SilcsBio User Guide

Release 2023.1

Dec 2022

CONTENTS

SilcsBio LLC
1100 Wicomico Street, Suite 323
Baltimore, MD 21230
info@silcsbio.com
https://silcsbio.com

CHAPTER

ONE

ABOUT THIS DOCUMENT

SilcsBio offers a suite of computer-aided drug design software including: the patented SILCS functional group mapping approach with capabilities for (a) small-molecule database screening, fragment-based drug design, and lead optimization, and (b) excipient screening for biomacromolecular therapeutics; and the CGenFF force field parameter assignment engine. This documentation covers how to install and use the software.

CHAPTER

TWO

RELEASE NOTES

2.1 Version 2023.1

This version adds the following:

- GCMC sampling utilizing GPU resources. Prior versions utilized GPU resources for the MD portion of the GCMC/MD sampling in SILCS simulations whereas GCMC sampling was done using CPUs only. The new GPU GCMC code can make runs 50% faster overall while also enabling more efficient probe sampling due to algorithmic improvements for mu_ex oscillation.
- SILCS protocol specifically tailored for RNA targets
- SILCS-Biologics workflow setup, run control, and data analysis from the SilcsBio Graphical User Interface (GUI)
- SILCS support for membrane proteins other than GPCRs, such as transporters and ion channels
- SILCS-Hotspots output validation capability by docking of a diverse subset of FDA-approved drugs
- Mechanism for checking internet connection, server connection, and SILCS server version compatibility from within the SilcsBio GUI
- Integrated SILCS-Biologics documentation access from within the SilcsBio GUI
- Updated graphical user interface design
- Compatibility with GROMACS version 2022.x
- Improved jobs control within the SilcsBio GUI
- Scripting to subtract one set of FragMaps from another, including automatic handling of differences in FragMap grid dimensions. This can be useful for qualitative analysis of functional group binding preferences between two homologous proteins.

as well as bug fixes and stability improvements.

2.2 Version 2022.2

This version adds the following:

- SilcsBio Graphical User Interface (GUI) support for Halogen SILCS simulation
- SilcsBio GUI support for inclusion of Halogen SILCS FragMaps in ligand posing and scoring for SILCS-MC: Docking and Pose Refinement and SILCS-MC: Ligand Optimization
- Support for custom temperature choice for SILCS simulations
- Support for using existing setup/trajectory files for starting SILCS simulations via the SilcsBio GUI
- Expanded SilcsBio GUI visualization capabilities including: atomic GFE values and their sums for ligands scored with SILCS-MC; comprehensive atom selection options for visualization; and residue labels on proteins
- Usability improvement to the SilcsBio GUI including: showing project creation dates; ability to sort by date or name during file/directory selection; and progress bar for file transfers
- CGenFF program is version 2.5.2 and CGenFF parameters are version 4.6

as well as bug fixes and stability improvements.

2.3 Version 2022.1

This version adds the following:

- SilcsBio Graphical User Interface (GUI) support for SILCS-Hotspots
- Distance measurement between SILCS-Pharm pharmacophore features in the SilcsBio GUI
- Charting results from SILCS-MC jobs (ligand optimization, docking, pose refinement, hotspots) prior to full job completion, and recalculation of results at any time through the SilcsBio GUI
- Selection of number of processors and number of independent runs for SILCS-MC through the SilcsBio GUI
- SILCS-Hotspots report generation script update to show additional measurements
- SILCS-MC docking and pose refinement expansion to include Halogen FragMaps
- SILCS simulation with iron ions can be with +2 (default) or +3 charge
- Dimethylether replaces acetaldehyde in Standard SILCS simulations
- CGenFF program update to version 2.5.1 and CGenFF parameters update to version 4.6

The CGenFF program version 2.5.1 can optionally retain partial charges from the input Mol2 file instead of assigning them. CGenFF parameters version 4.6 is updated with training that adds numerous new compounds:

Model compounds for nonstandard amino acids (206 new residues encompassing 87 new bond parameters, 437 new angle parameters, 1698 new dihedral terms, and 34 new improper dihedral parameters) [18]

Flavin and flavin-related model compounds

as well as bug fixes and stability improvements.

2.4 Version 2021.1

This version adds the following:

- The SILCS-Biologics method for excipient screening for biomacromolecular therapeutics
- CGenFF program update to version 2.5 and CGenFF parameters update to version 4.5

  CGenFF coverage has been extended to the following compounds that have been explicitly parameterized and validated, allowing CGenFF to generate parameters for more diverse compounds.

  ASBB: (1-[(2-aminoethyl)sulfanyl]butan-1-one): charges and bonded parameters

  2-phenylthiazole and 5-methyl-3-phenyl-1,2,4-oxadiazole: bonded parameters

  Fentanyl: bonded parameters [16]

  (E)-1,2-di-p-tolyldiazene: bonded parameters [17]

as well as bug fixes and stability improvements.

2.5 Version 2020.2

This version adds the following:

- SilcsBio Graphical User Interface (GUI) improvements for file and directory selection allowing for input files to be chosen from remote servers as well as the local computer
- SilcsBio Graphical User Interface (GUI) support for visualization of Halogen SILCS FragMaps
- A new plug-in for visualizing SILCS FragMaps in MOE
- CGenFF program and parameters update to version 2.4.0

CGenFF coverage has been extended to amide bases and molecules containing boron. The functional groups in these molecules were not previously accessible in CGenFF. The following compounds have been explicitly parameterized and validated, allowing CGenFF to generate parameters for more diverse compounds.

N-methyl acetamide (deprotonated amide); N-ethyl acetamide (deprotonated amide); N-methyl benzamide (deprotonated amide); N-phenyl acetamide (deprotonated amide)

Methyl boronic acid (neutral, -1, -2); Ethyl boronic acid (neutral, -1, -2); Phenyl boronic acid (neutral, -1, -2)

Additionally, 112 naturally-occurring modified nucleotides, especially modified bases with heterocycles not previously covered by CGenFF, have been parametrized. Quantum mechanical calculations on model compounds, including geometries, dipole moments, and interactions with water, provided target data. Selected parameters were validated with extensive molecular dynamics simulations. Details of the parameter optimization and the complete list of nucleotides can be found in [15].

as well as bug fixes and stability improvements.

Validation of the SILCS-HotSpots approach has been published [13].

2.6 Version 2020.1

This version adds the following:

- The SILCS-Hotspots method for identifying all potential ligand binding sites on a protein
- SilcsBio Graphical User Interface (GUI) support for SILCS-Pharm
- SilcsBio Graphical User Interface (GUI) support for SILCS-MC Docking and Pose Generation
- Performance improvements to GCMC-MD
- CGenFF program and parameters update to version 2.3.0

  CGenFF version 2.3.0 adds explicit parametrization for the following molecules. In prior versions, functional groups in these molecules were accessible through analogy to related functional groups. Now, explicit parametrization and validation yields more accurate treatment and decreases the associated CGenFF penalty scores.

  1H-tetrazole; 5-methyl-1H-tetrazole; 5-ethyl-1H-tetrazole

  2-oxetanone; 3-oxetanone

  ammonium; dimethylammonium; trimethylammonium (Note: Protonated amine parameters were previously based on methylammonium. While the present explicit parametrization of secondary and tertiary amines leads to smaller penalty scores and improvements in performance, electrostatic interactions will continue to be dominated by the +1 monopole.)

1-butyne; 1-pentyne; 1-hexyne; 1-heptyne; 1-octyne; but-1-ene-3-yne (Note: Alkyne parmeters were previously based on ethene and propene. Extension to longer alkynes and the ene/yne combination validates the parameters and leads to improved treatment of the intramolecular interactions.)

CGenFF version 2.3.0 also includes improved halogen–protein interactions. Quantum mechanical calculations on chloro- and bromobenzene with model compounds representative of protein functional groups were used as target data to optimize atom-pair specific Lennard-Jones parameters for selected atoms in the model compounds. Application of the parameters in molecular dynamics simulations of eight ligand-protein systems demonstrated systematic improvement in interaction geometries.

as well as bug fixes and stability improvements.

2.7 Version 2019.2

This version adds the following:

- The SILCS-Pharm method for generating 3D receptor-based pharamcophore models from FragMaps in an automated manner

as well as bug fixes and stability improvements.

2.8 Version 2019.1

This version adds the following:

- The SILCS-Pharm method for generating 3D receptor-based pharamcophore models from FragMaps in an automated manner (early access)

as well as bug fixes and stability improvements.

2.9 Version 2018.2

This version adds the following:

- A new Graphical User Interface (GUI) capable of preparing and launching SILCS, SILCS-MC, and SSFEP jobs on remote computing clusters and analyzing and visualizing the job outputs
- A new CGenFF Parameter Optimizer with functionality for dihedral parameter fitting

as well as bug fixes and stability improvements.

2.10 Version 2018.1

This version adds the following:

- A new Graphical User Interface (GUI) capable of preparing and launching SILCS, SILCS-MC, and SSFEP jobs on remote computing clusters and analyzing and visualizing the job outputs (early access)
- Improved GPCR and other transmembrane protein support for SILCS
- Improved GPCR and other transmembrane protein support for SSFEP
- CGenFF program and parameters update to version 2.2.0

  CGenFF version 2.2.0 extends support to a large variety of drug-like molecules to be used routinely in computer-aided drug design projects. Specifically, 1) it improves treatment of halogen bonds by introducing lone-pairs onto the halogen atoms of aromatic systems. 2) Support is extended to four-membered oxetane and glycoluril. 3) Improved predictions with partial charge distributions around ammonium ions and primary amines.
- A new CGenFF Parameter Optimizer with functionality for dihedral parameter fitting (early access)

as well as bug fixes and stability improvements.

2.11 Version 2017.2

This version adds the following:

- GPCR support for SILCS
- GPCR support for SSFEP
- CGenFF program and parameter update to version 2.1.0

  CGenFF version 2.1.0 extends support to S-P bond found in the GTP-gamma like molecules. Bonded parameters and charge-distribution along the S-P bond were modeled using the CHARMM nucleic acid force-field patches for mono- and di-thio substitutions (top-par_all36_na_reactive_rna.str). Three new atom-types have been added to the CGenFF force-field : SG2P1, SG2P2 to support the mono- and di-thio substitutions, along with OG2S1 to model the terminal oxygen connected to the S-P bond.

as well as bug fixes and stability improvements.

2.12 Version 2017.1

This version is the initial release. This version includes the following packages:

- SILCS command line interface
- SILCS-MC command line interface
- SSFEP command line interface
- CGenFF program and parameter version 2.0.0

  CGenFF version 2.0.0 is the second release of CGenFF, extending support to a larger variety of drug-like molecules to be used routinely in computer-aided drug design projects. Specifically, it improves treatment of halogen bonds, by introducing lone-pairs onto the halogen atoms of aromatic systems. Additionally, support is now extended to four-membered oxetane and glycoluril moieties. This is driven largely by improvements in the newly released version 4.0 of CGenFF force field.

CHAPTER

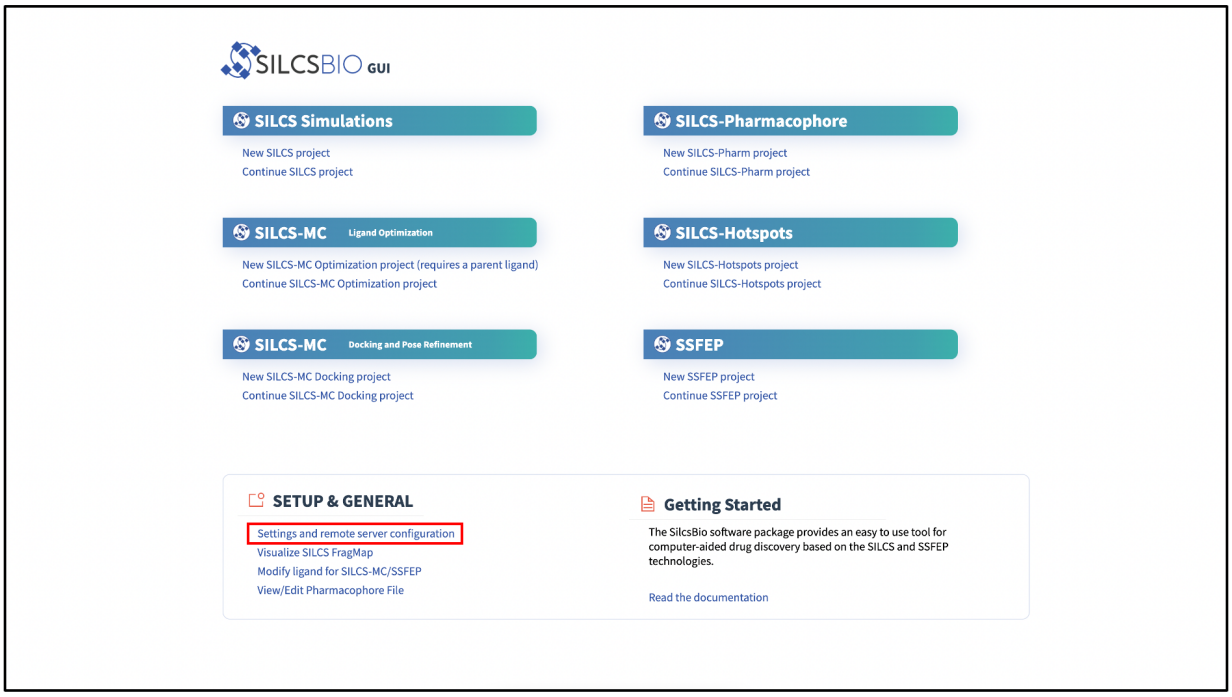
THREE

GRAPHICAL USER INTERFACE QUICKSTART

This chapter provides a step-by-step introduction on how to use the SilcsBio Graphical User Interface (GUI).

3.1 Remote server setup

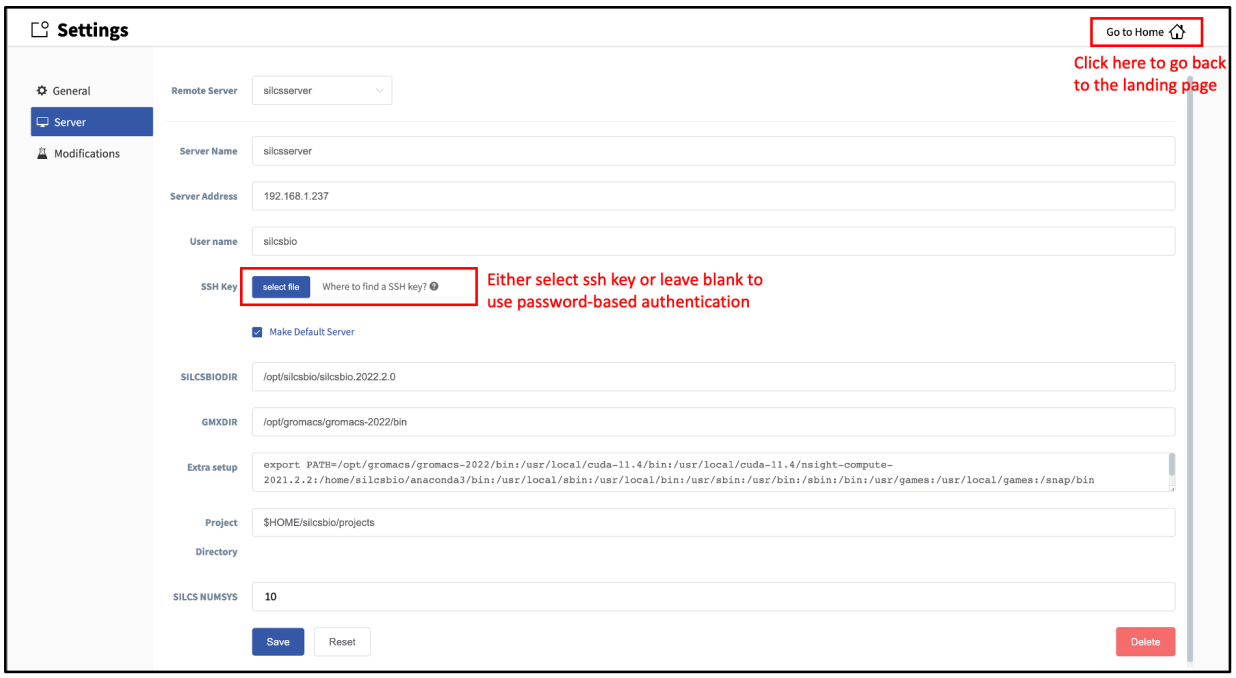
The SilcsBio GUI is designed to work with the server installation of the SilcsBio software. When you launch the GUI, you will be presented with the Home page. From the Home page, select *Settings and remote server configuration*.



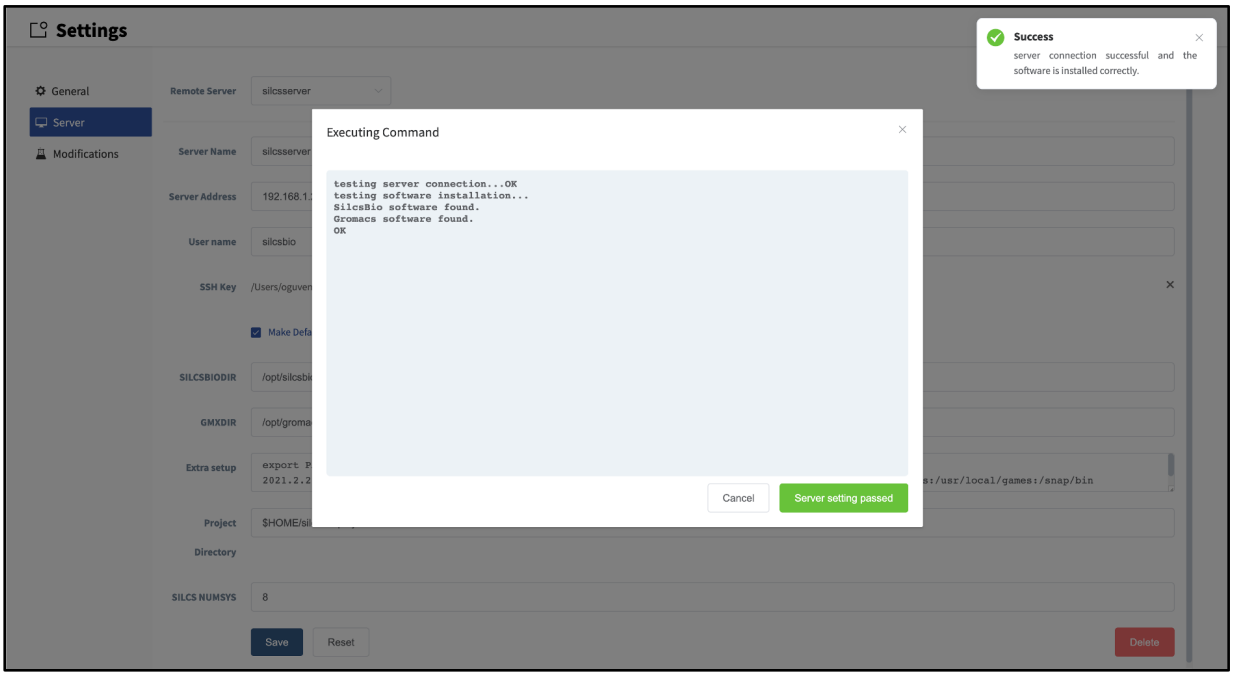
Within the "Settings" page, select *Server* and enter the remote server information, such as Server Address (IP address), Username, and an SSH key to the server. If you do not have an SSH key to

the server, leave it blank. The GUI will ask you the password to the remote server instead. Select the "Make Default Server" checkbox if you would like to set this server as your default server. This server will be selected as a default in other parts of the GUI.

You will also need to enter `SILCSBIODIR` and `GMXDIR` file path values. These should match the values on your server. You may use the "Extra setup" field to pass additional commands to the server, such as exporting environment variables or loading modulefiles. You may use the "Project Directory" field to define the directory on your server where server job input files will be created and server job outputs stored. Typical SILCS simulations produce output files in excess of 100 GB, so please select a project directory file folder with appropriate storage capacity. The "SILCS NUMSYS" field determines how many SILCS jobs are launched for creating FragMaps. Set this to an even integer; we recommend "10" to maximize convergence or "8" to minimize resource use.



Once all information is entered, click the "Save" button. The GUI will save your entries and confirm that you have a working connection to the server.



Please contact support@silcsbio.com if you need additional assistance.

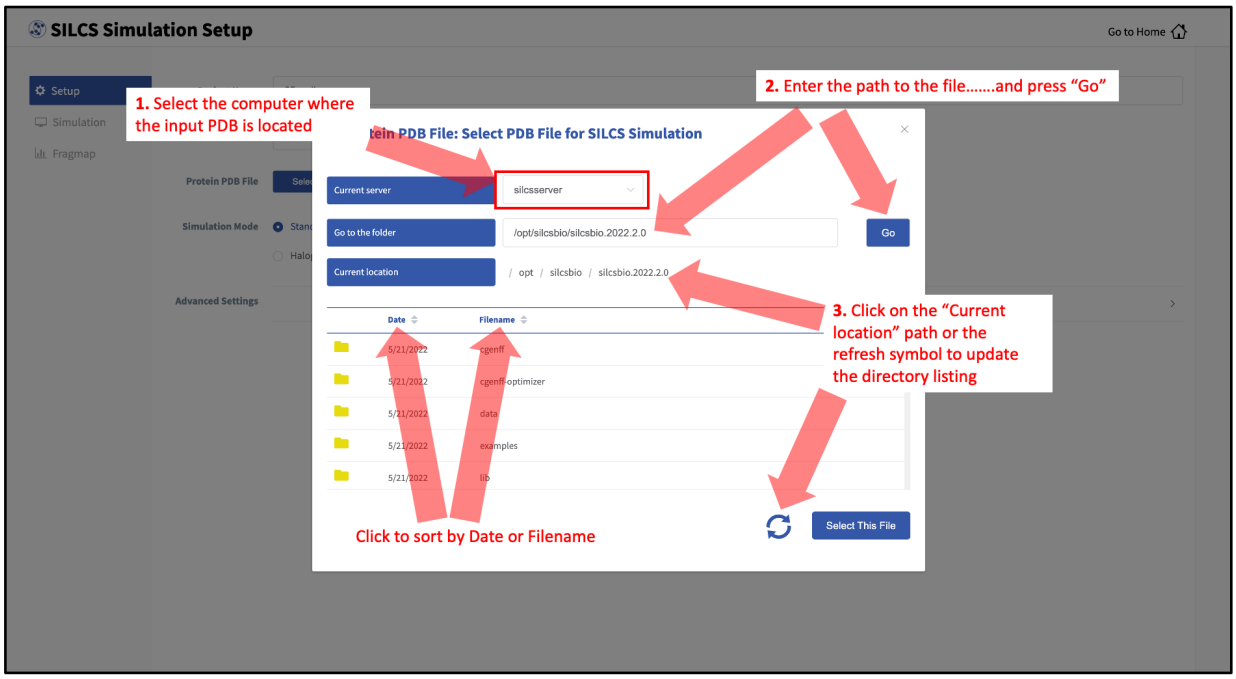
3.2 File and directory selection

To run the SilcsBio software, you will need to provide protein and/or ligand input files. The SilcsBio GUI has a standardized user interface that allows you to choose these input files from either the computer on which you are running the GUI or from a remote server you have previously configured. Steps to do this, illustrated in the below figure, are as follows.

1. Choose the "Current server," which is the physical location of the input file. If you would like to load a file located on the computer on which you are running the GUI, select "localhost"; otherwise, select the remote server name of a server you have previously configured through the Remote server setup process.

2. Type in the folder where the file is located using the "Go to the folder" field and then click the associated "Go" button. This will update the "Current location" to this folder. Click on the last portion of the "Current location" path (e.g. "silcsbio.2020.2.0" in "/opt/silcsbio/silcsbio.2020.2.0" in the below example) to update the directory listing located under that path. If you click on any other portion of the "Current location" path, you will be taken to that folder. Double clicking on a folder in the directory listing will move you into that folder.

3. In the directory listing, click on a file to highlight it and click the "Select This File" button at the bottom to finish your file selection.



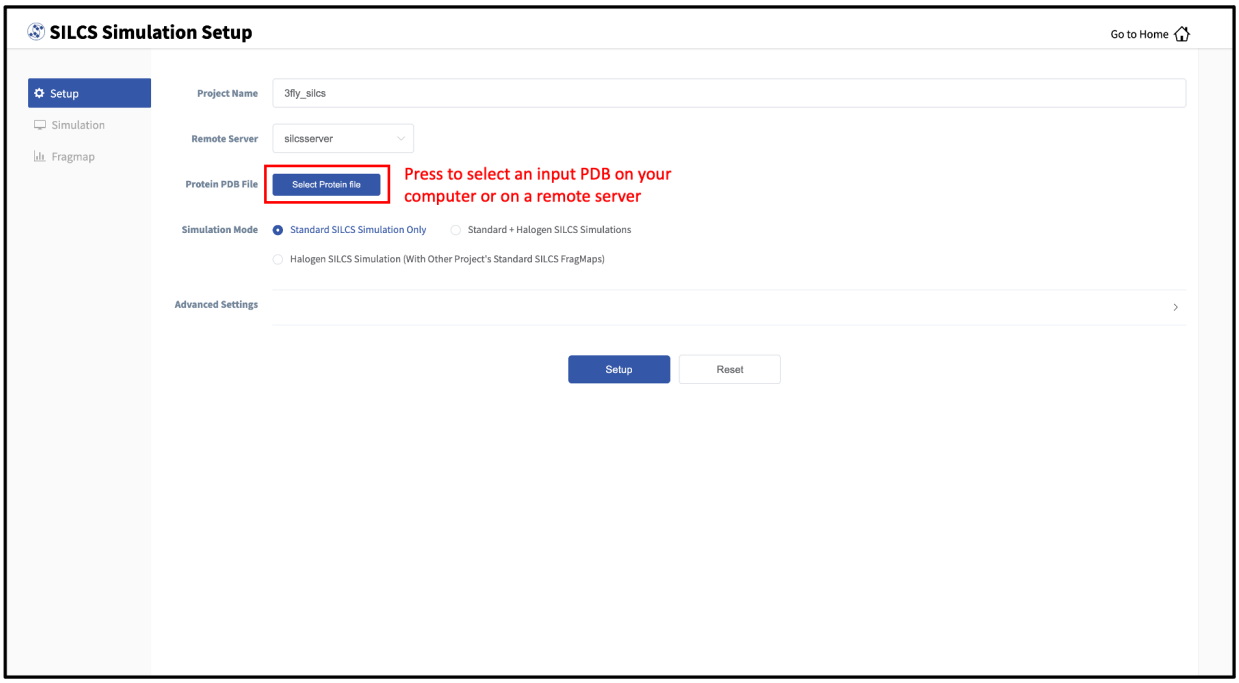
Follow a similar process to select a directory for those tasks requiring directory selection.

Tip: For those tasks needing FragMaps as inputs, "FragMap Location" needs to be a directory. The SilcsBio GUI assumes a directory path of the form `<basename>/maps/`. Select `<basename>` for your "FragMap Location". The GUI will automatically look for the `maps/` subdirectory and load FragMaps from that subdirectory. It will also automatically load the protein pdb file if one is located in the `<basename>` directory. It may take a few seconds for the GUI to download your input FragMaps if they are not on localhost.

3.3 SILCS simulation from the GUI

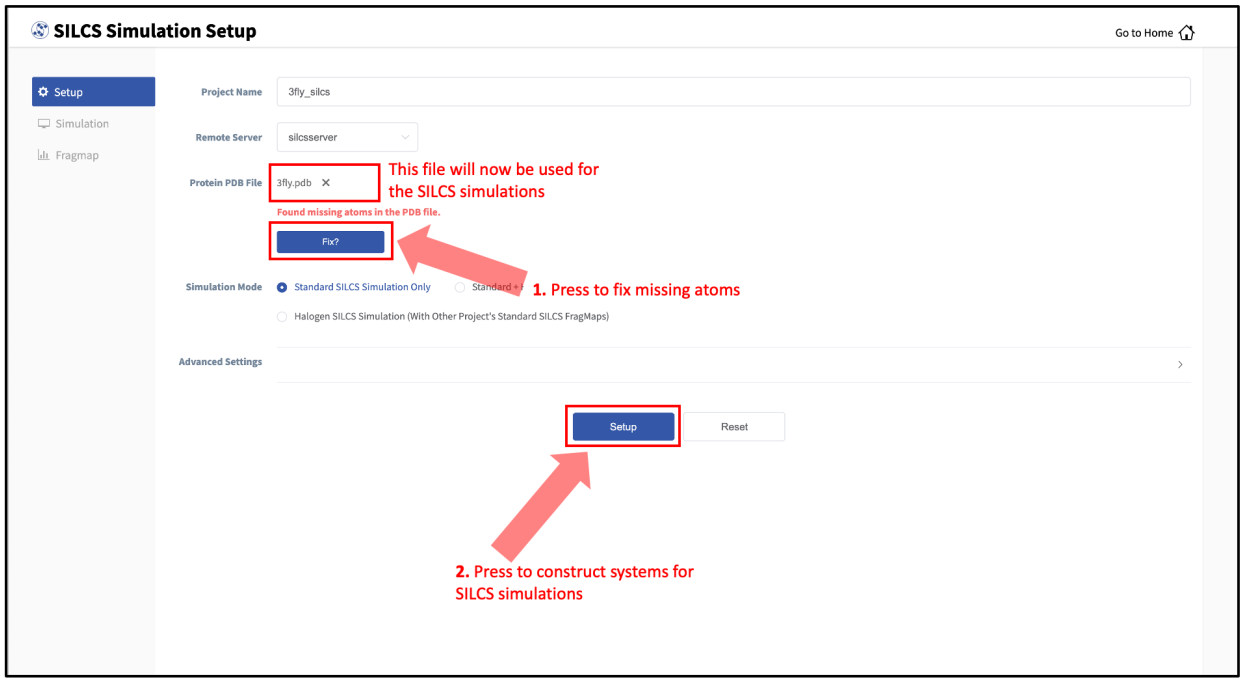
To begin a new SILCS project, follow these steps:

1. Select *New SILCS project* from the Home page.
2. Enter a project name and select the remote server where the SILCS jobs will run. Input and output files from the SILCS jobs will be stored on this server based on your choice of "Project Directory" during the Remote server setup process.

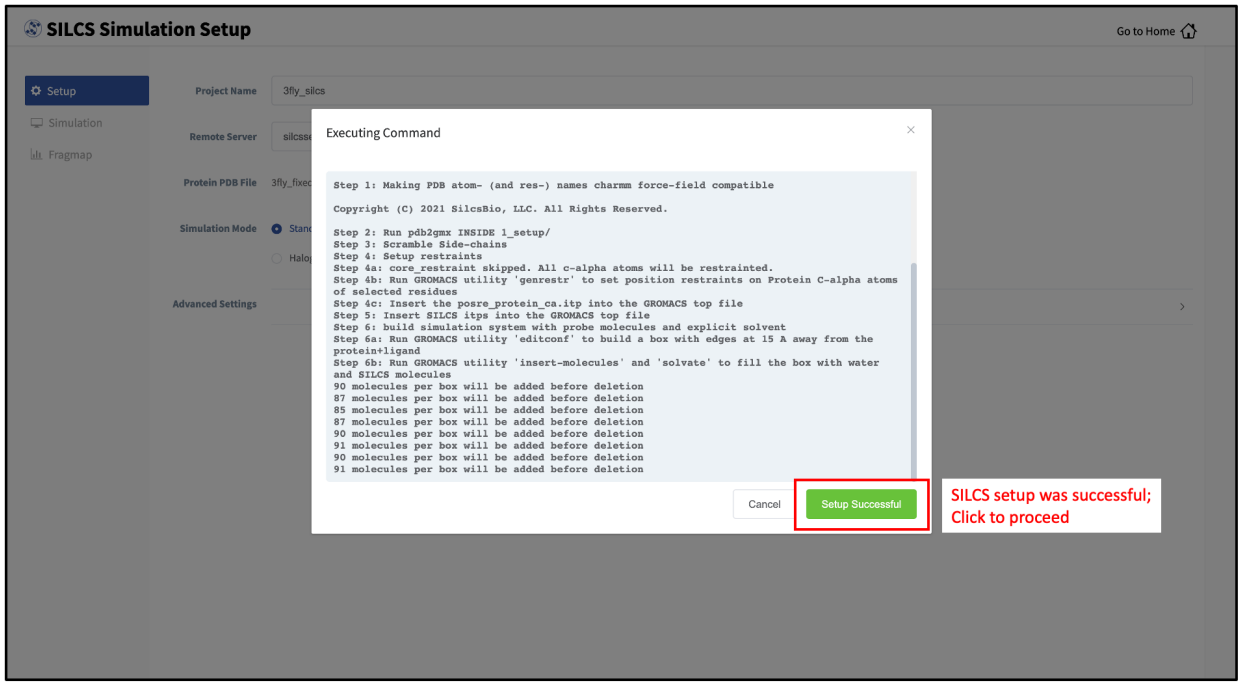


Next, select a protein PDB file. As described in File and directory selection, choose a file from your local computer ("localhost") or from any server you had configured through the Remote server setup process. We recommend cleaning your input PDB file before use, incuding keeping only those protein chains that are necessary for the simulation, removing all unnecessary ligands, renaming non-standard residues, filling in missing atomic positions, and, if desired, modeling in missing loops.

If the GUI detects missing non-hydrogen atoms, non-standard residue names, or non-contiguous residue numbering, it will inform the user and provide a button labeled "Fix?". If this button is clicked, a new PDB file with these problems fixed and with `_fixed` added to the base name will be created and used. Press the "Setup" button at the bottom of the page. The GUI will contact the remote server and perform the SILCS GCMC/MD setup process.

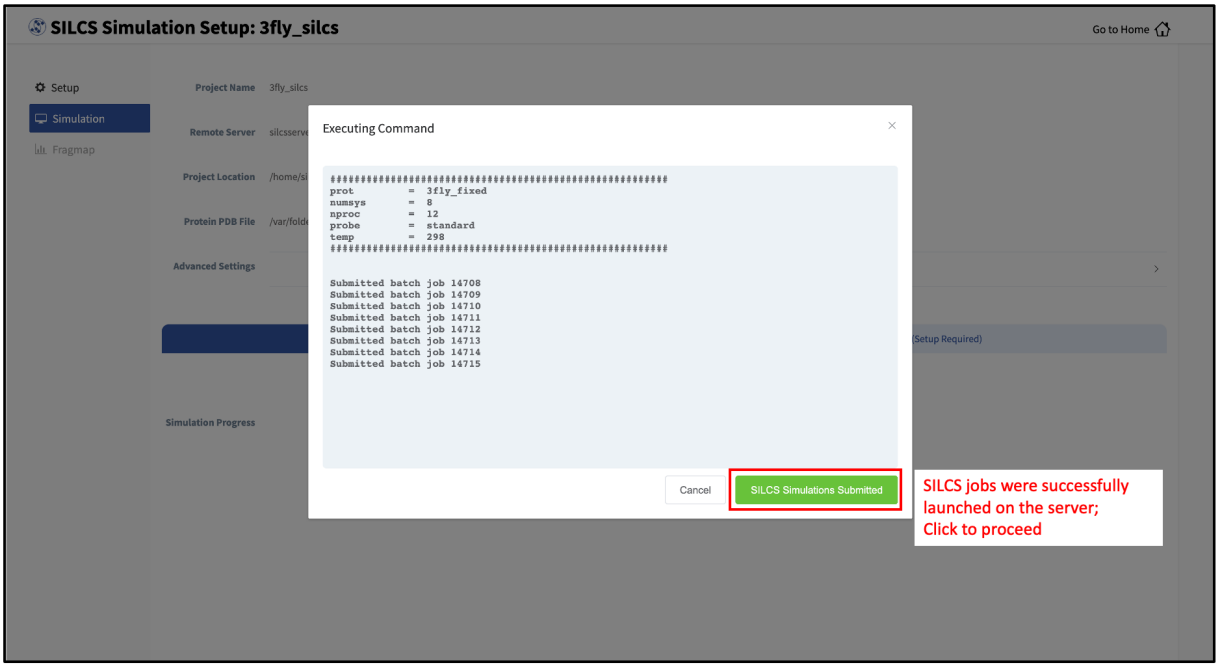


3. During setup, the program automatically performs several steps: building the topology of the simulation system, creating metal-protein bonds if metal ions are found, rotating side chain orientations to enhance sampling, and putting probe molecules around the protein. To complete the entire process may take up to 10 minutes depending on the system size. A green "Setup Successful" button will appear once the process has successfully completed. Press this button to go to the next step.

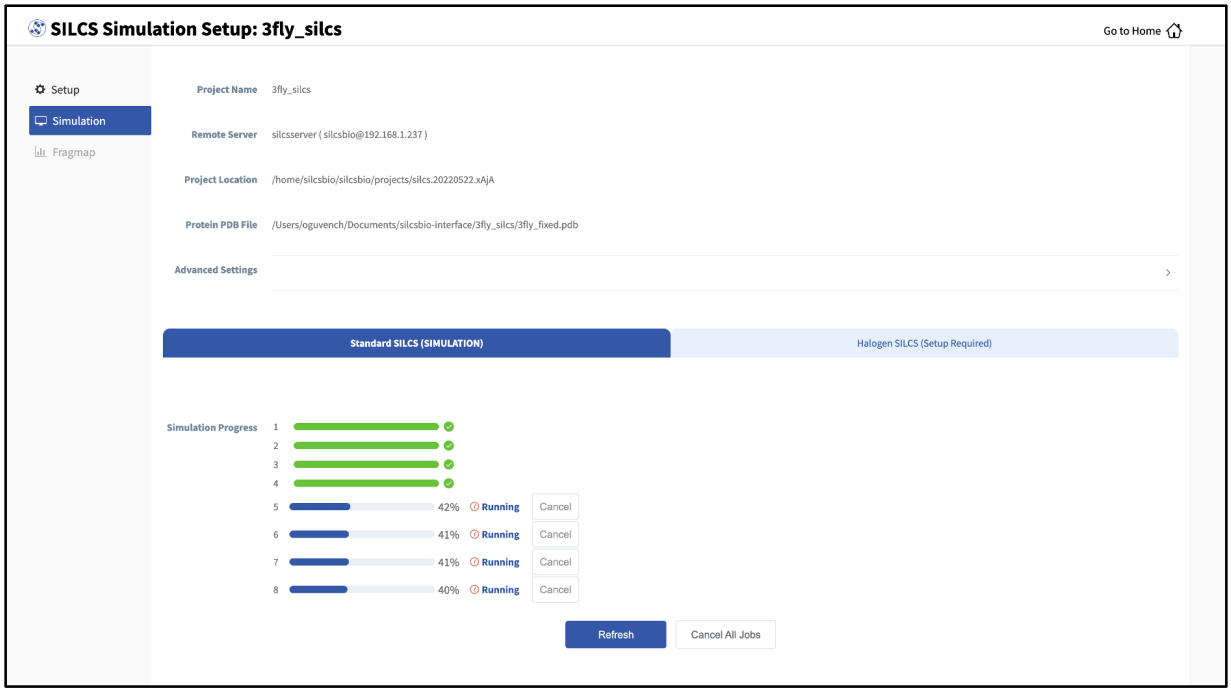


4. Your SILCS GCMC/MD simulation can now be started by clicking the "Run SILCS Simulation" button. Before running, you may wish to double check that you have chosen the desired file folder on the remote server and that it has 100+ GB of storage space.

Compute jobs will be submitted to the queueing system on the server.



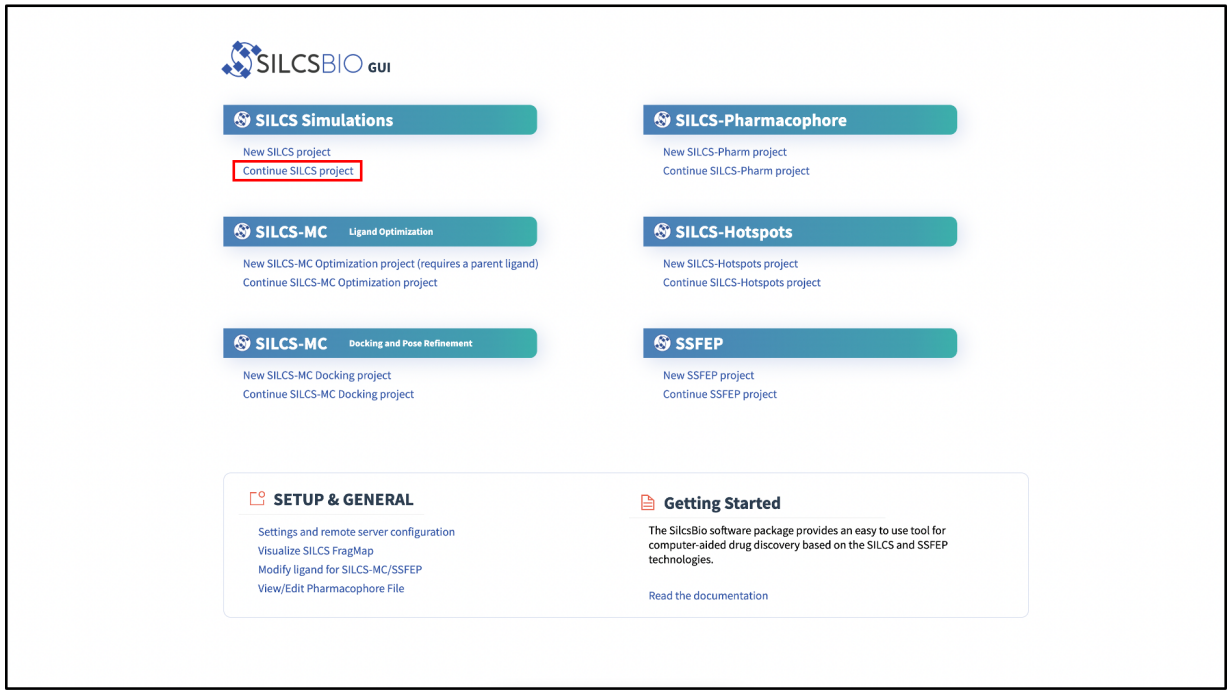
The status of each job is shown next to its progress bar: “Q” for queued and “R” for running. When a job successfully completes, its progress bar will turn green.



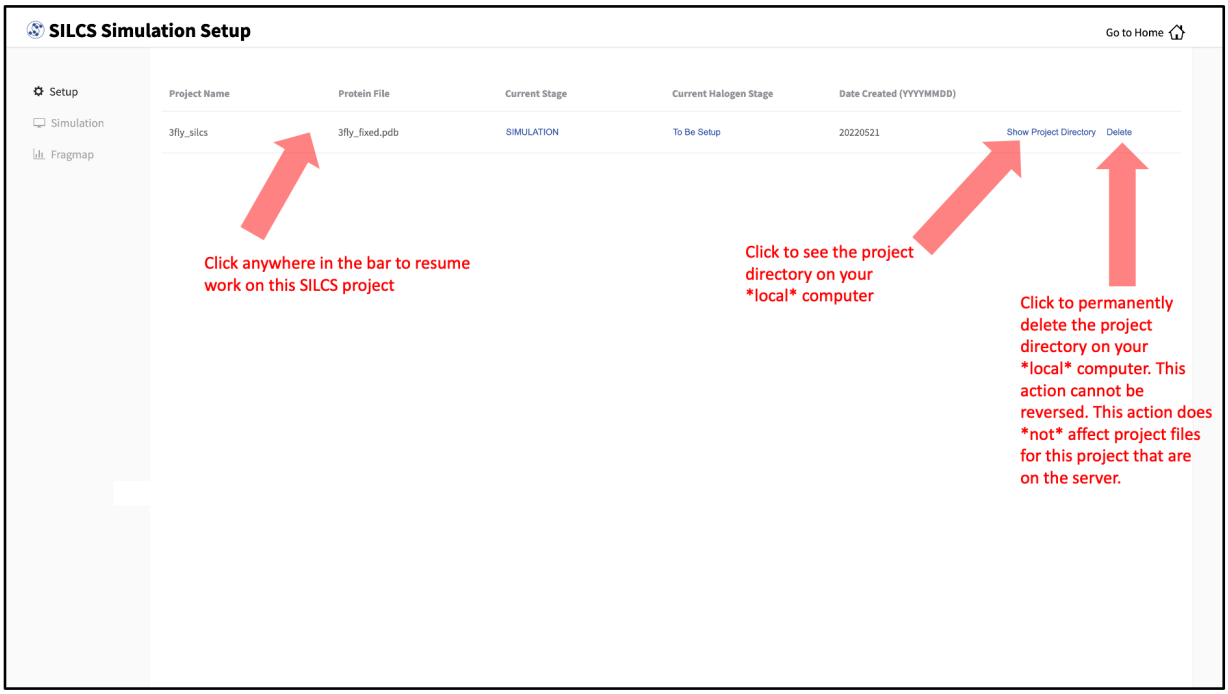
Once your jobs are in progress, in either the queued or running state, you may safely quit the SilcsBio GUI or go back to the Home page to do other tasks.

If a job encounters an error or you cancel it before it finishes, its progress bar will turn red. “Restart” will appear next to the progress bar. Pressing “Restart” will resubmit the job to the queue and continue from the last cycle of the calculation so that previous progress on that job is not lost.

5. To see a full listing of all of your projects, select *Continue SILCS project* from the Home page. This will show the complete list of all SILCS projects you have set up on the local machine where you are currently running the SilcsBio GUI, as well as the status of each project.



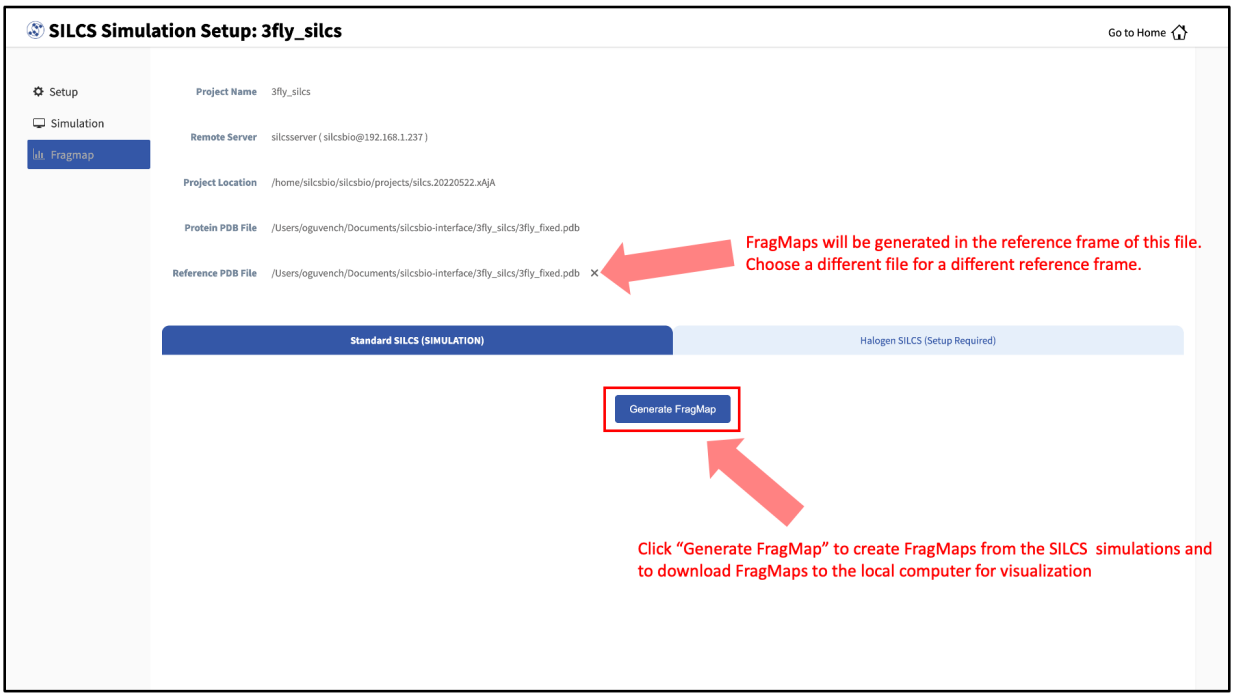
To resume work on a project or check the status of associated compute jobs you previously started, simply click the project name in the list.



