note that if you provide a docking definition file using the `--docking-definition` option the number of conformations will be read from the respective `.flexx` file and used as default. However, you can overwrite it with the `--max-nof-conf` option. For the virtual screening of large libraries, the default of 10 conformations will often be too high and produce large amounts of data, so that you may want to reduce this to 1-2 conformations that are usually sufficient to enrich promising binders. However, for smaller libraries and for re-docking of compounds it may be beneficial to increase the maximum number of conformations up to 100 in order to generate more poses for the re-scoring with HYDE (see Section

6.4). After re-scoring, it is now more likely to find the near-native conformation among the top-ranked poses.

7.3 Enumeration of Stereo Isomers

`--stereo-mode (0)` This option allows you to control the treatment of acyclic stereo centers during the docking by giving an integer as argument. The default value is 0, e.g. stereo centers are maintained during docking. The following behavior can be selected:

- `0` Do not flip stereo centers. All stereo centers of the input molecules are maintained. This is the default value.
- `1` Flip R/S stereo centers.
- `2` Flip E/Z stereo centers.
- `3` Flip both R/S and E/Z stereo centers.

Flipping R/S and/or E/Z stereo centers may help to identify the isomer with the most promising binding geometry for your target. Keep in mind that flipping every stereo center can lead to a combinatorial explosion for compounds with multiple stereo centers! This may lead to extended runtimes for large docking libraries.

8 General Options

`-h` / `--help` Displays the commandline help with short descriptions for every argument option. For more information see Section 3.

`--license-info` Displays detailed information about the license setup you currently use. For more information see Section 2.2.

`--thread-count` Specify the maximum number of threads used for the docking calculations. By default, all available logical cores of your computer are used. You may want to reduce the number of threads used if you want to run other computations on your computer at the same time, or if you share the compute resource.

`--version` Displays information on the version of FlexX on the commandline. In quoting FlexX, please mention this version number.

`-v` / `--verbosity` You can set the verbosity level, e.g. the level of console output, by giving an integer as argument. The default value is 2. The following options are available:

- `0` Silent. No messages will be displayed in the console during the docking run. Errors will be ignored whenever possible.
- `1` Error. Only error messages leading to failures of docking runs will be displayed.
- `2` Warning. The default setting. Warnings and error messages will be displayed.
- `3` Workflow. In addition to errors and warnings, the different steps of the docking workflow are displayed.
- `4` Steps. In addition to the 'Workflow' option, the progress of the docking run is displayed in detail.

9 Further Reading, References, How to Cite

The original ideas behind the FlexX incremental docking method have been cited hundreds of times; they are covered in the original publication by Matthias Rarey.[3] Additional information on performance and scoring can be found below in the references ([1] and[4]). Over the years, FlexX has undergone steady further developments and improvements, amongst them, for example, parallelization, extensions for covalent binders, fuzzy template-driven dockings and pharmacophore-guided dockings.

Additional information on the tool is available at

`https://biosolveit.de/products/#FlexX`.

Cite FlexX as:

```
FlexX Version 6.0.0, BioSolveIT GmbH,
St. Augustin, Germany, 2023, biosolveit.de/FlexX
```

Complementary tools, especially also the graphical platforms SeeSAR and infiniSee, can be obtained from the BioSolveIT website (`https://biosolveit.com`).

We wish you great success and much joy with FlexX!

References

[1] Marcus Gastreich, Markus Lilienthal, Hans Briem, and Holger Claussen. Ultrafast de novo docking combining pharmacophores and combinatorics. *Journal of Computer-Aided Molecular Design*, 20:717–734, 12 2006.

[2] Sally A Hindle, Matthias Rarey, Christian Buning, and Thomas Lengauer. Flexible docking under pharmacophore type constraints. *Journal of Computer-Aided Molecular Design*, 16:129–149, 2002.

[3] Matthias Rarey, Bernd Kramer, Thomas Lengauer, and Gerhard Klebe. A fast flexible docking method using an incremental construction algorithm. *Journal of Molecular Biology*, 261(3):470–489, 1996.

[4] Gregory L. Warren, C. Webster Andrews, Anna-Maria Capelli, Brian Clarke, Judith LaLonde, Millard H. Lambert, Mika Lindvall, Neysa Nevins, Simon F. Semus, Stefan Senger, Giovanna Tedesco, Ian D. Wall, James M. Woolven, Catherine E. Peishoff, and Martha S. Head. A critical assessment of docking programs and scoring functions. *Journal of Medicinal Chemistry*, 49(20):5912–5931, 2006. PMID: 17004707.