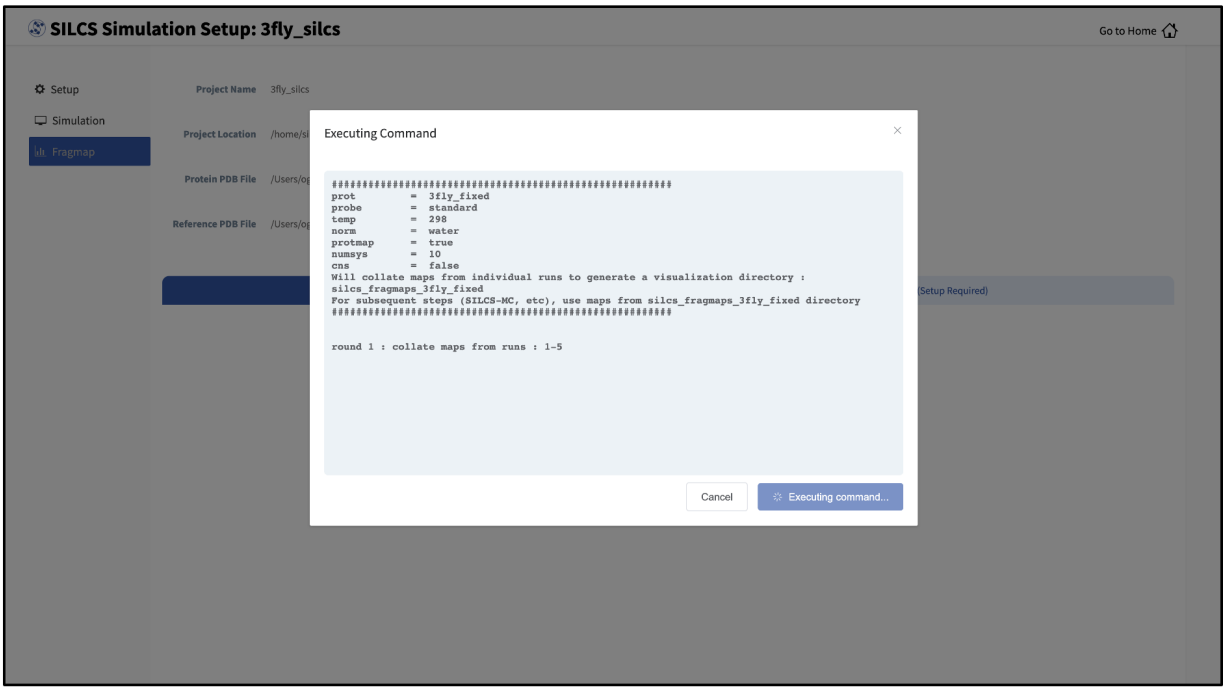
6. Once your SILCS project compute jobs are finished, the GUI can be used to create FragMaps and visualize them. Green progress bars indicate successful job completion. Once all progress bars are green, the "Prepare FragMap" button will appear.

Press this button, and on the next screen, confirm your "Reference PDB File." FragMaps will be created in the coordinate reference frame of this file. Generally, this reference PDB file is the same as the protein PDB file. A different file with the protein in a different orientation can be selected if you want to generate FragMaps relative to that orientation. Click "Generate FragMap" to generate FragMaps and download them to the local computer for visualization.

Tip: If you plan to compare FragMaps from two different protein structures, you will want to generate them with the same orientation. In that case, pre-align your input structures with each other and use these aligned coordinates as your Reference PDB Files.



Processing the GCMC-MD trajectories will take 10-20 minutes, with the GUI providing updates on progress from the server during the process.



Once completed, the GUI will automatically copy the files from the server to the local computer and load them for visualization. Please see *Visualizing SILCS FragMaps* for

details on how to use the SilcsBio GUI, as well as external software (MOE, PyMol, VMD), to visualize SILCS FragMaps.

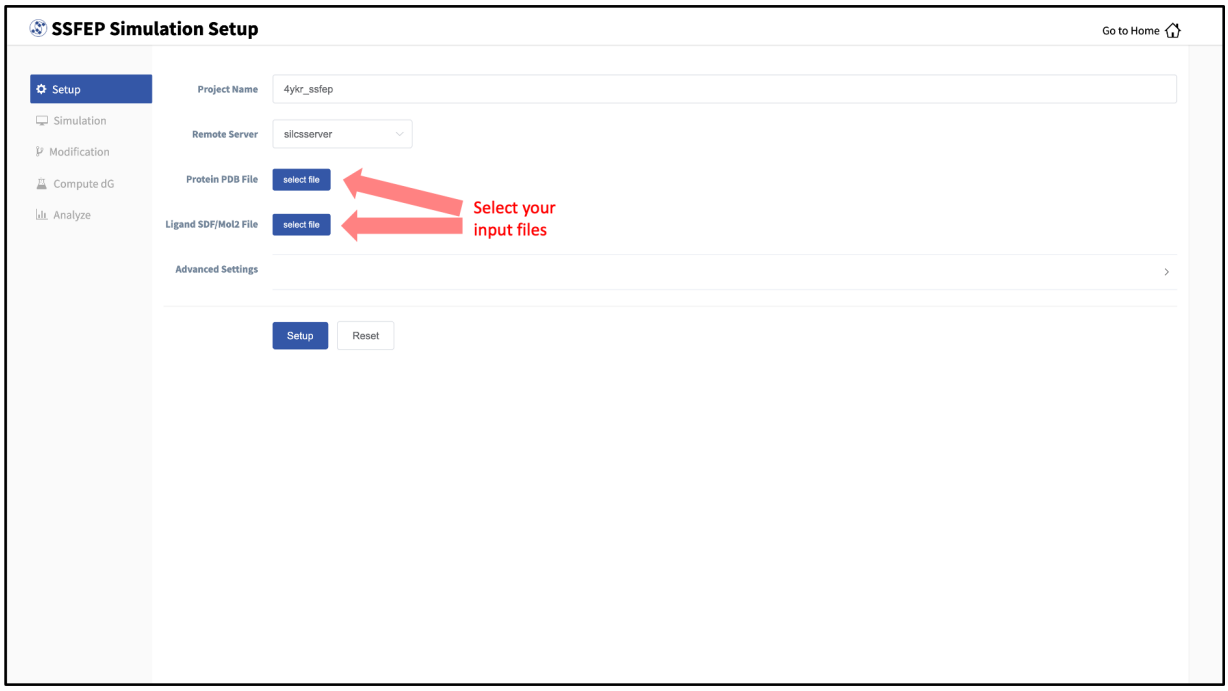
SILCS FragMaps are the basis for a host of functionality. The FragMaps can be used for optimization of a parent ligand (*SILCS-MC: Ligand Optimization*), docking of ligands and refinement of existing docked poses (*SILCS-MC: Docking and Pose Refinement*), creation of pharmacophore models (*SILCS-Pharm: Receptor-Based Pharmacophore Models from FragMaps*), and detection of hotspots and fragment-based drug design (*SILCS-Hotspots: Fragment Binding Sites Including Allosteric Sites*).

For additional details on SILCS, please see *SILCS: Site Identification by Ligand Competitive Saturation*.

3.4 SSFEP simulation from the GUI

To begin a new SSFEP project, follow these steps:

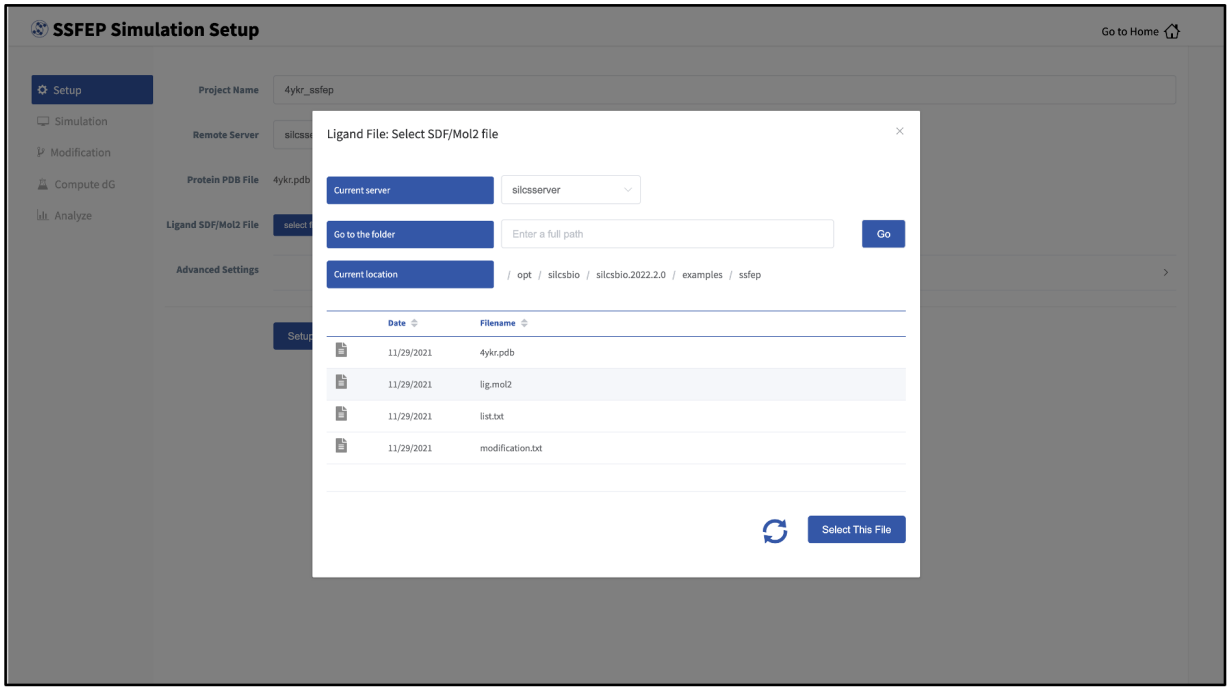
1. Select *New SSFEP project* from the Home page.
2. Enter a project name, and select the remote server where compute jobs will run. Typical SSFEP simulations produce output files in excess of 20 GB, so please select a project location file folder with appropriate storage capacity.



Next, select a protein PDB file and a ligand file as described in File and directory selection. The ligand should be aligned to the binding pocket in the accompanying protein PDB

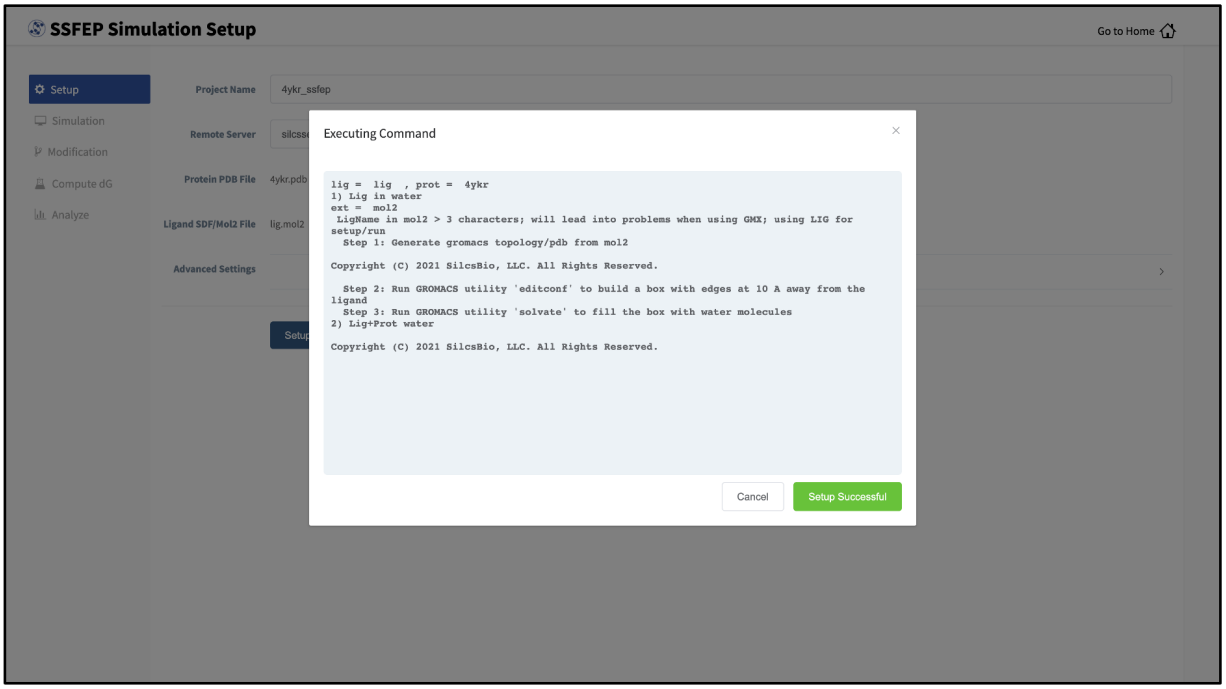
file. We recommend cleaning your PDB file before use, incuding keeping only those protein chains that are necessary for the simulation, removing all unnecessary ligands, renaming non-standard residues, filling in missing atomic positions, and, if desired, modeling in missing loops.

If the GUI detects missing non-hydrogen atoms, non-standard residue names, or non-contiguous residue numbering, it will inform the user and provide a button labeled "Fix?". If this button is clicked, a new PDB file with these problems fixed and with `_fixed` added to the base name will be created and used in the SSFEP simulation.

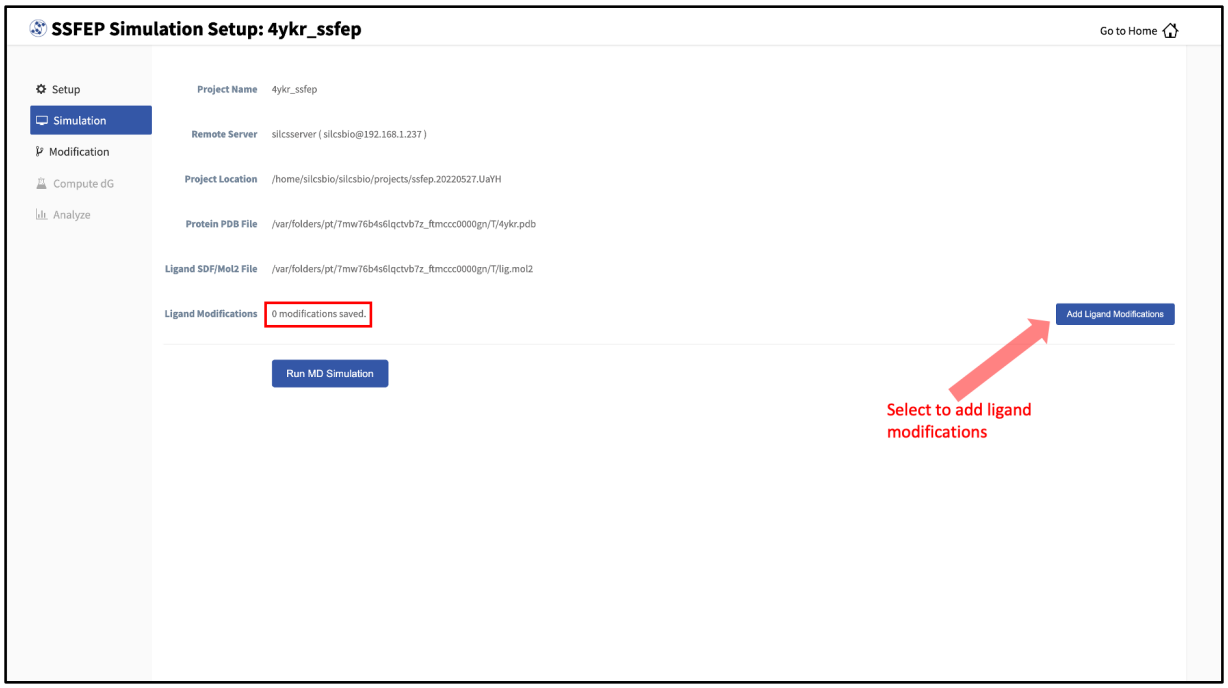


3. Once all information is entered correctly, press the "Setup" button at the bottom of the page. The GUI will contact the remote server and perform the SSFEP setup process.

   During setup, the program automatically performs several steps including building the topology of the simulation system and creating metal-protein bonds if metal ions are found. To complete the entire process may take up to 10 minutes depending on the system size. A green "Setup Successful" button will appear once the process has successfully completed. Press this button to proceed.

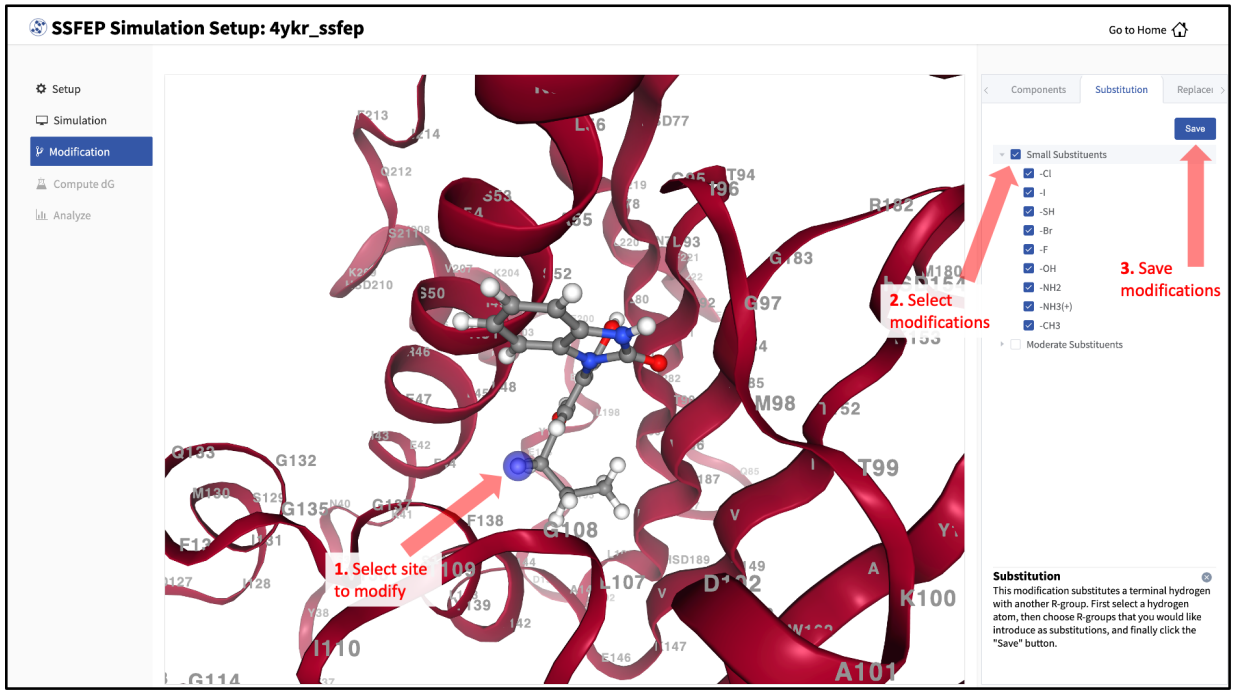


4. You can now prepare your ligand modifications with the "Add Ligand Modifications" button.



There are two major modification types, Substitution and Replacement, available in the GUI. Substitution is used to substitute a hydrogen with another functional group. Replacement is used to replace an atom in a ring with another functional group that preserves the ring.

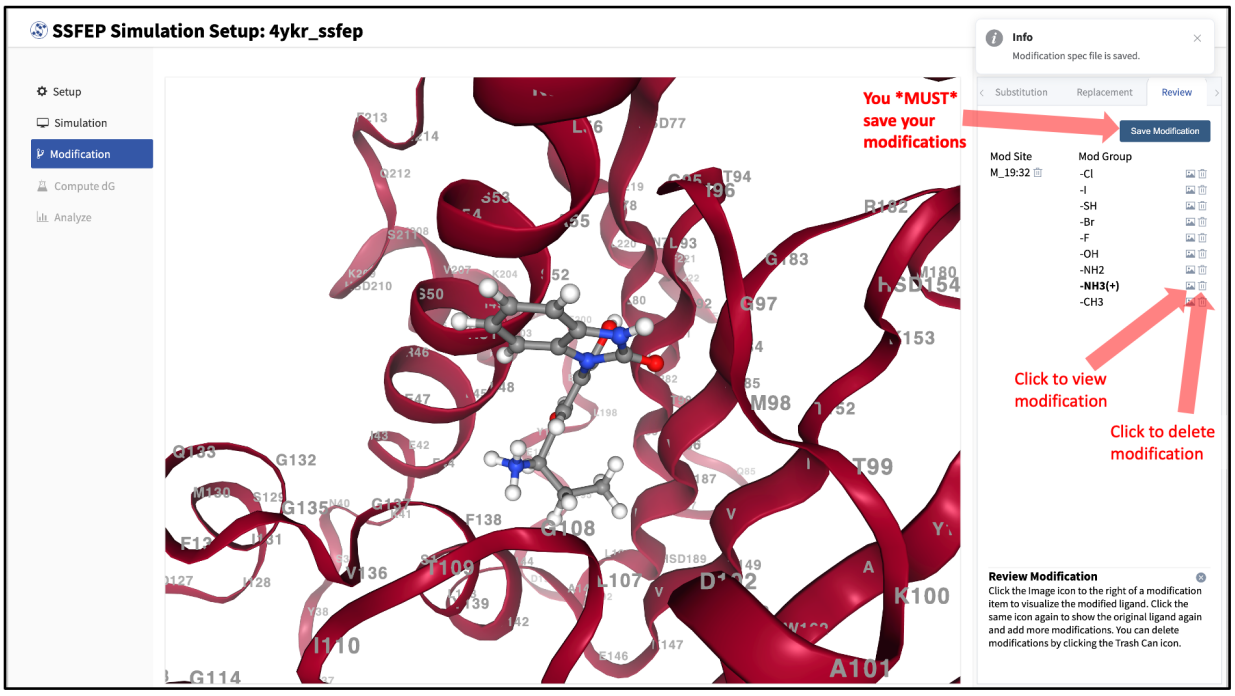
In the visualization window, select the atom to be modified. Then, select your desired modifications from the "Substitution" or the "Replacement" tab in the right-hand panel. Pressing the "Save" button in the panel will update your list of modifications.



The list of modification types in the GUI covers a broad range of chemical functionality. Custom modifications can be made using the Command Line Interface (CLI) as detailed in *SSFEP: Single Step Free Energy Perturbation*.

6. Use the "Review" tab to confirm your desired modifications.

   Valid modifications will have a small Image icon as well as a small Trash Can icon. Clicking on the Image icon will show the modification in the center panel. Clicking on it again will show the parent ligand. Clicking on the Trash Can icon will delete the proposed modification from your list. You can go back to the "Substitution" and "Replacement" tabs to add to your list. Once you have completed your list of modifications, **you must press the "Save Modification" button in the "Review" tab to actually save the list of modifications for your project**.

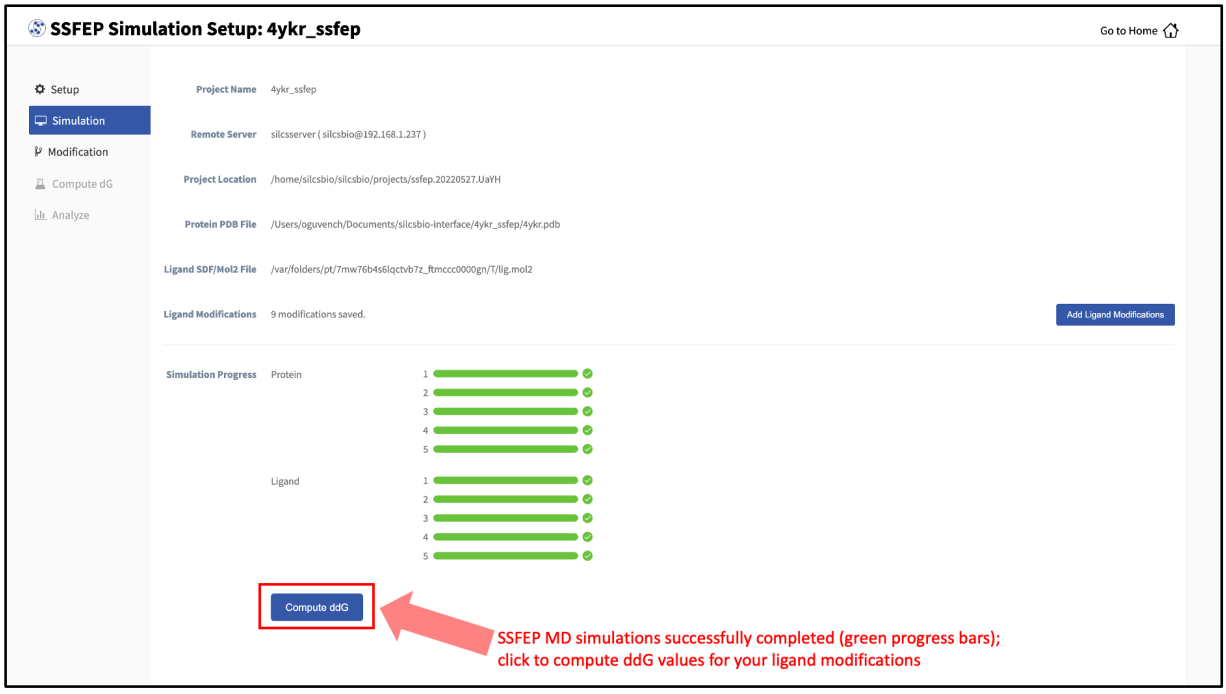


4. Your SSFEP simulation can now be started by selecting "Simulation" from the left-hand pane and then clicking the "Run MD Simulation" button in the resulting screen. Before running, you may wish to double check that you have chosen the desired file folder on the remote server and that it has 20+ GB of storage space. There are two parts to SSFEP: a compute-intensive MD simulation and a very rapid $\Delta\Delta G$ calculation. The compute-intensive MD may take several hours, and is done only once. The rapid $\Delta\Delta G$ calculation part relies on the MD results and is able to test thousands of functional group modifications to your parent ligand in under an hour. Should you wish to test additional modifications to your parent ligand at a later time in the project, there is no need to re-run the compute-intensive MD, which makes SSFEP a very efficient method.

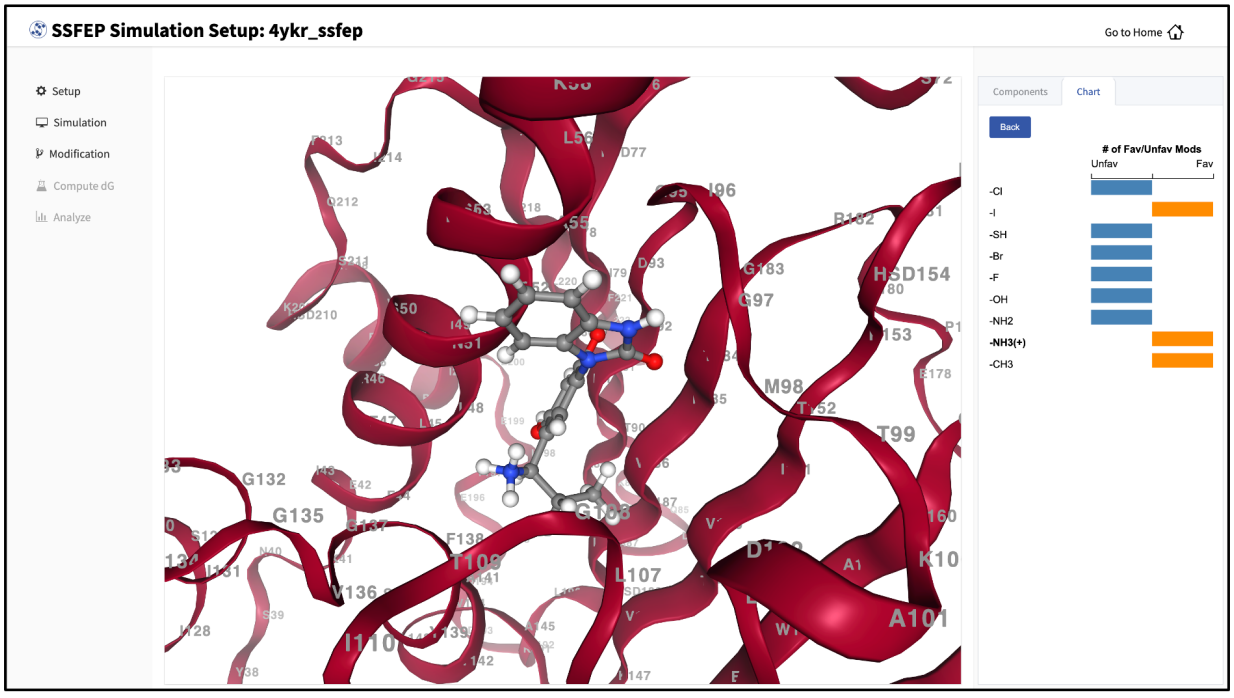
10 compute jobs will be submitted to the queueing system (five for the ligand and five for the protein:ligand complex) on your remote server. Job progress will be displayed in this same window. The status of each job is shown next to its progress bar: "Q" for queued and "R" for running, At this point, your jobs are in progress and you may safely quit the SilcsBio GUI or go back to the Home page to do other tasks.

If a job encounters an error and does not finish, a "restart" button will appear next to the status. If the restart button is used, the job will be resubmitted to the queue and continue from the last cycle of the calculation.

8. Once the MD simulation stage has finished, use the GUI to automatically analyze your list of modifications and create a chart.



SSFEP is designed to evaluate small modifications and results are best interpreted qualitatively. Therefore GUI-created charts indicate the predicted change in direction of the binding affinity relative to the parent ligand.



For additional details on SSFEP, please see *SSFEP: Single Step Free Energy Perturbation*.

CHAPTER

FOUR

COMMAND LINE INTERFACE QUICKSTART

This chapter provides a step-by-step introduction on how to use the SilcsBio Command Line Interface (CLI). Example commands assume a Bash shell is being used.

4.1 SILCS simulation from the command line

1. Set the environment variables required to access the software:

```
export GMXDIR=<gromacs/bin>
export SILCSBIODIR=<silcsbio>
```

2. Set up the SILCS simulations:

```
${SILCSBIODIR}/silcs/1_setup_silcs_boxes prot=<Protein PDB>
```

   To determine if the setup is complete check that the 10 PDB files required for the simulations are available using the command `ls 1_setup/*_silcs.*.pdb`

3. Submit the SILCS GCMC/MD jobs to the queue:

```
${SILCSBIODIR}/silcs/2a_run_gcmd prot=<Protein PDB>
```

   This will submit 10 jobs to the queue. To check job progress, use:

```
${SILCSBIODIR}/silcs/check_progress
```

   This will provide a summary list of the SILCS jobs consisting of the full job path, the job number, the task ID, the job status, and the current/total number of GCMC/MD cycles. Job status values are: Q queued, R running, E successfully completed, F failed, NA not submitted.

4. When the GCMC/MD jobs are finished, generate FragMaps:

```
${SILCSBIODIR}/silcs/2b_gen_maps prot=<Protein PDB>
```

This command will submit 10 jobs to the queue for calculating the occupancy maps from individual runs. Once they are done, use the following command to create the FragMaps:

```
${SILCSBIODIR}/silcs/2c_fragmap prot=<Protein PDB>
```

This command will create a `silcs_fragmap_<Protein PDB>` folder, which contains the final FragMap files as well as scripts for visualization with external software. Please see *Visualizing SILCS FragMaps* for detailed instructions on visualizing your SILCS FragMaps.

For additional details, please see *SILCS: Site Identification by Ligand Competitive Saturation*.

4.2 SSFEP simulation from the command line

1. Set the environment variables required to access the software:

```
export GMXDIR=<gromacs/bin>
export SILCSBIODIR=<silcsbio>
```

2. Set up the SSFEP simulations:

```
${SILCSBIODIR}/ssfep/1_setup_ssfep lig=<Ligand Mol2/SDF file> prot=
↪<Protein PDB>
```

To determine if the setup is completed check that the PDB files required for the simulations are available using the command `ls 1_setup/*/*_gmx_wat.pdb`. The listing should show PDB files for the ligand alone and for the protein/ligand complex.

3. Submit the SSFEP MD simulation jobs to the queueing system:

```
${SILCSBIODIR}/ssfep/2_run_md_ssfep lig=<Ligand Mol2/SDF file>␣
↪prot=<Protein PDB>
```

This will submit 10 jobs to the queue, 5 for the protein:ligand complex and 5 for the ligand. To check job progress, use:

```
${SILCSBIODIR}/ssfep/check_progress
```

This will provide a summary list of the SSFEP jobs consisting of the full job path, the job number, the task ID, the job status, and the current/total number of SSFEP cycles. Job status values are: Q queued, R running, E successfully completed, F failed, NA not submitted.

4. When the SSFEP MD simulation jobs are finished, use the ligand modification files to submit the $\Delta\Delta G$ calculations:

```
${SILCSBIODIR}/ssfep/3a_setup_modifications lig=<Ligand Mol2/SDF␣
↪file> prot=<Protein PDB> mod=<modification file>
```

This command will submit 10 jobs, each of which processes one of the MD trajectories for all modifications in the modification file. Depending on the number and sizes of the modifications, this step may take minutes to several hours to complete.

Once completed, use the following command to collate the results for all of the modifications:

```
${SILCSBIODIR}/ssfep/3b_calc_ddG_ssfep mod=<modification file>
```

This command will create a `lig_decor.csv` file, which contains the free energy change for each modification relative to the parent ligand. SSFEP is designed to evaluate small modifications and results are best interpreted qualitatively. Therefore it is recommended that only the sign of the change, and not the magnitude, be used to inform decision making: values < 0 indicate a modification predicted to be favorable.

For additional details, please see *SSFEP: Single Step Free Energy Perturbation*.

CHAPTER

FIVE

SILCSBIO SOFTWARE INSTALLATION

5.1 Minimum hardware requirement

SilcsBio software requires relatively robust computational resources for the molecular dynamics (MD) components of the SILCS and SSFEP protocols. For example, computing SILCS FragMaps for a 35 kDa target protein takes 80-90 hours of walltime when run in parallel on ten compute nodes, each equipped with 8 3-GHz CPU cores. The software can take advantage of GPU acceleration: the addition of a single NVIDIA GeForce RTX 2070 GPU to each node will reduce the walltime to 24-48 hours. In the case of SSFEP, walltime using these GPU-equipped nodes will be 3-4 hours.

SilcsBio software is designed to run the compute-intensive MD on a cluster using a cluster queue management system such as OpenPBS, Sun Grid Engine, or SLURM. With both SILCS and SSFEP, subsequent evaluation of relative ligand affinities takes seconds to minutes on a single CPU core, allowing for modifications to be rapidly evaluated for a large number of ligands to a given target.

For customers without ready access to an appropriate in-house computing cluster, SilcsBio offers three possible solutions. The first solution is the SilcsBio Workstation. The SilcsBio Workstation combines the SilcsBio server and GUI software with high-performance GPU-computing hardware in a quiet, sleek form factor for use in an office setting. The SilcsBio Workstation is a complete turn-key solution that is ready to plug in to a standard wall electrical socket. The second solution is for SilcsBio to perform computations as a service and supply data to the customer for subsequent in-house analysis. For this service, in the case of SILCS, SilcsBio requires only the structure of the target, and, in the case of SSFEP, only the structure of the protein-parent ligand complex. Depending on the choice of SSFEP or SILCS, no intellectual property disclosure to SilcsBio in the form of proposed chemical modifications to the parent ligand (SSFEP) or even the parent ligand itself (SILCS) is required. The third solution is for SilcsBio to assist customers with setting up their own virtual cluster using Amazon Web Services. Please contact info@silcsbio.com for additional information on these solutions.

5.2 Python 3 requirement

The SilcsBio server software requires a working Python 3 installation. We recommend using Miniconda (https://docs.conda.io/en/latest/miniconda.html) for installing Python 3. Once Python 3 is installed, you will need to install some additional Python packages. To do so, run the following command:

```
pip install -r $SILCSBIODIR/utils/python/requirements.txt
```

5.3 GROMACS requirement

Both SILCS and SSFEP molecular dynamics simulations require the GROMACS molecular dynamics software. We recommend GROMACS version 2021.4. GROMACS can be obtained at https://manual.gromacs.org/documentation/2021.4/download.html and must be installed in order to use the SilcsBio software package.

We strongly recommend building GROMACS with GPU acceleration enabled. Below is a sequence of commands for GROMACS v. 2021.4 installation with GPU acceleration:

```
cd <GROMACS source directory>
mkdir build
cd build
cmake .. -DGMX_BUILD_OWN_FFTW=on \
  -DREGRESSIONTEST_DOWNLOAD=on \
  -DGMX_GPU=CUDA \
  -DGMXAPI=off \
  -DBUILD_SHARED_LIBS=off \
  -DGMX_PREFER_STATIC_LIBS=on \
  -DGMX_BUILD_SHARED_EXE=off \
  -DCMAKE_INSTALL_PREFIX=<GROMACS install location>
make
make install
```

If your compute nodes do not have GPUs, use the following commands:

```
cd <GROMACS source directory>
mkdir build
cd build
cmake .. -DGMX_BUILD_OWN_FFTW=on \
  -DREGRESSIONTEST_DOWNLOAD=on \
  -DGMXAPI=off \
  -DBUILD_SHARED_LIBS=off \
  -DGMX_PREFER_STATIC_LIBS=on \
  -DGMX_BUILD_SHARED_EXE=off \
```

(continues on next page)

(continued from previous page)

```
  -DCMAKE_INSTALL_PREFIX=<GROMACS install location>
make
make install
```

Please refer to http://manual.gromacs.org/documentation/current/install-guide/index.html for further details.

5.4 Installing the SilcsBio server software

The SilcsBio server software package is delivered as a zip-compressed file. Unzip and place the files to an accessible location. The files have the following directory structure

```
silcsbio.$VERSION/
  data/
  examples/
  lib/
  programs/
  silcs/
  silcs-hotspots/
  silcs-mc/
  silcs-memb/
  silcs-pharm/
  ssfep/
  ssfep-memb/
  templates/
  utils/
  VERSION
```

The top-level `silcsbio.$VERSION/` folder contains software for running SILCS and SSFEP simulations. The `programs/`, `silcs*/`, and `ssfep*/` folders contain executable code, and the `templates/` folder contains templates for job handling and input scripts. Some template files may need to be edited with information for your queuing system.

If you are a system administrator, place the top-level `silcsbio.$VERSION/` folder where it can be accessed by other users, such as `/opt/silcsbio/`. If you are a single user, you may place the folder in your home directory.

For SilcsBio server software to work, the two shell environment variables `GMXDIR` and `SILCSBIODIR` need to be set. To do so, replace `<gromacs/bin>` and `<silcsbio>` with the complete file paths for the corresponding folders:

```
# bash
export GMXDIR=<gromacs/bin>
export SILCSBIODIR=<silcsbio>
```

Currently, the SilcsBio server software is compatible only with the Bash shell environment. You may insert the above environment variable settings in `.bashrc` for convenience.

5.5 Installing the SilcsBio Graphical User Interface

The SilcsBio Graphical User Interface (GUI) enables running SILCS and SSFEP simulations and analyzing results through a GUI instead of the command line. The SilcsBio GUI is available for Windows, macOS, and Linux. Please download and install the software on your local desktop or laptop computer.

In addition to providing standalone features such as FragMap visualization and ligand modification, the SilcsBio GUI can set up, launch, manage, and analyze compute-intensive SILCS and SSFEP simulations. To enable this functionality requires a simple configuration step to allow the GUI to communicate with your SilcsBio server software.

Please follow the Remote server setup process as described in *Graphical User Interface Quickstart*. Contact support@silcsbio.com if you need help with this process.

CHAPTER

SIX

FREQUENTLY ASKED QUESTIONS

6.1 I installed the software, how do I test if it is correctly installed?

Because different users have different settings and requirements for their clusters or workstations, we provide a general job handling script for you to customize to your needs.

To assist with job handling script customization, example input files are available under the `$SILCSBIODIR/examples` folder.

For SILCS, use the following commands to make sure the software is correctly installed and the job handling script is working. If you are only interested in SSFEP simulations, you may skip to the SSFEP section below.

```
mkdir -p test/silcs
cd test/silcs
cp $SILCSBIODIR/examples/silcs/p38a.pdb .
$SILCSBIODIR/silcs/1_setup_silcs_boxes prot=p38a.pdb
$SILCSBIODIR/silcs/2a_run_gcmd prot=p38a.pdb numsys=1 nproc=1
```

If this set of commands runs without error, confirm that the SILCS job is running with the `check_progress` command:

```
$SILCSBIODIR/silcs/check_progress
```

and then go ahead and stop the successfully running SILCS job:

```
$SILCSBIODIR/silcs/2a_run_gcmd cancel=true sys=1
```

and confirm it is stopped:

```
$SILCSBIODIR/silcs/check_progress
```

Otherwise, if you experienced an error with the `1_setup-silcs_boxes` step, the software is not correctly installed, whereas if you experienced an error with the `2a_run_gcmd` step, the job

handling scripts need to be edited. The job handling scripts for SILCS are:

- `templates/silcs/job_mc_md.tmpl`
- `templates/silcs/job_gen_maps.tmpl`
- `templates/silcs/pymol_fragmap.tmpl`
- `templates/silcs/vmd_fragmap.tmpl`
- `templates/silcs/job_cleanup.tmpl`

Typically the header portion of a job handling script requires editing. Please contact support@silcsbio.com if you need assistance.

For SSFEP, use the following commands to make sure the software is correctly installed and the job handling script is working.

```
mkdir -p test/ssfep
cd test/ssfep
cp $SILCSBIODIR/examples/ssfep/* .
$SILCSBIODIR/ssfep/1_setup_ssfep prot=4ykr.pdb lig=lig.mol2
$SILCSBIODIR/silcs/2_run_md_ssfep prot=4ykr.pdb lig=lig.mol2 nproc=1
```

If this set of commands runs without error, confirm that the SSFEP job is running with the `check_progress` command:

```
$SILCSBIODIR/ssfep/check_progress
```

and then go ahead and stop the successfully running SSFEP job:

```
$SILCSBIODIR/ssfep/2_run_md_ssfep cancel=true target=lig
$SILCSBIODIR/ssfep/2_run_md_ssfep cancel=true target=prot
```

and confirm it is stopped:

```
$SILCSBIODIR/ssfep/check_progress
```

Otherwise, if you experienced an error with the `1_setup_ssfep` step, the software is not correctly installed, and if you experienced an error with the `2_run_md_ssfep` step, the job handling script needs to be edited. The job handling scripts for SSFEP are:

- `templates/ssfep/job_lig_md.tmpl`
- `templates/ssfep/job_prot_lig_md.tmpl`
- `templates/ssfep/job_dG.tmpl`

Typically the header portion of a job handling script requires editing. Please contact support@silcsbio.com if you need assistance.

6.2 I don't have a cluster but I have a GPU workstation. What can I do?

You may be able to practically run the SilcsBio software if your GPU workstation has sufficient resources. An appropriate workstation may have at least 24 CPU cores, 4 GPUs, 64 GB of RAM, and 10 TB of disk space. Installing a job queueing system, such as the Slurm Workload Manager, will allow the SilcsBio server software to run on the workstation.

The SilcsBio Workstation is a turn-key GPU workstation hardware+software solution developed by SilcsBio that comes with all necessary software pre-installed. The SilcsBio workstation has a quiet, sleek form factor for use in an office setting and comes ready to plug in to a standard wall electrical socket. Please contact info@silcsbio.com for details.

6.3 I compiled my GROMACS with MPI and my job is not running

Please contact us so we can repackage the files with the appropriate command using `mpirun` instead.

Alternatively, you may edit the job handling script to edit the GROMACS command.

For example, the `mdrun` command is specified at the top of `templates/ssfep/job_lig_md.tmpl` file:

```
mdrun="${GMXDIR}/gmx mdrun -nt $nproc"
```

You may edit this to

```
mdrun="mpirun -np $nproc ${GMXDIR}/gmx mdrun"
```

6.4 GROMACS on the head node does not run because the head node and compute node have different operating systems

In this case, we recommend compilng GROMACS on the head node and compiling `mdrun` only on the compute node.

Building only `mdrun` can be done by supplying the `-DGMX_BUILD_MDRUN_ONLY=on` keyword to the `cmake` command in the build process. Once the `mdrun` program is built, place it in the same $GMXDIR folder. Now template files needs to be edited to use the `mdrun` command properly on the compute node.

For example, the `mdrun` command is specified at the top of the `templates/ssfep/job_lig_md.tmpl` file:

```
mdrun="${GMXDIR}/gmx mdrun -nt $nproc"
```

You may edit this to

```
mdrun="${GMXDIR}/mdrun -nt $nrpoc"
```

6.5 I get the "error while loading shared libraries: libcudart.so.8.0: cannot open shared object file: No such file or directory" message during my setup

If you encounter this error, the most likely reason is that GROMACS was compiled on a machine having a GPU whereas the current machine where the command is being executed does not have a GPU.

It may be possible that the necessary library is already available for the machine even though it does not have a GPU. So, check if the `libcudart.so` file exists on the current machine. The most likely place is `/usr/local/cuda/lib64`. If the file exists in that location, add that path to your `LD_LIBRARY_PATH` environment variable..

If the library is not available on the current machine, we recommend following FAQ #4 to compile GROMACS and `mdrun` separately.

6.6 I want to modify the force field and topology files for SILCS simulation

As an example, if there is the need to add extra bonds that are not present in the standard force field definitions, this is the procedure to make the necessary modifications. For example, some proteins contain two metals ions adjacent to each other, in which case it may be useful to place a bond connecting the ions. The protein 3bi0 has two Zn ions adjacent to each other, and adding such a bond is useful to restrain the distance between the ions to that in the crystal structure. Please refer to the GROMACS documentation regarding to modify the .top and ffbonded.itp files.

First, run the following command. This will copy the basic force field to and generate an initial topology file in `1_setup/`, allowing you to edit them.

```
$SILCSBIODIR/silcs/1_setup_silcs_boxes prot=<prot PDB>
```

Then edit the force field parameter file `1_setup/charmm36.ff/ffbonded.itp`.

If you want to modify the topology file (e.g. to add an explicit bond between the ions), copy `1_setup/<prot>_gmx.top.1.bak` to `1_setup/<prot>_gmx.top`. Then edit `<prot>_gmx.top` and add the desired bond between the two ions in the `[ bond ]` list.

Once the files are edited, re-run the 1_setup command with the `skip_pdb2gmx=true` keyword. This will preserve your edits and create the necessary files to run the SILCS simulations.

```
$SILCSBIODIR/silcs/1_setup_silcs_boxes prot=<prot PDB> skip_
↪pdb2gmx=true
```

Once this completes, run the `$SILCSBIODIR/silcs/2a_run_gcmd` script to initiate your SILCS simulations.

6.7 I want to visualize FragMaps using MOE

By default, SILCS FragMaps are in the MAP grid file format. However, this file format is not supported in MOE. Please see *FragMaps in MOE* for detailed instructions on creating FragMaps in a MOE-compatible format.

6.8 How do I handle phosphorylated amino acids?

The following phosphorylated amino acids are supported:

- pSer
- pThr
- pTyr

To create a phosphorylated amino acid, rename that amino acid in your input pdb file as follows:

- SER => SP1 or SP2
- THR => THP1 or THP2
- TYR => TP1 or TP2

The number at the end of the amino acid name refers to whether the phosphate group has mono- or divalent charge.

6.9 What if my protein has a glycan attached to it?

While setting up a glycan-containing protein directly from a PDB file is not currently supported, you can set up your simulation system for SILCS if you have a PSF file created with the CHARMM36 force field.

An example can be found in the `$SILCSBIODIR/examples/glycan` folder. Running the `setup.sh` script in that directory will run the example and create a folder named `1_setup`.

For your own system, copy the `gromacs` folder and `setup.sh` file and edit the copied `setup.sh` file before running it:

```
psffile="psf/step1_pdbreader.psf" # PSF file
pdbfile="psf/step1_pdbreader.pdb" # PDB file
prefix="5vgp" # prefix for the SILCS simulation
```

6.10 What happens when I set up SILCS simulations with an input structure containing a metal ion?

SILCS simulation supports a variety of metal ions, including calcium, copper, iron, magnesium, manganese, nickel, and zinc. If an ion in the input structure is located close to protein atoms (~ 3A), the setup script will automatically create covalent linkages between the metal ion and nearby protein residues so as to ensure the coordination structure is maintained throughout the SILCS GCMC/MD simulations.

6.11 My protein contains iron and I want to set a +3 charge state

By default, the SILCS setup assigns a +2 charge to iron. If you want to change the charge of the iron ion, change the residue name of the ion in the input PDB to FE3. The setup script will then assign a +3 charge to that ion.

6.12 How do I fit my membrane protein in a bilayer as suggested by the OPM server?

1. Prepare your bare membrane protein with Alphafold, MOE etc. as <prot PDB>
2. Upload the <prot PDB> to the OPM server (PPM 3.0) and get the output.
3. Download the <OPM output pdb>, open it with PyMOL or VMD, and determine the required translation along the z-axis:

   PyMOL

   - Select protein atoms: `sele all and polymer`
   - Calculate center of mass (COM): `centerofmass sele`

- Copy the **Z-coordinate_of_COM** and go to the next step

VMD

- Open VMD Main >> Extensions >> TkConsole
- Create selection with protein atoms: `set sel1 [atomselect top "all and protein"]`
- Calculate center of mass (COM): `set com1 [measure center $sel1 weight mass]`
- Copy the **Z-coordinate_of_COM** and go to the next step

4. Align <prot PDB> to <OPM output pdb> for subsequent use OR extract protein from <OPM output pdb> for subsequent use.
5. Run setup with the additional `offset_z`:

```
$SILCSBIODIR/silcs-memb/1a_fit_protein_in_bilayer prot=<aligned␣
↪prot PDB> orient_principal_axis=false offset_z=<Z-coordinate_of_
↪COM>
```

6.13 How do I calculate the difference between two sets of SILCS FragMaps?

Copy the two silcs_fragmaps_xxx directories you wish to use to a new directory.

Please make sure:

- *Either* both sets of FragMaps have the same grid setup, which means the header of the `silcs_fragmaps_xxx/maps/*.map` files are the same.
- *Or* both sets of FragMaps have were created relative to proteins having the same rotational and translational alignments. If the input proteins were not aligned to each other prior to running SILCS, you must use the **"ref="** option with the `2b_gen_maps` command to ensure alignment. A map cutting algorithm will be used to make the FragMaps compatible, with the smaller dimensions of the two grids used to calculate the difference maps.

Run the following command:

```
$SILCSBIODIR/utils/calc_difference_maps.sh [silcs_fragmaps_#1] [silcs_
↪fragmaps_#2]
```

The difference maps will be saved in a directory `DIFF_MAPS_fragmap#1_VS_fragmap#2` and can be visualized in the same way as regular FragMaps (*Visualizing SILCS FragMaps*).

Tip: In the difference maps, if the value at a grid point is negative, the grid free energy (GFE) of the grid point in fragmap_#1 is more favourable than in fragmap_#2 and if it is positive, then fragmap_#2 is more favourable than fragmap_#1.

Note: The difference maps are intended ONLY for visualization and NOT for any quantitative calculations.

6.14 How do I include a covalently bound ligand/cofactor in SILCS simulations?

While creating a covalent bond between your protein and the ligand/cofactor during SILCS setup is not supported, it is possible to achieve a good approximation of the structure by using positional restraints to maintain relative geometries during the SILCS simulation. To do so, provide the ligand/cofactor as an individual ligand using the option **lig=lig.mol2** during setup. The mol2 file should contain only the ligand/cofactor molecule and must have all hydrogen atoms and a appropriate three-dimensional coordinates that provide a reasonable internal geometry and place it correctly relative to the protein. Weak position restraints will be automatically added on the non-hydrogen atoms of ligand/cofactor, so it will be positionally restrained. (Note: you will also need to set the option **scramblesc=false** when running `1_setup_silcs_boxes` in order to keep protein sideschain conformations from being scrambled.)

```
$SILCSBIODIR/silcs/1_setup_silcs_boxes prot=<prot PDB> lig=lig.mol2␣
↪scramblesc=false
```

However, the amino acid sidechain to which the ligand/cofactor is meant to be covalently bound will be unrestrained. To address this, you will need to manually add weak position restraints on non-hydrogen atoms of the sidechain. To do this, edit the `1_setup/posre_protein_ca.itp` file which restraints C-alpha atoms and append it with the non-hydrogen atoms from your sidechain.

```
; position restraints for (atomname_CA_or_atomname_"C1'")

[ position_restraints ]
;  i funct       fcx        fcy        fcz
   5    1     50.208     50.208     50.208
  20    1     50.208     50.208     50.208
  46    1     50.208     50.208     50.208
  58    1     50.208     50.208     50.208
.
```

(continues on next page)

(continued from previous page)

```
.
.
5571    1     50.208     50.208     50.208
5589    1     50.208     50.208     50.208
5603    1     50.208     50.208     50.208
5617    1     50.208     50.208     50.208
; **ADD YOUR SIDECHAIN ATOMS BELOW**
xxxx    1     50.208     50.208     50.208
xxxx    1     50.208     50.208     50.208
xxxx    1     50.208     50.208     50.208
xxxx    1     50.208     50.208     50.208
```

Please visualize the `1_setup/<prot>_silcs.1-10.pdb` files to make sure the ligand and sidechain are correctly prepared before you run the `2a_run_gcmd` command.

CHAPTER

SEVEN

SILCS: SITE IDENTIFICATION BY LIGAND COMPETITIVE SATURATION

7.1 SILCS simulation background

The design of small molecules that bind with optimal specificity and affinity to their biological targets, typically proteins, is based on the idea of complementarity between the functional groups in a small molecule and the binding site of the target. Traditional approaches to ligand identification and optimization often employ the one binding site/one ligand approach. While such an approach might be straightforward to implement, it is limited by the resource-intensive nature of screening and evaluating the affinity of large numbers of diverse molecules.

Functional group mapping approaches have emerged as an alternative, in which a series of maps for different classes of functional groups encompass the target surface to define the binding requirements of the target. Using these maps, medicinal chemists can focus their efforts on designing small molecules that best match the maps.

Site Identification by Ligand Competitive Saturation (SILCS) offers rigorous free energy evaluation of functional group affinity pattern for the entire 3D space in and around a protein [7]. The SILCS method yields functional group free energy maps, or FragMaps, which are precomputed and then used to rapidly facilitate ligand design. FragMaps are generated by molecular dynamics (MD) simulations that include protein flexibility and explicit solvent/solute representation, thus providing an accurate, detailed, and comprehensive set of data that can be used in database screening, fragment-based drug design, and lead optimization of small molecules.

In the context of biological therapeutics, the comprehensive nature of the FragMaps is of utility for excipient design as all possible binding sites of all possible excipients and buffers can be identified and quantified.

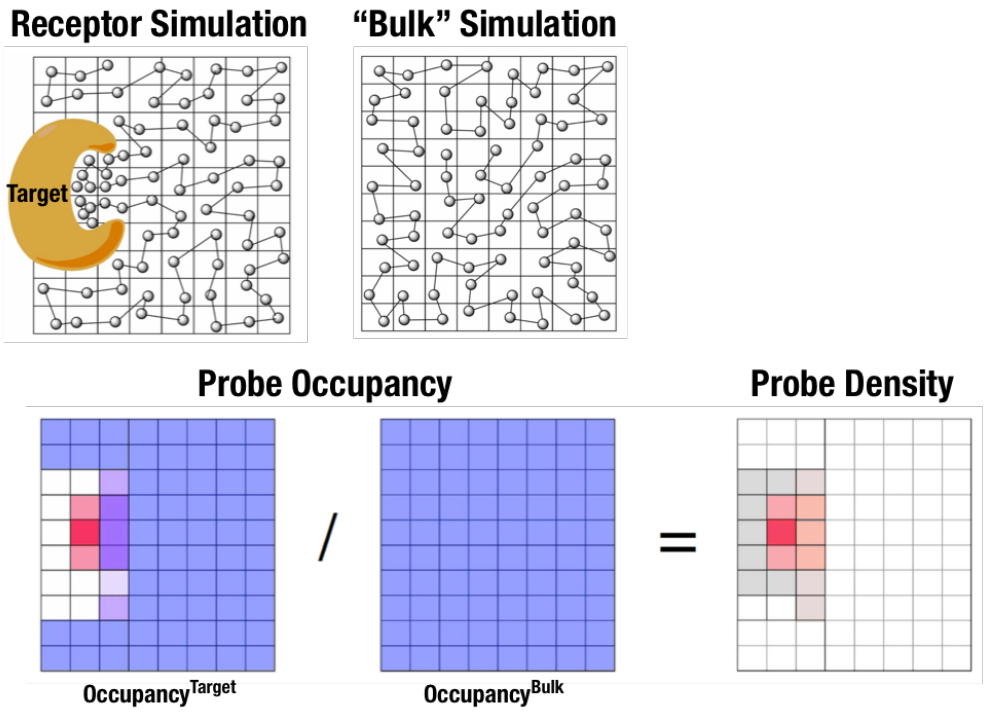


Fig. 7.1: Illustration of fragment density maps.

Fig. 7.1 illustrates FragMaps generation. Two MD simulations of a probe molecule (e.g., benzene) are performed in the presence of a protein and in aqueous solution (without protein), resulting in two occupancy maps, O^{target} and O^{bulk}, respectively. The GFE (grid free energy) FragMaps are then derived by the following formula

$$\text{GFE}^{T}_{x,y,z} = -RT \log \frac{O^{\text{target}}_{x,y,z}}{\langle O^{\text{bulk}} \rangle}$$

By generating FragMaps using various small solutes (probes) that include a diversity of atoms and chemical functional groups, it is possible to map the functional group affinity pattern of a protein. FragMaps encompass the entire protein such that all FragMap types are in all regions. Thus, information on the affinity of all the different types of functional groups is available in and around the full 3D space of the protein or other target molecule.

A Standard SILCS simulation uses benzene, propane, methanol, imidazole, formamide, dimethylether, methylammonium, acetate, and water as probes to generate Standard SILCS FragMaps. A Halogen SILCS simulation uses fluorobenzene, chlorobenzene, bromobenzene, fluoroethane, chloroethane, and trifluoroethane as probes to generate Halogen SILCS FragMaps. Halogen SILCS FragMaps are used to augment the information provided by Standard SILCS FragMaps.

FragMaps may be used in a qualitative fashion to facilitate ligand design by allowing the medicinal chemist to readily visualize regions where the ligand can be modified or functional groups added to improve affinity and specificity. As the FragMaps include protein flexibility, they indicate regions of the target protein that can "open" thereby identifying regions under the protein surface accessible

for ligand binding. In addition to FragMaps, an Exclusion Map is generated based on regions of the system that are not sampled at all by any probe molecules, including water, during the MD simulations. Therefore, the Exclusion Map represents a strictly inaccessible surface.

Many proteins contain binding sites that are partially or totally inaccessible to the surrounding solvent environment that may require partial unfolding of the protein for ligand binding to occur. SILCS sampling of such deep or inaccessible pockets is facilitated by the use of a Grand Canonical Monte Carlo (GCMC) sampling technique in conjunction with MD simulations [10]. Thus, the SILCS technology is especially well-suited for targeting the deep or seemingly inaccessible binding sites found in targets like GPCRs and nuclear receptors.

Once a set of ligand-independent SILCS FragMaps are produced, they can be used for various purposes that rapidly rank ligand binding in a highly computationally efficient manner for multiple ligands **WITHOUT** recalculating the FragMaps. Applications of SILCS methodology include:

- Binding site identification
  - Binding pocket searching with known ligands
  - Binding pocket identification via pharmacophore generation
  - Binding pocket identification via fragment screening
- Database screening
  - SILCS 3D pharmacophore models
  - Ligand posing using available techniques (Catalyst, etc)
  - Ligand ranking
- Fragment-based ligand design
  - Identification of fragment binding sites
  - Estimation of ligand affinity following fragment linking
  - Expansion of fragment types
- Ligand optimization
  - Qualitative ligand optimization by FragMaps visualization
  - Quantitative evaluation of ligand atom contributions to binding
  - Quantitative estimation of relative ligand affinities
  - Quantitative estimation of large numbers of ligand chemical transformations

SILCS generates 3D maps of interaction patterns of functional group with your target molecule, called FragMaps. This unique approach simultaneously uses multiple different small solutes with various functional groups in explicit solvent MD simulations that include target flexibility to yield 3D FragMaps that encompass the delicate balance of target-functional group interactions, target desolvation, functional group desolvation and target flexibility. The FragMaps may then be used to

both qualitatively direct ligand design and quantitatively to rapidly evaluate the relative affinities of large numbers of ligands following the precomputation of the FragMaps.

This chapter goes over the workflow for generating SILCS FragMaps.

7.2 FragMap names and underlying probe atoms

Each FragMap generated from a SILCS simulation is the result of computing occupancies for one or more kinds of atoms associated with the SILCS probes. The tables below provide the complete lists for Standard SILCS and for Halogen SILCS simulations.

Table 7.1: Standard SILCS FragMaps

FragMap name	generated from probe atom(s)
acec	acetate carboxylate carbon
apolar	generic apolar carbons (benc+prpc)
benc	benzene carbons
dmeo	dimethyl ether oxygen
forn	formamide nitrogen
foro	formamide oxygen
hbacc	generic hydrogen bond acceptors (foro+dmeo+imin)
hbdon	generic hydrogen bond donors (forn+iminh)
imin	imidizole nitrogen (h-bond acceptor)
iminh	imidizole nitrogen (h-bond donor)
mamn	methylamine nitrogen
meoo	methanol oxygen
prpc	propane carbon
tipo	water oxygen

Table 7.2: Halogen SILCS FragMaps

FragMap name	generated from probe atom
brbx	bromobenzene bromine
clbx	chlorobenzene chlorine
clex	chloroethane chlorine
fetx	fluoroethane fluorine
flbc	fluorbenzene fluorine
tfec	trifluoroethane carbon bonded to fluorines

The FragMap with the name "excl" is a special case. It is the SILCS Exclusion Map, and it enumerates all voxels where no probe molecule (including water) sampling occurred.

7.3 Running SILCS simulations from the SilcsBio GUI

Please see SILCS simulation from the GUI as described in *Graphical User Interface Quickstart* for a step-by-step description for using the SilcsBio GUI for SILCS simulations.

FragMap generation using SILCS begins with a well-curated protein structure file (without ligand) in PDB file format. We recommend keeping only those protein chains that are necessary for the simulation, removing all unnecessary ligands, renaming non-standard residues, filling in missing atomic positions, and, if desired, modeling in missing loops. Please contact support@silcsbio.com if you need assistance with protein preparation.

A typical SILCS simulation can produce output totaling in excess of 100 GB, so please select a "Project Directory" folder on your server with appropriate free space. The default is determined by the "Project Directory" setting during Remote server setup.

The setup process automatically builds the topology of the simulation system, creating metal-protein bonds if metal ions are present, rotates side chain orientations to enhance sampling, and places probe molecules around the protein. If missing loops were not modeled into the input PDB file, the amino acids on either side of each missing loop will be automatically capped with neutral functional groups.

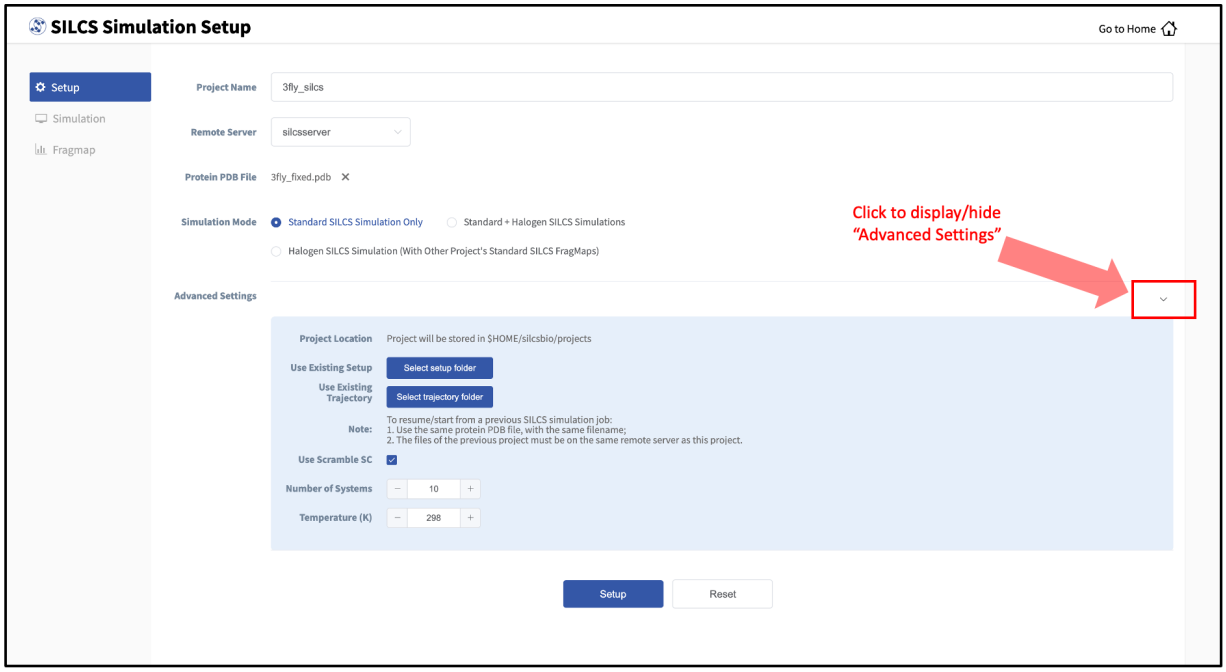
During setup, you will need to decide whether to run a Standard SILCS Simulation, Standard + Halogen SILCS Simulations, or a Halogen SILCS Simulation (with another project's Standard SILCS FragMaps). All functionality in the SILCS platform requires FragMaps from a Standard SILCS Simulation. FragMaps from Halogen SILCS Simulations can supplement Standard SILCS FragMaps for silcs/silcs-mc-optimization and for silcs/silcs-mc-docking. Visualization of Halogen SILCS FragMaps can be very useful for qualitatively informing ligand design (see silcs/visualization). See FragMap names and underlying probe atoms for a complete listing of both Standard and Halogens SILCS probe molecules and the resulting FragMaps.

Running a Halogen SILCS simulation requires the same computation time as running a Standard SILCS simulation, so selecting "Standard + Halogen SILCS Simulations" will consume double the computational resources as selecting "Standard SILCS Simulation". The default is to generate and run 10 systems each for a Standard SILCS simulation and for a Halogen SILCS simulation. Therefore, if your computing resource has 10 nodes, it will take twice as much wall time to complete a "Standard + Halogen SILCS Simulations" job as to complete a "Standard SILCS Simulation". However, if you have access to 20 nodes simultaneously, the wall time will be the same, since each individual system is run independently of the others.

While the same identical input protein coordinates are used for constructing each of these systems, each system is unique because the probe molecules are randomly placed. Additional diversity is added by default to initial coordinates by scrambling the sidechains of exposed amino acids.

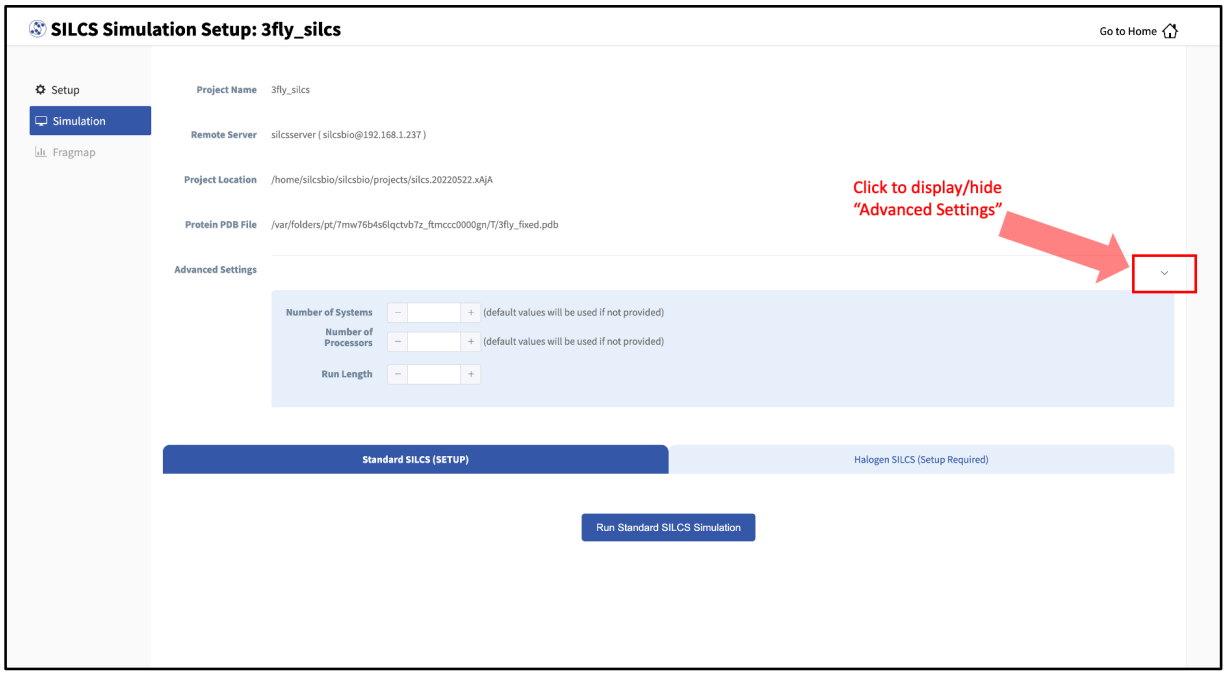
Tip: Advanced Settings options during setup allow you to change the scrambling of sidechains (default: on), the number of systems (default: 10), and the simulation temperature (default 298 K). We STRONGLY urge you to use these default settings, which have been selected based on extensive experience.

It is also possible to choose an existing setup folder or an existing trajectory folder at this time.



Once the systems are set up, you will need to start the SILCS simulations. There are separate tabs for running the Standard SILCS simulation and the Halogen SILCS simulation, and you will need to use each tab and the "Run…" button underneath that tab to run the corresponding simulation type.

Tip: Advanced Settings options allow you to change the number of systems that are run, as well as the number of processors used for each run and the length of each run. Leaving these fields blank will result in default values being used. We STRONGLY urge you to use the default settings. If you have concerns about the number of processors per run, please contact your system administrator.



Tip: If a simulation stops prematurely, the progress bar will turn red and show a "Restart" option. When the job is restarted, completed cycles will be automatically skipped and the progress will update to where the job had stopped. That is to say, restarting a failed or stopped job will *not* cause you to lose existing job progress.

7.4 Running SILCS simulations from the command line interface

Please see SILCS simulation from the command line as described in *Command Line Interface Quickstart* for a step-by-step description for using the Command Line Interface (CLI) for SILCS simulations.

FragMap generation using SILCS begins with a well-curated protein structure file (without ligand) in PDB file format. We recommend keeping only those protein chains that are necessary for the simulation, removing all unnecessary ligands, renaming non-standard residues, filling in missing atomic positions, and, if desired, modeling in missing loops. Please contact support@silcsbio.com if you need assistance with protein preparation.

7.4.1 SILCS simulation setup

```
${SILCSBIODIR}/silcs/1_setup_silcs_boxes prot=<Protein PDB>
```

Warning: The setup program internally uses the GROMACS utility `pdb2gmx`, which may have problems processing the protein PDB file. The most common reasons for errors at this stage involve mismatches between the expected residue name/atom names in the input PDB and those defined in the CHARMM force-field.

To fix this problem: Run the `pdb2gmx` command manually from within the `1_setup` directory for a detailed error message. The setup script already prints out the exact commands you need to run to see this message.

```
cd 1_setup
pdb2gmx -f <PROT PDB NAME>_4pdb2gmx.pdb -ff charmm36 -water tip3p␣
↪-o test.pdb -p test.top -ter -merge all
```

Once all the errors in the input PDB are corrected, rerun the setup script.

The setup command offers a number of options. The full list of options can be reviewed by simply entering the command without any options:

```
$SILCSBIODIR/silcs/1_setup_silcs_boxes
This script will setup Topology/PDB for SILCS simulations

Usage: $SILCSBIODIR/silcs/1_setup_silcs_boxes prot=<protein PDB file>

Optional Parameters:
   numsys=<# of simulations; default=10>
   skip_pdb2gmx=<true/false; default=false>
   lig=<lig.mol2; default=empty/NULL>
   core_restraints_string=<restrained residues, e.g., "r 17-21 | r 168-
↪174"; default=all>
   scramblesc=<true/false; default=true>
   margin=<system margin in Angstrom; default=15>
   halogen=<true/false; default=false>
```

- *core_restraints_string*: By default, a weak harmonic positional restraint, with a force constant of ~0.1 kcal/mol/Å^2, is applied to all C-alpha atoms during SILCS simulations. This option lets you modify this default.

  For example, if `core_restraints_string="r 17-160 | r 168-174"` is supplied, then only the C-alpha atoms in residues 17 to 160 and residues 168 to 174 will be restrained.

- *lig*: Use this option if your input structure contains a cofactor and you wish to compute your

FragMaps in the presence of that cofactor. Remove the cofactor from the input PDB, and provide the structure of the cofactor in `.mol2` format using this option. The cofactor needs to be aligned appropriately with the protein structure.

Note: If your input structure contains a metal ion, please see *What happens when I set up SILCS simulations with an input structure containing a metal ion?*

7.4.2 SILCS GCMC/MD simulation

Following completion of the setup, run 10 GCMC/MD jobs using the following script:

```
${SILCSBIODIR}/silcs/2a_run_gcmd prot=<Protein PDB>
```

This script will submit 10 jobs to the predefined queue. Each job runs out 100 ns of GCMC/MD with the protein and the solute molecules.

Tip: SILCS simulations are run at a temperature of 298 K by default, and most published work with SILCS has used this default temperature. It is possible to set a custom temperature by adding the `temp=<simulation temperature; default=298>` option to the `2a_run_gcmd` command. For example, `temp=310` would run the SILCS simulations at 310 K, which may enhance protein conformational sampling.

Warning: It is important to pass the same `temp=` option to `2c_fragmap` (see below) as was used for `2a_run_gcmd`. The simulation temperature is required for Boltzmann conversion of occuppany FragMaps to Grid Free Energy (GFE) FragMaps, and needs to be the same across both commands for correctness. The `temp=` option can be safely ommitted from both of these steps if the default temperature of 298 K is desired.

Once the simulations are complete, the `2a_run_gcmd/[1-10]` directories will contain `*.prod.125.rec.xtc` trajectory files. If these files are not generated, then your simulations are either still running or stopped due to a problem. Look in the log files within these directories to diagnose problems.

7.4.3 FragMap generation

Once your simulations are done, generate the FragMaps using the following command:

```
${SILCSBIODIR}/silcs/2b_gen_maps prot=<Protein PDB>
```

This will submit 10 single-core jobs that will build occupancy maps (FragMaps) spanning the simulation box for select probe molecule atoms representing different functional groups. For a ~50K atom simulation system, this step takes approximately 10-20 minutes to complete. The FragMaps have a default grid spacing of 1 Å.

The next step is to combine the occupancy FragMaps generated from individual simulations and convert them into GFE FragMaps. Each voxel in a GFE FragMap contains a Grid Free Energy (GFE) value for the probe type that was used to create the corresponding occupancy FragMap.

```
${SILCSBIODIR}/silcs/2c_fragmap prot=<Protein PDB>
```

Warning: It is important to pass the same `temp=` option to `2c_fragmap` as was used for `2a_run_gcmd` (see above). The simulation temperature is required for Boltzmann conversion of occuppany FragMaps to Grid Free Energy (GFE) FragMaps, and needs to be the same across both commands for correctness. The `temp=` option can be safely ommitted from both of these steps if the default temperature of 298 K is desired.

GFE FragMaps will be created in the `silcs_fragmap_<protein PDB>/maps` directory, and PyMOL and VMD scripts to load these file will be created in the `silcs_fragmap_<protein PDB>` directory.

7.4.4 Cleanup

Raw trajectory output files are large. To reduce disk usage due to these files, use the cleanup command:

```
${SILCSBIODIR}/silcs/2d_cleanup prot=<Protein PDB>
```

The `2d_cleanup` command deletes unnecessary files, such as those from the equilibration phase, and deletes hydrogen atoms from production trajectory files for each run directory, `1/`, `2/`, etc. It then creates a `.tar.gz` file of that directory. That is to say, the contents of `1.tar.gz` and `1/` after running cleanup are identical. You may choose to remove either one of those without loss of information, though we strongly recommend comparing the output of `tar tvzf <run>.tar.gz` with the output of `find <run> -type f`, where you replace `<run>` with `1`, `2`, etc. and confirm the contents are identical before deleting anything.

7.5 SILCS simulation setup with membrane proteins

You may use either a bare protein structure or your own pre-built protein/bilayer system with SILCS.

If you are beginning with a bare protein, SilcsBio provides a command line utility to embed transmembrane proteins, such as GPCRs, in a bilayer of 9:1 POPC/cholesterol for subsequent SILCS simulations:

```
${SILCSBIODIR}/silcs-memb/1a_fit_protein_in_bilayer prot=<Protein PDB>
```

This command will align the first principal axis of the protein with the bilayer normal (Z-axis) and translate the protein center of mass to the center of the bilayer (Z=0).

If the protein is already oriented as desired, `orient_principal_axis=false` will suppress automatic principal axis-based alignment.

```
${SILCSBIODIR}/silcs-memb/1a_fit_protein_in_bilayer prot=<Protein PDB>␣
↪orient_principal_axis=false
```

By default the system size is set to 120 Å along the X and Y dimensions. To change the system size, use the `bilayer_x_size` and `bilayer_y_size` options.

```
${SILCSBIODIR}/silcs-memb/1a_fit_protein_in_bilayer prot=<Protein PDB>␣
↪bilayer_x_size=<X dimension> bilayer_y_size=<Y dimension>
```

By default, SILCS fits the bare protein into the lipid bilayer such that the center of mass of the protein is aligned with the center of mass of the bilayer. This may not be the best choice for your membrane proteins, and you may therefore want to adjust the position of your protein according to a recommendation from an external program such as the OPM server. In this case you can adjust the relative position of the protein with respect to the position of the bilayer by using the `offset_z` option to specify the distance by which the center of mass of the protein should be offset in Z-direction from the center of the bilayer.

```
${SILCSBIODIR}/silcs-memb/1a_fit_protein_in_bilayer prot=<Protein PDB>␣
↪offset_z=<distance_in_Z-direction>
```

Note: For example, to use OPM server (https://opm.phar.umich.edu/ppm_server) output as your guide to build a transmembrane system for SILCS simulation, please see *How do I fit my membrane protein in a bilayer as suggested by the OPM server?*

The resulting protein/bilayer system will be output in a PDB file with suffix `_popc_chol`. It is important to use molecular visualization software at this stage to confirm correct orientation of the protein and size of the bilayer before using this output as the input in the next step.

Alternatively, you may use a protein/bilayer system that has been built using other software tools.

The following command will prepare the SILCS simulation box by adding water and probe molecules to the protein/bilayer system:

```
${SILCSBIODIR}/silcs-memb/1b_setup_silcs_with_prot_bilayer prot=
↪<Protein/bilayer PDB>
```

SILCS GCMC/MD simulation can now be launched with the following command:

```
${SILCSBIODIR}/silcs/2a_run_gcmd prot=<Protein/bilayer PDB> memb=true
```

The SILCS GCMC/MD will begin with a 6-step pre-equilibration to slowly relax the bilayer followed by 10 ns of relaxation of the protein into the bilayer prior to the production simulation.

Tip: GPCR simulation systems are typically large and can require significant storage space (> 200 GB) and simulation time (> 14 days with GPUs).

FragMap generation from the GCMC/MD simulation data follows the same protocol as for non-bilayer systems. Refer to previous instructions for `$SILCSBIODIR/silcs/2b_gen_maps` and `$SILCSBIODIR/silcs/2c_fragmap`.

If you encounter a problem, please contact support@silcsbio.com.

7.6 SILCS simulation setup with halogen probes

SILCS probes with halogen-containing functional groups can be useful for extending the diversity of chemical space covered by FragMaps. While a common approach is to consider halogen atoms as non-polar groups, research on non-covalent "halogen bonding" suggests halogen atoms warrant special treatment.

The CHARMM General Force Field CGenFF v.2.3.0+ allows halogen atoms to be treated with lonepairs to achieve directionality in polar interactions. These force field improvements are included in SILCS simulations from version 2020.1 of the SilcsBio software. Halogen-protein interactions are determined using a set of halogenated probes in SILCS simulations: chlorobenzene, fluorobenzene, bromobenzene, chloroethane, fluoroethane, and trifluoroethane.

To use these halogenated SILCS probes in your simulation instead of the standard SILCS probes, add the `halogen=true` keyword:

```
${SILCSBIODIR}/silcs/1_setup_silcs_boxes prot=<Protein PDB>␣
↪halogen=true
${SILCSBIODIR}/silcs/2a_run_gcmd prot=<Protein PDB> halogen=true
${SILCSBIODIR}/silcs/2b_gen_maps prot=<Protein PDB> halogen=true
${SILCSBIODIR}/silcs/2c_fragmaps prot=<Protein PDB> halogen=true
```

This will create folders ending with `_x` (i.e., `1_setup_x/`, etc.) to denote SILCS-Halogen simulation systems, as opposed to standard SILCS simulations. The resulting halogen FragMaps will be

included in the `silcs_fragmaps_<PROT>/maps` folder. To use halogen FragMaps in SILCS-MC jobs, refer to the `halogen=true` option in *SILCS-MC: Docking and Pose Refinement*.

While running SILCS with halogen probes is currently supported only through the command line interface, the SilcsBio GUI will automatically load halogen FragMaps from these simulations alongside standard FragMaps.

If you encounter a problem, please contact support@silcsbio.com.

7.7 Resuming stopped SILCS jobs

Situations such as exceeding workload manager (job queue) walltime limits can cause SILCS jobs to stop before running to completion. Resuming such jobs from where they stopped is straightforward. On the server where the job was running, go to the directory `<Project Location>/2a_run_gcmd/i`, where `i` is an integer from 1 to 10 and corresponds to the stopped job among the ten SILCS GCMC/MD jobs that were being run for that particular target. In that directory, directly use the file `job_mc_md.cmd` to submit the job to the workload manager, for example `qsub job_mc_md.cmd` for Sun Grid Engine (SGE) and `sbatch job_mc_md.cmd` for Slurm.

To ensure a successful restart, make sure to issue any required extra setup commands (for example, as listed in the "Extra setup" field under *Settings and remote server configuration* from the SilcsBio GUI Home screen) such as `export LD_LIBRARY_PATH=<path/to/cuda/libraries>` or `module load <name of cuda module>` before submitting the job.

CHAPTER

EIGHT

SILCS-RNA: TARGETING RNAS WITH SMALL MOLECULES

8.1 Background

SILCS-RNA has been developed to tailor the SILCS methodolgy towards targeting RNA molecules with small molecules. The fundamental idea of complementarity between the functional groups in a small molecule and the binding site of the target, originally developed for protein targets, has been further developed and validated for RNA targets. Extensions to the method include an enhanced oscillating excess chemical potential protocol for the Grand Canonical Monte Carlo calculations and individual simulations of the neutral and charged solutes from which the SILCS functional group affinity maps (FragMaps) are calculated for subsequent binding site identification, pharmacophore, docking etc. calculations. Development and validation of SILCS-RNA has taken into consideration the complexities of RNA tertiary structure and the effects of the highly negatively charged phosphate backbone on sampling of neutral and charged probes. Full details have been published in [14].

8.2 Preparation of input RNA PDB file

SILCS-RNA requires a clean input RNA PDB file for best results.

The input RNA PDB file:

- Should NOT contain any ions, solvents or water molecules.
- Should NOT contain more than one MODEL. For example, PDB files from NMR studies may contain multiple models, and user must remove unwanted models and keep only one.
- Should NOT contain alternate conformations of residues.

8.3 SILCS-RNA simulation setup

Set the environment variables required to access the software:

```
export GMXDIR=<gromacs/bin>
export SILCSBIODIR=<silcsbio>
```

Set up the SILCS-RNA simulations

```
${SILCSBIODIR}/silcs-rna/1_setup_silcs_boxes prot=<RNA PDB file>
```

To determine if the setup is complete, check that the 20 PDB files required for the simulations are available using the command `ls 1_setup_*/*_silcs.*.pdb` It is also a good idea to visualize these files to confirm the presence of the RNA molecule and the SILCS solute molecules (probes) within a box of water molecules. In contrast to standard SILCS with protein targets, SILCS-RNA simulations split the GCMC/MD simulations into two separate sets, one having neutral probes (`1_setup_neutral`) and the other having charged probes (`1_setup_charged`).

The setup command offers a number of options. The full list of options can be reviewed by running the command without any arguments:

```
$SILCSBIODIR/silcs-rna/1_setup_silcs_boxes
This script will setup Topology/PDB for SILCS simulations

Usage: $SILCSBIODIR/silcs-rna/1_setup_silcs_boxes prot=<RNA PDB file>

Optional Parameters:
   numsys=<# of simulations; default=10>
   skip_pdb2gmx=<true/false; default=false>
   lig=<lig.mol2; default=empty/NULL>
   margin=<system margin in Angstrom; default=15>
```

- `lig`: Use this option if your input structure contains a ligand and you wish to compute your FragMaps in the presence of that ligand. Remove the ligand from the input RNA PDB file, and provide the structure of the ligand in `.mol2` format using this option. The ligand needs to be aligned appropriately with the RNA structure.
- `core_restraints_string`: By default, a weak harmonic positional restraint, with a force constant of ~0.1 kcal/mol/Å^2, is applied to all MG, C1', URA-N3, CYT-N3, ADE-N1 and GUA-N1 atoms during simulations. **The option to change this default is not available in the current version of SILCS-RNA**.

Warning: The setup program internally uses the GROMACS utility `pdb2gmx`, which may have problems processing the RNA PDB file. The most common reasons for errors at this stage involve mismatches between the expected residue name/atom names in the input PDB

and those defined in the CHARMM force-field.

To fix this problem: Run the `pdb2gmx` command manually from within the `1_setup_neutral` AND the `1_setup_charged` directories for detailed error messages. For guidance on syntax, the setup script already prints out the exact commands you need to run.

```
cd 1_setup_neutral
pdb2gmx -f <RNA PDB NAME>_4pdb2gmx.pdb -ff charmm36 -water tip3p -o␣
↪test.pdb -p test.top -ter -merge all
cd ../1_setup_charged
pdb2gmx -f <RNA PDB NAME>_4pdb2gmx.pdb -ff charmm36 -water tip3p -o␣
↪test.pdb -p test.top -ter -merge all
cd ..
```

Once all errors in the input PDB are corrected, rerun the setup script with `skip_pdb2gmx=true`.

8.4 SILCS-RNA GCMC/MD simulation

Following completion of the setup, start the 20 GCMC/MD jobs: script:

```
${SILCSBIODIR}/silcs-rna/2a_run_gcmd prot=<RNA PDB file>
```

This script will submit 20 jobs to the predefined queue. Each job runs 100 ns of GCMC/MD with the RNA and the solute molecules; 10 jobs are with neutral solute molecules (probes), and 10 are with charges solute molecules (probes).

Tip: SILCS simulations are run at a temperature of 298 K by default. It is possible to set a custom temperature by adding the `temp=<simulation temperature; default=298>` option to the `2a_run_gcmd` command. For example, `temp=310` would run the SILCS simulations at 310 K.

Once the simulations are complete, the `2a_run_gcmd_neutral/[1-10]` AND `2a_run_gcmd_charged/[1-10]` directories will contain `*.prod.100.rec.xtc` trajectory files. If these files are not generated, then your simulations are either still running or stopped due to a problem. Look in the log files within these directories to diagnose problems.

To check job progress, use:

```
${SILCSBIODIR}/silcs-rna/check_progress
```

This will provide a summary list of the SILCS jobs consisting of the job path, the job number, the task ID, the job status, and the current/total number of GCMC/MD cycles. Job status values are:

Q [queued], R [running], E [successfully ended], F [failed], NA [not submitted].

8.5 FragMap generation

Once your simulations are done, generate the SILCS FragMaps using the following command:

```
${SILCSBIODIR}/silcs-rna/2b_gen_maps prot=<RNA PDB file>
```

This will submit 20 single-core jobs that will build occupancy maps (FragMaps) spanning the simulation box for select solute molecule atoms representing different functional groups. For a ~50K atom simulation system, this step takes approximately 10-20 minutes to complete. The FragMaps have a default grid spacing of 1 Å.

The next step is to combine the occupancy FragMaps generated from individual simulations and convert them into GFE FragMaps. Each voxel in a GFE FragMap contains a Grid Free Energy (GFE) value for the probe type that was used to create the corresponding occupancy FragMap.

```
${SILCSBIODIR}/silcs-rna/2c_fragmap prot=<RNA PDB file>
```

GFE FragMaps will be created in the `silcs_fragmap_<RNA PDB NAME>/maps` directory, and PyMOL and VMD scripts to load these file will be created in the `silcs_fragmap_<RNA PDB NAME>` directory. Please see *Visualizing SILCS FragMaps* for details on how to use the SilcsBio GUI, as well as external software (MOE, PyMol, VMD), to visualize SILCS FragMaps.

As with FragMaps for protein targets, your SILCS-RNA FragMaps for RNA targets can be used for optimization of a parent ligand (*SILCS-MC: Ligand Optimization*), docking of ligands and refinement of existing docked poses (*SILCS-MC: Docking and Pose Refinement*), creation of pharmacophore models (*SILCS-Pharm: Receptor-Based Pharmacophore Models from FragMaps*), and detection of hotspots and fragment-based drug design (*SILCS-Hotspots: Fragment Binding Sites Including Allosteric Sites*).

8.6 Cleanup

Raw trajectory output files are large. To reduce disk usage due to these files, use the cleanup command:

```
${SILCSBIODIR}/silcs-rna/2d_cleanup prot=<RNA PDB file>
```

The `2d_cleanup` command deletes unnecessary files, such as those from the equilibration phase, and deletes hydrogen atoms from production trajectory files for each run directory, `1/`, `2/`, etc. It then creates a `.tar.gz` file of that directory. That is to say, the contents of `1.tar.gz` and `1/` after running cleanup are identical. You may choose to remove either one of those without loss of

information, though we strongly recommend comparing the output of `tar tvzf <run>.tar.gz` with the output of `find <run> -type f`, where you replace `<run>` with `1`, `2`, etc. and confirm the contents are identical before deleting anything.

CHAPTER

NINE

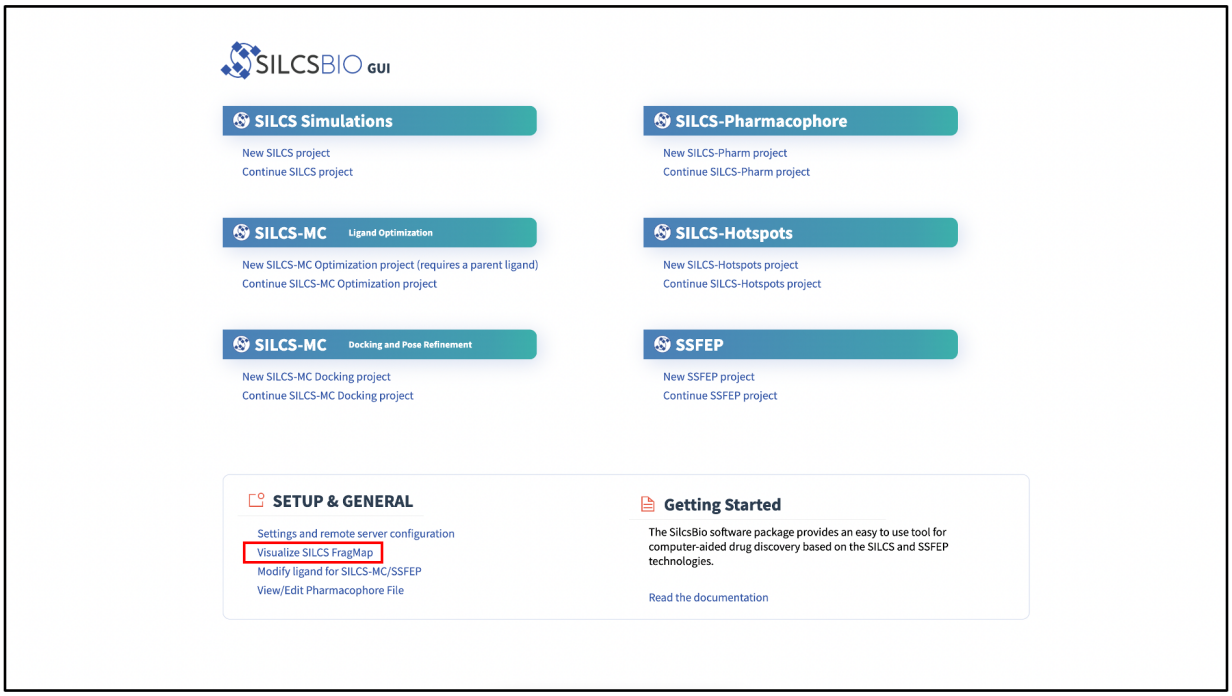
VISUALIZING SILCS FRAGMAPS

Whether you used the SilcsBIO GUI or the `2c_fragmap` command line utility to create SILCS FragMaps from your SILCS simulation trajectories, you will have a folder `silcs_fragmap_<protein PDB>/maps` that contains SILCS grid free energy (GFE) FragMaps for you system. SILCS FragMaps are defined using non-hydrogen atoms, and for certain solute types (imidazole, formamide) there are multiple FragMaps. FragMaps for water oxygen atoms (tipo) are created to identify water molecules that can be difficult to displace. Generic FragMaps for apolar (benzene+propane), H-bond donor (neutral H-bond donors), and H-bond acceptor (neutral H-bond acceptors) probe types are included to simplify visual analysis.

The easiest way to visualize FragMaps is with the SilcsBio GUI. SilcsBio also provides convenient FragMap visualization using MOE, PyMOL, and VMD.

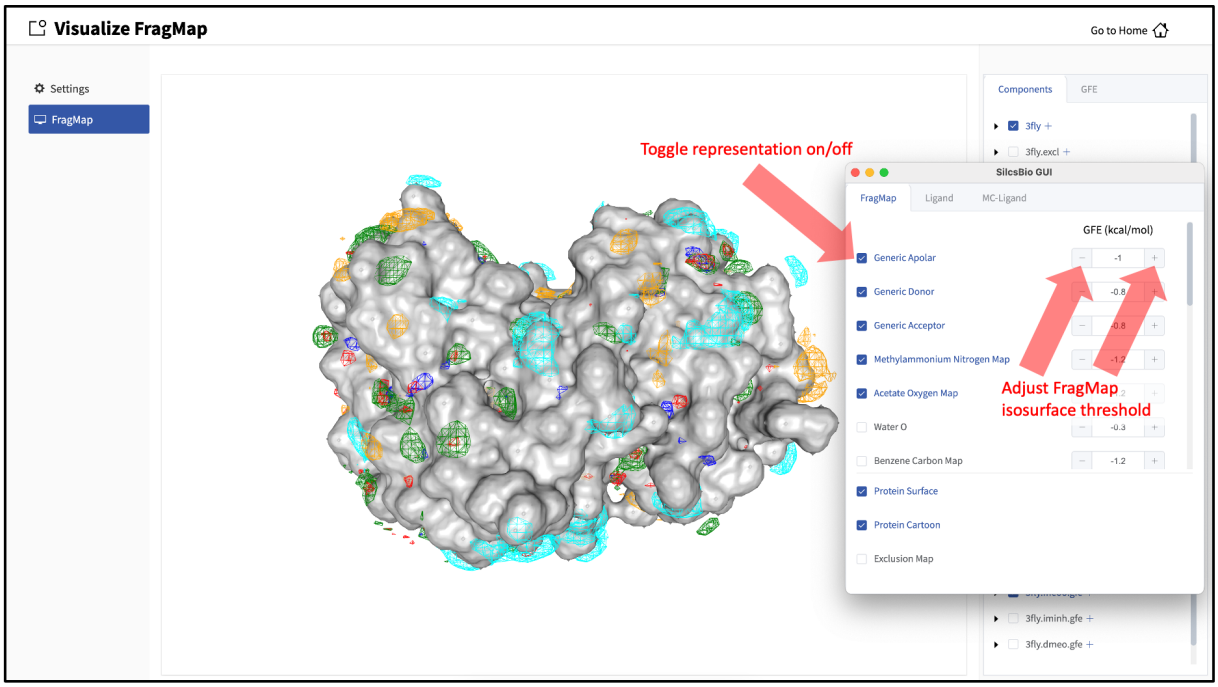
9.1 FragMaps in the SilcsBio GUI

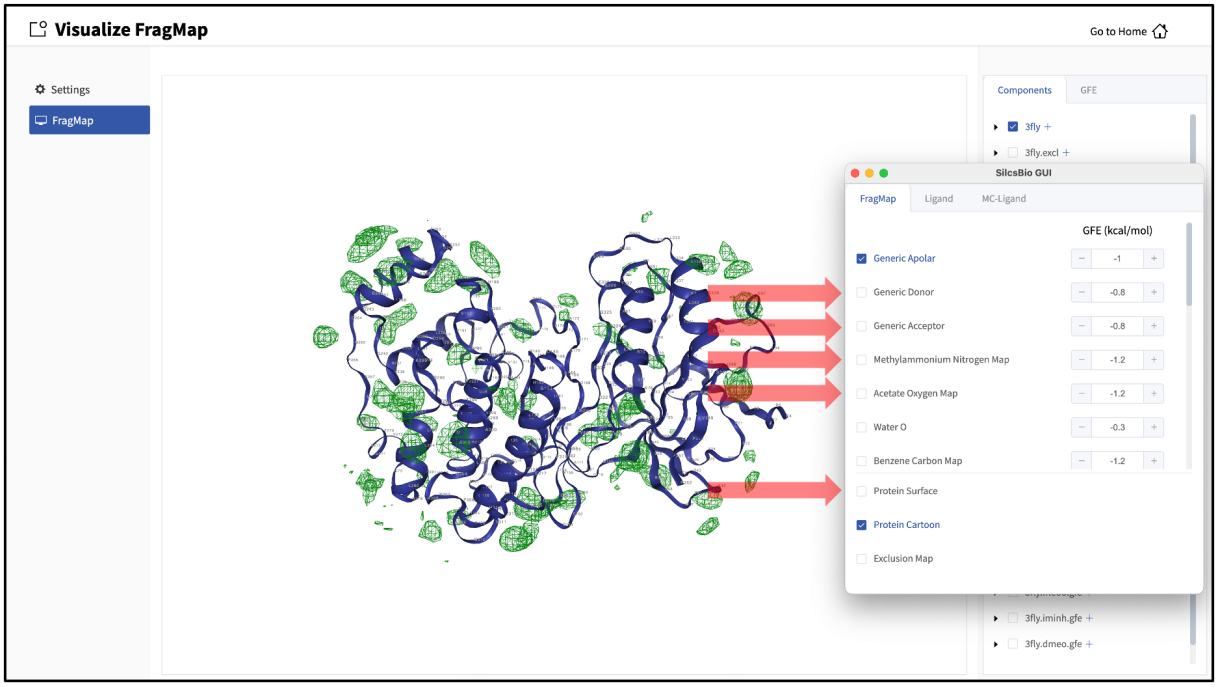
If you have used the SilcsBio GUI to run your SILCS simulations, once the simulations are complete, you will click the "Generate FragMap" button to automatically create FragMaps from the SILCS simulations, download them onto your local computer, and load them, along with the input PDB file, for visualization. Alternatively, you can load SILCS FragMaps from the Home page of the SilcsBio GUI:



If you are loading SILCS FragMaps from the Home page, you will be asked to select your FragMap directory (see File and directory selection). This directory has a standard name, `silcs_fragmaps_<protein PDB>`, where `<protein PDB>` is the name of the input PDB file. This directory was created on the server where you ran the SILCS jobs either when you clicked the "Generate FragMap" button or when you ran the `${SILCSBIODIR}/silcs/2c_fragmap prot=<protein PDB>` command.

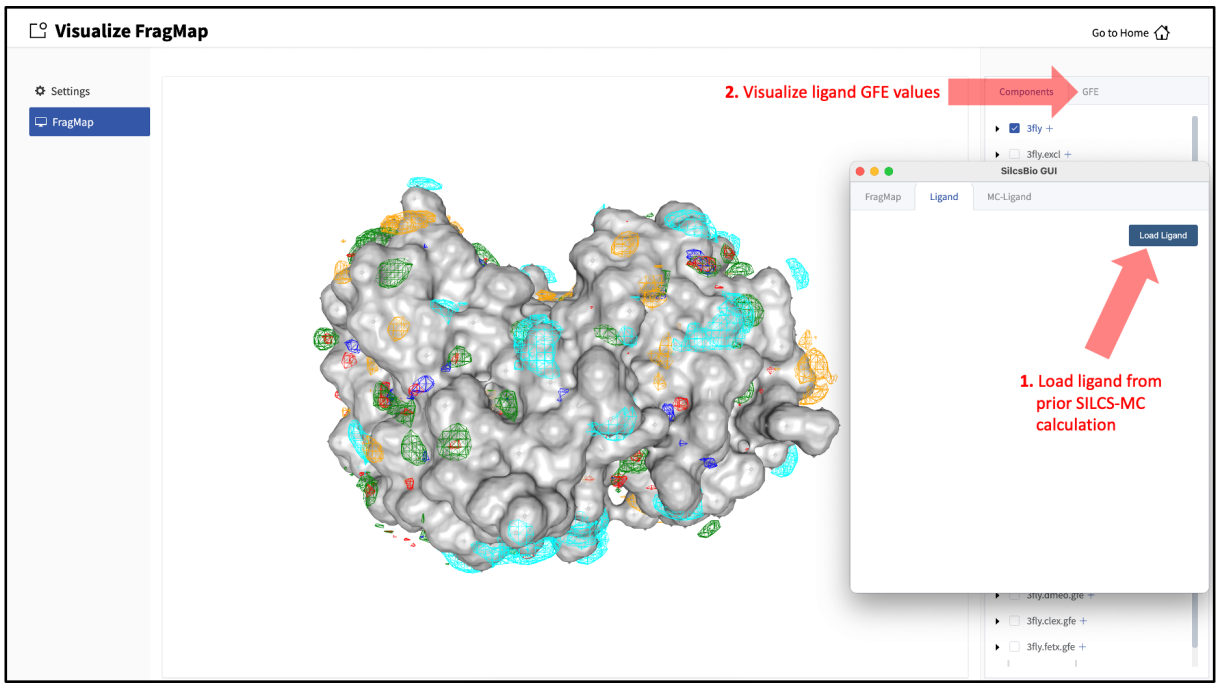
Note: In addition to the FragMaps, the `silcs_fragmaps_<protein PDB>` directory contains the PDB file used to run the SILCS simulations. The SilcsBio GUI will automatically detect and load this PDB file. Additionally, if you have run Halogen SILCS, this directory will also contain the Halogen SILCS FragMaps, which will be automatically detected and loaded by the GUI.


Visualize FragMap
Go to Home
Settings
FragMap
Components
GFE
3fly
3fly.excl
Toggle representation on/off
SilcsBio GUI
FragMap
Ligand
MC-Ligand
GFE (kcal/mol)
Generic Apolar
Generic Donor
Generic Acceptor
Methylammonium Nitrogen Map
Acetate Oxygen Map
Water O
Benzene Carbon Map
Protein Surface
Protein Cartoon
Exclusion Map
Adjust FragMap isosurface threshold
3fly.iminh.gfe
3fly.dmeo.gfe


Visualize FragMap
Go to Home
Settings
FragMap
Components
GFE
3fly
3fly.excl
SilcsBio GUI
FragMap
Ligand
MC-Ligand
GFE (kcal/mol)
Generic Apolar
Generic Donor
Generic Acceptor
Methylammonium Nitrogen Map
Acetate Oxygen Map
Water O
Benzene Carbon Map
Protein Surface
Protein Cartoon
Exclusion Map
3fly.iminh.gfe
3fly.dmeo.gfe

When adding an additional representation, the default is to apply it to all atoms. It is possible to apply it to only a select set of atoms by typing the atom selection in the blank box above the "cancel" and "confirm" buttons. The selection syntax is the same as for the NGL molecular structure viewer (see *Atom selection in the SilcsBio GUI*).

It is also possible to load ligands for visualization. If the ligand file was the output of a SILCS-MC calculation, GFE values for the ligand atoms will be embedded within that file. These values can be visualized and their sum automatically computed by first loading the ligand file, and then using the "GFE" tab as detailed at the end of the section Running SILCS-MC ligand optimization from the SilcsBio GUI.



9.2 FragMaps in MOE

To visualize your SILCS FragMaps with MOE, first create FragMaps in the MOE-readable CNS format. For this example, if your jobs ran on your server in the directory `~/silcsbio/projects/silcs.5tGP`:

```
cd ~/silcsbio/projects/silcs.5tGP
$SILCSBIODIR/silcs/2c_fragmap prot=<protein PDB> cns=true
```

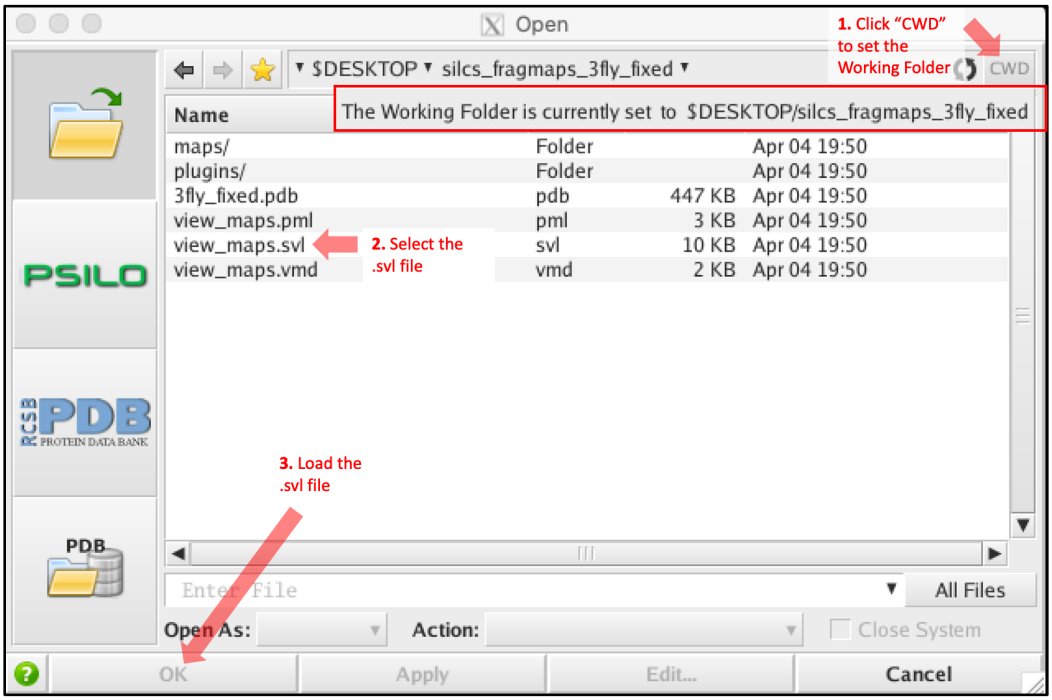
Note: This step is required ONLY for visualization with MOE. CNS format grid files are approximately 10x the size of the default-format grid files generated and used by your SilcsBio software. PyMol and VMD are capable of reading the default-format files, and, to save disk space, CNS format files are not generated unless specifically specified using the above command.

Running this command will create CNS format FragMap files in the `silcs_fragmap_<protein PDB>/maps` directory. It will also create the file `silcs_fragmap_<protein PDB>/view_maps.svl`. First, copy the entire

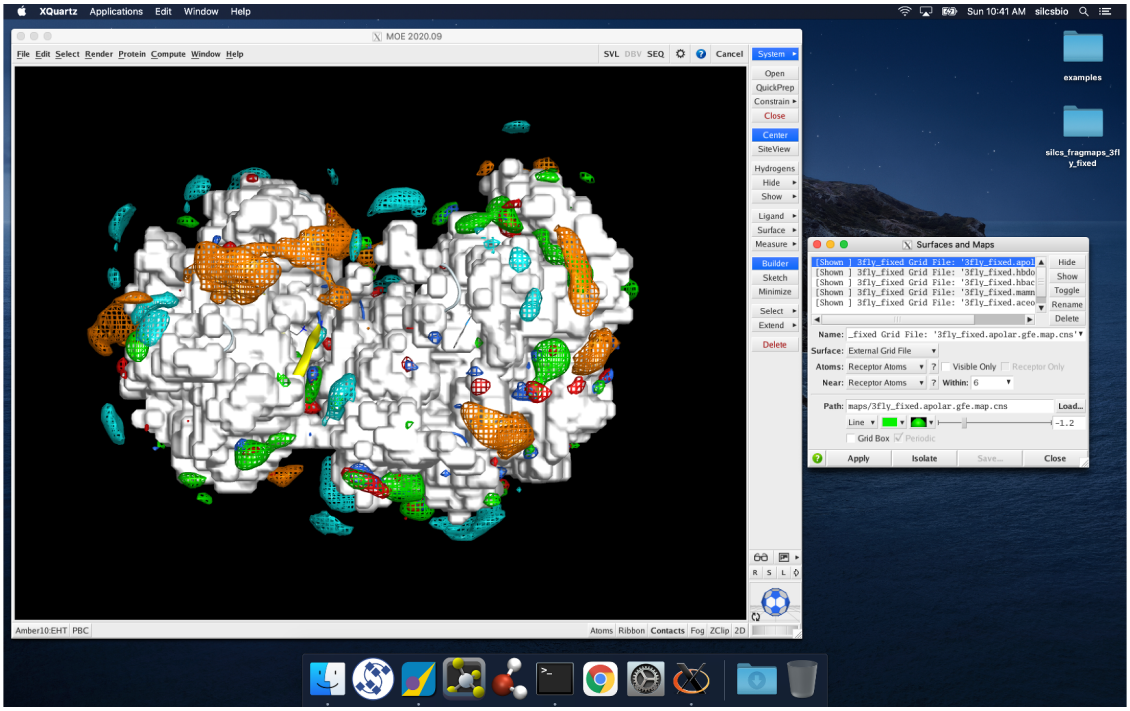
`silcs_fragmap_<protein PDB>` directory from your server to a convenient location on the desktop or laptop machine where you will be doing the visualization. For example, on Linux or MacOS, you may use a command like

```
scp -rp silcsbio@silcsserver:~/silcsbio/projects/silcs.5tGP/silcs_
↪fragmaps_3fly_fixed ~/Desktop
```

Then, use MOE to open the `silcs_fragmap_<protein PDB>/view_maps.svl` file by selecting *File* → *Open*, which will bring up the "Open" window. In this window, browse to the `silcs_fragmap_<protein PDB>` directory, which, in this example, was downloaded from the server to the Desktop of the machine on which we are doing the visualization. Click the "CWD" button in the upper right corner to set this directory as the "Working Folder." Then, select the `view_maps.svl` file and click the "Open" button:



MOE will process the `view_maps.svl` file to load your input PDB structure `<protein PDB>.pdb` used for the SILCS simulations, the CNS format FragMap files `<protein PDB>.<fragmaptype>.map.cns`, and the Exclusion Map `<protein PDB>.excl.map.cns`. The visualization of the FragMaps and Exclusion Map can be controlled in the "Surfaces and Maps" window, which is also automatically loaded by `view_maps.svl`.



9.3 FragMaps in PyMol

First, copy the entire `silcs_fragmap_<protein PDB>` directory from your server to a convenient location on the desktop or laptop machine where you will be doing the visualization. For example, on Linux or MacOS, you may use a command like

```
scp -rp silcsbio@silcsserver:~/silcsbio/projects/silcs.5tGP/silcs_
↪fragmaps_3fly_fixed ~/Desktop
```

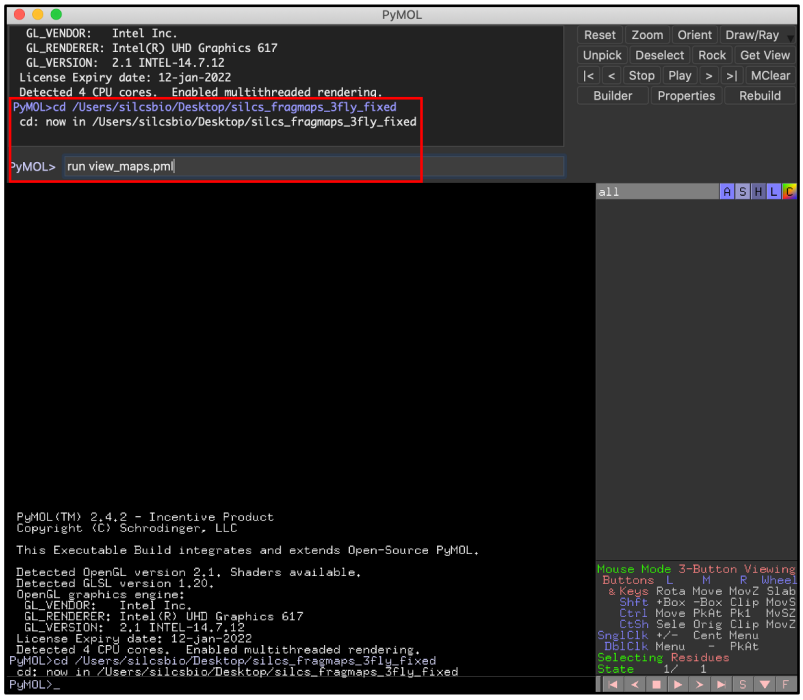
Open PyMol, then select *File* → *Open...*, and open the `view_maps.pml` file located within the `silcs_fragmaps_<protein PDB>` folder. It is also possible to do this by typing commands into the PyMol console. For the `silcs_fragmaps_3fly_fixed` example here, the commands would be

```
cd /Users/silcsbio/Desktop/silcs_fragmaps_3fly_fixed
```

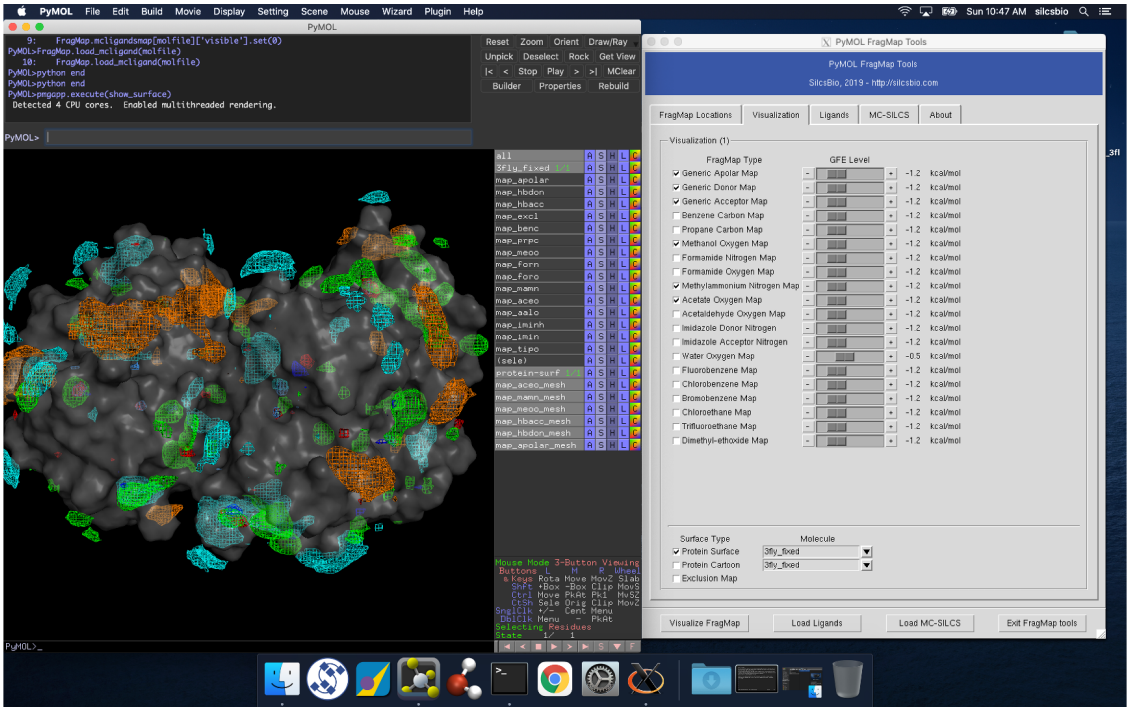
which sets the working directory, and

```
run view_maps.pml
```

which runs the `view_maps.pml` file in that directory.



Whichever of these two methods you use to load `view_maps.pml`, PyMol wil load the PDB structure `<protein PDB>.pdb` used for the SILCS simulations, the FragMap files, and the Exclusion Map. The visualization of the FragMaps and Exclusion Map can be controlled in the "PyMol FragMap Tools" window, which is also automatically loaded by `view_maps.pml`.



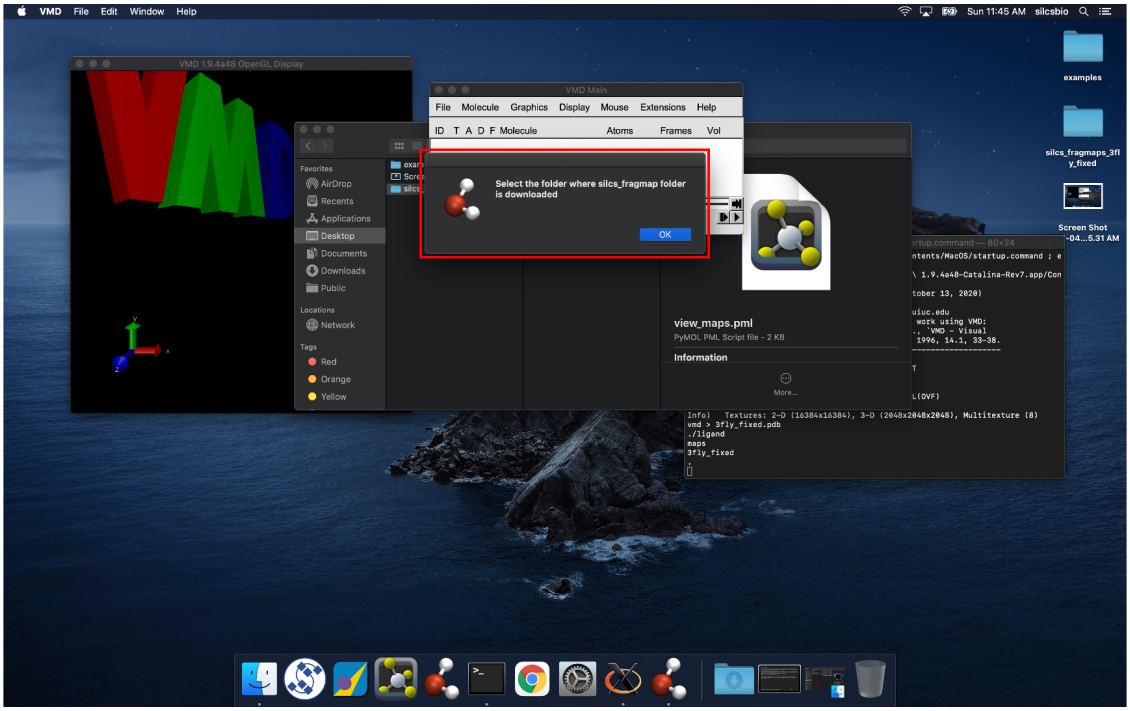
Warning: If your operating system associates the `.pml` extension with PyMol, it is possible simply to double click on the `view_maps.pml` file in the `silcs_fragmaps_<protein PDB>` folder. However, PyMol may fail to open multiple FragMaps. Specifically, if you are opening two different FragMaps having the same protein name, using double clicking to open `view_maps.pml` will result in the first FragMap being loaded again instead of the second one. Doing *File* → *Open...* will direct you to provide the correct path for the second FragMap set.

9.4 FragMaps in VMD

First, copy the entire `silcs_fragmap_<protein PDB>` directory from your server to a convenient location on the desktop or laptop machine where you will be doing the visualization. For example, on Linux or MacOS, you may use a command like

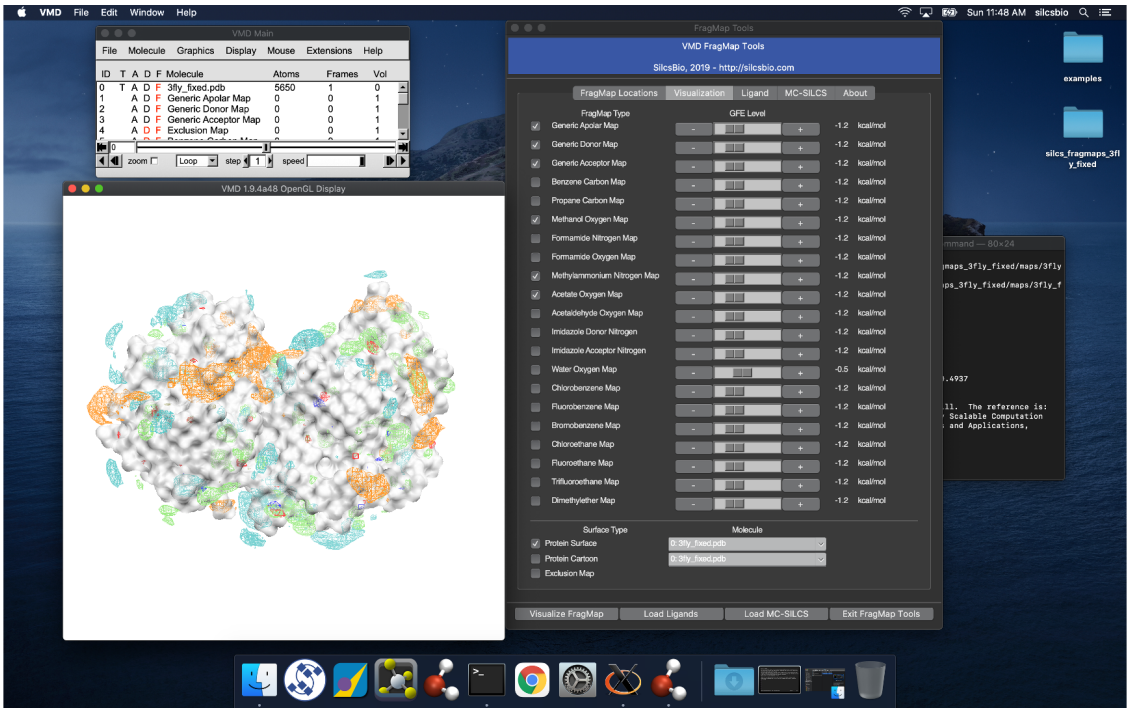
```
scp -rp silcsbio@silcsserver:~/silcsbio/projects/silcs.5tGP/silcs_
↪fragmaps_3fly_fixed ~/Desktop
```

Open VMD, select *File* → *Load Visualization State...*, and open the `view_maps.vmd` file located in the `silcs_fragmap_<protein PDB>` directory you downloaded. A new window will pop up that says, "Select the folder where silcs_fragmap folder is downloaded". Click the "OK" button and select the `silcs_fragmap_<protein PDB>` you downloaded.



Tip: Be careful to select the `silcs_fragmap_<protein PDB>` folder and NOT the `silcs_fragmap_<protein PDB>/maps` folder.

VMD wil load the PDB structure `<protein PDB>.pdb` used for the SILCS simulations, the FragMap files, and the Exclusion Map. The visualization of the FragMaps and Exclusion Map can be controlled in the "VMD FragMap Tools" window, which is also automatically loaded by `view_maps.vmd`.



Note: VMD (version 1.9.4 or older) does not read .map files correctly. It shifts the grid map by 1.73 angstrom. Visualization in PyMol is preferred.

CHAPTER

TEN

SILCS-MC: LIGAND OPTIMIZATION

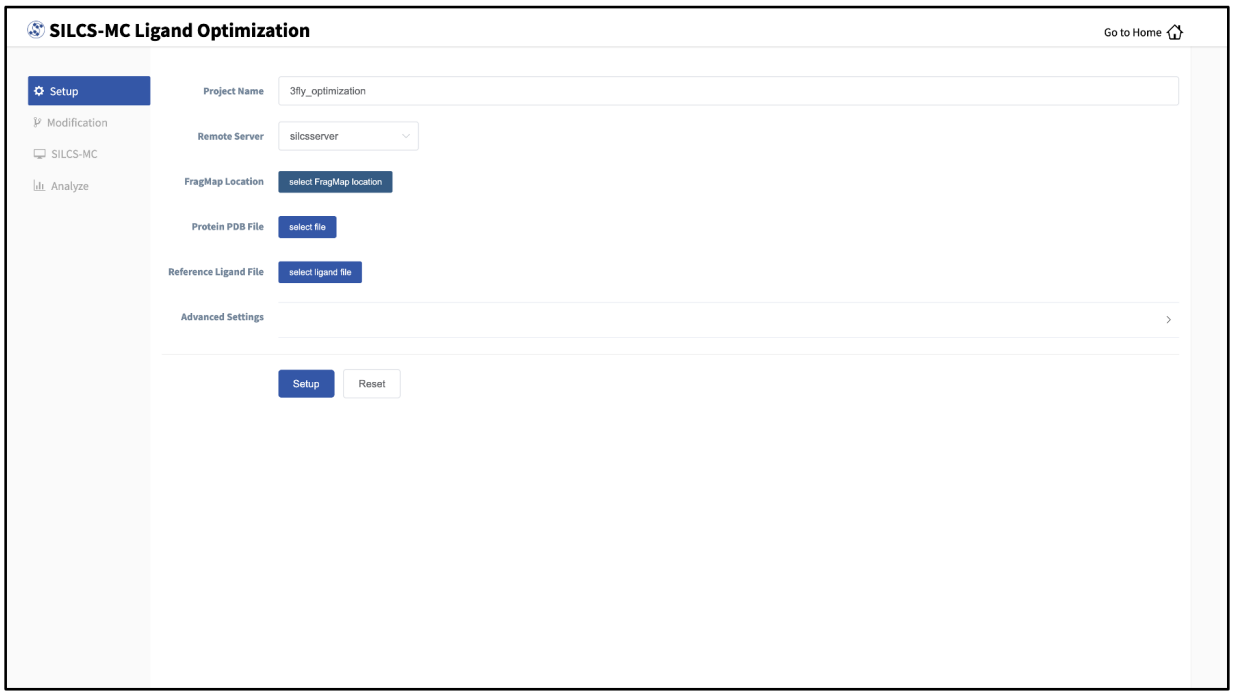
10.1 Background

The power of SILCS lies in the ability to use FragMaps to rapidly evaluate binding of diverse ligands to a target. SILCS-MC is Monte-Carlo (MC) sampling of ligands in translational, rotational, and torsional space in the field of FragMaps. MC sampling uses CGenFF intramolecular energies and the Ligand Grid Free Energy (LGFE), which is the sum of atomic Grid Free Energies (GFEs). The Exclusion Map prevents ligand sampling where no probe or water molecules visited during SILCS simulations. SILCS-MC allows for rapid conformational sampling of the ligand while accounting for protein flexibility in a mean-field-like fashion since ligand affinity and volume exclusion are embedded in the combination of FragMaps and the Exclusion Map. Final ligand scoring is based on the LGFE score [9][22].

FragMaps previously generated for a target (see *SILCS: Site Identification by Ligand Competitive Saturation*) can be used for optimization of a parent ligand by rapidly evaluating LGFE scores for derivatives of the parent ligand. The SILCS-MC approach has two principal advantages over free energy perturbation (FEP): functional group modifications to the parent ligand are very quick to evaluate, and there is no upper size limit on these modifications. Because the calculations are done in the context of pre-computed FragMaps, it is easy to decide which functional groups are candidates for modification simply by viewing the binding pose of the parent along with the FragMaps. FragMap densities adjacent to but unoccupied by parent ligand atoms offer opportunities for growing the parent ligand. Conversely, parent ligand atoms not in favorable regions of the FragMap densities can be candidates for modification or deletion without affecting binding affinity. Quantitative evaluation of individual functional group contributions to binding affinity can be achieved by analysis of atomic GFE scores. These capabilities are particularly useful when modifying functional groups to optimize pharmacokinetic properties or attempting to reduce the ligand size while maintaining affinity.

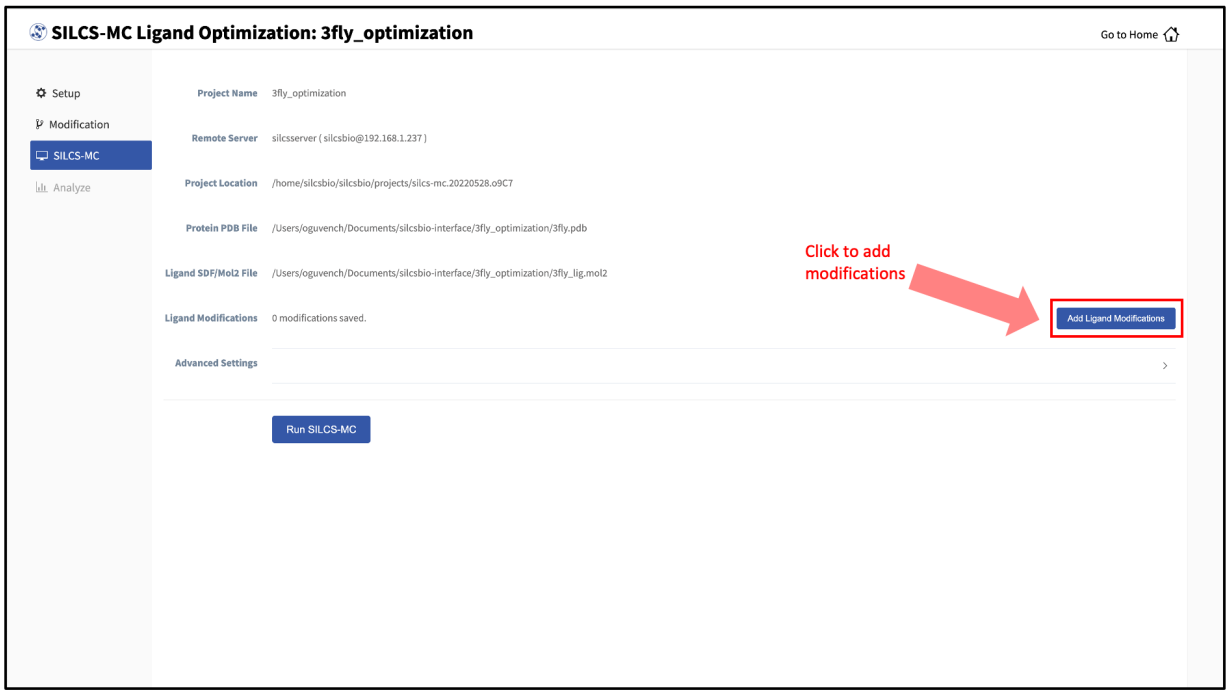
10.2 Running SILCS-MC ligand optimization from the SilcsBio GUI

1. Select *New SILCS-MC Optimization project* from the Home page.
2. Enter a project name and select the remote server where your SILCS-MC jobs will run. Next, provide FragMap, protein, and ligand input files. You may choose these files from the computer where you are running the SilcsBio GUI ("localhost") or from any server you have previously configured, as described in File and directory selection.



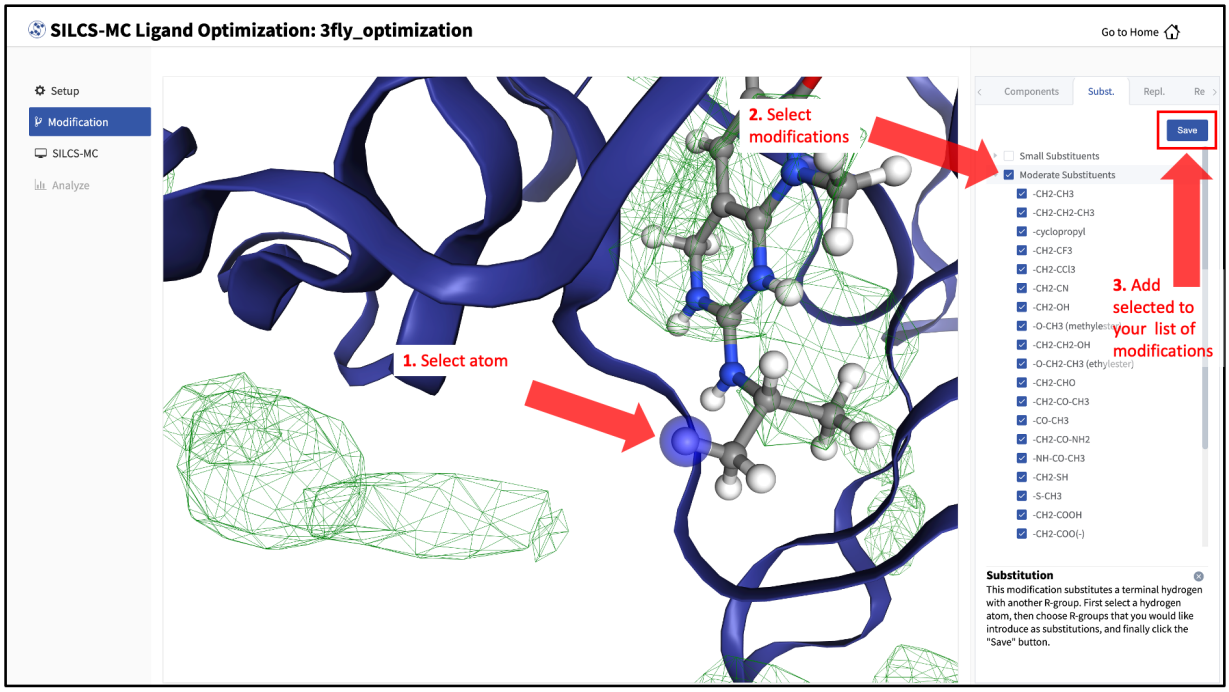
Tip: The "Reference Ligand File" must be an SD or Mol2 format file that contains the parent ligand aligned in the binding pocket of the input protein.

3. Once all information is entered correctly, press the "Setup" button at the bottom of the page. The GUI will contact the remote server and upload the your input files to the "Project Location" directory on the remote server. A green "Setup Successful" button will appear once the upload has successfully completed. Press this button to proceed.
4. The GUI will display a summary screen with the Project Name, Remote Server, Project Location, Protein PDB File, Ligand SDF/Mol2 File, and Ligand Modifications. The entry for Ligand Modifications will state "0 modifications saved" and have a button labeled "Add Ligand Modifications." Press this button to select modifications to the parent ligand for evaluation by SILCS-MC.



5. You will now see the parent ligand in the binding pocket. Rotate the view until you can easily see the functional group you wish to modify. There are two major modification types, Substitution and Replacement, available in the GUI. Substitution is used to substitute an atom with a functional group. Replacement is used to replace an atom in a ring or aliphatic acyclic group with another functional group that preserves the connectivity and valence of the ring or chain.

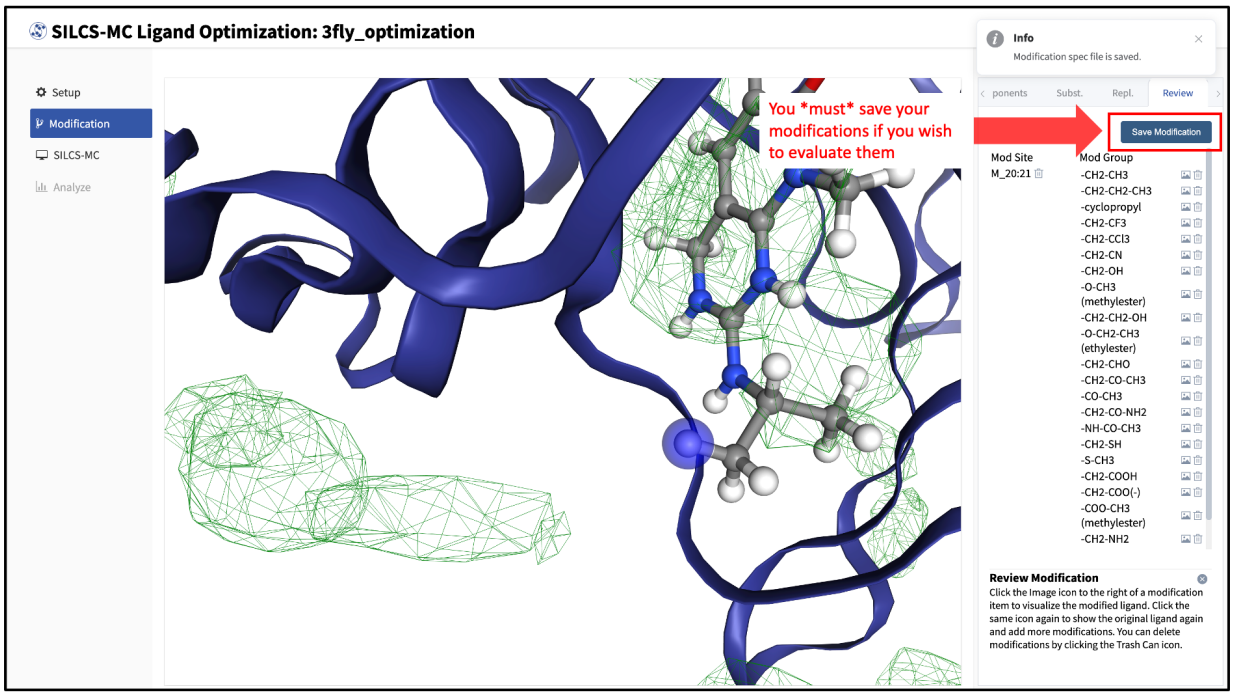
   In the visualization window, select the atom to be modified. Then, select your desired modifications from the "Substitution" or the "Replacement" tab in the right-hand panel. Pressing the "Save" button in the panel will take you to the "Review" tab.



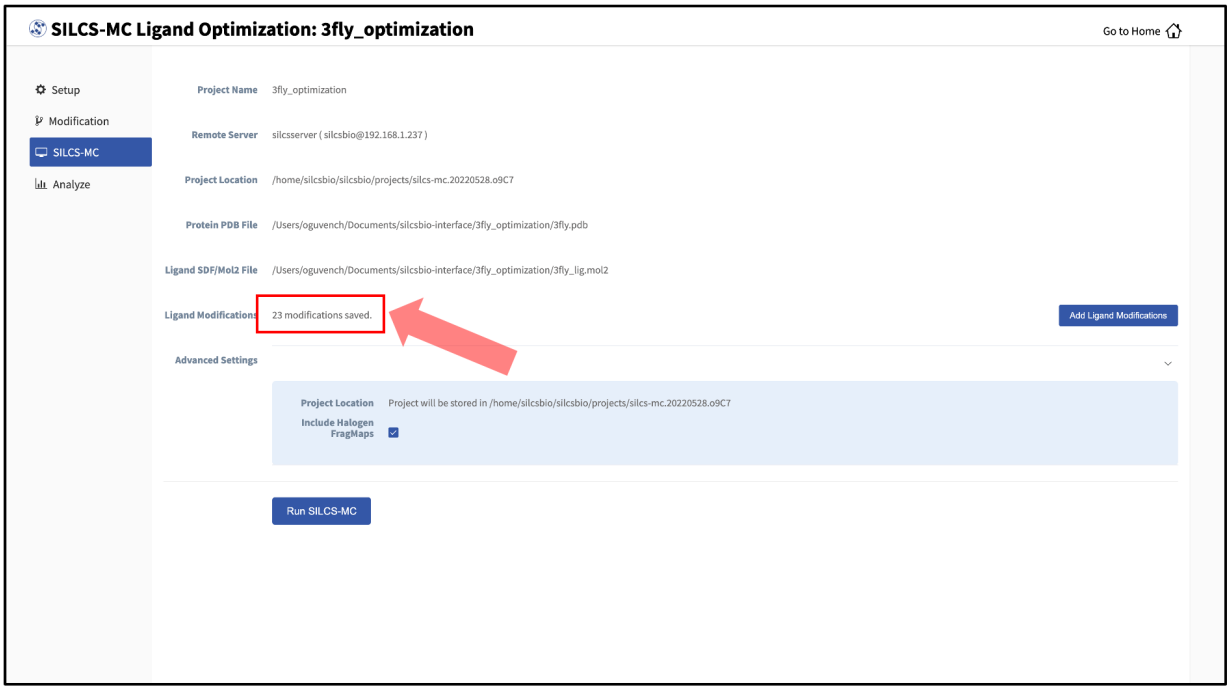
The list of modification types in the GUI covers a very broad range of chemical functionality and size.

6. Use the "Review" tab to confirm your desired modifications.

   Valid modifications will have a small Image icon as well as a small Trash Can icon. Clicking on the Image icon will show the modification in the center panel. Clicking on it again will show the parent ligand. Clicking on the Trash Can icon will delete the proposed modification from your list. You can go back to the "Substitution" and "Replacement" tabs to add to your list. Once you have completed your list of modifications, **you must press the "Save Modification" button in the "Review" tab to actually save the list of modifications for evaluation by SILCS-MC**.



7. Once you have saved all your desired modifications, click on *SILCS-MC* in the left-hand panel to go back to the summary screen. The "Ligand Modifications" will have been updated to reflect the number of saved modifications that will be evaluated by SILCS-MC. Because SILCS-MC is a fast calculation, it is feasible to select a very large number of modifications (hundreds+) for quick evaluation in a single run.

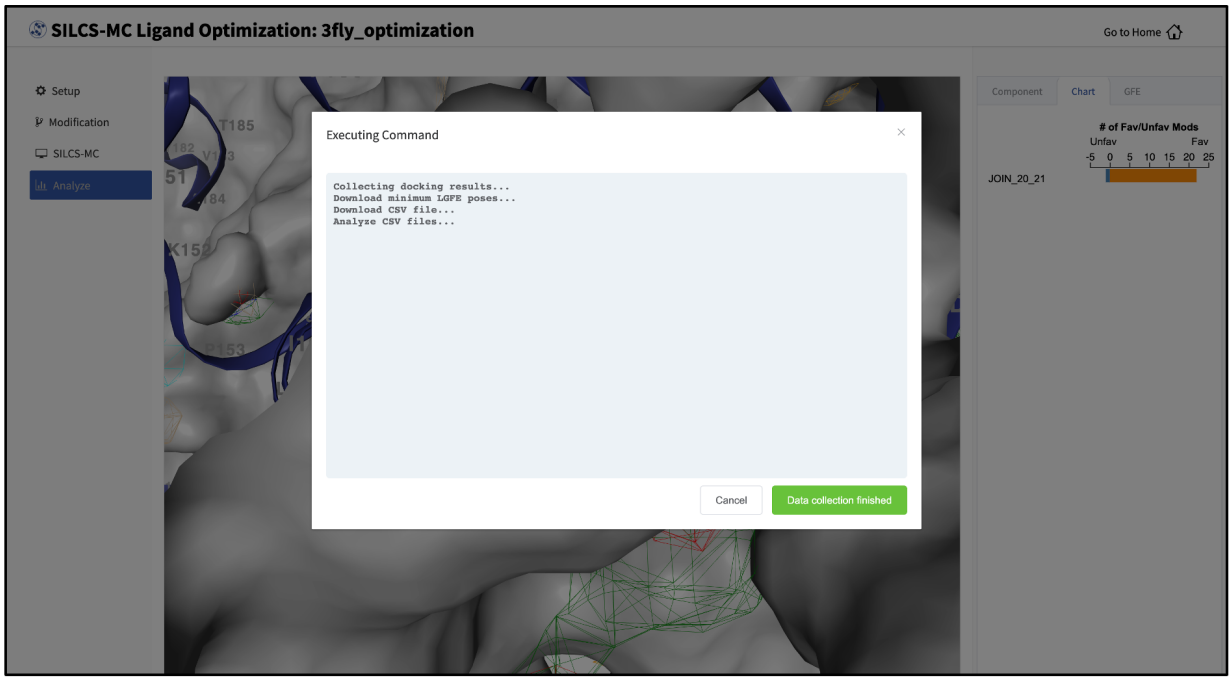


Tip: If you ran Halogen SILCS simulations for your target, you can include the Halogen SILCS FragMaps in the SILCS-MC posing and scoring by checking the "Include Halogen FragMaps" box.

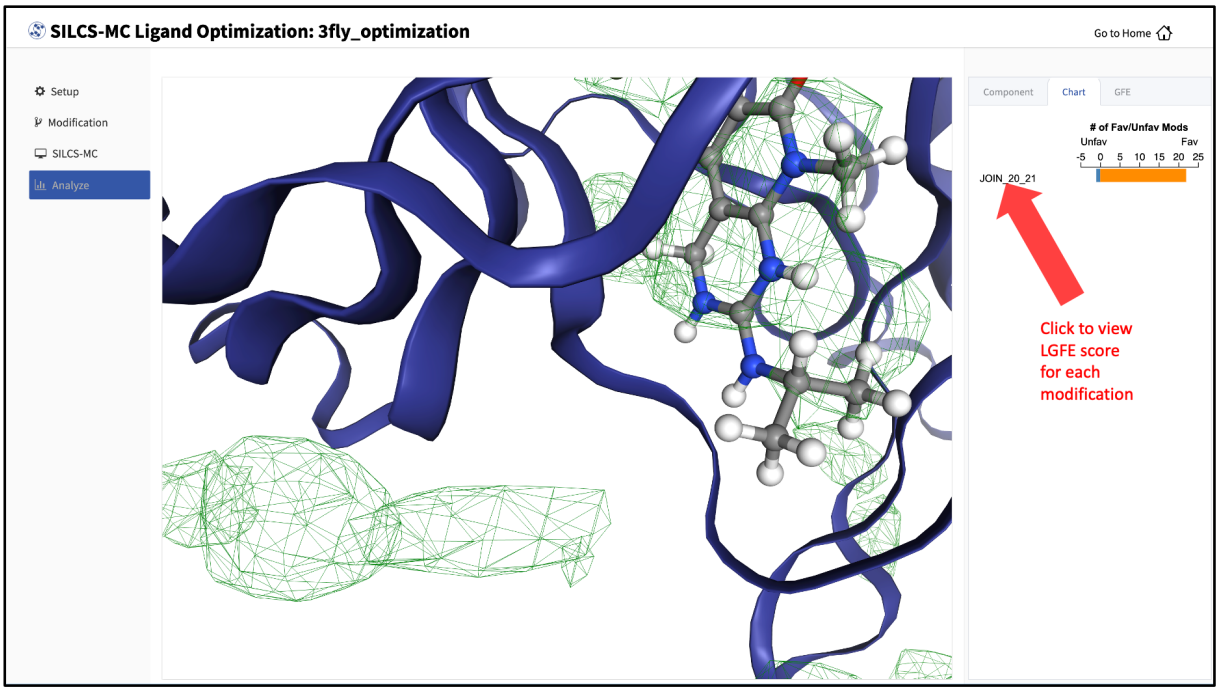
Click the "Run SILCS-MC" button to launch your jobs on the remote server.

You can monitor job progress in the SilcsBio GUI, and click on the "Show Chart" button once all job progress bars are at 100%.

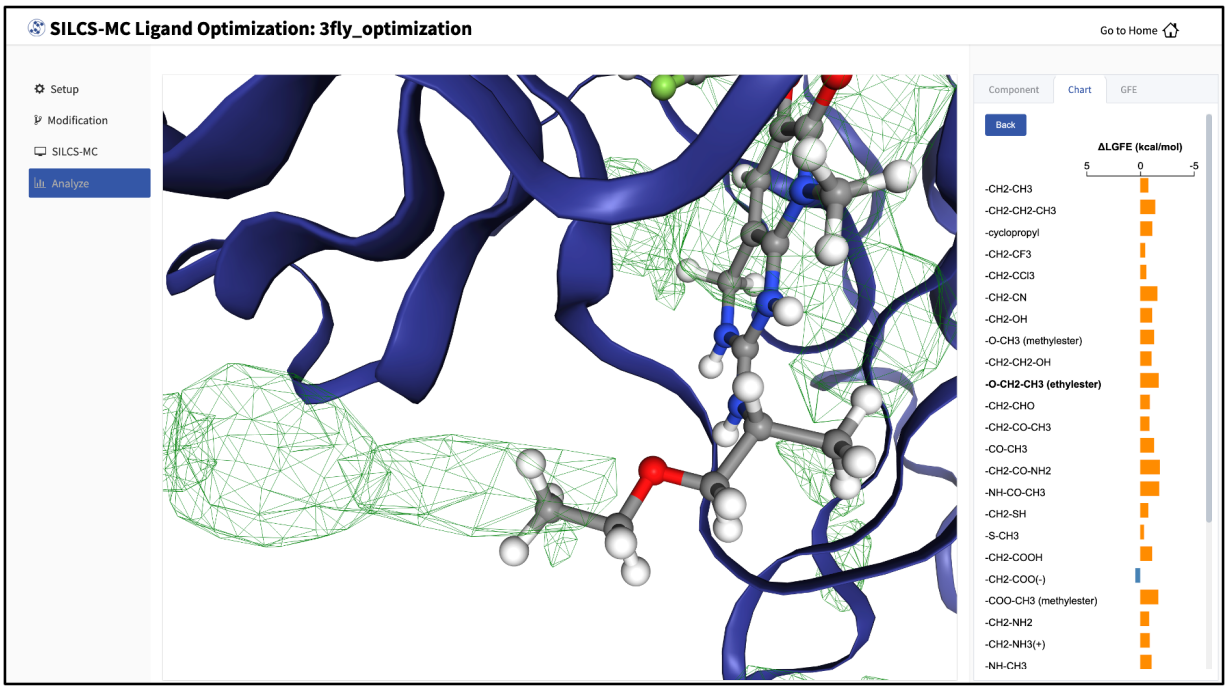
8. Clicking the "Show Chart" button after jobs have reached 100% will download the SILCS-MC results from the server. Click on "Data collection finished" to proceed.



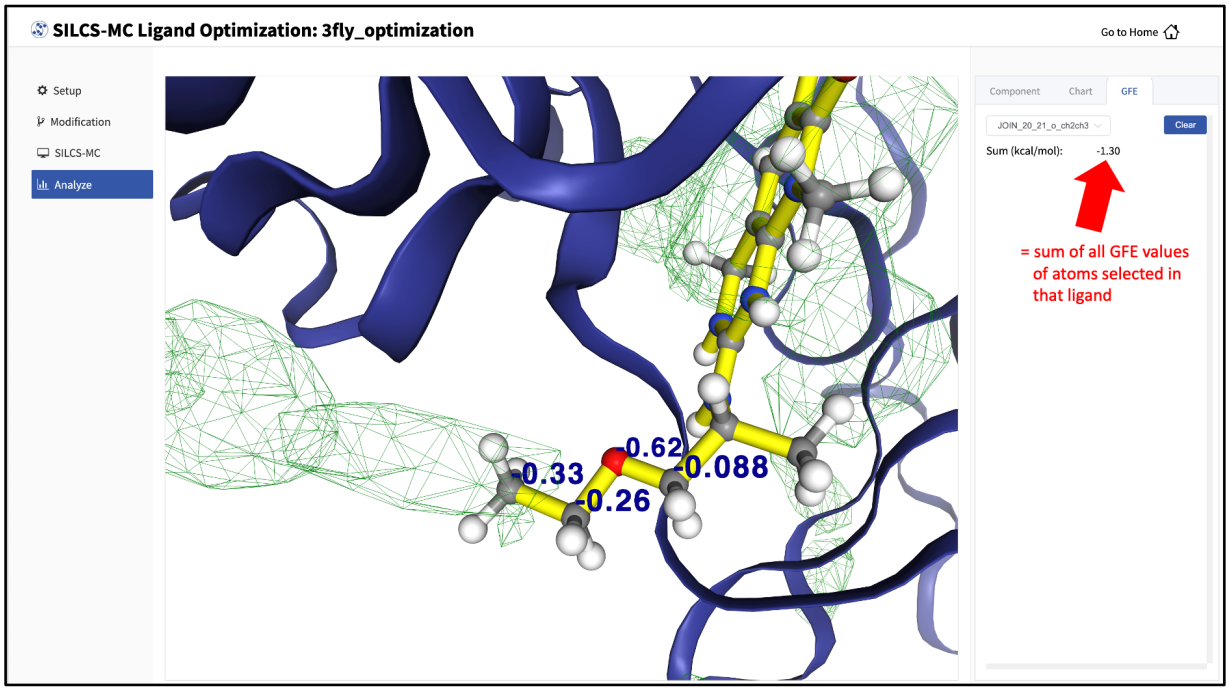
9. The "Chart" tab in the right-hand panel shows a summary of the number of favorable and unfavorable modifications associated with a particular modification site. Click on the name of the modification site in that panel to see a detailed list of the change in Ligand Grid Free Energy ($\Delta LGFE$) relative to the parent ligand for each modification at that site.



Clicking on the name of a modification will display that modification in the central panel.



The "GFE" tab in the right-hand panel can be used to reveal the contribution of each scored atom in the ligand to the overall LGFE score. Clicking on a ligand atom in the central panel will display its atomic GFE score and add it to list of atoms that are used to compute the "Sum" of GFE values for that ligand in the right-hand panel.



Detailed analysis of atomic GFE scores is an intuitive and powerful means to enable deci-

sions regarding ligand modifications. Such analysis can reveal the affinity contribution of a new functional group. It can also reveal how the new functional group may have led to a change in the orientation of the ligand in the binding pocket relative to the parent compound. In such a case, comparison of atomic GFE scores for atoms preserved between the parent and derivative can reveal the preserved atoms' contributions to the change in binding pose. See Table 2 of [23] and Figure 6 of [24] for examples.

10.3 Running SILCS-MC ligand optimization from the command line interface

Running SILCS-MC ligand optimization from the command line interface consists of two steps: creating a set of Mol2 files with the desired modifications to the parent ligand and then running SILCS-MC pose refinement on these files. Details for these steps can be found in *Chemical group transformations* and in *Running SILCS-MC pose refinement from the command line interface*, respectively.

CHAPTER

ELEVEN

SILCS-MC: DOCKING AND POSE REFINEMENT

11.1 SILCS-MC Background

The power of SILCS lies in the ability to use FragMaps to rapidly evaluate binding of diverse ligands to a target. SILCS-MC is Monte-Carlo (MC) sampling of ligands in translational, rotational, and torsional space in the field of FragMaps. MC sampling uses CGenFF intramolecular energies and the Ligand Grid Free Energy (LGFE), which is the sum of atomic Grid Free Energies (GFEs). The Exclusion Map prevents ligand sampling where no probe or water molecules visited during SILCS simulations. SILCS-MC allows for rapid conformational sampling of the ligand while accounting for protein flexibility in a mean-field-like fashion since ligand affinity and volume exclusion are embedded in the combination of FragMaps and the Exclusion Map. Final ligand scoring is based on the LGFE score [9][22].

SILCS-MC can be used to generate and score binding poses for a ligand to a target using SILCS FragMaps that have been previously computed for that target (see *SILCS: Site Identification by Ligand Competitive Saturation*). This can readily be done for a single ligand or a database of ligands. Two default conformational sampling protocols are available, "docking" and "pose refinement," as described in detail below. The docking protocol is useful when no information is available about the binding pose, as it entails extensive translational, rotational, and intramolecular conformational sampling. The pose refinement protocol is useful when a reasonable starting pose for the ligand is available, for example to re-score poses output by another high-throughput in silico screening tool. Instructions for developing custom SILCS-MC ligand conformational sampling protocols are provided at the end of this chapter.

11.2 Running SILCS-MC docking

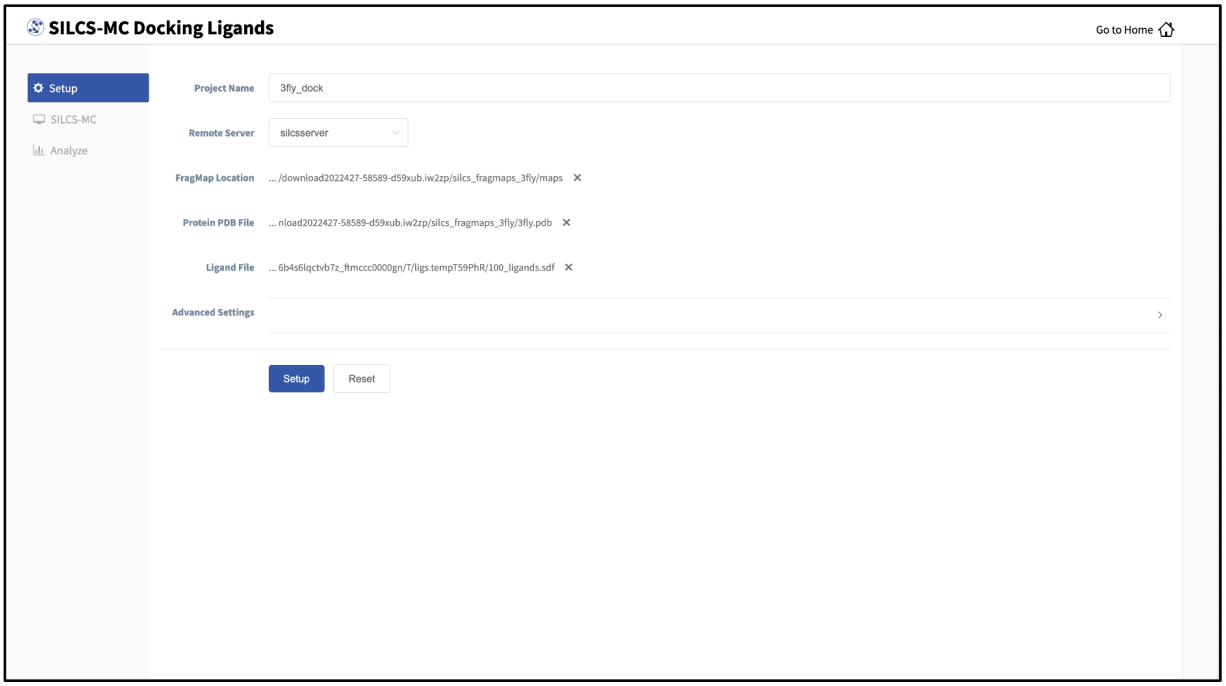
Docking is exhaustive sampling of a ligand's conformation in a given pocket to determine its most favorable orientation and internal geometry as defined by LGFE scoring. The pocket is predefined as a 10 Å sphere and the center of the pocket is taken as either the center of the supplied ligand molecule coordinates or explicitly given by the user. This protocol entails five independent MC runs with the ligand.

This protocol is recommended for ligands with diverse chemotypes and unknown binding poses. When the pose of a parent ligand is known and SILCS-MC evaluations are to be performed over a congeneric series, the pose refinement protocol is recommended instead (see below).

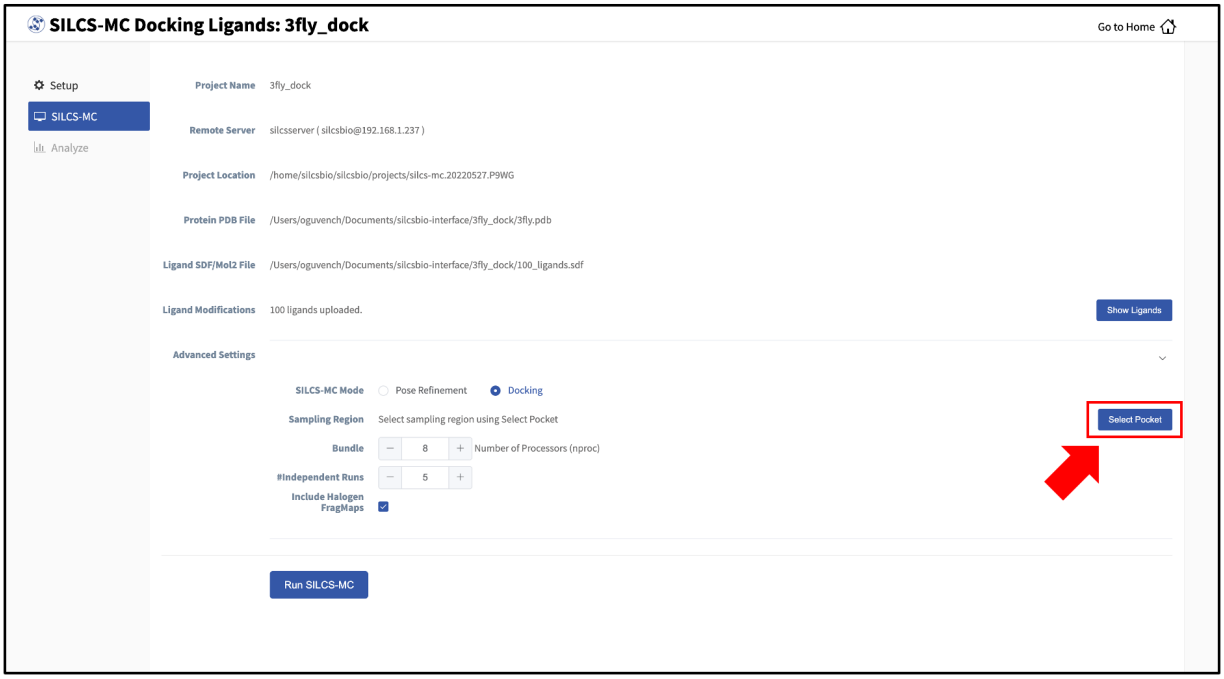
11.2.1 Running SILCS-MC docking from the SilcsBio GUI

1. Select *New SILCS-MC Docking project* from the Home page.
2. Enter a project name and select the remote server where the SILCS-MC docking jobs will run. Also, provide FragMap and protein input files. You may choose these files from the computer where you are running the SilcsBio GUI ("localhost") or from any server you have previously configured, as described in File and directory selection. You will additionally need to provide a "Ligand SDF File" that contains the database of ligands to be docked.

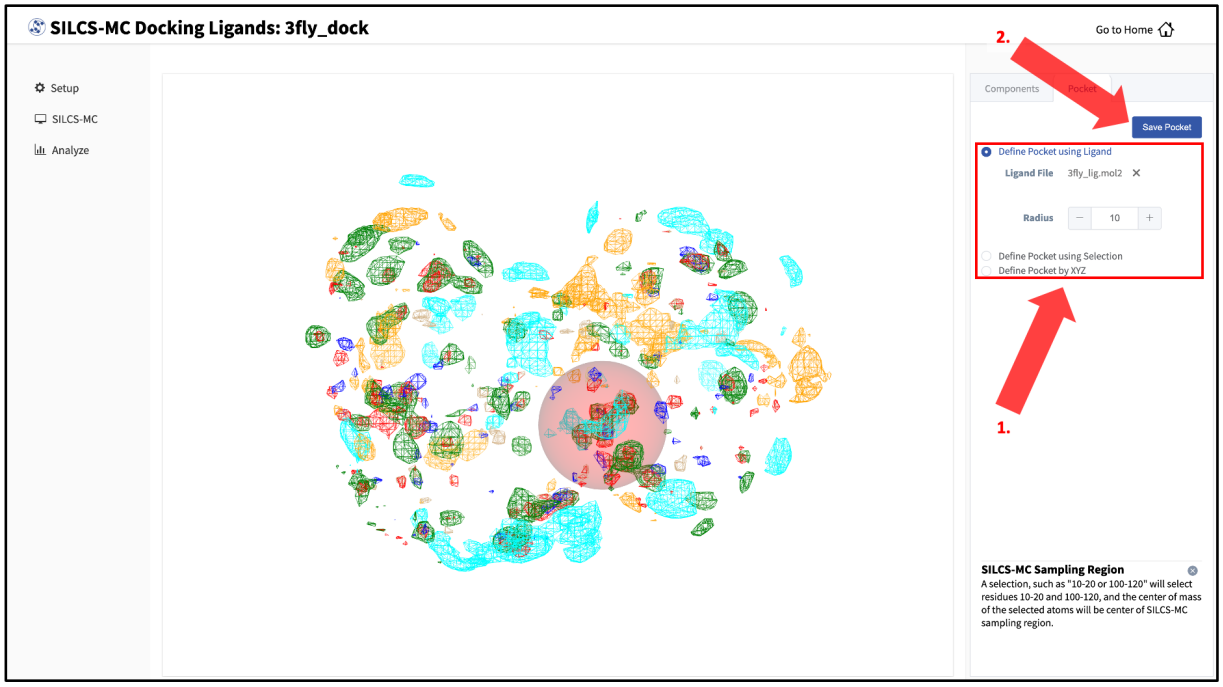
Warning: Ligands in the "Ligand SDF File" must include all hydrogens, including pH-appropriate (de)protonations, and must have reasonable three dimensional conformations.



3. Once all information is entered correctly, press the "Setup" button at the bottom of the page. The page will update to list the number of ligands and show options for the sampling protocol ("Docking" or "Pose Refinement") and the sampling region. Select the "Docking" option and then press the "Select Pocket" button.

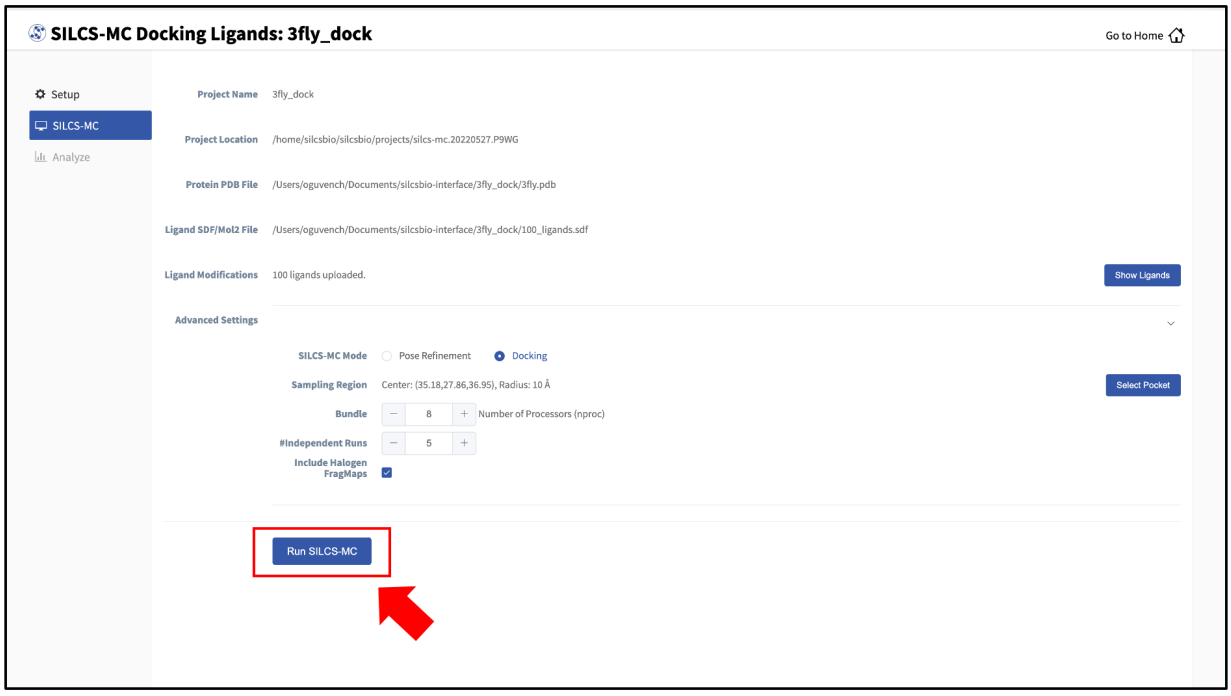


4. The GUI will now be showing the protein molecular graphic in the center pane. On the right-hand side in the "Pocket" tab, you can define the pocket center based on the center-of-geometry of a ligand pose ("Define Pocket using Ligand"), or a target residue selection ("Define Pocket using Selection"), or by directly entering an x, y, z coordinate ("Define Pocket by XYZ"). You will also need to choose a radius (default value "10") to complete the definition of the spherical pocket. If it is difficult to see the spherical pocket definition in the center pane, hide the protein surface representation. Click on the "Save Pocket" button and the "OK" acknowledgement to continue.



5. You will be returned to the previous screen, which now includes "Sampling Region" information consisting of the spherical pocket center and radius.

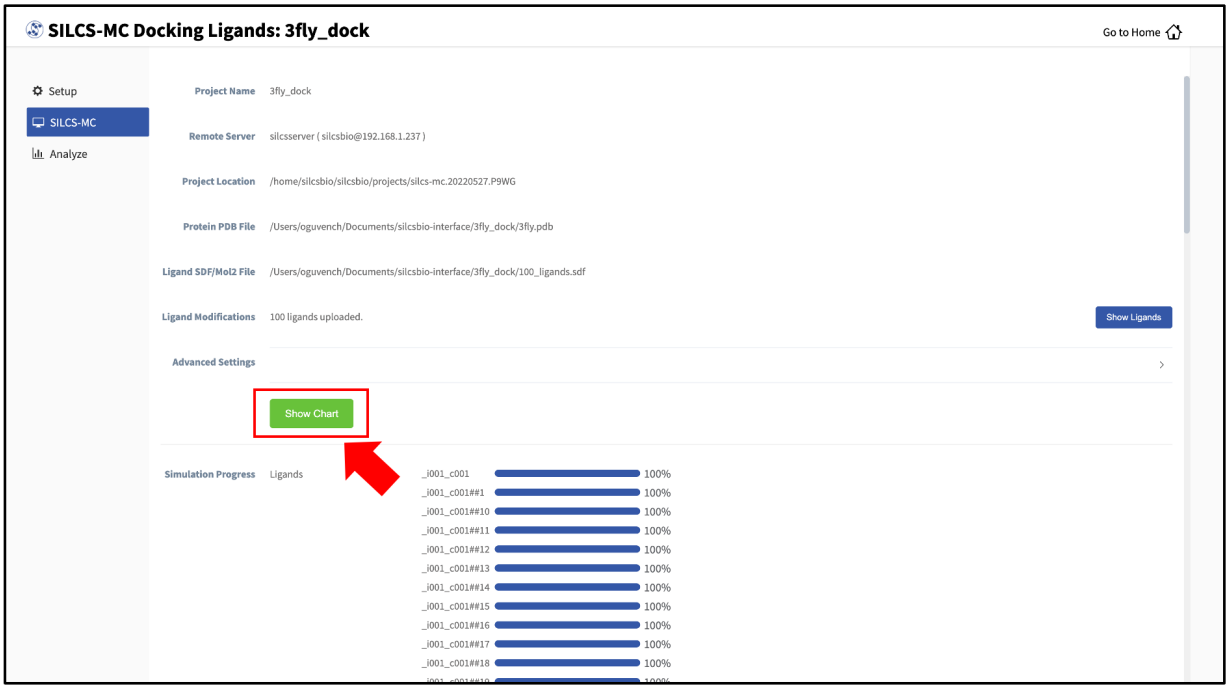
Tip: If you ran Halogen SILCS simulations for your target, you can include the Halogen SILCS FragMaps in the SILCS-MC posing and scoring by checking the "Include Halogen FragMaps" box.



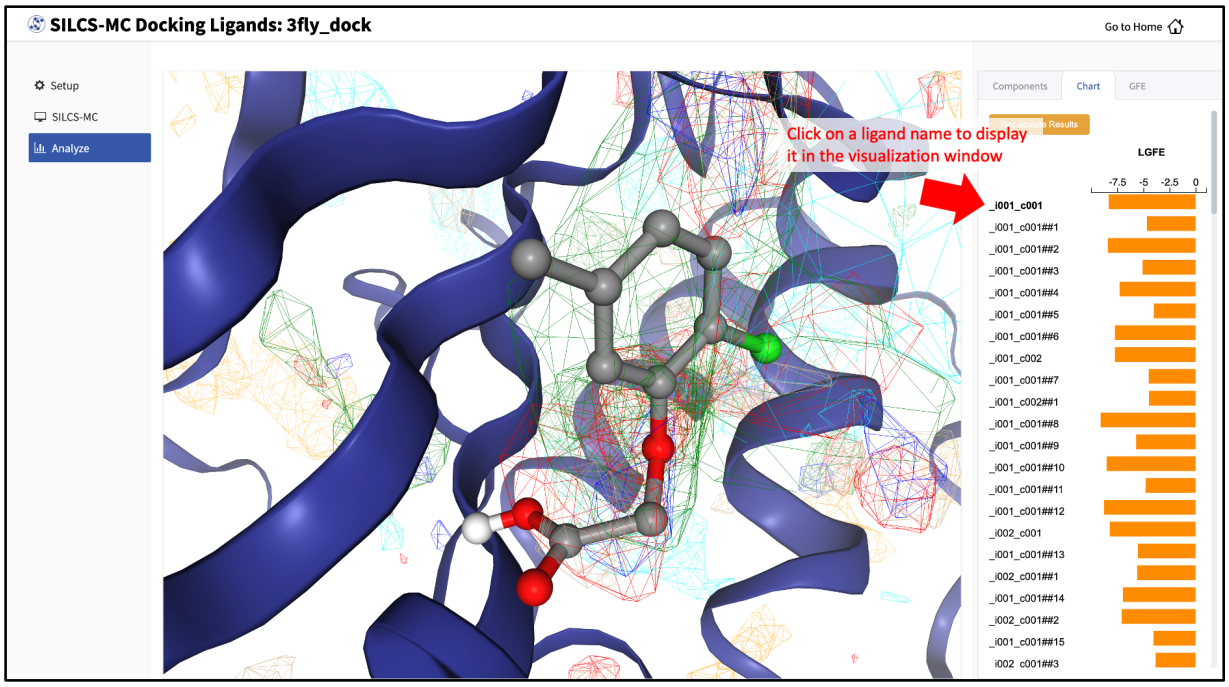
You may now click the "Run SILCS-MC" button to start the SILCS-MC docking. Doing so will submit jobs to the remote server job schedular and list them in a pop-over window.

Once all jobs are submitted, you may click on the "Setup Successful" button to dismiss the pop-over window and return to the previous screen, which will now show a "Simulation Progress" section. You may update this section by scrolling to the bottom of the screen and clicking the "Refresh" button. This will update the progress bars for all of the ligands being docked.

6. Once progress bars for all ligands reach 100%, click on the "Show Chart" button above the "Simulation Progress" section to proceed.

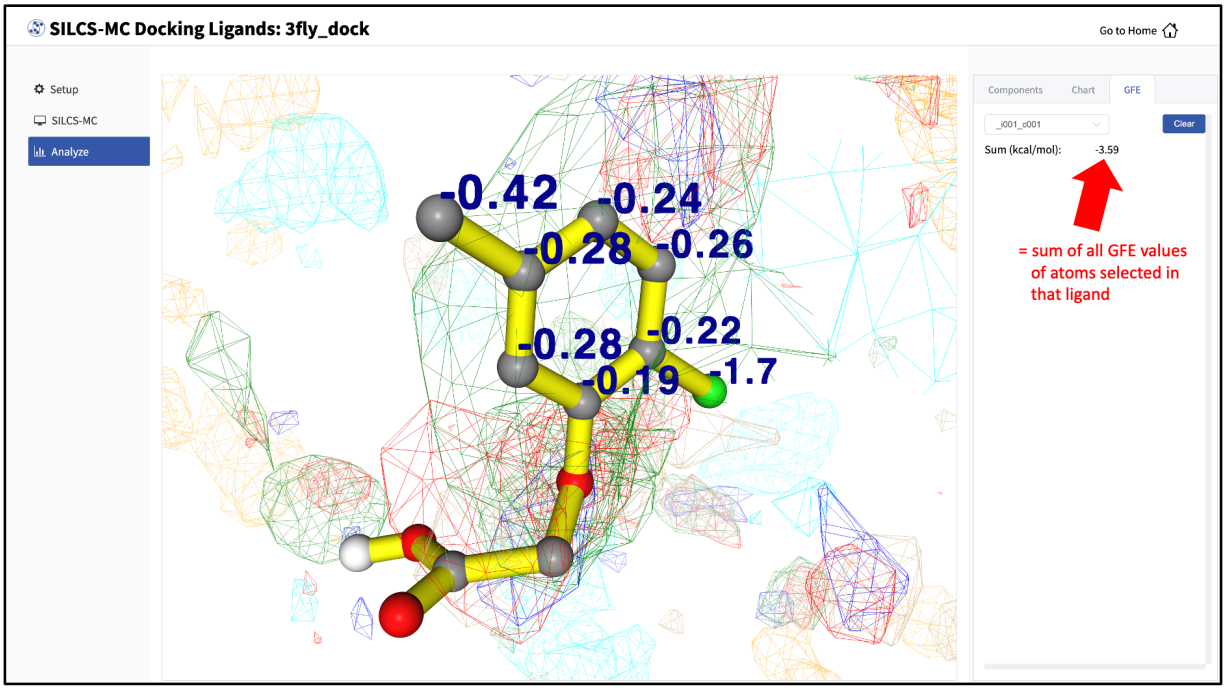


Upon successful completion of this command in the pop-over window, you may click on the green "Data collection finished" button to return to the GUI. A new tab, labeled "Chart" will have been created in the right-hand panel. Under that tab, clicking on a ligand name will display that ligand.



Choosing the "GFE" tab in the right-hand panel will allow you to click on individual atoms

within a ligand to see their atomic GFE values. The "GFE" tab will also display the sum of all GFE values of atoms selected in that ligand.



11.2.2 Running SILCS-MC docking from the command line interface

To set up and run SILCS-MC docking from the command line interface, create a directory containing all the ligands to be evaluated. Each ligand can be stored as a separate SDF or Mol2 file. Alternatively, all the ligands can be combined into a single SDF file.

```
${SILCSBIODIR}/silcs-mc/1_run_silcsmc_exhaustive \
    prot=<prot pdb> \
    ligdir=<directory containing ligand mol2/sdf> \
    mapsdir=<directory containing SILCS FragMaps> \
    center="x,y,z"
```

Note: `.sdf`, `.sd`, or `.mol2` files can be placed in the `ligdir` directory, and SILCS-MC will read a single molecule from each file. Note that if a file contains multiple molecules, use of the `ligdir` option will result in only the first molecule in the file being processed.

If you have an SDF file with multiple molecules in it, replace `ligdir=<directory containing ligand mol2/sdf>` with `sdfile=<path to sdfile>` to process all molecules in the file.

Warning: Ligands, regardless of file format, must include all hydrogens, including pH-appropriate (de)protonations, and must have reasonable three dimensional conformations.

If halogen FragMaps have been generated (see *SILCS simulation setup with halogen probes*), they can be included in the SILCS-MC calculation to improve the scoring of halogen-containing compounds. To do so, add the `halogen=true` option to the `1_run_silcsmc_exhaustive` command.

11.2.3 SILCS-MC docking protocol details

Docking spawns five independent single-core serial jobs per ligand. Each run involves a maximum of 250 cycles and a minimum of 50 cycles of Monte Carlo/Simulated Annealing (MC/SA) sampling of the ligand within the defined 10 Å sphere. Each of these 250 cycles consists of 10,000 steps of MC at a high temperature followed by 40,000 steps of SA towards a lower temperature. At the beginning of each cycle, the ligand will be reoriented within the predefined sphere. When no sphere center is defined by the user, the ligand orientation in the input SDF or Mol2 file will be used as the starting pose for each cycle. The MC sampling has three types of moves: i) molecular translations with a maximum step size of 1 Å, ii) molecular rotation with a maximum step size of 180 degrees, and iii) intramolecular dihedral rotations with a maximum step size of 180 degrees. For intramolecular dihedral rotations, only the rotatable dihedral angles are selected for MC moves. The lowest LGFE scoring pose from the MC sampling is used as starting pose in the following SA sampling. The SA sampling also involves the same three types of moves, but with a smaller step size compared to the MC sampling: i) molecular translations with a maximum step size of 0.2 Å, ii) molecular rotation with a maximum step size of 9 degrees and, iii) intramolecular dihedral rotations with a maximum step size of 9 degrees. The lowest LGFE scoring pose from the SA is saved in a multi-frame PDB file: `3_silcsmc/<run>/pdb/<lig>_mc.<run>.pdb`.

In each run, after a minimum of 50 MC/SA cycles, if the LGFE score difference between the top three poses (defined by lowest LGFE scores) are less than 0.5 kcal/mol, then that run is considered converged and terminated. If the top three scored poses are separated by more 0.5 than kcal/mol, the MC/SA procedure continues either until the convergence criterion is met or until a maximum of 250 MC/SA cycles have been completed.

11.3 Running SILCS-MC pose refinement

The pose refinement protocol is designed to limit conformational sampling near the ligand input pose supplied by the user. Pose refinement is appropriate when there is high confidence in the input parent ligand pose and SILCS-MC evaluations are to be performed over a congeneric series. The sphere center for the pocket definition is the center-of-geometry from the input ligand pose.

11.3.1 Running SILCS-MC pose refinement from the SilcsBio GUI

Please see the previous section on running SILCS-MC docking from the SilcsBio GUI, and, in Step 3, select the "Pose Refinement" option. Note that there will be no "Select Pocket" step, as pose refinement assigns the pocket center based on the center-of-geometry of the input ligand. For multiple input ligands in a single `.sdf` file, each ligand will have its own center-of-geometry used to define the pocket center.

11.3.2 Running SILCS-MC pose refinement from the command line interface

To set up and run SILCS-MC pose refinement, create a directory containing all the ligands to be evaluated. Each ligand can be stored as a separate `.mol2` or `.sdf` file. Alternatively, all the ligands can be combined into a single `.sdf` file.

```
${SILCSBIODIR}/silcs-mc/1_run_silcsmc_local prot=<prot pdb> \
    ligdir=<directory containing ligand mol2/sdf> \
    mapsdir=<directory containing SILCS FragMaps>
```

Note: `.sdf`, `.sd`, or `.mol2` files can be placed in the `ligdir` directory, and SILCS-MC will read a single molecule from each file. Note that if a file contains multiple molecules, use of the `ligdir` option will result in only the first molecule in the file being processed.

If you have an SDF file with multiple molecules in it, replace `ligdir=<directory containing ligand mol2/sdf>` with `sdfile=<path to sdfile>` to process all molecules in the file.

Warning: Ligands, regardless of file format, must include all hydrogens, including pH-appropriate (de)protonations, and must have reasonable three dimensional conformations.

If halogen FragMaps have been generated (see *SILCS simulation setup with halogen probes*), they can be included in the SILCS-MC calculation to improve the scoring of halogen-containing compounds. To do so, add the `halogen=true` option to the `1_run_silcsmc_local` command.

11.3.3 SILCS-MC pose refinement protocol details

Pose refinement spawns five independent single-core serial jobs per ligand, and each run involves a maximum of 10 cycles of MC/SA sampling of the ligand within a 1 Å sphere. The center of the sphere is defined as the center-of-geometry of the input ligand pose. Each of cycles consists of 100 steps of MC at high temperature followed by 1000 steps of SA towards a lower temperature.

At the beginning of each cycle, the ligand orientation/conformation will be reset to the one found in the input file. MC sampling moves are: i) molecular translation with a maximum step size of 1 Å, ii) molecular rotation with a maximum step size of 180 degrees, and iii) intramolecular dihedral rotation with a maximum step size of 180 degrees. For intramolecular dihedral rotation, only the rotatable dihedral angles are selected for MC moves. The lowest LGFE scoring pose from the MC sampling is used as starting pose in the following SA sampling. SA sampling moves are smaller than for the MC phase: i) molecular translation with a maximum step size of 0.2 Å, ii) molecular rotation with a maximum step size of 9 degrees, iii) intramolecular dihedral rotation with a maximum step size of 9 degrees. The lowest LGFE scoring pose from the SA is saved in the multi-frame PDB file `3_silcsmc/<run>/pdb/<lig>_mc.<run>.pdb`.

11.4 Best-pose retrieval

Once the SILCS-MC simulation is finished, retrieve the LGFE scores for each ligand subjected to SILCS-MC using:

```
${SILCSBIODIR}/silcs-mc/2_calc_lgfe_min_avg_sd ligdir=<directory␣
↪containing lig mol2>
```

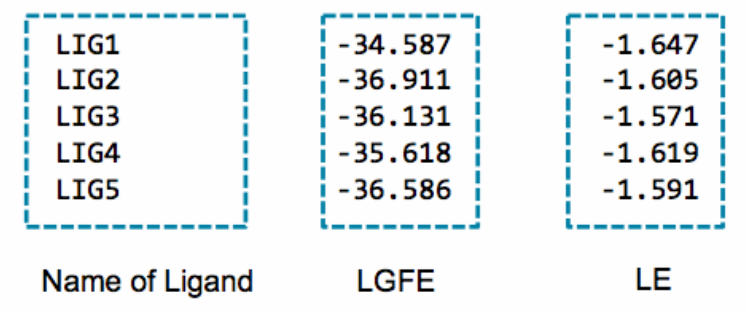
Note: If you used `sdfile=<path to sdfile>` option in the previous step, use `sdfile=<path to sdfile>` in this step as well, instead of `ligdir` option.

The above command will collect the SILCS-MC result and create `3_silcsmc/lgfe.csv` file, which contains the best LGFE scores from SILCS-MC sampling per ligand molecule. The command will collect top 10 best scoring poses and put them in `3_silcsmc/minconfpdb/` folder in SDF format.

Note: To change the number of poses, add `npose=<number>` at the end of `2_calc_lgfe_min_avg_sd` command.

To output PDB file format for best scoring poses, add `pdb=true` at the end of `2_calc_lgfe_min_avg_sd` command.

An example of the output of this script is:



An alternative to the LGFE score is the ligand efficiency (LE). The LE is calculated as the LGFE score divided by the number of heavy atoms in each ligand.

$$LE = \frac{LGFE}{N_{\text{HeavyAtoms}}}$$

11.5 User-defined protocols

In addition to the default docking and pose refinement protocols, users can define their own SILCS-MC protocols. To do so, copy `${SILCSBIODIR}/templates/silcs-mc/params_custom.tmpl` to the location where you intend to run your custom SILCS-MC protocol. Edit this copy to reflect your customization; see below for a detailed description of user-definable parameters. Parameter values in angle brackets in this file, such as `<SILCSBIODIR>`, will be replaced automatically at runtime where possible.

After you have edited `params_custom.tmpl`, use the following command to set up and run SILCS-MC. Each input ligand can be stored as a separate SDF or Mol2 file. Alternatively, all input ligands can be combined in a single SDF file.

```
${SILCSBIODIR}/silcs-mc/1_run_silcsmc_custom prot=<prot pdb> \
    ligdir=<directory containing ligand sdf/mol2> \
    mapsdir=<directory containing SILCS FragMaps> \
    paramsfile=<params_custom.tmpl>
```

The number of runs that will be spawned can also modified with the command-line parameter `totruns`:

```
${SILCSBIODIR}/silcs-mc/1_run_silcsmc_custom prot=<prot pdb> \
    ligdir=<directory containing ligand sdf/mol2>
    mapsdir=<directory containing SILCS FragMaps>
    paramsfile=<params_custom.tmpl>
    totruns=<# of runs>
```

The full list of user-definable parameters for `params_custom.tmpl` is:

- `CGENFF_RULES <cgenff rules_file>` (required)

This file is needed by the internal CGenFF library to determine the correct force-field parameters for the ligand. The default value is `${SILCSBIODIR}/data/cgenff/cgenff.rules`

- `CGENFF_PAR <cgenff parameter file>` (required)

  Along with the CGENFF_RULES file, this file is needed by the internal CGenFF library to determine the correct force-field parameters for the ligand. The default value is `${SILCSBIODIR}/data/cgenff/par_all36_cgenff.prm`

- `SILCS_RULES <silcs rules file>` (required)

  This rules file is used to map the different atoms in the ligand to the corresponding SILCS FragMap types. This mapping is used to determine the appropriate "field" that will be applied to the different atoms in the ligand when attempting an MC-move. The default value is `${SILCSBIODIR}/data/silcs/silcs_classification_rules_2021_generic_apolar_scale_1e.dat`

  When `silcs_classification_rules_2021_generic_apolar_scale_1e.dat` is used, ligand atoms are assigned using generic classifications for mapping back to the FragMaps.

  Additional rules files are available in the `${SILCSBIODIR}/data/silcs/` for other mapping schemes, including for more specific classifications and for using halogen FragMaps (see *SILCS simulation setup with halogen probes*).

- `GFE_CAP <default: 3.0>`

  Maximum allowable unfavorable GFE (kcal/mol) in the MC calculation.

- `RDIE <default: true>`

  When true, the distance dependent dielectric (RDIE) scheme is used to treat intramolecular electrostatics. When false, CDIE (constant dielectric scheme) is used.

- `DIELEC_CONST <default: 4>`

  Dielectric constant used in the intramolecular electrostatic interactions calculations.

`#MINIMIZE_INPUT <default: false>`

Perform minimization of input structure.

`#MINIMIZE_BFGS <default: false>`

Perform minimization of input structure using BFGS algorithm.

- `MIN_STEPS <default: 10000>`

  Maximum number of steps of minimization performed using the steepest-descent algorithm with the ligand, before initiating MC simulation.

- `EMTOL <default: 0.01>`

Minimization is converged when the diff in total energy (totE) across the last 10 steps is smaller than this value. Once this criteria is satisfied minimization terminates.

- `MC_MOVE_RANGE <default:1.0 180.0 180.0>`.

  Maximum range of translation, rigid body rotation and dihedral rotation per step of MC simulation.

- `MC_PRNT_FRQ <default:  0>`

  Number of intermediate steps of MC to be written into OUTMCPDBFILE.

- `MC_STEPS <default:  10000>`

  Number of steps of MC simulation to be performed per cycle.

- `SIM_ANNEAL_MOVE_RANGE <default:0.2 9.0 9.0>`

  Maximum range of translation, rigid body rotation and dihedral rotation per step of simulated annealing simulation after MC simulation.

- `SIM_ANNEAL_STEPS <default:  40000>`

  Number of steps of simulated annealing to be performed per cycle.

- `INIT_RUNS <default:  50>`

  Number of MC/SA cycles before initiating checks for convergence.

- `NUM_TOL <default:  3>`

  Number of top-scoring cycles with differences in LGFE less than DELTAE_TOLERANCE, before this simulation (run) is considered converged.

- `DELTAE_TOLERANCE <default:  0.5>`

  When differences in LGFE of NUM_TOL most-favorable cycles are less than this defined tolerance value, convergence is reached and the program exits

- `DELTAE_BUFF <default:  10>`

  Progression of MC+SA from one cycle to next is such that LGFE (of lowest conf) from MC should be less than (prev_min+deltae_buff). This ensures that N cycles are proceeding towards a minimum lower than that previously discovered lowest energy conformation.

- `TOTE_CRITERIA <default:  false>`

  When true, instead of LGFE, total energy (totE) of the system is used for convergence checks. Useful when running vacuum-phase MC simulations of the ligand.

- `TOT_RUNS <default:  250>`

  Maximum number of MC simulation cycles. The program terminates if the `DELTAE_TOLERANCE` criteria is satisfied before reaching `TOT_RUNS`. Alternately, even if the `DELTAE_TOLERANCE` criteria is not satisfied when the number of cycles executed reaches `TOT_RUNS`, the program terminates.

- RANDOM_SEED: <default: system-time>

  Seed used in MC simulation. When not set, system-time is used as a seed.

- SIMULATION_CENTER: <x,y,z>

  Cartesian coordinates of where the MC simulation should be performed.

- SIMULATION_RADIUS: <default: 10.0 A>

  Radius of the sphere within which MC simulation will be performed.

- RANDOM_INIT_ORIENT: <true/false>

  When set to TRUE, SIMULATION_CENTER should also be set. The ligand is placed within a sphere of size SIMULATION_RADIUS in a random orientation and a random conformation.

  When set to FALSE, then the center-of-geometry of the ligand is used as the center for the MC simulation. This is useful when the ligand pose in the pocket is well known.

- ATOM_TO_RESTRAIN: <atom number in sdf/mol2>

  When set, a spherical potential is applied to restrain the defined atom within the sphere during MC moves. This enables geometrically restraining a particular pharmacophore feature. Note, when using this feature, supply the full molecule with explicit hydrogens already added. Rrandom pocket pose and placement using RANDOM_INIT_ORIENT true is incompatible with this option.

  When not set, the entire molecule is free to rotate/move/translate

- ATOM_RESTRAINT_CENTER: <x,y,z>

  To be used in conjunction with ATOM_TO_RESTRAIN option. This value is used to defined the center of the spherical potential.

- ATOM_RESTRAINT_RADIUS: <default: 1.0 A>

  To be used in conjunction with ATOM_TO_RESTRAIN option. This value is used to defined the radius of the spherical potential. When not defined, then a default of 1 A is used.

- OUTRMSDFILE <output RMSD file>

  This file stores the RMSD and LGFE of the lowest energy conformation from each run of the MC/SA simulation. To be used in conjunction with RANDOM_INIT_ORIENT set to true.

- SILCSMAP <MapType> <map name> <scaling factor> (required)

  Multiple SILCSMAP entries are typically defined, with each entry pointing to a FragMap filename <map name>. <scaling factor> is used to scale atomic Grid Free Energies (GFEs) for <MapType> atoms in the ligand being scored. <MapType> entries here must correspond to those defined in the SILCS_RULES rules file described above.

- OUTPUT_FORMAT [SDF|PDB] (required)

Choice of output format. SDF or PDB is supported.

- `OUTPUT_FILE <output file name>` (required)

  This file stores the lowest energy conformation from each cycle of the MC/SA simulation.

- `LOGFILE <output log file>`

  This file stores the energy statistics of the lowest energy conformation from each cycle of the MC/SA simulation.

CHAPTER

TWELVE

SILCS-PHARM: RECEPTOR-BASED PHARMACOPHORE MODELS FROM FRAGMAPS

12.1 Background

Three-dimensional (3-D) pharmacophore models (also called "hypotheses") offer an intuitive and powerful approach for virtual screening (VS). 3-D pharmacophore VS works by searching for matches between the 3-D pattern of functional groups constituting a pharmacophore model and the 3-D arrangement of functional groups in ligand conformers in a virtual database. Ligands having conformers that closely match the pharmacophore model are considered hits. Traditionally, 3-D pharmacophore models were developed using experimental knowledge of the binding poses for ligands to a receptor. This is in contrast to energy function-based VS ("docking"). An advantage of docking approaches was that they only require experimental knowledge of the receptor structure, and not of bound ligands. However, it is possible to develop pharmacophore models using only the receptor structure. One means to generate such receptor-based pharmacophore models is with data from SILCS simulations.

SILCS-Pharm converts Grid Free Energy (GFE) FragMaps into 3-D pharmacorphore models. GFE FragMaps from SILCS simulations are used as inputs for the four-step SILCS-Pharm process:

1. voxel selection;
2. voxel clustering and FragMap feature generation;
3. FragMap feature to pharmacophore feature conversion;
4. generation of a pharmacophore model for virtual screening (VS).

Here, "feature" refers to the identity and location of a chemical functional group and "pharmacophore model" is a collection of pharmacophore features.

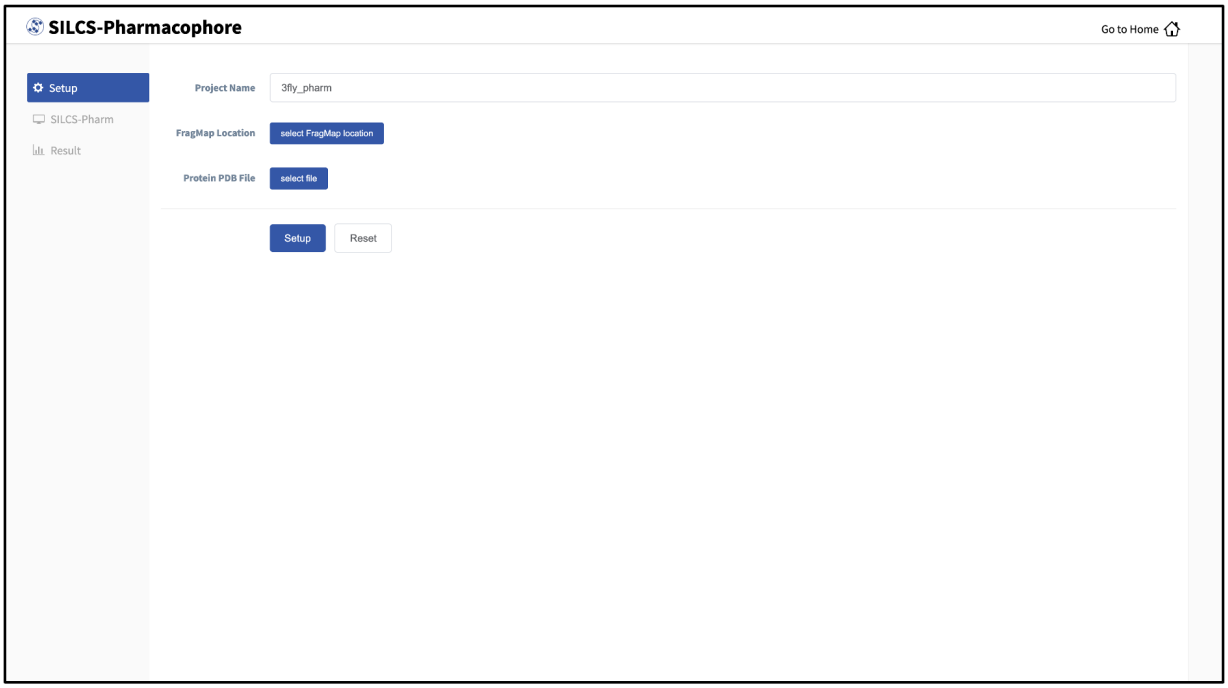
FragMap generation from SILCS MD entails partitioning 3-D space into uniform cubic voxels and enumerating fragment binding probabilities for each voxel. The voxels are retained during Boltzmann-inversion of probabilities to create GFE FragMaps. As the GFE voxel value represents the binding strength of a functional group at that specific location on the protein surface, the first step identifies the most favorable interactions based on a particular GFE cutoff value. In the second step, clustering is performed to group adjacent voxels, with each unique cluster becoming a

FragMap feature. In the third step, the FragMap features are classified and converted into pharmacophore features. Finally, the pharmacophore features are prioritized using Feature GFE (FGFE) scores to create a SILCS 3-D pharmacophore model [11]. In the current scheme, the program searches through five FragMap types: Generic Apolar, Generic Donor, Generic Acceptor, Methylammonium N, and Acetate O. Additionally, instead of using a rigidly-defined protein surface to determine regions that ligands cannot sample because of protein volume occlusion, the SILCS Exclusion Map is used. The Exclusion Map has been validated to be a better alternative to more traditional representations of the occluded volume of the protein, and it takes protein flexibility into account in an explicit way.

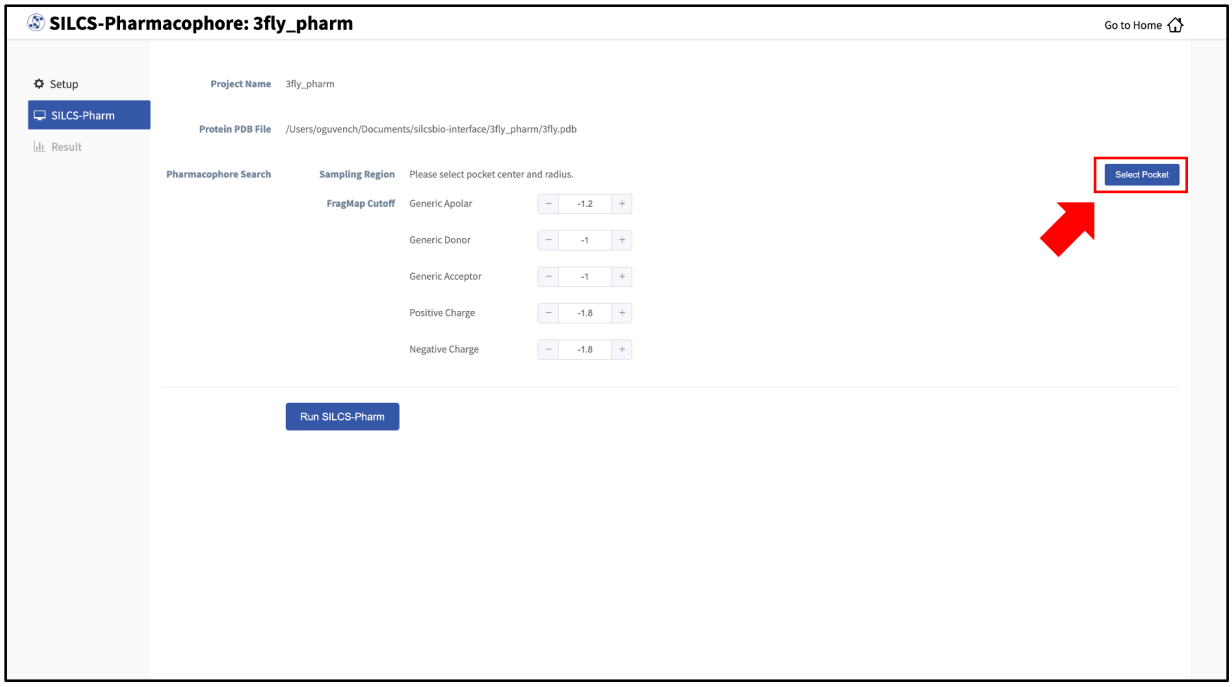
12.2 Running SILCS-Pharm

12.2.1 Running SILCS-Pharm from the SilcsBio GUI

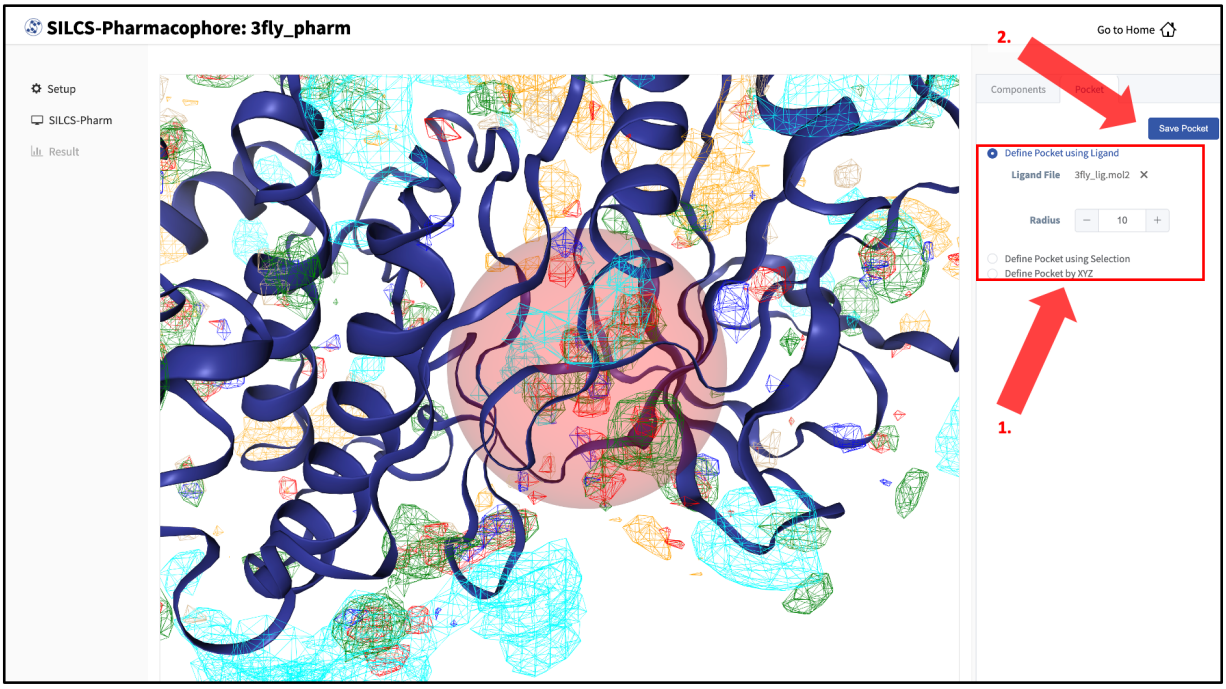
1. Select *New SILCS-Pharm project* from the Home page.
2. Enter a project name. Then, provide FragMap and protein input files. You may choose these files from the computer where you are running the SilcsBio GUI ("localhost") or from any server you have previously configured, as described in File and directory selection.



3. Once all information is entered correctly, press the "Setup" button at the bottom of the page. The screen will update to add "Pharmacophore Search" to the list of information. Click the "Select Pocket" button on the right-hand side of the window.

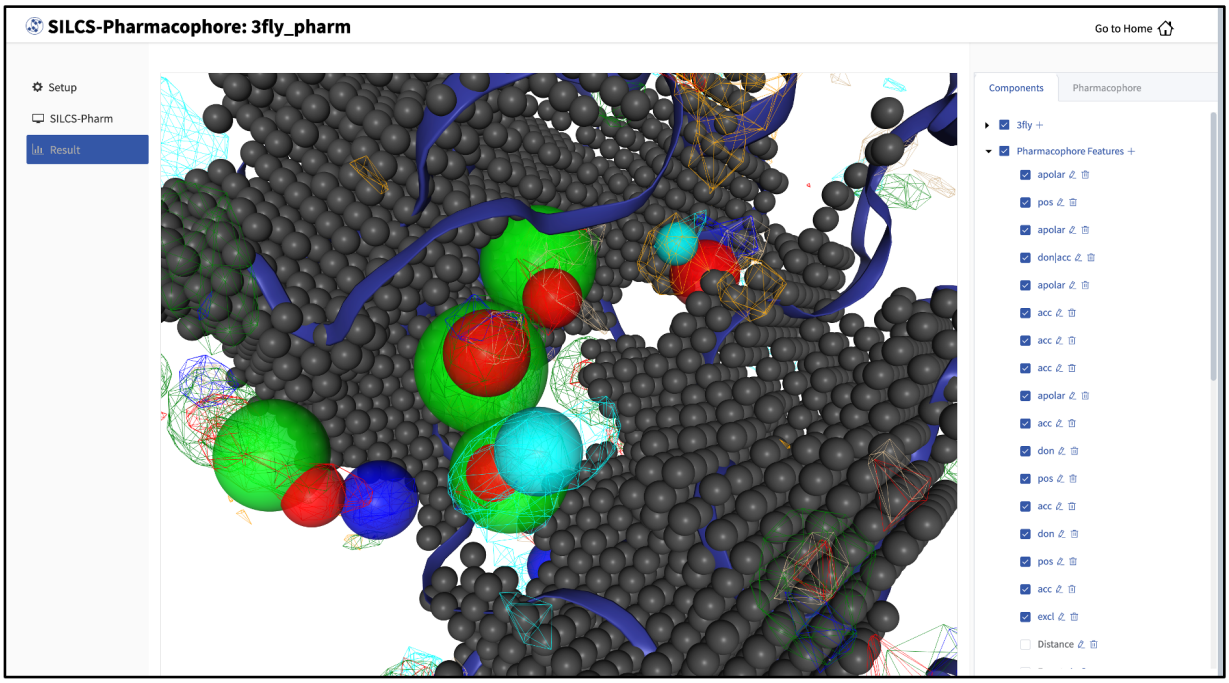


4. The GUI will now be showing the protein molecular graphic in the center pane. On the right-hand side, in the "Pocket" tab, you will need to define the pocket center based on the center-of-geometry of a ligand pose ("Define Pocket using Ligand"), or a target residue selection ("Define Pocket using Selection"), or by directly entering an x, y, z coordinate ("Define Pocket by XYZ"). You will also need to choose a radius (default value "10") to complete the definition of the spherical pocket. If it is difficult to see the spherical pocket definition in the center pane, hide the protein surface representation. Click on the "Save Pocket" button and the "OK" acknowledgement to continue.

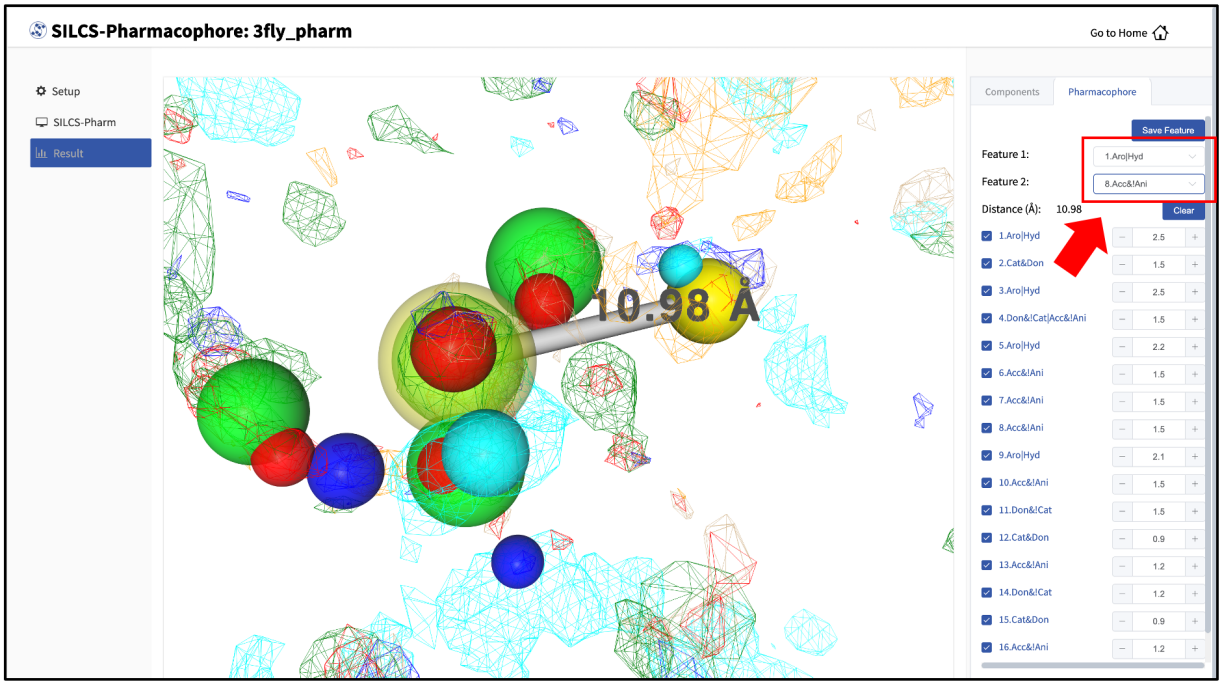


5. You will be returned to the previous screen, which now includes "Sampling Region" information consisting of the spherical pocket center and radius. You will also see default FragMap cutoff values listed for selection of FragMap densities for creation of pharmacophore features. You may adjust those values if you desire. Click on the "Run SILCS-Pharm" button to run the four-step SILCS-Pharm process.

A pop-over window will appear and show job output, and the job will run to completion in a matter of seconds. You may wish to click on the "Components" tab in the right-hand panel and deselect the protein for easier viewing.



The "Pharmacophore" tab will list all the pharmacophore features automatically generated from the FragMaps. You may adjust the radii of the individual features or deselect them entirely. You can also determine the distance between any two features to assist your analysis and choices.



Once finished with pharmacophore feature selection and adjustment, click the "Save Feature" button. This will allow you to save your pharmacophore model containing your selected/adjusted features in `.ph4` format on your local computer. By selecting/adjusting different subsets of the pharmacophore features, including the Exclusion Map, you may create multiple different pharmacophore models and save each one as a separate `.ph4` file. This capability allows for different pharmacophore model screens of a binding site. The resulting diverse hits from those screens can be combined and, for example, subjected to SILCS-MC Pose Refinement for rescoring.

12.2.2 Running SILCS-Pharm from the command line interface

A single command line interface command performs the four-step SILCS-Pharm process:

```
${SILCSBIODIR}/silcs-pharm/1_calc_silcs_pharm prot=<prot pdb> center=
↪"x,y,z"
```

The input arguments are the PDB file used for the SILCS run and the absolute position of the center of a 10 Å sphere to be used to define the boundaries of the pharmacophore model. Two output files result from this command. `<prot>.keyf_<#features>.ph4` can be directly used for 3-D pharmacophore VS by compatible programs (see below for generating Pharmer-compatible ph4 files). `<prot>_silcspharm_features.pdb` provides output in PDB format for easy visualization using standard molecular graphics packages. Running the command with no arguments

```
${SILCSBIODIR}/silcs-pharm/1_calc_silcs_pharm
```

will list additional options. Along with the required arguments of `prot=<prot pdb>` and `center="x,y,z"`, the following additional options can be set:

1) FragMap directory path:

```
mapsdir=<location and name of directory containing FragMaps;␣
↪default=2b_gen_maps>
```

By default, the program looks for FragMaps in the `2b_gen_maps` directory.

2) Output directory path:

```
outputdir=<location and name of output directory; default=5_pharm>
```

By default, the program creates the directory `5_pharm` and places all output files there.

3) Radius:

```
radius=<default: 10>
```

By default, a radius of 10 Å centered at center="x,y,z" is searched to generate pharmacophore features from the input FragMaps.

4) Generic Apolar FragMap cutoff:

```
apolar_cutoff=<default: -1.2>
```

By default, Generic Apolar FragMap voxels having a GFE value <= -1.2 kcal/mol are selected.

5) Generic Donor FragMap cutoff:

```
hbdon_cutoff=<default: -1.0>
```

By default, Generic Donor FragMap voxels having a GFE value <= -1.0 kcal/mol are selected.

6) Generic Acceptor FragMap cutoff:

```
hbacc_cutoff=<default: -1.0>
```

By default, Generic Acceptor FragMap voxels having a GFE value <= -1.0 kcal/mol are selected.

7) Methylammonium N FragMap cutoff:

```
mamn_cutoff=<default: -1.8>
```

By default, Methylammonium N FragMap FragMap voxels having a GFE value <= -1.8 kcal/mol are selected.

8) Acetate O FragMap cutoff :

```
aceo_cutoff=<default: -1.8>
```

By default, Acetate O FragMap FragMap voxels having a GFE value <= -1.8 kcal/mol are selected.

In addition to visualization, the output PDB file `<prot>_silcspharm_features.pdb` can be used for easy editing of the pharmacophore model. Using a text editor, modify/reduce the features in this file as desired and save the revised file as `<prot>_silcspharm_features_revised.pdb`. Using this revised PDB file and the original ph4 file `<prot>.keyf_<#features>.ph4` as input, create a new ph4 file with:

```
${SILCSBIODIR}/programs/revise_ph4 <prot>.keyf_<#features>.ph4
↪<prot>_silcspharm_features_revised.pdb
```

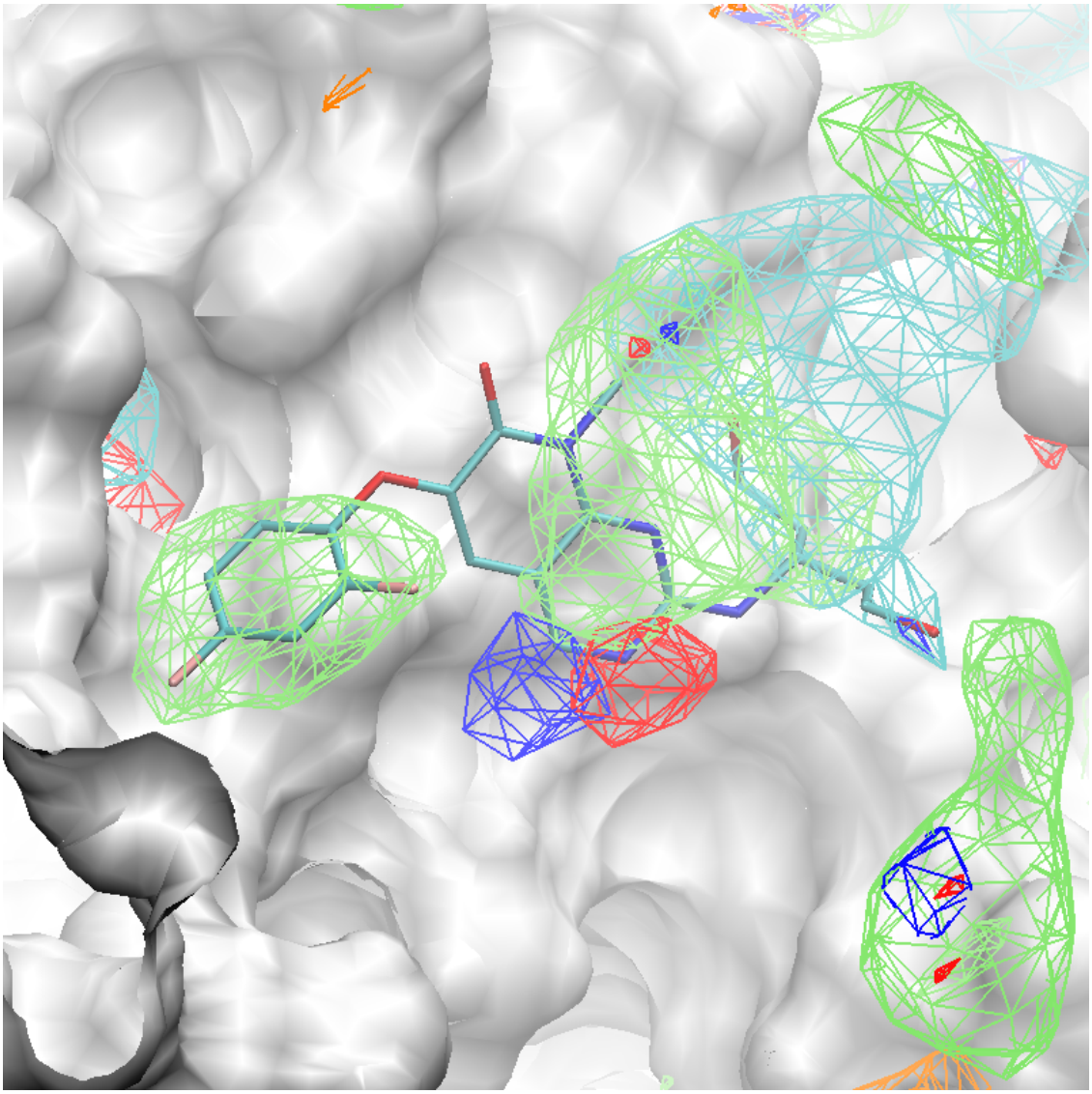
The output of this command will be the revised ph4 file `<prot>.keyf_<#features>_revised.ph4`. To create Pharmer-compatible ph4 output, add the `pharmer` option:

```
${SILCSBIODIR}/programs/revise_ph4 <prot>.keyf_<#features>.ph4
↪<prot>_silcspharm_features_revised.pdb pharmer
```

12.2.3 Example

The following example demonstrates use of SILCS-Pharm from the command line interface to generate a pharmacophore model for p38 MAP kinase. Input files, including FragMaps, are provided in `${SILCSBIODIR}/examples/silcs/` (these same input files may also be used for a trial run of SILCS-Pharm with the SilcsBio GUI).

The x,y,z coordinates of the center of the complexed ligand are used to create a SILCS-Pharm 3-D pharmacophore model encompassing the ligand binding site (shown below). These coordinates define the center of the sphere within which FragMap voxels are searched and clustered to generate features.



Using the ligand center coordinates `center="35.24, 27.48, 37.73"`, generate the 3-D pharmacophore model with:

```
${SILCSBIODIR}/silcs-pharm/1_calc_silcs_pharm prot=3fly.pdb center="35.
↪24,27.48,37.73" mapsdir=${SILCSBIODIR}/examples/silcs/silcs_fragmaps_
↪3fly/maps
```

The command will complete after several seconds, and the output will note, "A total of 13 features have been detected." All output files will be in a new subdirectory `5_pharm`. Standard molecular graphics software can be used to visualize the output file `3fly_silcspharm_features.pdb`, which has one `ATOM` entry for each of the 13 pharmacophore features:

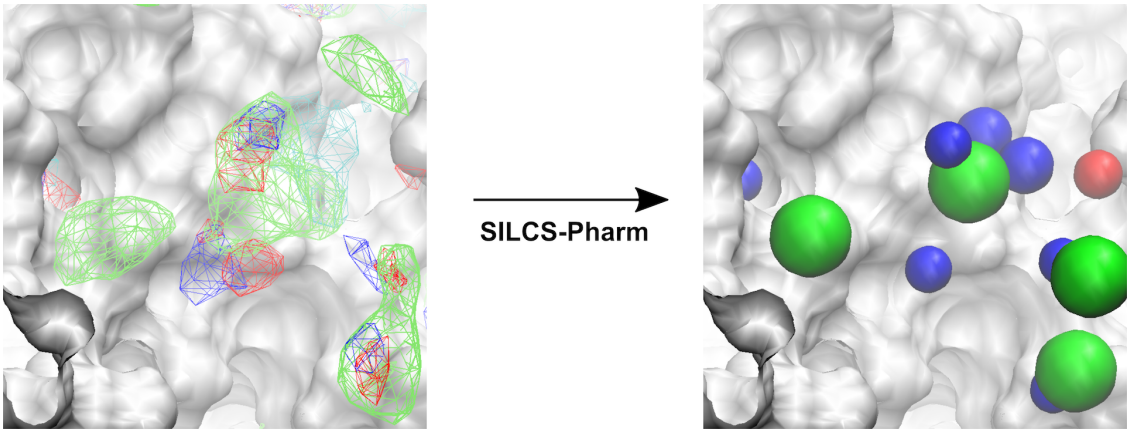


Fig. 12.1: Conversion of FragMaps (left) to SILCS pharmacophore features (right).

To modify/reduce the features, edit the output file `3fly_silcspharm_features.pdb` and save it as `3fly_silcspharm_features_revised.pdb`, then run `${SILCSBIODIR}/programs/revise_ph4`:

```
${SILCSBIODIR}/programs/revise_ph4 3fly.keyf_13.ph4 3fly_silcspharm_
↪features_revised.pdb
```

The output `3fly.keyf_13_revised.ph4` will reflect your revisions in `3fly_silcspharm_features_revised.pdb`.

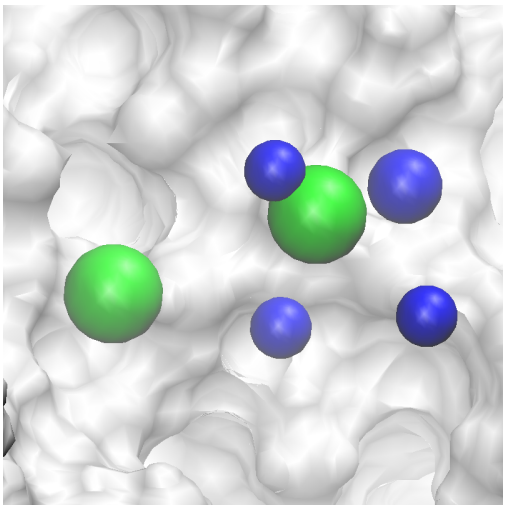


Fig. 12.2: Revised SILCS pharmacophore features.

CHAPTER

THIRTEEN

SILCS-HOTSPOTS: FRAGMENT BINDING SITES INCLUDING ALLOSTERIC SITES

13.1 Background

SILCS-Hotspots identifies all potential fragment binding sites on a target by doing SILCS-MC sampling across the entire target structure. The protein or other macromolecular target is partitioned into a collection of subspaces in which the fragment is randomly positioned and subjected to extensive SILCS-MC to identify favored local poses. This may be performed 1000 times or more per fragment in each subspace. The identified fragment positions are then RMSD-clustered followed by selection of the lowest energy pose in each cluster to define a binding site for further analysis.

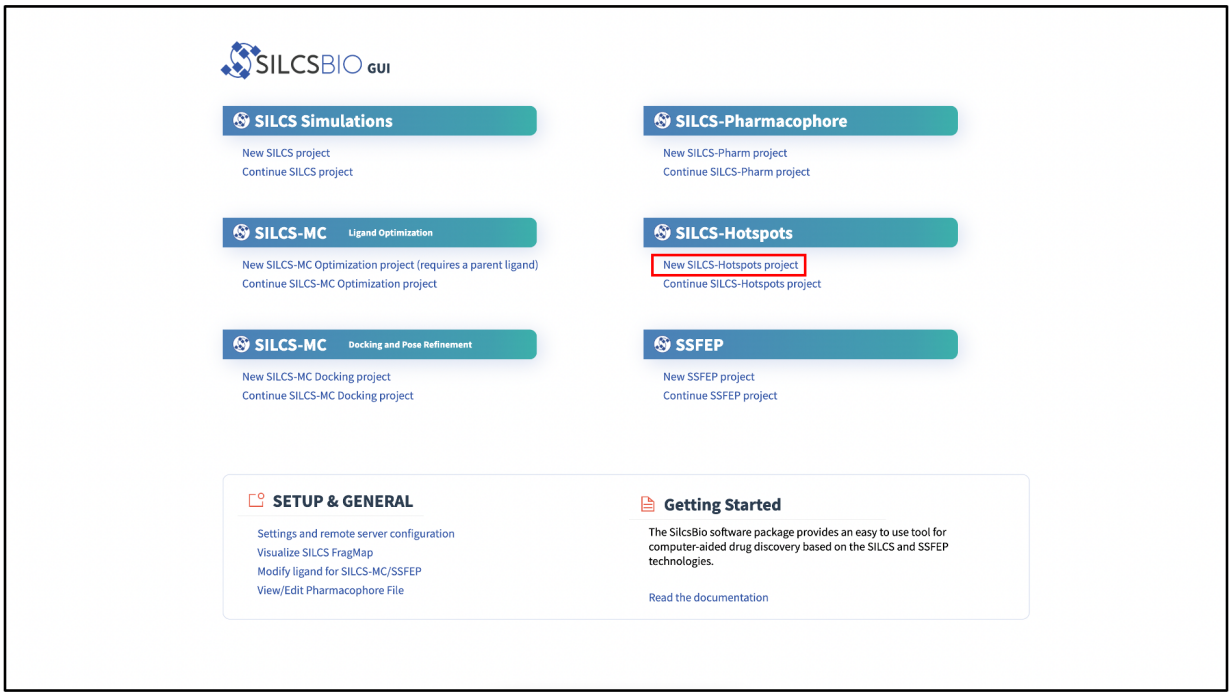
SILCS-Hotspots may be applied to small fragment-like molecules as well as larger drug-like molecules. When multiple fragments are used, a second round of clustering may be performed over the different fragment types to identify binding sites occuppied by one or more of the fragment types included in the calculation.

Results from SILCS-Hotspots include all potential binding poses of the individual fragments, binding sites that contain one or more of the fragment types, and ranking of fragment poses and binding sites based on LGFE scores. Applications of SILCS-Hotspots include identifying putative fragment binding sites and performing fragment-based drug design. Putative binding sites identified by SILCS-Hotspots can be used for rapid database screening with SILCS-Pharmacophore. Alternatively, sites identified by SILCS-Hotspots can be considered for fragment-based design by linking poses of fragment-like molecules in adjacent sites to create drug-like molecules.

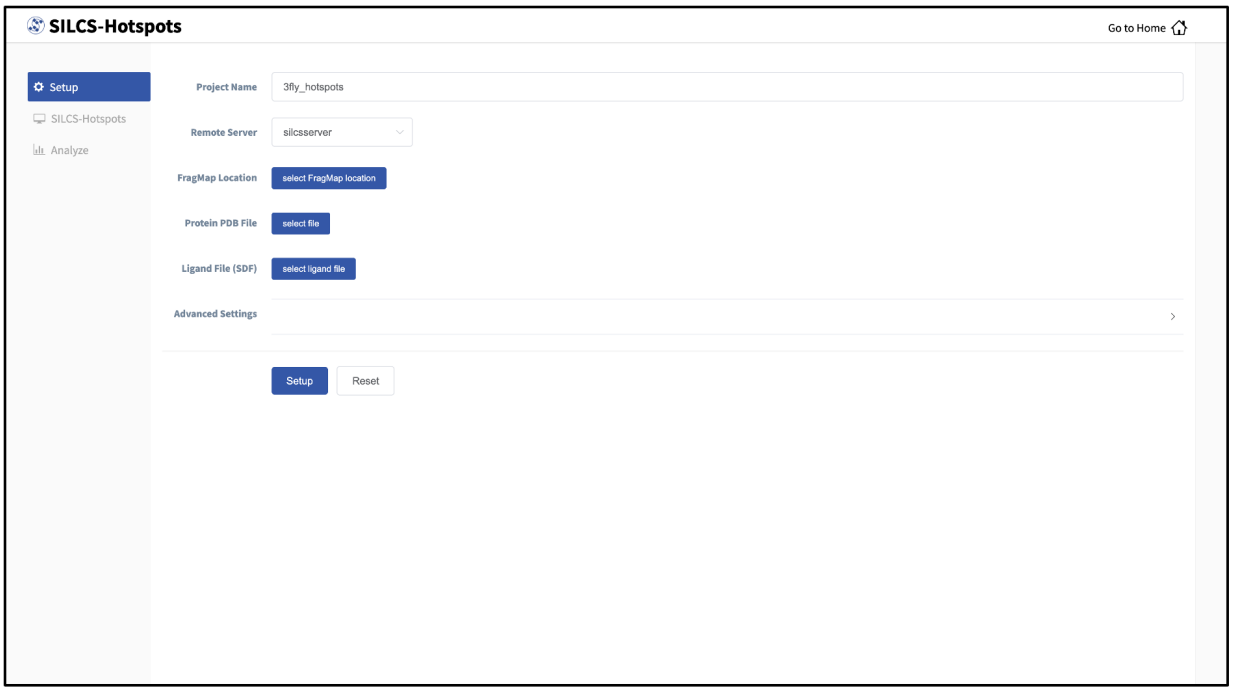
Inputs for SILCS-Hotspots are the protein or other macromolecular target used for the SILCS simulations, the SILCS FragMaps resulting from those simulations, and the fragment(s) to be used for sampling. SilcsBio provides fragment files appropriate for SILCS-Hotspots (see below). Alternatively, you may use your own fragment file(s).

13.2 Running SILCS-Hotspots from the SilcsBio GUI

1. Select *New SILCS-Hotspots project* from the Home page.

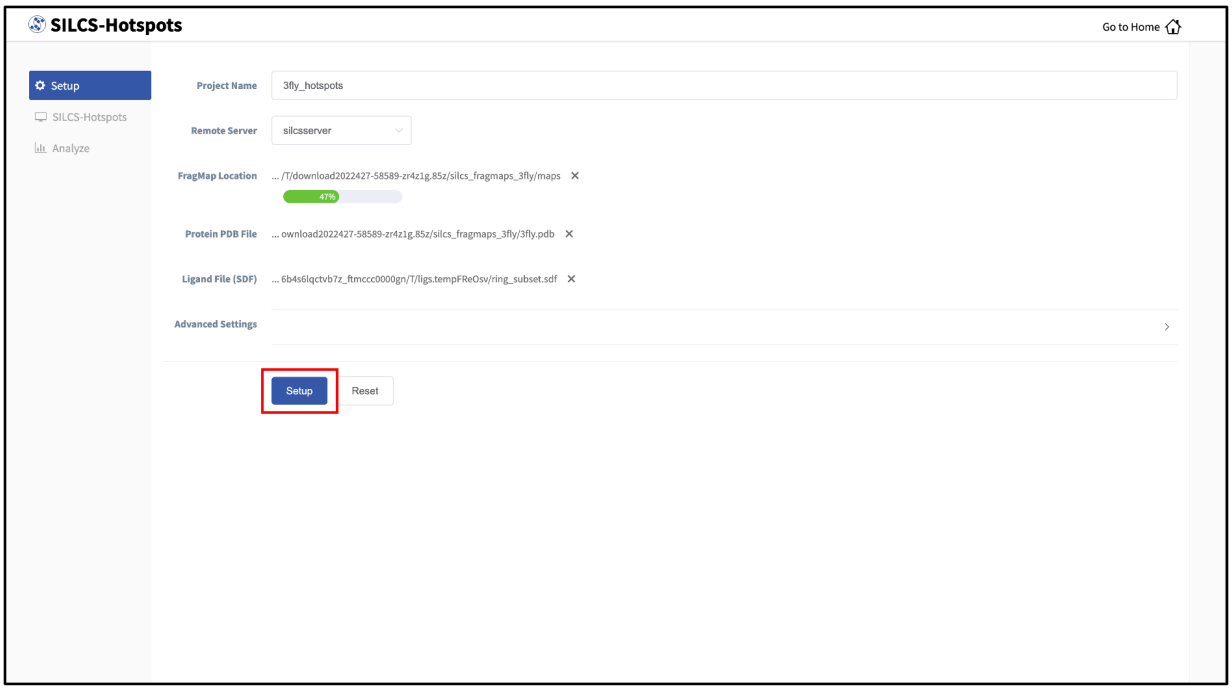


2. Enter a project name and select the remote server where the SILCS-Hotspots jobs will run. You may choose these files from your local machine where you are running the SilcsBio GUI ("localhost") or from any server you have previously configured, as described in File and directory selection.

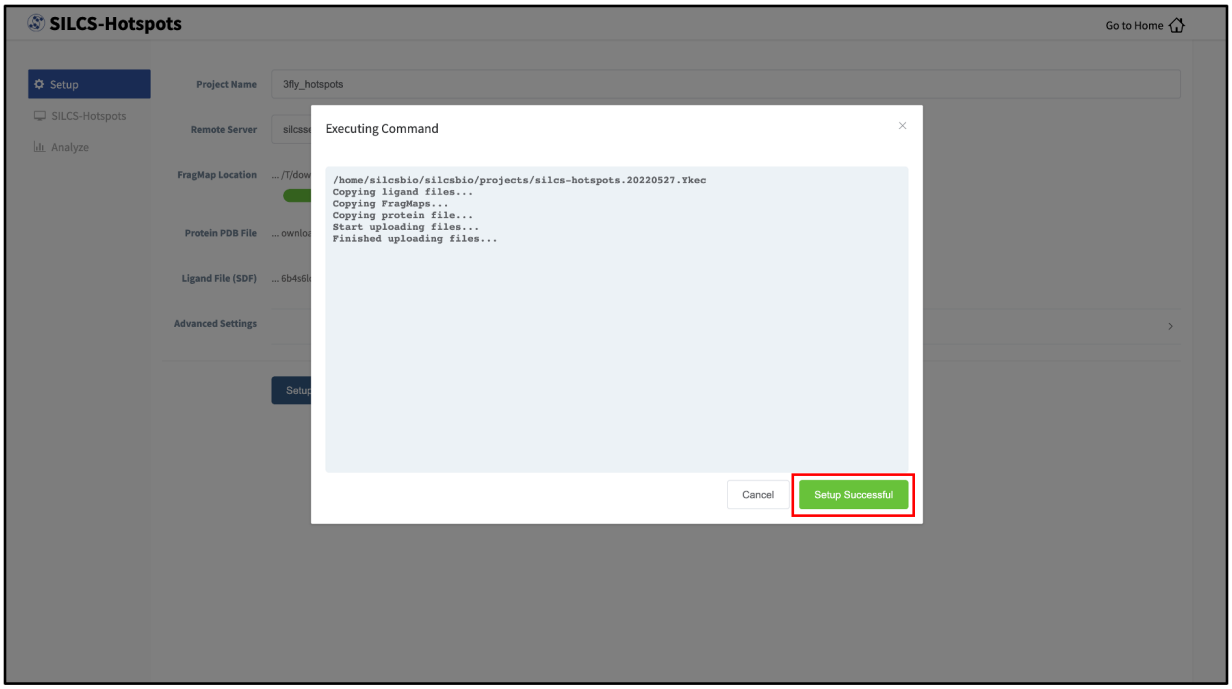


3. Provide FragMap and protein input PDB files. You will additionally need to provide a "Ligand file (in SDF format)" that contains the database of ligands to be used for sampling (see *Default databases of fragment-like molecules*). Once the information is entered correctly, click the "Setup" button at the bottom of the page.

Warning: Ligands in the "Ligand file (in SDF format)" must include all hydrogens, including pH-appropriate (de)protonations, and must have reasonable three dimensional conformations.

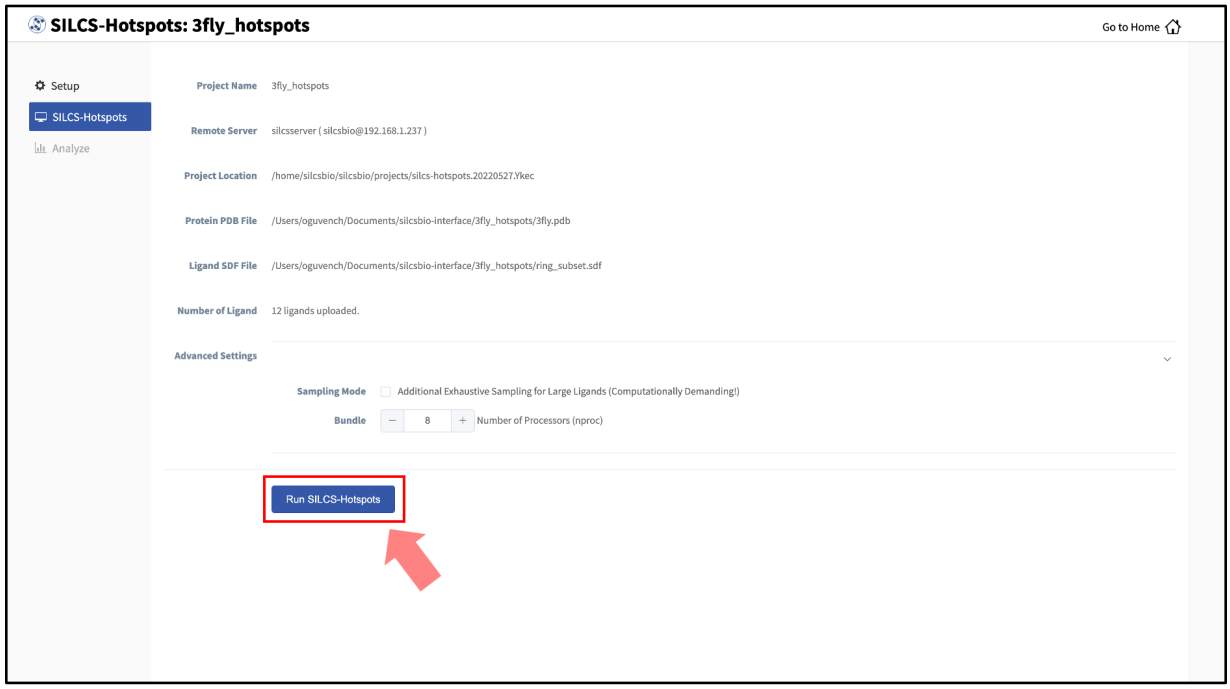


4. The GUI will contact the remote server and upload the input files to the "Project Location" directory on the remote server. A green "Setup Successful" button will appear once the upload has successfully completed. Press this green button to proceed.

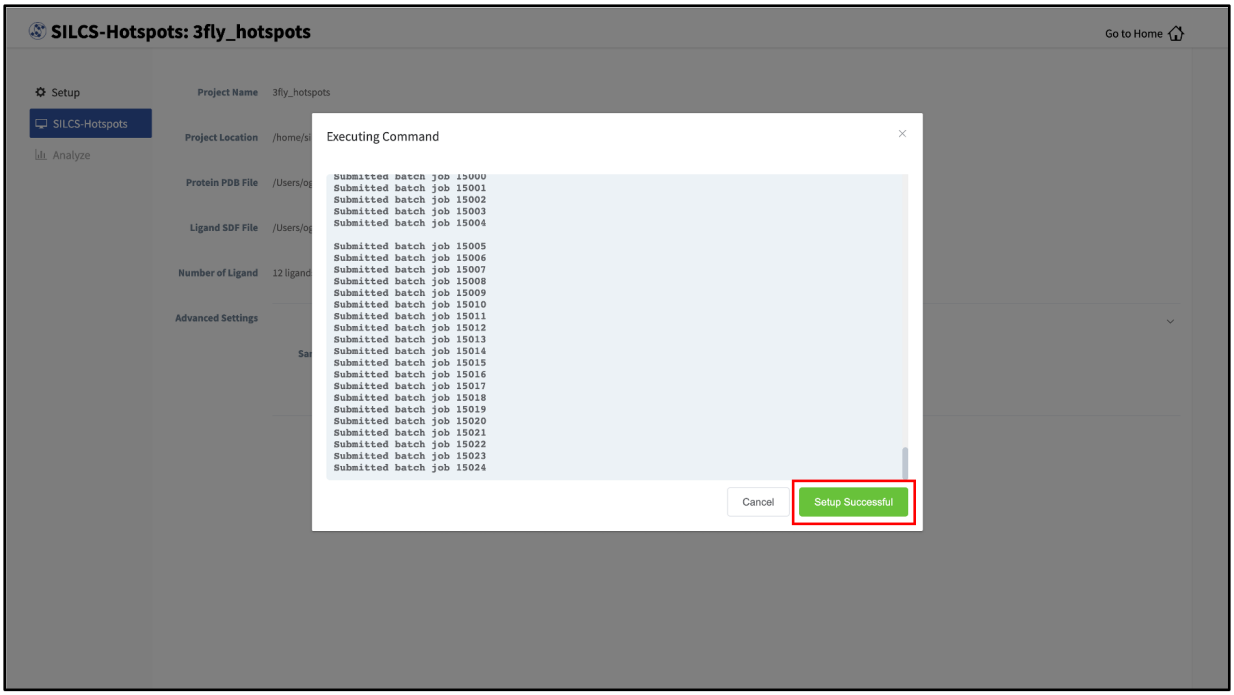


5. The GUI will display a summary screen with the Project Name, Remote Server, Project Location, Protein PDB file, Ligand SDF file, and the Number of Ligands. You will also see

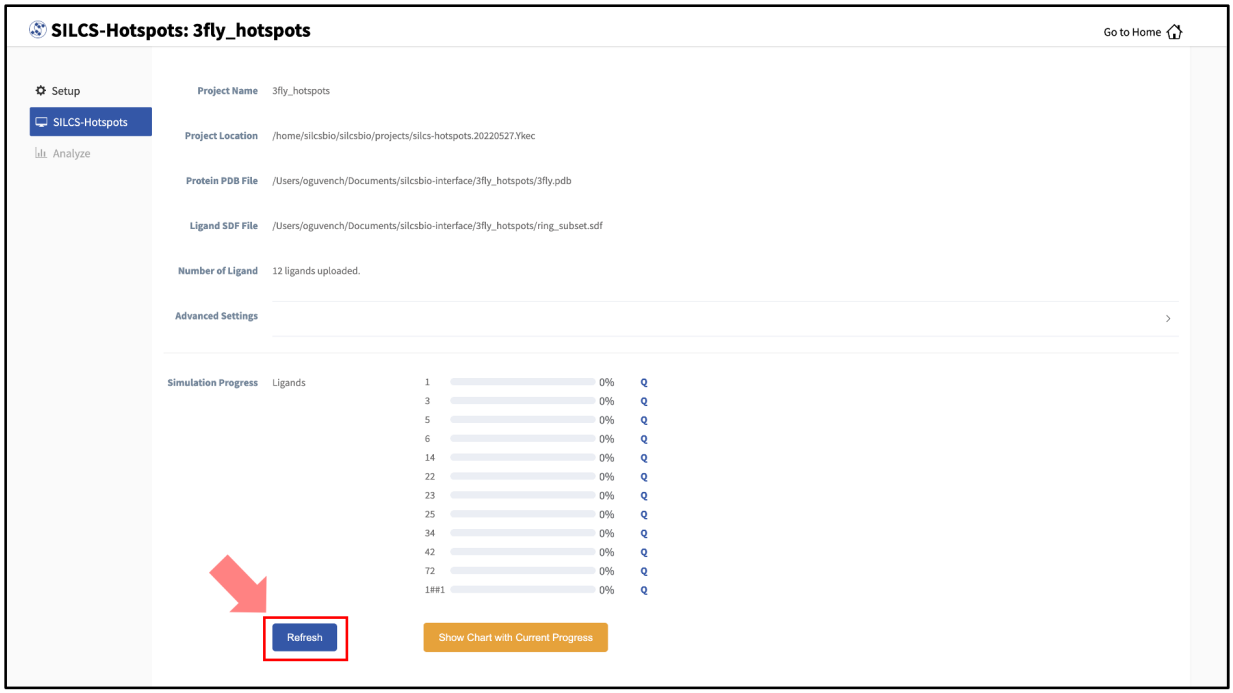
"Exhaustive sampling mode" and "number of processors" under "Advanced Settings". You may select/adjust these if you desire. Click on the "Run SILCS-Hotspots" button. Doing so will submit jobs to the remote server and list them in a pop-over window.



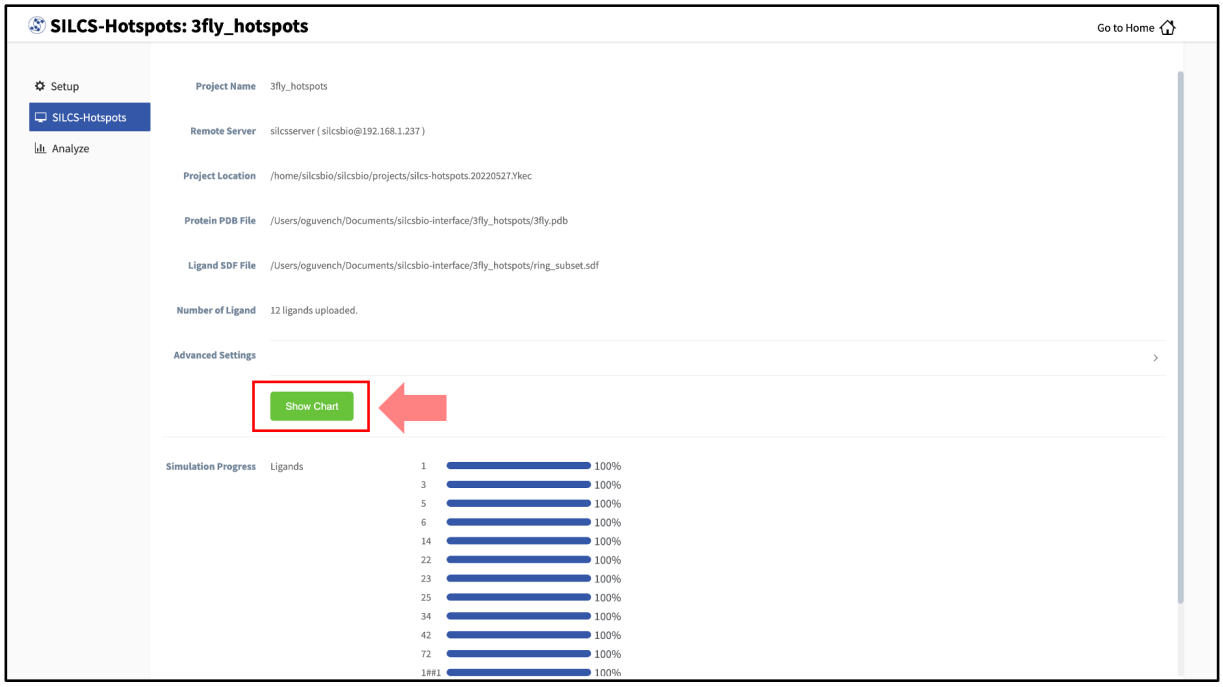
6. Once the jobs are submitted, you may click on the "Setup Successful" button to dismiss the pop-over window and return to the previous screen.



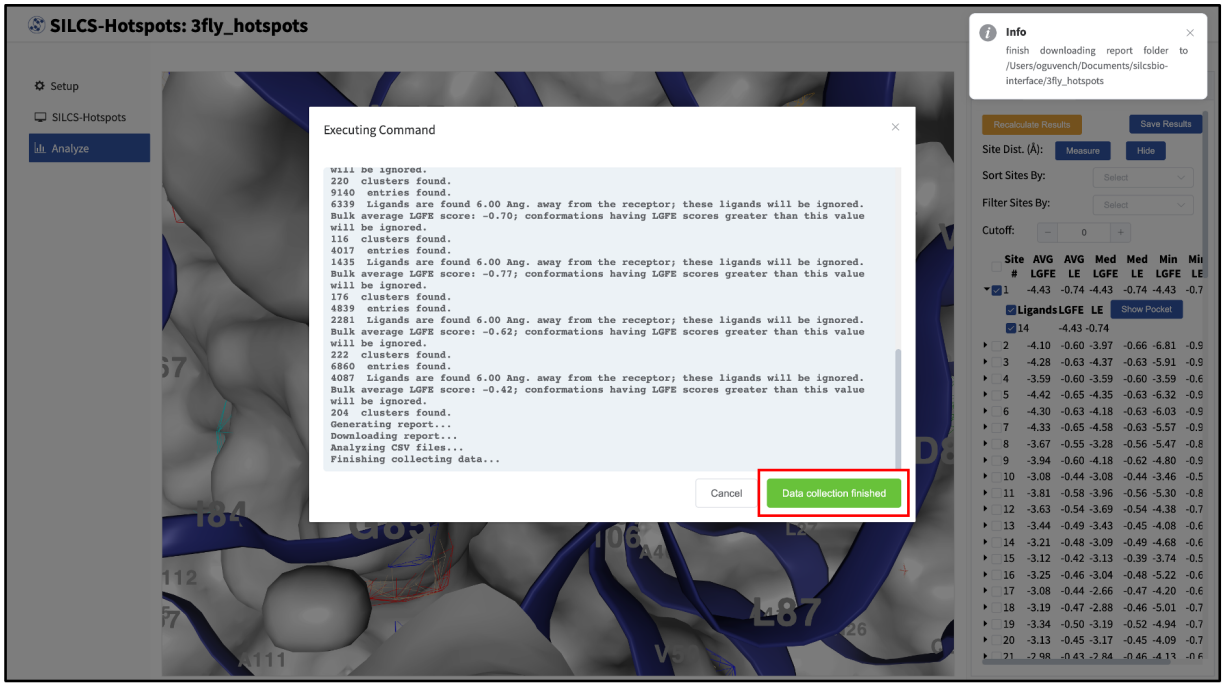
7. The screen will now show a "Simulation Progress" section. You can update this section by clicking the "Refresh" button. This will update the job progress bars in the SilcsBio GUI.



8. Once the progress bars reach 100%, you will see a green "Show Chart" button. Click it to proceed. You may also click on the "Show Chart with Current Progress" button while some of the jobs are still running.

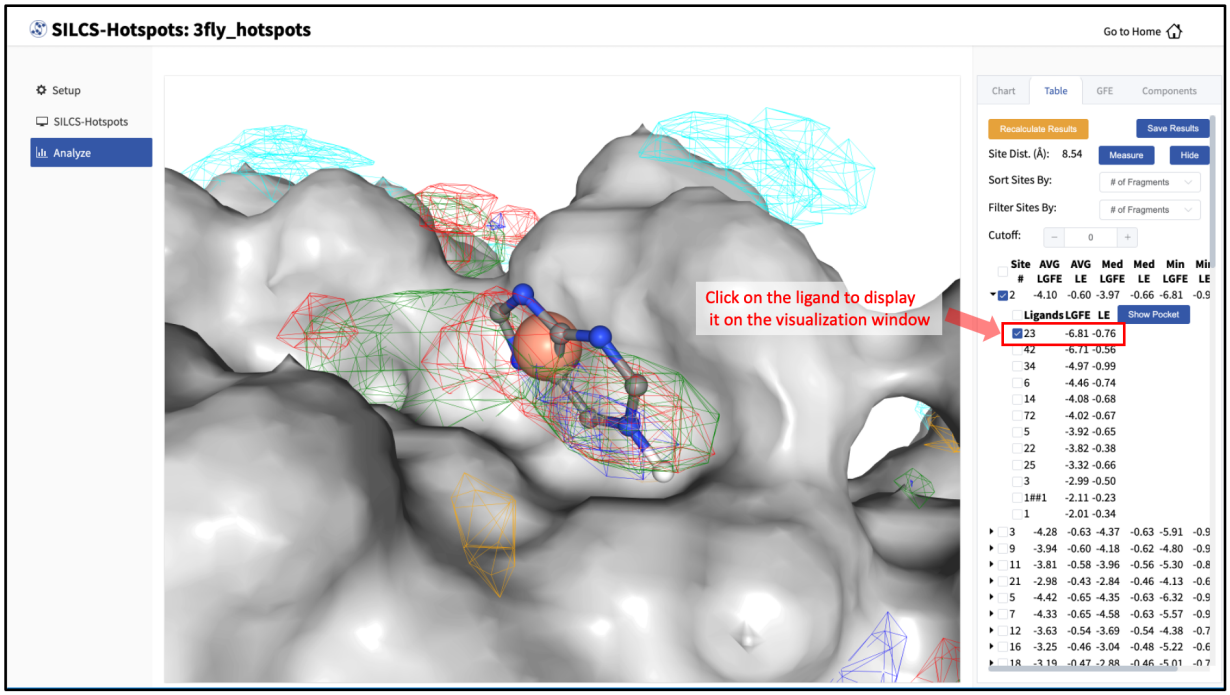


9. Clicking the "Show Chart" button will download a report folder from the server. Click on "Data collection finished" to proceed.

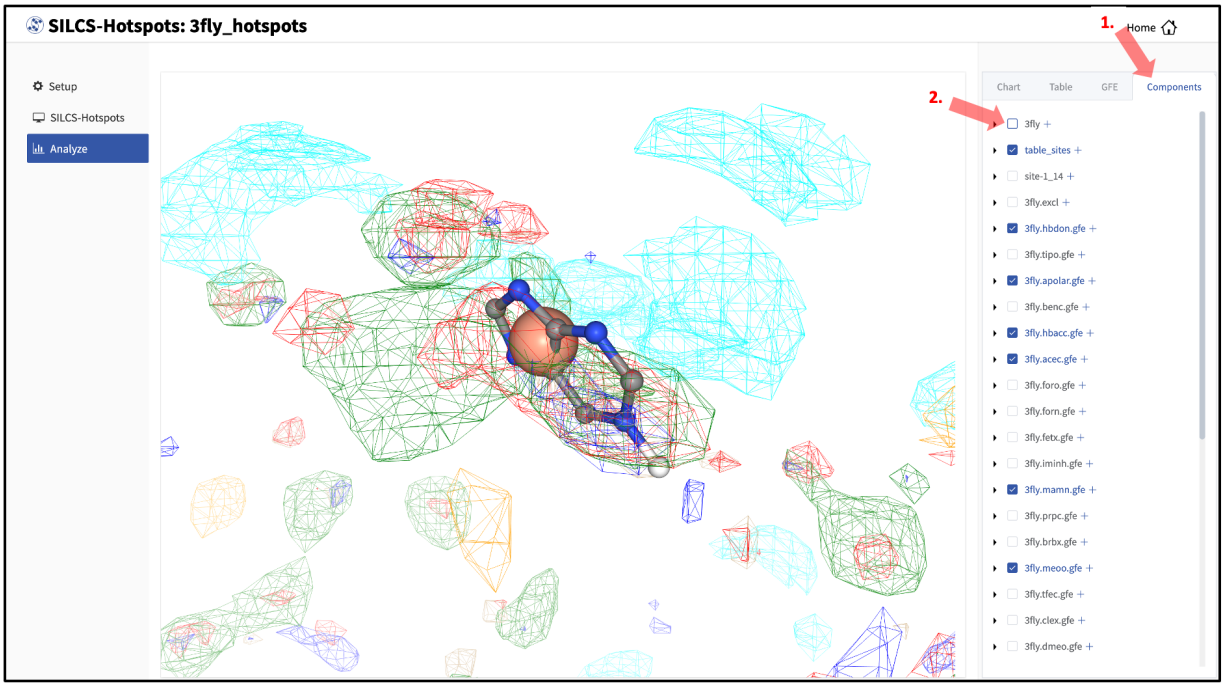


10. A new tab, labeled "Table" will have been created in the right-hand panel. The "Table" tab lists the LGFE and Ligand Efficiency (LE) scores of each hotspot (referred to as "site"). By

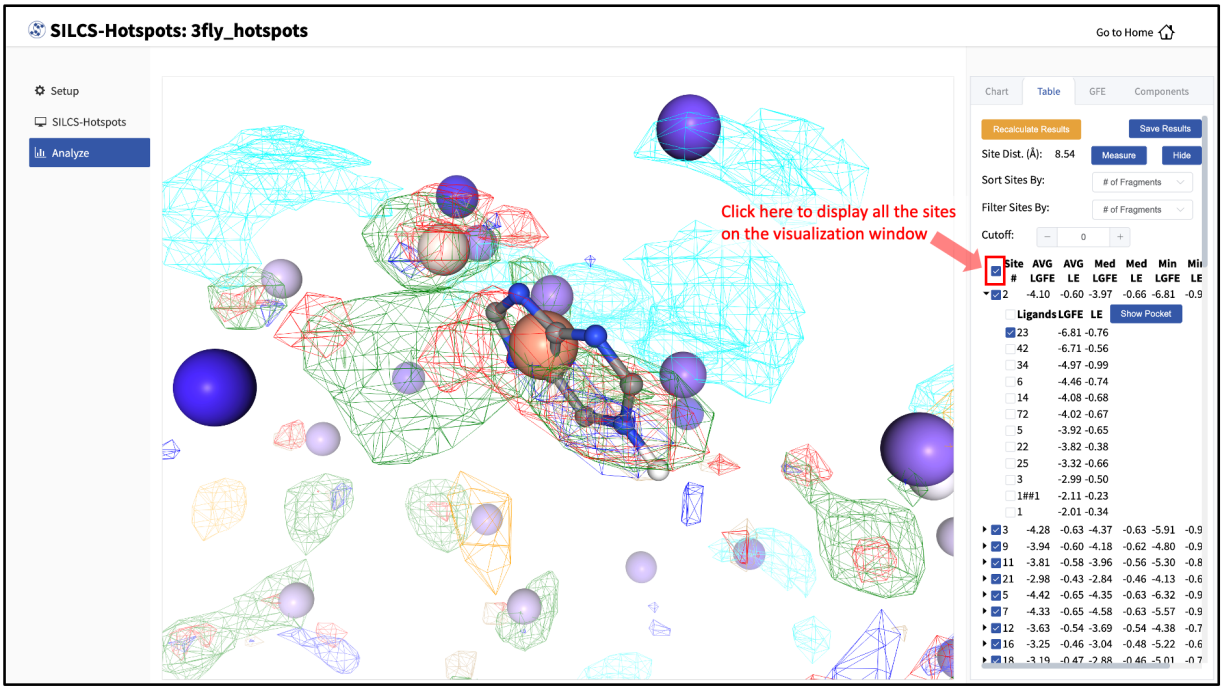
clicking on the ligand under the "Table" tab, you can center the visualization window at the ligand.



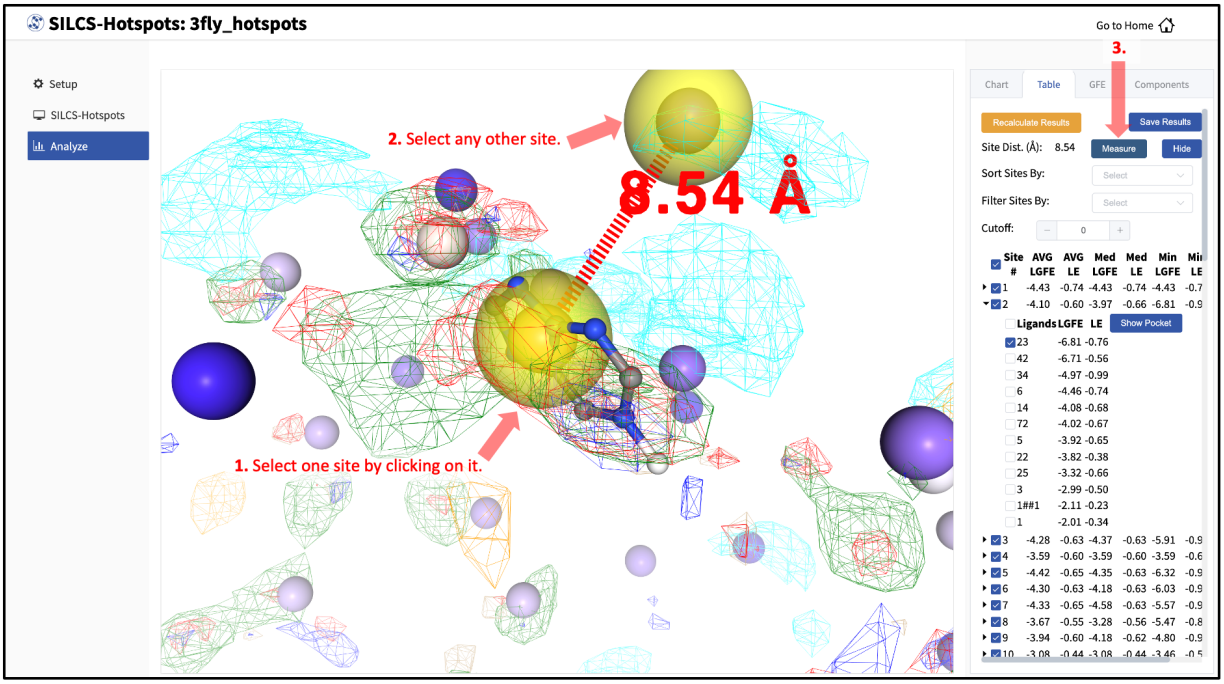
11. You may also click on the "Components" tab in the right-hand panel and deselect the protein for easier viewing.



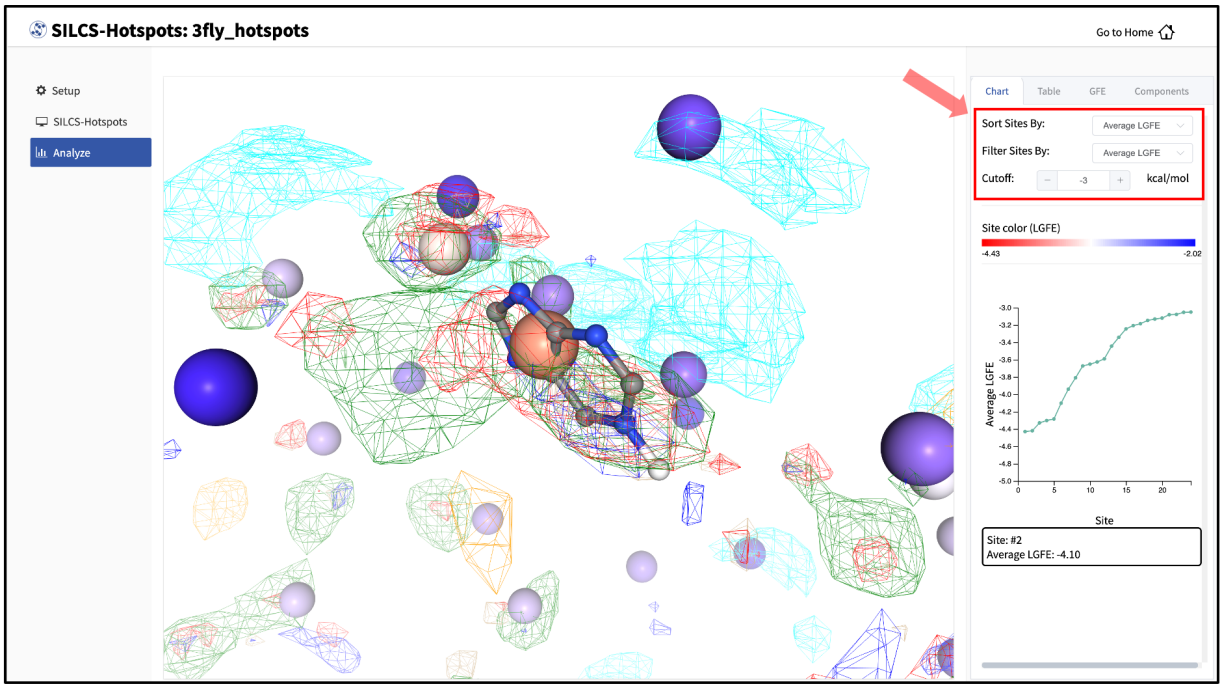
12. Click on the box next to "Site #" to display all the hotspot sites on the visualization window. Click the "Save Results" button to save the SILCS-Hotspots results for all your selected hotspot sites to your local computer. You can sort the table using the "Sort Site By:" drop-down menu. You can additionally display only those sites meeting a cutoff criterion by using the "Filter Sites By:" drop-down menu and adjusting the "Cutoff:" value.



13. To measure the distance between two hotspot sites, select their solid spheres and click the "Measure" button in the right-hand panel. This will display a dashed line in the visualization window with the measured distance (in Å) from one hotspot site to the other.



14. The “Chart” tab in the right-hand panel converts the information from the “Table” tab into plots. The plot can be adjusted by sorting and/or filtering sites using the “Sort Sites By:” and “Filter Sites By:” drop-down menus. Changing the “Cutoff:” value will adjust the y-axis. The plot is interactive: hovering your mouse over a point will display the x and y values for that point.



13.3 Running SILCS-Hotspots from the command line interface

13.3.1 Launching SILCS-Hotspots SILCS-MC jobs

Input arguments for launching the SILCS-Hotspots SILCS-MC jobs are: the PDB file used for the SILCS run (`prot`), the location of the fragment files (`ligdir`), and location of the SILCS FragMaps (`mapsdir`). When SILCS-Hotspots calculations are performed, the subdirectory `4_hotspots/` is created to store output from the exhaustive SILCS-MC runs.

```
$SILCSBIODIR/silcs-hotspots/1_setup_silcs_hotspots prot=<prot PDB>␣
↪ligdir=<fragment database> mapsdir=<mapsdir> bundle=<true/false>
```

Note: The above command will submit a large number of jobs, which can strain job queueing systems. The `bundle` keyword launches bundled jobs instead of individual jobs.

Options include:

1) FragMap directory path:

```
mapsdir=<location and name of directory containing FragMaps;␣
↪default=maps>
```

By default, the program looks for FragMaps in the `maps/` directory.

2) Fragment database location:

```
ligdir=<location and name of directory containing fragment mol2/
↪sdf>
```

See *Default databases of fragment-like molecules* for detailed information.

Note: `.sdf`, `.sd`, or `.mol2` files can be placed in the `ligdir` directory, and SILCS-MC will read a single molecule from each file. Note that if a file contains multiple molecules, use of the `ligdir` option will result in only the first molecule in the file being processed.

If you have an SDF file with multiple molecules in it, replace `ligdir=<directory containing ligand mol2/sdf>` with `sdfile=<path to sdfile>` to process all molecules in the file.

Warning: Ligands, regardless of file format, must include all hydrogens, including pH-appropriate (de)protonations, and must have reasonable three dimensional conformations.

3) Number of processors to use for bundled jobs:

```
nproc=<# of processors used when bundle=true>
```

4) Directory containing output to be used in subsequent steps:

```
hotspotsdir=<location of hotspots data; default=4_hotspots>
```

5) Template file for SILCS-MC sampling:

```
paramsfile=<custom params file>
```

The default SILCS-MC job parameters file for SILCS-Hotspots is `$SILCSBIODIR/templates/silcs-hotspots/params.tmpl`. See User-defined protocols for customization details.

13.3.2 Post-run clustering

Once the SILCS-Hotspots SILCS-MC jobs are complete, the next step is to cluster fragment poses that were output by the completed jobs. The clustering algorithm iteratively finds clusters with

the largest number of members [19]. The process entails 1) computing the number of neighbors of each pose, 2) choosing the pose with the largest number of neighbors and marking it and its neighbors as cluster members, 3) removing cluster members identified in Step 2 from the pool of poses, and 4) repeating Steps 1-3 until there are no available poses left. The cluster "center" is defined as that fragment pose with the most neighbors in that cluster. This pose's LGFE score defines the LGFE score of the cluster. Run the command

```
$SILCSBIODIR/silcs-hotspots/2_collect_hotspots prot=<prot PDB> ligdir=
↪<fragment database>
```

to perform the clustering. You may also specify the following additional options:

1) Radius for RMSD clustering:

```
cutoff=<cutoff for clustering; default=3>
```

2) Maximum number of sites to be identified for each fragment:

```
maxsites=<maximum number of sites ordered by LGFE; default=200>>
```

3) Directory containing output from the previous steps:

```
hotspotsdir=<location of hotspots data; default=4_hotspots>
```

13.3.3 Site determination and report generation

The final step of SILCS-Hotspots is site determination and report generation. This involves a second round of clustering over all specified fragments to identify sites on the protein to which one or more fragments bind. Each site is given the average LGFE score over all the fragments. In addition, the sites themselves may be clustered to identify binding pockets that suggest multiple adjacent sites that may be linked to build larger fragments (under development). Information on all the fragments, the binding sites, and the binding pockets is included in `report.xlsx` and in PDB files that are output to the subdirectory `4_hotspots/report/`. Run the following command

```
$SILCSBIODIR/silcs-hotspots/3_create_report ligdir=<fragment database>
```

You may add these options:

1) Directory containing output from the previous steps:

```
hotspotsdir=<location of hotspots data; default=4_hotspots>
```

2) Cluster radius for binding site determination:

```
site_cutoff=<cluster radius for site determination; default=6.0>
```

3) LGFE cutoff for inclusion of fragment poses in site determination:

```
ligand_lgfe_cutoff=<Ligand LGFE cutoff for site determination;␣
↪default=-2.0>
```

4) Average LGFE cutoff for site determination:

```
site_lgfe_cutoff=<average LGFE cutoff for binding sites; default=-
↪2.0>
```

Sites having values less favorable than the cutoff will be discarded.

5) Flag to activate binding pocket calculation and associated clustering radius (under development):

```
pocket=<perform pocket analysis; default=False>
pocket_cutoff=<cluster radius for binding pocket determination;␣
↪default=12.0>
```

The `3_create_report` command will populate the `4_hotspots/report/` subdirectory with the following:

- `report_all.xlsx`: Spreadsheet file containing analysis information. Included are the LGFE and Ligand Efficiency (LE) scores for each copy of every fragment obtained from clustering, relative affinity analysis, and listing of the posed fragment that defines each site. If binding pocket analysis is performed, information on binding pockets is included.
- `hotspots_sites.pdb`: PDB file containing the identified sites. The B-factor column value includes the average LGFE score of that site.
- `pdb_by_ligands`: Directory containing PDB files for each fragment with multiple coordinates for each site identified for the fragment. In the PDB files, REMARK includes the LGFE scores, the B-factor column includes the GFE score for each atom, and the final column lists the SILCS atom type.
- `pdb_by_site`: Directory containing PDB files of the fragments located at each site in `hotspots_sites.pdb`. For example, the filename `site_all_1_8_1.pdb` indicates that at site 1 fragment 8 is present. The final 1 indicates that this is the first copy of fragment 8 at site 1. From the clustering alogrithm, it is possible that more than one copy of a fragment is included in a site. For example, `site_all_1_8_2.pdb` would be the second copy of fragment 8. In each PDB file, REMARK includes the LGFE scores, the B-factor column includes the GFE score for each atom, and the final column lists the SILCS atom type.

The following additional outputs are produced if binding pocket analysis is performed (`pocket=true`):

- `hotspots_pockets.pdb`: PDB file containing sites that define each binding pocket. In the following, pocket P01 or 1 is defined by four sites with LGFE values for each site shown in the B-factor column. The algorithm does not number pockets consecutively: in this example, there is no P02 pocket, and pocket P03 contains 2 sites.

```
ATOM      1   X  P01 A   1      20.980   7.230 -12.513  1.00 -4.39␣
 ↪          C
ATOM      2   X  P01 A   1      27.947  16.848  -6.986  1.00 -2.82␣
 ↪          C
ATOM      3   X  P01 A   1      17.606  12.224 -10.043  1.00 -2.67␣
 ↪          C
ATOM      4   X  P01 A   1      14.912  17.557  -5.282  1.00 -2.08␣
 ↪          C
ATOM      5   X  P03 A   3      16.182 -10.455  11.793  1.00 -2.96␣
 ↪          C
ATOM      6   X  P03 A   3      17.605   0.058  15.384  1.00 -2.83␣
 ↪          C
```

- `pdb_by_pocket`: Directory containing PDB files for fragments that comprise each pocket. For example, `pocket_all_8_site_12_10_1.pdb` indicates pocket 8 contains a fragment from site 12 and that fragment is fragment 10. `1` indicates that it is the first copy of fragment 10 in that pocket. REMARK includes the LGFE scores, the B-factor column includes the GFE score for each atom, and the final column is the SILCS atom type.

13.3.4 Practical considerations

`1_setup_silcs_hotspots` creates a subdirectory, `4_hotspots/`, (or user defined name using `hotspotsdir=`) that contains fragment pose and LGFE information. As SILCS-Hotspots makes use of large numbers of SILCS-MC calculations on each fragment, this directory will be filled with a substantial amount of data. It is suggested that, once all analyses are complete, these files either be deleted or archived, and `4_hotspots/` be renamed prior to additional hotspots runs. Remember, the data in `4_hotspots/` are used for post-run clustering and site determination and report generation.

If `1_setup_silcs_hotspots` is being rerun with new parameters, such as new fragments as specified by `ligdir=`, the subdirectory `4_hotspots/` should be renamed or an alternate name assigend using `hotspotsdir=` to avoid information from the original run being overwritten. However, this is not strictly necessary IF all the new fragments have unique filenames relative to the original run AND the SILCS-MC job parameters specified by `paramsfile=` are not changed, since the subsequent `2_collect_hotspots` command only performs analysis on Mol2/SDF files in the specified `ligdir` directory.

`1_setup_silcs_hotspots` launches a large number of jobs that, while the majority of jobs finish quickly individual jobs may take time (minutes to an hour) to complete. In some cases one or two jobs in the set may require additional time due to the robust SILCS-MC convergence criteria used in SILCS-Hotspots.

For the SILCS-MC sampling, especially with larger fragments or ligands, you may prefer not to include the SILCS Exclusion Map. This will permit fragment sampling of poses that would otherwise be rejected because of overlap with the exclusion region. Using this approach, final fragment poses will be based solely on scoring with SILCS FragMaps. To achieve this, set the weighting of the Exclusion Map to zero in your custom SILCS-MC job parameters file (`paramsfile=<custom params file>`). A simple means to this end is to use a copy of the default file `$SILCSBIODIR/silcs-mc/params_custom.tmpl` in which you replace the `1.000` in the below line with `0.000`.

```
SILCSMAP EXCL <MAPDIR>/<prot>.excl.map        1.000
```

Clustering of data in the `4_hotspots/` or equivalent subdirectory with the `2_collect_hotspots` command produces representative fragment poses on a per-cluster basis. If clustering is repeated, for example with a different clustering radius, the old PDB files created in `4_hotspots/` will be overwritten. Therefore, make sure to run site determination and report generation (the `3_create_report` command) on your original clustering output before repeating clustering.

`2_collect_hotspots` processes each of the fragments individually and typically requires minutes to complete for 100 fragments. During that process a number of warning messages, such as "Clustered PDB not found: 4_hotspots/2/subspace_1/pdb/2_clust_1_1.pdb," will be given. These are expected as they indicate subspaces for the fragments in which no favorable fragments poses were identified. For example, this may occur where the subspace encompasses the protein structure.

Site determination and report generation with the `3_create_report` command creates a subdirectory, `4_hotspots/report/`, and outputs `report.xls` and PDB files into this subdirectory. Rerunning `3_create_report` with different parameters, like an alternate value of the site clustering cutoff, will overwrite the original output. Therefore, prior to rerunning report generation, rename `4_hotspots/report/` to save your original report generation outputs. The clustering alogrithm for site determination does not consider the identity of the fragment when performing the clustering. Accordingly, in certain cases it is possible for the same fragment to be included two or more times in a given site.

13.3.5 Example

The following demonstrates use of SILCS-Hotspots on p38 MAP kinase. Input files, including FragMaps, are in `$SILCSBIODIR/examples/silcs/`.

```
cp -r $SILCSBIODIR/examples/silcs/silcs_fragmaps_p38a .
cd silcs_fragmaps_p38a
$SILCSBIODIR/silcs-hotspots/1_setup_silcs_hotspots prot=p38a.pdb␣
↪ligdir=$SILCSBIODIR/data/databases/ring_subset mapsdir=maps␣
↪bundle=true
```

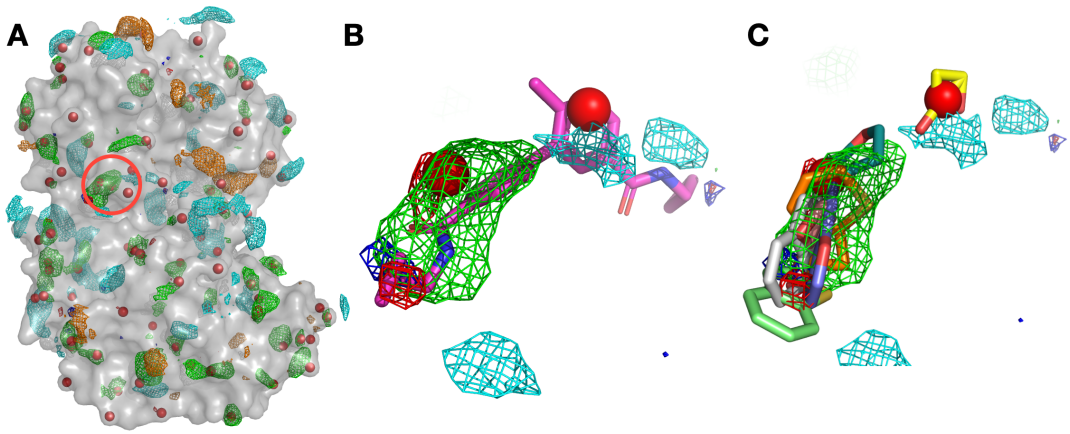
In this example, we use `$SILCSBIODIR/data/databases/ring_subset` as the fragment database (see *Default databases of fragment-like molecules*). Owing to the small size of this database, the example run finishes quickly.

Once the jobs spawned by `1_setup_silcs_hotspots` complete, use the following commands for post-run clustering and site determination and report generation.

```
$SILCSBIODIR/silcs-hotspots/2_collect_hotspots prot=p38a.pdb ligdir=
↪$SILCSBIODIR/data/databases/ring_subset
$SILCSBIODIR/silcs-hotspots/3_create_report ligdir=$SILCSBIODIR/data/
↪databases/ring_subset
```

This will have created the `4_hotspots/report/` subdirectory containing the following:

- `hotspots_sites.pdb`: Centroid positions of fragment clusters, that is, the "hotspots." Clusters are ranked using average LGFE scores, with the cluster rank listed in the residue number field and the average LGFE score in the B-factor field.
- `report_all.xlsx`: Report of hotspots in spreadsheet format.
- `pdbs_by_sites`: List of PDB files for each hotspot.



Panel (A) shows p38 (surface representation), SILCS FragMaps (wire frame), and all hotspots (red spheres) determined by SILCS-Hotspots. The crystal binding pocket is circled in red. Panel (B) has FragMaps, hotspots, and the crystallographic ligand. Two hotspots are identified within the crystal binding pocket and coincide with the rings of the fragment. Panel (C) shows fragment poses from SILCS-Hotspots calculations as well as FragMaps and hotspots.

13.4 Validating hotspots using FDA-approved drugs

A SILCS-Hotspots run will typically use a database of fragment-like molecules (see *Default databases of fragment-like molecules*) to determine probable small-molecule binding hotspots on

the input target. Further validation of a subset of these hotspots can be done by docking and scoring actual FDA-approved drugs at these hotspot locations. To this end, an automated workflow is provided for docking 348 diverse FDA-approved drugs at user-selected hotspots and calculating the drug molecules' relative solvent accessible surface areas (rSASA) before and after docking.

13.4.1 Suggested characteristics of hotspots to be selected for further validation

Users will need to judiciously choose a subset of the hotspots determined by SILCS-Hotspots for further validation by docking of FDA-appoved drugs. As these drug molecules are significantly larger and have more rotatable bonds than the fragment-like molecules used in determining hotspots, there is relatively more computational expense associated with docking them versus fragment-like molecules. This can make it impractical to apply the validation procedure to the entire set of hotspots. Suggested characteristics for choosing a hotspot to include in your subset for further validation include

- The hotspot is partially or fully buried
- The hotspot has other hotspots nearby (typically within 10 Å)
- The hotspot has 2 or more apolar FragMap regions in close vicinity
- The hotspot has multiple hydrogen bond Don/Acc FragMap regions in close vicinity

Once you have selected your subset of hotspots, you will need to EITHER supply them as a separate pdb file `<my_hotspots.pdb>` OR as a comma-separated list (e.g. `"2,7,10,22"`) along with the full `hotspots_sites.pdb` file from the SILCS-Hotspots standard report:

```
$SILCSBIODIR/silcs-hotspots/4a_fda_top20_analysis prot=<prot PDB>␣
↪hotspotspdb=<my_hotspots.pdb>
```

OR

```
$SILCSBIODIR/silcs-hotspots/4a_fda_top20_analysis prot=<prot PDB>␣
↪hotspotspdb=hotspots_sites.pdb myhotspots="2,7,10,22"
```

This first step will set up and run SILCS-MC simulations for each of the 348 FDA-approved drug molecules at each of your selected hotspots. The SILCS-MC in this first step entails exhaustive sampling of ligand orientations and conformations.

Other options include:

- A tag identifying the subset of hotspots you are analyzing:

  ```
  sitetype=<tag_name>
  ```

  The default `<tag_name>` is set to `"rings"`. If you use this option, make sure to use it for ALL the following steps as well.

- FragMap directory path:

```
mapsdir=<location and name of directory containing FragMaps;␣
↪default=maps>
```

  By default, the program looks for FragMaps in the `maps/` directory.

- Compound library/database location:

```
sdfile=<location and name of SDF containing all compounds; default=
↪$SILCSBIODIR/data/databases/fda_approved_348.sdf>
```

  or

```
ligdir=<location and name of directory containing compound mol2/
↪sdf>
```

  The default is the database of 348 diverse FDA-approved drug molecules, which is expected generally to work well. However, you may choose to supply a different database.

- Number of processors to use for bundled jobs:

```
nproc=<# of processors used when bundle=true>
```

  By default, the `bundle` keyword is set to `true` in this analysis.

The second step involves collecting poses from completed SILCS-MC simulations of the 348 FDA-approved drugs and rank ordering them, with the results being output to `lgfe.csv`:

```
$SILCSBIODIR/silcs-hotspots/4b_fda_top20_analysis prot=<prot PDB>␣
↪hotspotspdb=<my_hotspots.pdb>
```

The third step will set up and submit jobs to compute rSASA values for top compounds.

```
$SILCSBIODIR/silcs-hotspots/4c_fda_top20_analysis prot=<prot PDB>␣
↪hotspotspdb=<my_hotspots.pdb>
```

An additional option for this third step is:

- `top` : Number of top compounds with best LGFE scores to select for subsequent analysis. By default, this is set to `top=20` as suggested by the names of the job scripts in this section. If you change this default, make sure to include the same option in the next final step.

The fourth and final step will collect rSASA results, plot rSASA-vs-LGFE, and create a directory `fda_top20_analysis` containing the final results:

```
$SILCSBIODIR/silcs-hotspots/4d_fda_top20_analysis prot=<prot PDB>␣
↪hotspotspdb=<my_hotspots.pdb>
```

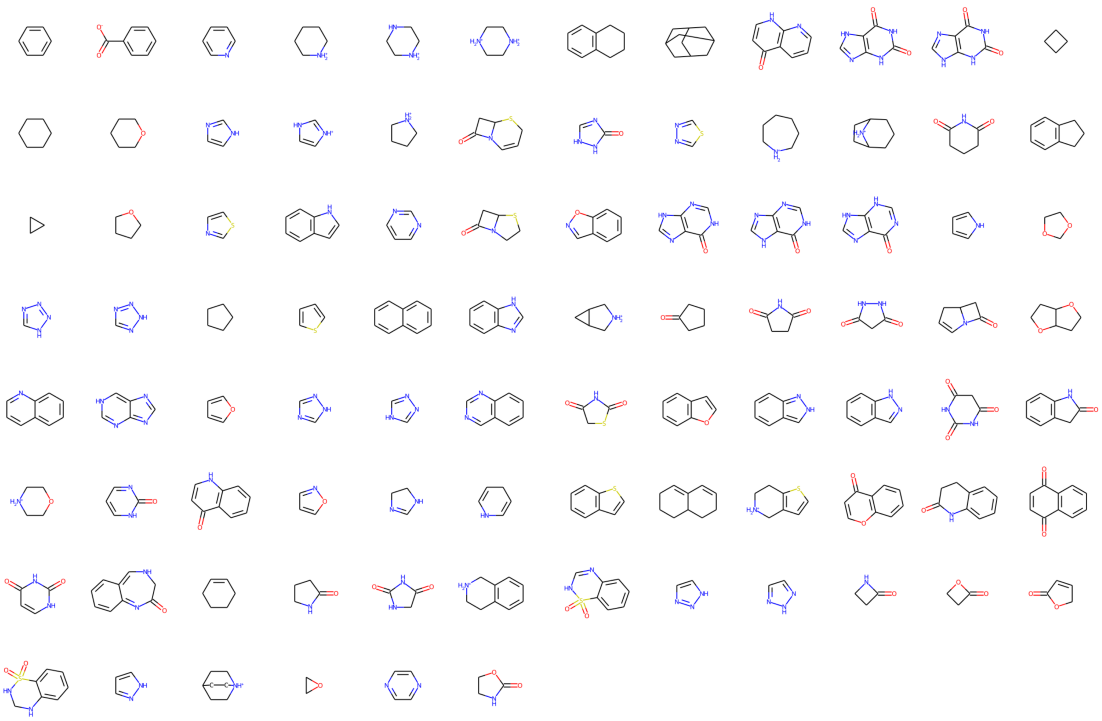
To evaluate a different subset of hotspots, simply run the above steps again with this different subset of hotspots and specify a different `sitetype` value.

Warning: Re-using the same `sitetype` value will cause your prior results to be overwritten.

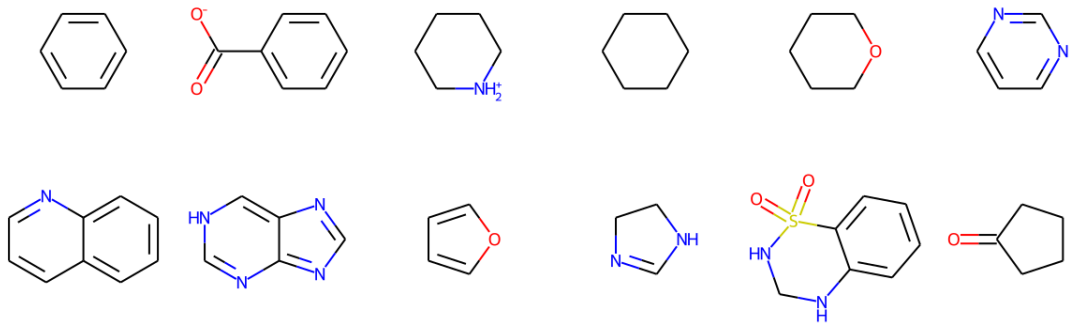
13.5 Default databases of fragment-like molecules

Three default databases of fragment-like molecules are provided with your SilcsBio software. You may choose to use one of these for your SILCS-Hotspots run, or use your own custom molecule or database of molecules. The databases are provided both as directories containing one molecule per Mol2 file and as a single SDF file containing all molecules in the database.

- `$SILCSBIODIR/data/databases/{ring_system,ring_system.sdf}`: Mono- and bicyclic rings commonly appearing in drug-like compounds [21].



- `$SILCSBIODIR/data/databases/{ring_subset,ring_subset.sdf}`: A small subset of the `ring_system` database.



- `$SILCSBIODIR/data/databases/{astex_mini_frag, astex_mini_frag.sdf}`: Polar fragments curated from Astex [20].

CHAPTER

FOURTEEN

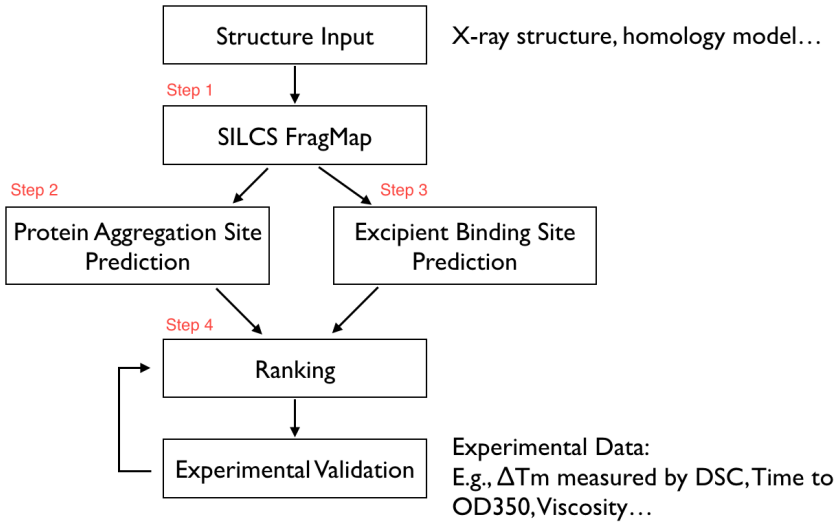
SILCS-BIOLOGICS: EXCIPIENT SCREENING FOR BIOMACROMOLECULAR THERAPEUTICS

14.1 Background

Biomacromolecular therapeutics, commonly called "biologics," are typically protein molecules that have been developed to selectively interact with a therapeutic target. Common examples of biologics are therapeutic antibodies. Biologics must be carefully formulated to maximize their stability, so as to ensure both efficacy and safety. Important determinants of stability are protein aggregation and protein denaturation. Toward maximizing stability, biologics can be formulated with excipients, which can help minimize aggregation and denaturation of the biologic in a solution formulation.

SILCS-Biologics applies the patented Site Identification by Ligand Competitive Saturation (SILCS) platform technology to the rational selection of excipients for biologics formulations. The first goal is to minimize protein-protein interactions between multiple molecules of the protein, all in the native state. This reduces the likelihood of aggregation of the protein in its native state. The second goal is to minimize denaturation by stabilizing the native (folded) state of the protein. This is accomplished by screening for excipients that can efficiently bind to the protein native state. Such binding drives the equilibrium toward the native state and away from denatured states. Achieving the second goal also contributes to minimizing aggregation, as denatured states of the protein can be aggregation-prone.

The SILCS-Biologics workflow consists of four steps: 1) SILCS simulation, 2) protein-protein interaction screening, 3) protein-excipient hotspot screening, and 4) analysis and ranking.



SILCS-Biologics is actualized as a unified tool, available through both the command line and the SilcsBio Graphical User Interface, that integrates all four steps. As such, SILCS-Biologics automatically takes care of preparing and running computing jobs that entail SILCS to generate FragMaps, SILCS-PPI to predict protein-protein interactions (PPI) that can drive aggregation, and SILCS-Hotspots to determine excipient binding sites. Please see *SILCS: Site Identification by Ligand Competitive Saturation* and *SILCS-Hotspots: Fragment Binding Sites Including Allosteric Sites* for additional details. SILCS-PPI works by aligning FragMaps from one protein with the functional groups on the surface of another protein to predict the likelihood of a PPI between the two proteins [1].

14.2 Installation

Please see *SilcsBio Software Installation* for installation directions for the SilcsBio server software and the SilcsBio Graphical User Interface.

Note: SILCS-Biologics is licensed separately from the SILCS platform. Please contact info@silcsbio.com for additional information.

Please contact support@silcsbio.com with installation questions.

14.3 Usage

In principle, SILCS-Biologics can be applied to any protein therapeutic. In practice, because SILCS is an all-atom explicit-solvent molecular dynamics methodology, the SILCS simulations underpinning SILCS-Biologics can become computationally prohibitive for large proteins, like

full-sequence antibodies or non-antibody protein therapeutics of large size. To enable application of SILCS-Biologics to larger proteins, you may split your full protein into two or three domains. SILCS-Biologics will automatically take care of managing the separate SILCS simulations for each domain and the subsequent SILCS-Hotspots and SILCS-PPI analyses, as well as collating the separate data into a single report for the full-length protein.

In what follows, we discuss three use cases ordered by increasing complexity. The first case involves a protein small enough to be run intact. We use as an example a single Fab fragment (~450 amino acids) from an antibody. The second use case involves a hypothetical fusion protein engineered by combining the sequences of two separate 500 amino acid proteins, with each one forming one domain of the fusion protein. In this second example, the full length protein is first split into its two domains, and each domain is processed as a separate input for computational expediency. The third example is a complete antibody molecule (~1300 amino acids). It is split into three domains: the two Fab regions and the one Fc region.

The three use cases are ordered by increasing complexity. In the first, only Fab-Fab PPI needs to be considered. Contrast this to the third, where FabA-FabA, FabA-FabB, FabA-Fc, FabB-FabB, FabB-Fc, and Fc-Fc PPI need to be considered. Despite this increased complexity, and the resulting need to keep track of multiple different simulations in the second and third use cases, SILCS-Biologics is easy to use in all three cases because it automatically manages all of the necessary simulations and resulting data.

Tip: In what follows, the three use cases are described using the command-line implementation of SILCS-Biologics. As of Release v 2023.1, the identical functionality is available through the SilcsBio Graphical User Interface (GUI), and is accessed by selecting *Suites* → *Biologics* from the menu bar. It is strongly recommended that users read and understand the descriptions of the use cases before using either the command line or the SilcsBio GUI to run SILCS-Biologics.

14.4 Running the complete workflow with a single command

SILCS-Biologics is designed to be self contained, allowing all calculations across all four steps to be performed with a single command. SILCS-Biologics can also be used in a modular manner, which allows the user to run each step to completion and inspect intermediate results before moving on to the next step. The three use case examples below demonstrate its application in a self-contained manner. Please make sure you read through and understand these uses case examples. Details of its modular use follow after these use case examples.

14.4.1 Use Case 1. Running SILCS-Biologics with a single protein domain

The simplest example with one input protein domain requires two input parameters, `step=1*2*3*4*` and `prot1=fab.pdb`. The first parameter requests that all four steps, including any substeps as indicated by the use of `*`, be run. The second parameter says to use use `fab.pdb` as the input protein domain file. `fab.pdb` contains coordinates for only the Fab portion of an antibody. To run this example, you can copy `fab.pdb` from `$SILCSBIODIR/examples/biologics/nist_fab/` to your local directory and run the following command:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1*2*3*4* \
    prot1=fab.pdb
```

Tip: Running the complete SILCS-Biologics workflow is a compute-intensive task. You can expect the `fab.pdb` example here to take 2 to 4 days total on a compute cluster with 10 GPU-enabled nodes. Step 1 (SILCS) will take 1 to 2 days and Step 2 (SILCS-PPI) and Step 3 (SILCS-Hotspots) will take 0.5 to 1 day each.

Tip: Step 2 (SILCS-PPI) is a RAM-intensive task. For the `fab.pdb` example here, a single SILCS-PPI job will require about 5 GB of RAM, and will fail if not enough RAM is available. You may need to adjust the job control parameters in `$SILCSBIODIR/templates/ppi/run.tmpl` to ensure that your PPI jobs will have enough RAM to successfully run.

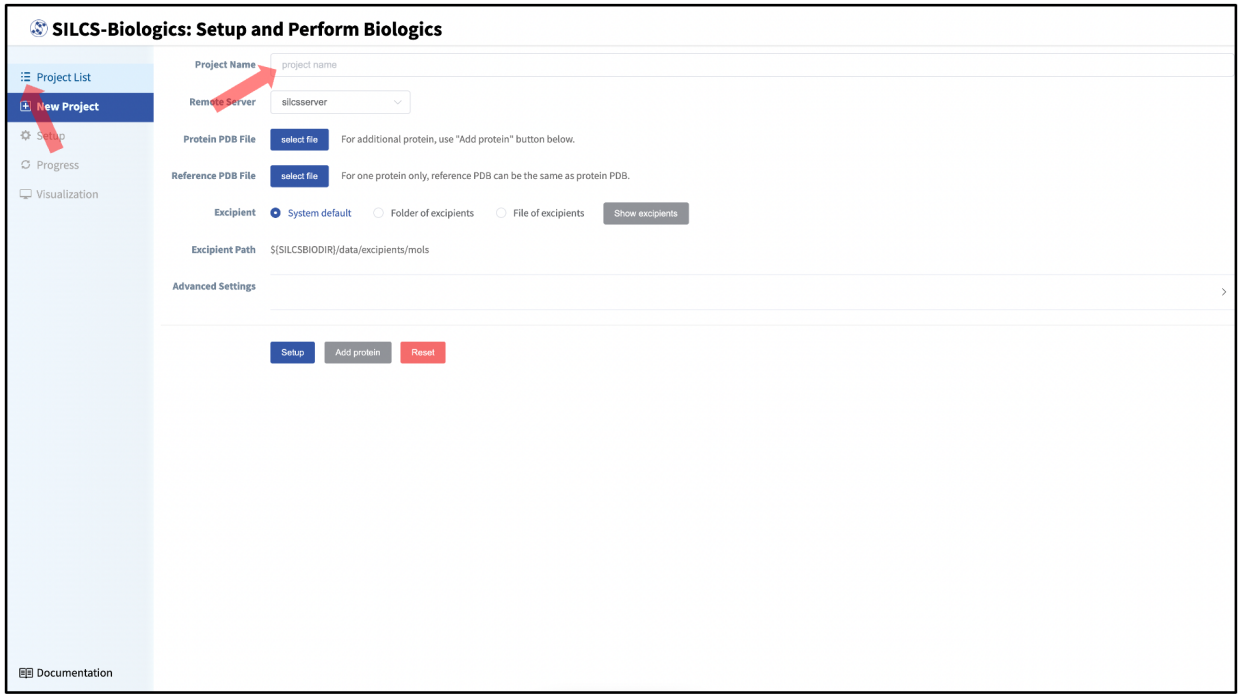
Tip: If you are unable to leave your terminal window open for the full duration of the `silcs-biologics` workflow, you can reply `y` when `silcs-biologics` asks "Do you want to run the workflow in the background using nohup?". This will launch `silcs-biologics` as a background job and allow it to keep running even if you logout from or close your terminal window. When you log back in, you can check the files `job_progress.$job_id` and

`silcs-biologics_main.$job_id.log` (see below for details on how `$job_id` is set).

By default, `silcs-biologics` uses the excipient molecules in `$SILCSBIODIR/data/excipients/mols/`: alanine, arginine, aspartate, citrate, glucose, glutamate, glycine, histidine, lactate, lysine, malate, mannitol, phosphate, proline, sorbitol, succinate, sucrose, threonine, trehalose, and valine. Each molecule is in mol2 format. If you prefer to provide your own excipients, create a directory and place a mol2 format file for each excipient you would like into that directory. Your mol2 files must contain optimized three-dimensional geometries as well as correct atom types. Additional excipients and buffers can be found in the `amino_acid/`, `buffers/`, and `sugars/` subdirectories within `$SILCSBIODIR/data/excipients/`. For example, you could create a directory `my_excipients/` in your working directory where you will run the `silcs-biologics` command, copy mol2 files of your choice from `$SILCSBIODIR/data/excipients/` into `my_excipients/`, and provide the optional parameter `molsdir=my_excipients` to run using these excipients:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1*2*3*4* \
    prot1=fab.pdb \
    molsdir=my_excipients
```

The SILCS-Biologics setup page in the SilcsBio GUI provides a convenient interface to the command line functionality described above. When using the SilcsBio GUI to run a SILCS-Biologics workflow on a protein, you will need to provide a "Project Name" that will allow you to uniquely identify the workflow run from other runs. All of your runs are accessible through the "Project List" in the left-hand column of the GUI, and you may quit the SilcsBio GUI once you have launched your workflow run and then return to it at any time and monitor progress by selecting it in the "Project List."



It is possible to differentiate excipients from the buffer. Without this distinction, all of the provided mol2 files are posed and scored using SILCS-Hotspots, and the final report includes Ligand Grid Free Energies (LGFEs) for each mol2. If a buffer mol2 is specified, it is likewise posed and scored using SILCS-Hotspots. However, the final reporting is done relative to the buffer. That is, the buffer molecule is not included in the reporting and each score for the non-buffer molecules is computed relative to the buffer molecule. For example, if you wish to use `phosphate.mol2` as the buffer molecule, you can include it in your `my_excipients/` directory and indicate it with the option `buffer=my_excipients/phosphate.mol2`:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics
    step=1*2*3*4* \
    prot1=fab.pdb \
    molsdir=my_excipients \
    buffer=my_excipients/phosphate.mol2
```

Tip: When using the SilcsBio GUI to run SILCS-Biologics, designation of an excipient molecule as buffer will be available once the compute-intensive parts of the workflow have successfully finished and the results are ready for analysis. Do make sure that any molecule you intend to analyze as a buffer is included with your other excipient molecules when you set up and perform your SILCS-Biologics workflow using the the GUI.

14.4.2 Viewing Step 4 reports

Once all four steps have completed, reports will be ready for viewing and analysis.

If you wish to open these data in a spreadsheet application, you can find them in `$WORKDIR/4_report.$job_id/`, where `$WORKDIR` is the top-level directory containing all the `silcs-biologics` workflow outputs. By default, `$job_id` is set based on the system date and time, and `$WORKDIR` is set to the directory in which the `silcs-biologics` command was executed. You can override these defaults with the options `job_id=` and `workdir=` when executing `silcs-biologics`.

Tip: See *SILCS-Biologics directory structure* for a complete description of how `silcs-biologics` organizes and names the directories and files that it creates.

Tip: If you ran your SILCS-Biologics workflow using the SilcsBio GUI, these reports will also be available for viewing via the SilcsBio GUI.

`$WORKDIR/4_report.$job_id/report_all.xlsx` has a tab named "Ranking" that contains a number of properties on a per-excipient basis:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Excipient	# Binding Site (LGFE < -1)	# Binding Site (LE < -0.25)	# Binding Site (RAA < 1000)	# Binding Site (PPIP > 0.10)	# Binding Site (LE < -0.25 and PPIP > 0.10)	Ave LGFE	Ave LE	Lowest LGFE	Sum(PPIP) /# Sites	# Binding Site ∩ Buffer	# Binding Site ¢ Buffer	# Binding Site (RAA_BUFFER < 1)
2	histidine	79	40	24	6	2	-2.79	-0.25	-5.87	0.04	0	79	1
3	proline	103	70	41	6	2	-2.44	-0.3	-5.26	0.04	0	103	0
4	sorbitol	64	53	18	3	3	-4.52	-0.38	-8.17	0.04	0	64	20
5	aspartate	67	40	34	4	4	-2.56	-0.28	-5.169	0.04	0	67	0
6	glutamate	71	45	23	4	3	-2.92	-0.29	-5.876	0.04	0	71	1
7	threonine	98	84	43	4	3	-2.74	-0.34	-5.49	0.04	0	98	1
8	valine	101	72	37	5	3	-2.55	-0.32	-5.38	0.04	0	101	1
9	alanine	115	85	69	4	4	-1.93	-0.32	-4.32	0.04	0	115	0
10	succinate	52	50	35	3	2	-2.93	-0.37	-5.17	0.04	0	52	0
11	glycine	105	94	105	3	3	-1.83	-0.37	-3.71	0.03	0	105	0
12	phosphate	117	105	113	8	8	-1.8	-0.36	-3.9	0.04	0	117	0
13	arginine	55	31	7	3	0	-3.34	-0.28	-6.94	0.04	0	55	2
14	sucrose	42	23	6	1	1	-6.06	-0.26	-10.446	0.03	0	42	26
15	mannitol	56	51	20	4	3	-4.59	-0.38	-7.88	0.04	0	56	16
16	glucose	79	66	23	4	2	-4.21	-0.35	-7.48	0.04	0	79	16
17	citrate	52	41	31	2	1	-4.1	-0.32	-6.277	0.04	0	52	10
18	lactate	106	106	106	7	7	-2.35	-0.39	-4.18	0.04	0	106	0
19	lysine	60	50	3	3	2	-3.39	-0.34	-7.47	0.04	0	60	1
20	malate	68	66	40	3	3	-3.34	-0.37	-5.57	0.04	0	68	1
21	trehalose	42	24	8	1	1	-6.21	-0.27	-10.86	0.04	0	42	25

The contents of the columns are:

A: The excipient.

B: The number of binding sites where the excipient binds with an Ligand Grid Free Energy (LGFE)

< -1 kcal/mol. LGFE approximates the binding affinity.

C: The number of binding sites where the excipient Ligand Efficiency (LE) < -0.25 kcal/mol. LE provides a metric of the binding affinity normalized for molecular size. LE = LGFE / N_heavy_atom, where N_heavy_atom is the number of non-hydrogen atoms in the excipient.

D: The number of binding sites found with a Relative Affinity Analysis metric (RAA) < 1000. RAA indicates the number of "strong" binding sites accessible to each molecule. The RAA is a ratio of dissociation constants (Kd's), where the numerator is the Kd for the excipient at a given pocket and the denominator is the Kd for the best-scoring (highest affinity) binding pose for that excipient across the entire protein. Kd is computed from LGFE, where LGFE is taken to be the free energy.

E: The number of binding sites with a sum of PPI preference for residues in the binding site (PPIP) > 0.10. A high PPIP value for a binding site suggests that site is more likely to contribute to a PPI.

F: The number of binding sites having both LE < -0.25 kcal/mol and PPIP > 0.10.

G: The average LGFE for the excipient.

H: The average LE for the excipient.

I: The LGFE for the best-scoring (highest affinity) pose of the excipient.

J: Sum(PPIP)/# sites is the sum of the PPIP values for all of the binding sites in column B divided by the value of column B. This gives an average PPIP value computed accross the excipient-favored binding sites.

K: The number of binding sites that overlap with buffer binding sites. If no buffer was specified, this number will be zero. If all excipient binding sites are also capable of binding buffer, this number will be equal to the value in column B.

L: The number of excipient binding sites that are not also buffer binding sites. If no buffer was specified, this number will be equal to the value in column B.

M: The number of binding sites where the excipient has a higher binding affinity (i.e, lower Kd or, equivalently, more negative LGFE) than the buffer at that site. In otherwords, the number of binding sites where this excipient will out-compete buffer for binding. If no buffer was specified, this computation is done relative to a hypotheical buffer with an LGFE of -4 kcal/mol.

`$WORKDIR/4_report.$job_id/report_output.xlsx` contains fractional occupancy data for excipient binding sites. Individual sheets or tabs in this file correspond to separate input formulations. Information includes the excipients and buffer that bind at each site identified for that formulation along with selected SILCS metrics associated with the occupancy calculations and PPI interactions. The top row identifies the buffer and excipients used in that particular formulation along with their concentrations.

With regard to occupancies,

- Kd: dissociation constant, such that LGFE=R*T*ln(Kd)
- [L]: concentration of the ligand/excipient, with unit Wt/V (g/mL) % => mM

- Occupancy = [L]/([L] + Kd) and has range (0, 1). A high occupancy value (close to 1) means that excipient molecule has a high probability of binding at the site. NOTE: Occupany values relative to buffer are also accessible.

Below is an example of fractional occupancy data. The example metrics assume an occupancy cutoff of 0.8. This and other terms may be varied for reporting.

Excipient	# Binding Site (PPIP > 0.10)	# Binding Site (Occup. > 0.80)	# Binding Site (Occup. > 0.80 and PPIP > 0.10)	# Binding Site (Excipient1) (Occup. > 0.80)	# Binding Site (Excipient1) (Occup. > 0.80 and PPIP > 0.10)	# Binding Site (Excipient2) (Occup. > 0.80)	# Binding Site (Excipient2) (Occup. > 0.80 and PPIP > 0.10)	# Binding Site (Excipient3) (Occup. > 0.80)	# Binding Site (Excipient3) (Occup. > 0.80 and PPIP > 0.10)
A-Met	23	161	23	161	23	0	0	0	0
A-Pro-Met	43	180	31	137	20	43	11	0	0
A-Met-Arg	35	175	27	130	19	45	8	0	0
A-Pro-Tre-Iso	53	188	32	92	16	85	12	11	4
A-Pro-Iso	43	185	30	158	23	27	7	0	0
A-Pro-Ala	47	131	27	105	22	26	5	0	0
A-Pro-Tre-Arg	54	181	30	85	12	59	9	37	9
A-Pro-Tre-Ala	57	150	27	86	12	53	13	11	2
A-Pro-Tre	47	170	33	86	12	84	21	0	0
A-Pro-Lys-Tre	54	174	31	85	12	61	12	28	7
A-Pro	37	176	33	176	33	0	0	0	0
A-Iso-Tre	43	220	36	135	24	85	12	0	0
A-Pro-Lys	44	160	31	92	16	68	15	0	0
A-Pro-Arg-Glu	50	161	27	83	15	40	9	38	3
A-Ala	39	134	27	134	27	0	0	0	0
A-Lys	22	132	22	132	22	0	0	0	0
A-Arg	23	122	21	122	21	0	0	0	0
A-Arg-Glu	34	147	24	76	13	71	11	0	0
A-Tre	12	86	12	86	12	0	0	0	0
A-Su	15	87	15	87	15	0	0	0	0
A-NaCl	12	11	3	11	3	0	0	0	0

"# Binding Site (Excipient1),(Occup. > 0.80)" suggests the number of excipient binding sites with occupancy > 0.8. If the formulation has multiple excipients, then "excipient1" also represents the excipient with the most binding sites, n comparison with other excipient(s).

"# Binding Site (Excipient2),(Occup. > 0.80)" suggests the number of sites occupied by the excipient with 2nd most binding sites.

"# Binding Site (Excipient3),(Occup. > 0.80)" suggests the number of sites occupied by the excipient with 3rd most binding sites.

"# Binding Site (Excipient1), (Occup. > 0.80 and PPIP > 0.10)" suggests the number of excipient binding sites with high occupancy that coincide with likely PPI. If the formulation has multiple different excipients, then "excipient1" also represents the excipient with the most binding sites, in comparison with other excipient(s).

"# Binding Site (Excipient2),(Occup. > 0.80 and PPIP > 0.10)" suggests the number of second most excipient binding sites with high occupancy that coincide with likely PPI.

"# Binding Site (Excipient3),(Occup. > 0.80 and PPIP > 0.10)" suggests the number of third most excipient binding sites with high occupancy that coincide with likely PPI.

"# Binding Site (Occup. > 0.80)" is the sum of any excipient binding sites with high occupancy. Each binding site with higher occupancy will be counted once.

"# Binding Site (PPIP > 0.1)" suggests the number of all excipient binding sites that coincide with potential PPI.

"# Binding Site (Occup. > 0.80 and PPIP > 0.10)" is the sum of all excipient binding sites with high occupancy that coincide with likely PPI.

While a spreadsheet application or the SilcsBio GUI are the preferred means to view `.xlsx` files, a web view option is available for `report_all.xlsx`:

```
cd $WORKDIR/4_report.$job_id/view
sh run.sh
```

`sh run.sh` will print out a message like:

```
* Serving Flask app "main.py" (lazy loading)
* Environment: development
* Debug mode: on
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 222-574-256
```

Open a web browser and go to the URL displayed in the output (e.g., http://127.0.0.1:5000 or http://<IP address>:5000).

If you are running the web view from a remote server and if the server is behind a firewall, you may not able to directly access the web view. In this case, you can use a method called SSH port forwarding. This uses an SSH connection to the remote server to access the web view securely.

If you are using Linux, MacOS, or Linux subsystem on Windows as your desktop:

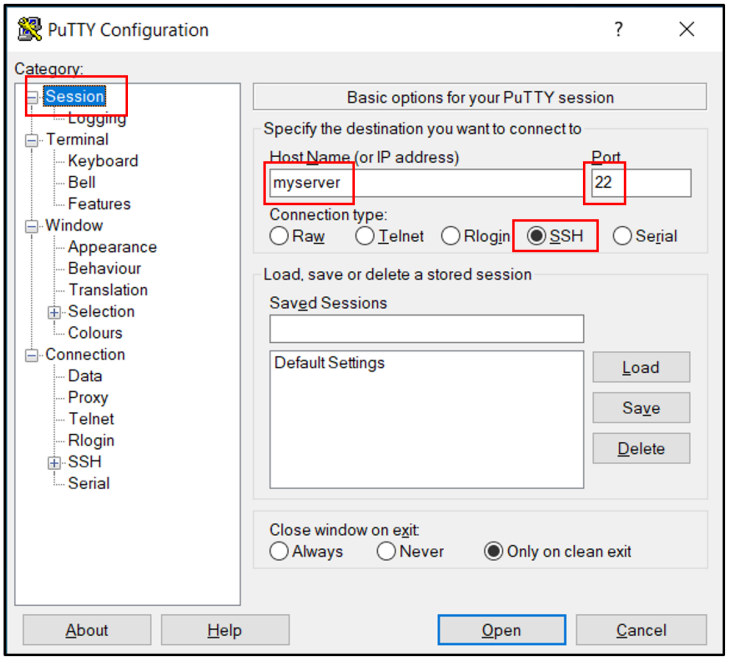
1. Open a terminal;
2. SSH to the remote server using the following example SSH command in the terminal;

```
ssh -L 5000:localhost:5000 username@servername
```

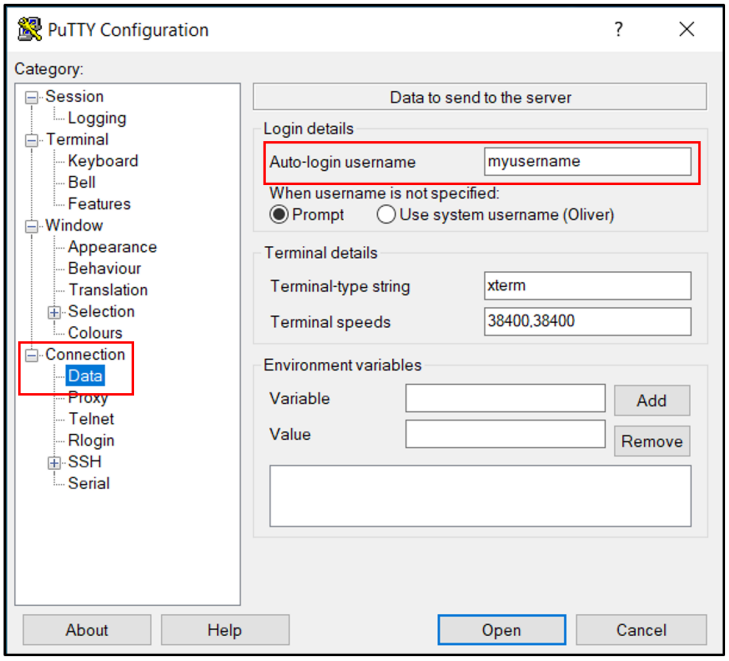
3. Open a tab in the web browser, and navigate to `localhost:5000`.

If you are using Windows as your desktop and and the PuTTY application for SSH access:

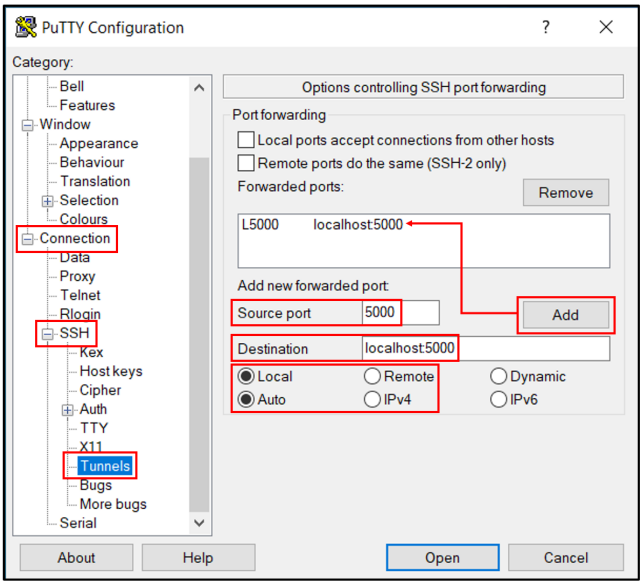
1. Open PuTTY
2. Select "Session", and
   - Type in the target host name and port (default: 22);
   - Choose SSH as the connection type (default);



3. Select "Connection -> Data", and
   - Type in your username as the auto-login username



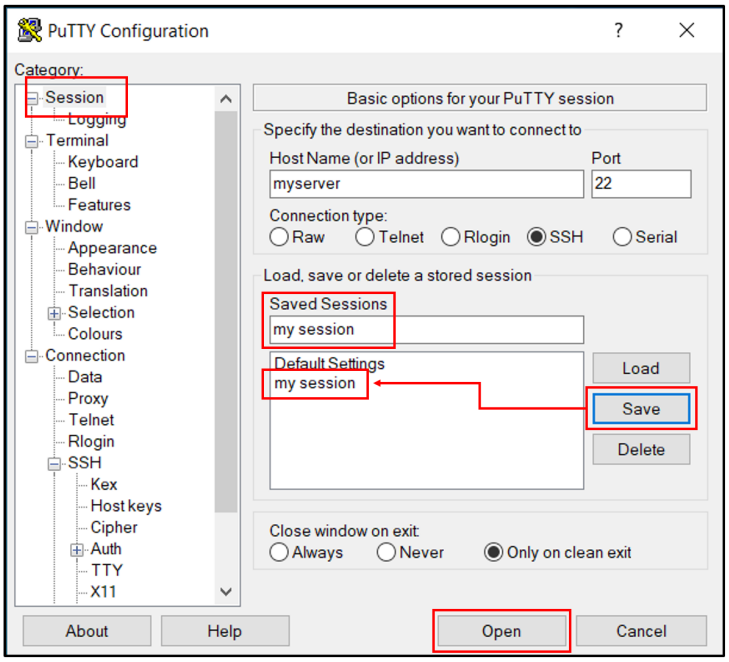
- Select “Connection -> SSH -> Tunnels”, and
  - In “Source port”, type in `5000`;
  - In “Destination”, type in `localhost:5000`;
  - Click the “Add” button on the right side of “Source port” box;
  - Turn on the *Local* and *Auto* radio buttions (default) below the “Destination” box;



4. Select “Session”, and
   1. Choose a name for the session;

2. Click the "Save" button to save the session with all your parameters;

5. Click "Open" to open a terminal to connect to your remote server.



6. Open a tab in the web browser, and navigate to `localhost:5000`.

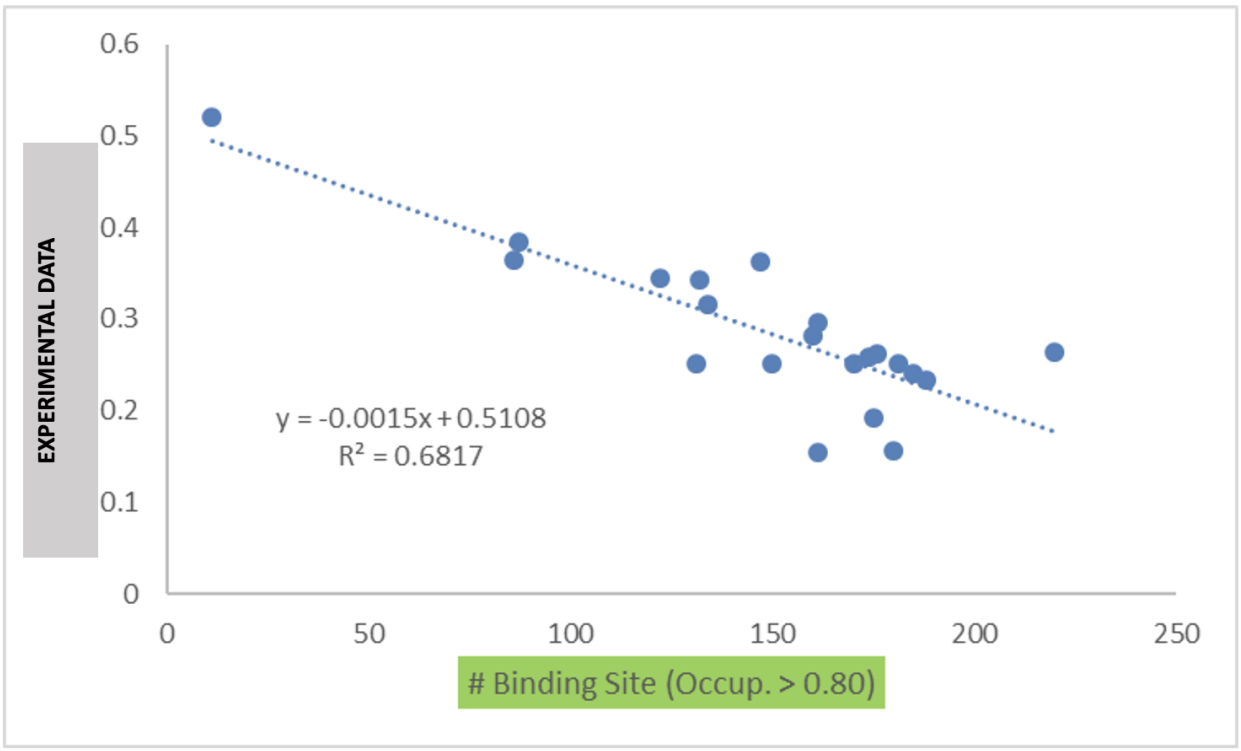
14.4.3 Data interpretation

The data output from Step 4 can help connect experimental observables to the molecular details of protein-excipient interaction. In doing so, these simulation data can help both rationalize known trends for existing data on excipients and suggest the choice of new excipients for additional testing. With regard to the former, the most straightforward approach is to compare available experimental data to the Step 4 output and note correlations:

Excipient	# Binding Site (LGFE < -1)	# Binding Site (LE < -0.25)	# Binding Site (PPIP > 0.10)	# Binding Site (LE < -0.20 and PPIP > 0.10)	Experimental Data
trehalose	86	61	12	11	
alanine	210	207	38	37	
isoleucine	197	193	32	31	
arginine	129	90	23	14	
methionine	161	159	22	22	
glutamate	113	108	15	14	
proline	203	176	36	32	
sucrose	87	62	15	9	
lysine	135	131	22	21	
Excipient					

Please see [2] for a real-world example.

Below is an example of blinded data from a collaboration case study that demonstrates the utility of the binding site occupancy metrics computed by SILCS-Biologics and available in `report_output.xlsx`:



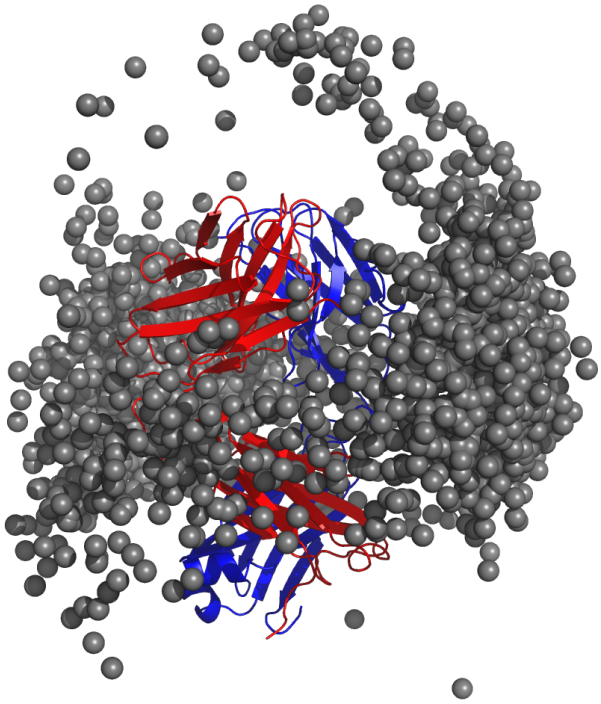
14.4.4 Visualizing SILCS FragMaps and SILCS-PPI results

Tip: If you ran your SILCS-Biologics workflow using the SilcsBio GUI, SILCS FragMaps and SILCS-PPI results visualization will be available to you directly in the SilcsBio GUI upon successful completion of your SILCS-Biologics job.

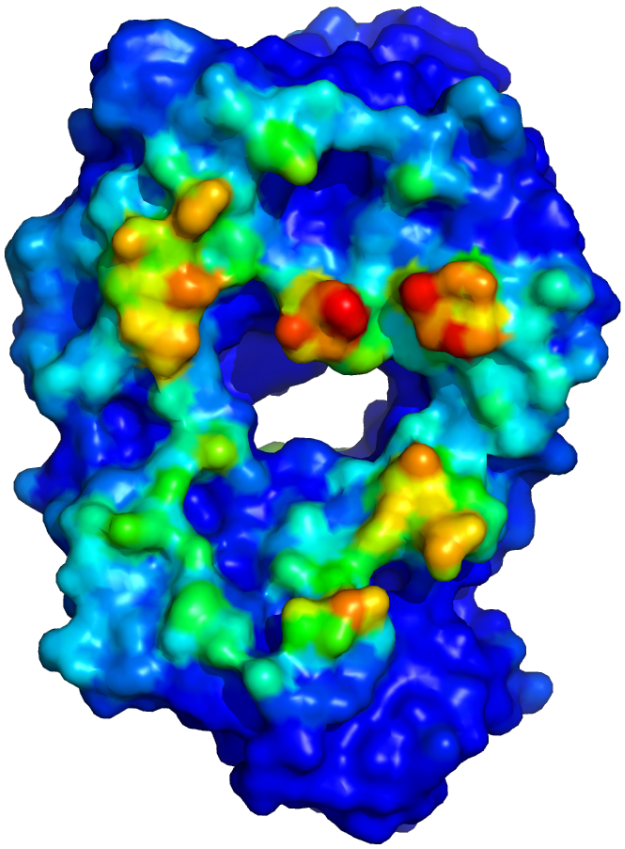
SILCS FragMaps generated during Step 1 of the SILCS-Biologics workflow can be easily visualized using your choice of MOE, PyMol, or VMD molecular graphics software packages. FragMaps will be located in `$WORKDIR/1_fragmap.$job_id/fab/silcs_fragmaps_fab`. Please see *Visualizing SILCS FragMaps* for detailed instructions.

SILCS-Biologics Step 2 SILCS-PPI results can also be viewed using molecular graphics software. For the purposes of SILCS-PPI, one protein is considered the "receptor" and the other the "ligand". In the current use case with only one input protein domain, `prot1=fab.pdb`, `fab.pdb` data are used for both the "receptor" and the "ligand". For the purpose of SILCS-PPI, the ligand protein is docked to the receptor protein by comprehensively sampling locations and orientations of the ligand protein relative to the receptor protein. The relative locations and orientations are scored based on the overlap of the ligand functional groups and the receptor protein SILCS FragMaps (computed in Step 1). The docked poses of the ligand are then clustered as part of SILCS-PPI.

The `$WORKDIR/2_ppi.$job_id/fab_fab/3_ppi/receptor_clusters.pdb` file contains the receptor protein structure and the cluster centroids. The cluster centroids in this file all have the same chain name, `Z`. Below is a molecular graphics image of `receptor_clusters.pdb`, with the receptor protein shown as ribbons and the cluster centroids as spheres. The CDR loops of the Fab are pointing to the top of the image, heavy chain amino acids are in red, light chain amino acids are in blue, and cluster centroids in gray.



`$WORKDIR/2_ppi.$job_id/fab_fab/3_ppi/receptor_surf_contact.pdb` contains PPI propensity values mapped onto the receptor protein and recorded in the B-factor column, which provides an easy way to visualize which residues are most at risk for contributing to PPI, and, therefore, to aggregation of the biologic in its native (folded) state:



14.4.5 Use Case 2. Running SILCS-Biologics for a protein with two domains

In this example, we start with a full-length structure of a hypothetical protein, `fullpdb.pdb`, with two independent domains, domain **X** and **Y**. To begin, you must create two input protein domain files corresponding to each domain.

To do so, simply make two copies of `fullpdb.pdb` and name one `protx.pdb` and the other `proty.pdb`. Then, edit `protx.pdb` and delete the amino acids corresponding to the domain Y. Repeat the same process for `proty.pdb`; edit `proty.pdb` and delete the amino acids corresponding to the domain X. Make sure that there is no overlap between the amino acids contained in `protx.pdb` and `proty.pdb`. In general, all amino acids in `fullpdb.pdb` should be accounted for by the combination of `protx.pdb` and `proty.pdb`; however, if long flexible peptide connects `protx.pdb` to `proty.pdb`, the amino acids in that peptide region can be excluded from `protx.pdb` and `proty.pdb` for computational expediency with likely minimal impact on the final results.

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1*2*3*4* \
    prot1=protx.pdb \
    prot2=proty.pdb \
    fullpdb=fullpdb.pdb
```

Note that there are now two additional input parameters relative to Use Case 1: `prot2=proty.pdb` and `fullpdb=fullpdb.pdb`. The addition of `prot2=` indicates a second input protein domain must be considered and the addition of `fullpdb=` provides a reference structure for collating PPI contact data as well as for excluding surface-exposed amino acids in `protx.pdb` and `proty.pdb` that are in fact buried in the context of `fullpdb.pdb`. This latter point is important for both the SILCS-PPI and SILCS-Hotspots analysis to ensure that buried amino acids in `full.pdb` are not incorrectly noted as either contributing to PPI or having hotspots. Both of the additional input parameters, `prot2=` and `fullpdb=`, are required.

Note: If you do not have the full-length structure, but only have the structures of individual domains, then you will have to create the full-length structure using external molecular modeling tools, such as homology modeling software or simple alignment to a known full-length homologous crystal structure, to utilize this workflow. Save the resulting full-length structure as `fullpdb.pdb` and create `protx.pdb` and `proty.pdb` as described at the beginning of this example.

As with the Use Case 1, you may specify a directory containing a custom set of excipients by adding `molsdir=<path to my excipient directory>` and/or rank the excipients relative to a buffer molecule by adding `buffer=<path to my buffer mol2 file>`.

14.4.6 Use Case 3. Running SILCS-Biologics for a protein with three domains

A complete antibody molecule is a large protein, consisting of ~1300 amino acids. Additionally, for the purposes of molecular dynamics simulations, it requires a very large simulation box for solvation because of its extended Y-shaped conformation. Splitting it into three domains, specifically its two Fab regions and the one Fc region, makes the molecular dynamics-based SILCS simulations substantially more computationally tractable. Not only are the individual domains each ~1/3 the size of the full antibody, but also, when considered individually, the Fab and Fc regions are very compact and therefore can be simulated inside relatively small simulation boxes to achieve appropriate solvation.

We start with a full-length structure of the antibody, `antibody.pdb`. From this file, you must create three input protein domain files corresponding to the two Fab regions and the one Fc region, which we will call `faba.pdb`, `fabb.pdb`, and `fc.pdb`, respectively. Make three copies of `antibody.pdb` and name one `faba.pdb`, another `fabb.pdb`, and the third `fc.pdb`. Then, edit `faba.pdb` and delete the amino acids corresponding to the second Fab and the Fc regions.

Edit `fabb.pdb` and delete the amino acids corresponding to the first Fab and the Fc region. And edit `fc.pdb` and delete the amino acids corresponding to the first Fab and the second Fab regions. Make sure that there is no overlap between the amino acids contained in `faba.pdb`, `fabb.pdb` and `fc.pdb`.

In general, all amino acids in `fullpdb.pdb` should be accounted for by the combination of `faba.pdb`, `fabb.pdb`, and `fc.pdb`; however, if long flexible peptide connects `faba.pdb`, `fabb.pdb`, and/or `fc.pdb`, the amino acids in that peptide region can be excluded from `faba.pdb`, `fabb.pdb`, and `fc.pdb` for computational expediency with likely minimal impact on the final results. An example can be found in `$SILCSBIODIR/examples/nist_mab/` folder.

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1*2*3*4* \
    prot1=faba.pdb \
    prot2=fabb.pdb \
    prot3=fc.pdb \
    fullpdb=antibody.pdb
```

Note that there is one additional required input parameter relative to Use Case 2: `prot3=fc.pdb`. The addition of `prot3=` indicates a third input protein domain will be considered. As with Use Case 2, `fullpdb=` indicates a reference structure for collating PPI contact data as well as for excluding surface-exposed amino acids in individual input protein domains that are in fact buried in the context of `antibody.pdb`. Note that all of the input parameters in the above example are required.

Note: If you do not have full-length antibody structure, but only have the structures of Fab and Fc domains, then you will have to create the full-length structure using other molecular modeling tools (such as homology modeling software.

Alternatively, you can align the domains onto other full-length IgG structures (e.g., PDB:1HZH from RCSB database). Save the resulting full-length structure as `fullpdb.pdb` and create `faba.pdb`, `fabb.pdb`, and `fc.pdb` as described at the beginning of this example.

As with the other use cases, you may specify a directory containing a custom set of excipients by adding `molsdir=<path to my excipient directory>` and/or rank the excipients relative to a buffer molecule by adding `buffer=<path to my buffer mol2 file>`.

14.5 Running the workflow one step at a time

`silcs-biologics` can be used to run the SILCS-Biologics workflow in a stepwise fashion, with `step=1*` requesting only the SILCS simulations be run, `step=2*` requesting SILCS-PPI be run, `step=3*` requesting SILCS-Hotspots be run, and `step=4*` requesting processing of data from the prior steps and generation of the final report. Finer control at the level of the smaller substeps can also be requested, as detailed in the following example.

14.5.1 Stepwise Use Case 1. One input protein domain

In this example, we use `fab.pdb` as the input protein domain. You can find this file in `$SILCSBIODIR/examples/biologics/nist_fab/`.

1. Step 1: Run SILCS and generate FragMaps

   You can run all the substeps of Step 1 automatically:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1* \
    prot1=fab.pdb \
    job_id=try01
```

   The `job_id=` parameter is used to group together job inputs and outputs from different steps/substeps. Therefore, when using `silcs-biologics` in a stepwise fashion, you will need to provide the same value for this parameter across all steps/substeps for the system you are modeling.

   Alternatively, you can run substep by substep:

   - Step 1a: Set up SILCS simulations

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1a \
    prot1=fab.pdb \
    job_id=try01
```

   - Step 1b: Run SILCS simulations

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1b \
    prot1=fab.pdb \
    job_id=try01
```

     These SILCS simulations for `fab.pdb` will take 1 to 2 days on a cluster with 10 GPU-enabled compute nodes. If the simulation jobs fail due to external factors such as a power outage, server maintenance, etc., you can use the exact same command to resume the SILCS jobs from the point where they failed (as opposed to needing to restart them from the very beginning).

   - Step 1c: Generate SILCS FragMaps

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1c \
    prot1=fab.pdb \
    job_id=try01
```

2. Step 2: Run SILCS-PPI

To continue with Step 2 using your outputs from Step 1, run your commands in the same directory where you ran your Step 1 commands and use the same `$job_id` you used for Step 1.

You can run all the substeps of Step 2 automatically:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=2* \
    prot1=fab.pdb \
    job_id=try01
```

Alternatively, you can run each substep by using the following commands:

- Sub-step 2a: Run SILCS-PPI jobs

```
$SILCSBIODIR/silcs-biologics/silcs-biologics step=2a \
    prot1=fab.pdb \
    job_id=try01
```

- Sub-step 2b: Collect PPI results

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=2b \
    prot1=fab.pdb \
    job_id=try01
```

3. Step 3: Run SILCS-Hotspots

To continue with Step 3 using your outputs from Step 2, run your commands in the same directory where you ran your Step 2 commands and use the same `$job_id` you used for Step 1 and Step 2.

As described previously in *Running the complete workflow with a single command*, you can specify a custom set of excipient molecules using the optional `molsdir=` parameter.

You can run all the substeps of Step 3 automatically:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=3* \
    prot1=fab.pdb \
    job_id=try01
```

Alternatively, you can run each substep by using the following commands:

- Step 3a: Run excipient docking

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=3a \
    prot1=fab.pdb \
    job_id=try01
```

- Step 3b: Cluster the hotspots

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=3b \
    prot1=fab.pdb \
    job_id=try01
```

4. Step 4: Collate and analyze data from prior steps and generate report

   The SILCS-Biologics data can be processed into a web report or a spreadsheet report. As described previously in *Running the complete workflow with a single command*, you can specify a buffer molecule that will be used as a reference for ranking of the excipients using the optional `buffer=` parameter.

   - Generate a web report

```
$SILCSBIODIR/silcs-biologics/silcs-biologics
    step=4a \
    prot1=fab.pdb \
    job_id=try01
```

   - Run step 4b only (Spreadsheet_Report)

```
$SILCSBIODIR/silcs-biologics/silcs-biologics
    step=4b \
    prot1=fab.pdb \
    job_id=try01
```

14.5.2 Stepwise Use Case 2. Two input protein domains

Follow the instructions for *Use Case 1. Running SILCS-Biologics with a single protein domain*, and, in addition to `prot1=`, provide values for `prot2=` and `fullpdb=`.

14.5.3 Stepwise Use Case 3. Three input protein domains

Follow the instructions for *Use Case 1. Running SILCS-Biologics with a single protein domain*, and, in addition to `prot1=`, provide values for `prot2=`, `prot3=`, and `fullpdb=`.

14.6 Re-running a system with a different set of excipients

If, after having run the SILCS-Biologics workflow, you decide you would like results for additional excipients, you can simply reuse your existing results from Step 1 (SILCS) and Step 2 (SILCS-

PPI) without re-running these two steps. To do so, you will need to use a new `$job_id` for the new set of excipients. Let us assume your original simulations were in `$WORKDIR` and had the `$job_id` value `try01`. After initially completing the SILCS-Biologics workflow, you would have the following directories:

```
$WORKDIR/1_fragmap.try01
$WORKDIR/2_ppi.try01
$WORKDIR/3_excipients.try01
$WORKDIR/4_report.try01
```

To re-use your existing SILCS FragMap and SILCS-PPI data, copy the contents of their respective directories and associate the new directories with a new `$job_id`, `try02`:

```
cd $WORKDIR
cp -r 1_fragmap.try01 1_fragmap.try02
cp -r 2_ppi.try01 2_ppi.try02
```

Tip: To save disk space, you may create symbolic links to instead of making copies of your existing data.

```
cd $WORKDIR
ln -s 1_fragmap.try01 1_fragmap.try02
ln -s 2_ppi.try01 2_ppi.try02
```

However, be mindful that with symbolic links any changes you make to `1_fragmap.try02` or to `2_ppi.try02` (including deleting files) will also be made to `1_fragmap.try01` and `2_ppi.try01`. Therefore, we strongly recommend you use `cp -r` instead of `ln -s` if you have disk space available.

Now, re-run Step 3 and Step 4 using `job_id=try02`:

```
$SILCSBIODIR/silcs-biologics/silcs-biologics
    step=3*4* \
    prot1=fab.pdb \
    molsdir=my_excipients_new
```

The above command will use excipient files contained in `$WORKDIR/my_excipients_new` for running the SILCS-Hotspots calculations and creating reports, and these results will be in the new directories `3_excipients.try02` and `4_report.try02`, respectively. If you like, you can also add the `buffer=` option. This same approach will also work for two or three input protein domains. Simply use the `prot2=`, `prot3=`, and `fullpdb=` options as you used for your initial `job_id=try01` run through the SILCS-Biologics workflow.

14.7 Conserving computing resources for antibody simulations

The most straightforward way to apply SILCS-Biologics to an antibody is to follow the directions for *Use Case 3. Running SILCS-Biologics for a protein with three domains*, and we strongly recommend that new users use that approach. That said, it is possible to conserve computing resources by taking advantage of the fact that for a normal antibody (i.e., not a bi-specific antibody), the amino acid composition of the two Fab regions is identical. In other words, the amino acid sequence in `faba.pdb` is identical to `fabb.pdb`, and their structures are therefore also very similar. As such, a single set of SILCS FragMaps can be used for both Fab regions instead of computing FragMaps independently for both `faba.pdb` and for `fabb.pdb`.

Similar to *Use Case 3. Running SILCS-Biologics for a protein with three domains*, you will have to create `faba.pdb`, `fabb.pdb`, and `fc.pdb` files from the full-length antibody structure, `fulllength.pdb` before we begin.

1. Generate FragMaps for one of the Fab domain using the following command.

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1* \
    prot1=faba.pdb \
    job_id=try01
```

2. Generate FragMaps for `Fc` domain using the following command (This can be done in parallel with the step 1).

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=1* \
    prot1=fc.pdb \
    job_id=try01
```

3. Generate FragMaps for another Fab domain using the following command.

```
python $SILCSBIODIR/utils/python/reorient_maps.py faba.pdb fabb.
↪pdb \
    1_fragmap.try01/faba/silcs_fragmaps_faba \
    --outdir 1_fragmap.try01/fabb/silcs_fragmaps_fabb/maps
```

4. Continue running the remaining steps from 2 to 4 using the following command.

```
$SILCSBIODIR/silcs-biologics/silcs-biologics \
    step=2*3*4* \
    prot1=faba.pdb \
    prot2=fabb.pdb \
    prot3=fc.pdb \
    fullpdb=antibody.pdb
```

14.8 SILCS-Biologics directory structure

`silcs-biologics` creates the below directories in `$WORKDIR`. By default, `$WORKDIR` is the directory in which `silcs-biologics` was run. Otherwise, `$WORKDIR` is determined by the parameter passed to `silcs-biologics` through the command line option `workdir=`.

Step 1 SILCS FragMap results will be in `1_fragmap.$job_id/`,

```
                                    |>>> 1_setup
                      |>>> $prot1 >>> |>>> 2a_run_gcmd
                      |               |>>> 2b_gen_maps
                      |               |>>> silcs_fragmaps_$prot1
1_fragmap.$job_id >>> |>>> $prot2 ...
                      |>>> $prot3 ...
```

Step 2 SILCS-PPI results will be in `1_ppi.$job_id/`,

```
                                        |>>> maps1
                  |>>> $prot1_$prot1 >>> |>>> maps2
                  |                       |>>> 3_ppi
2_ppi.$job_id >>> |>>> $prot1_$prot2 ...
                  |>>> $prot1_$prot3 ...
                  |>>> $prot2_$prot1 ...
                  |>>> $prot2_$prot2 ...
                  |>>> $prot2_$prot3 ...
                  |>>> $prot3_$prot1 ...
                  |>>> $prot3_$prot2 ...
                  |>>> $prot3_$prot3 ...
```

Step 3 SILCS-Hotspots excipient docking results will be in `3_excipients.$job_id/`,

```
                          |>>> mols
                          |>>> $prot1 >>> 4_hotspots >>> ***
3_excipients.$job_id >>>  |>>> $prot2 ...
                          |>>> $prot3 ...
```

and Step 4 reporting will be in `4_report.$job_id`,

```
4_report.$job_id
```

CHAPTER

FIFTEEN

SSFEP: SINGLE STEP FREE ENERGY PERTURBATION

15.1 Background

Free energy perturbation (FEP) has long been considered the gold standard in calculating relative ligand-binding free energies. However, FEP is often impractical for evaluating large number of changes to a parent ligand due to the large computational cost. Single Step Free Energy Perturbation (SSFEP) is an alternative that can be orders of magnitude faster than conventional FEP when evaluating large number of changes to a parent ligand, while maintaining useful accuracy for small functional group modifications [5].

The SSFEP method involves post-processing of MD simulation data of a ligand in a given environment in the canonical ensemble to estimate the alchemical free energy change of chemically modifying the ligand. Zwanzig's FEP formula is used,

$$\Delta G^{\mathrm{env}}_{\mathrm{L1}\to\mathrm{L2}} = -k_{\mathrm{B}}T \ln \left\langle e^{-\beta \Delta E} \right\rangle_{\mathrm{L1}} \tag{15.1}$$

where k_{B} is the Boltzmann constant and T is the temperature. The angular brackets indicate an average of the exponential factor over the MD trajectory of ligand $L1$ in the given environment, `env`, which can be either the solvated protein or water. ΔE is the energy difference between the two systems involving L1 and L2, which in practice is computed as the difference in the interaction energies of the two ligands in the corresponding environment:

$$\Delta E = E_{L2-\mathrm{env}} - E_{L1-\mathrm{env}}$$

The environment `env` in each system is defined as all non-ligand atoms. As the environment is constant between the two ligands, the internal environmental energy cancels exactly during the computation of ΔE. In addition, as the difference between L1 and L2 involves a very small number of heavy atom modifications, we expect any differential intra-ligand energy terms to also cancel exactly between the solution and protein environments. Therefore, once $\Delta G^{\mathrm{protein}}_{L1\to L2}$ and $\Delta G^{\mathrm{water}}_{L1\to L2}$ are computed according to Eq. (15.1), the relative binding free energy is given by

$$\Delta\Delta G^{\mathrm{bind}}_{L1\to L2} = G^{\mathrm{protein}}_{L1\to L2} - G^{\mathrm{water}}_{L1\to L2}$$

The SSFEP approach allows the data from simulation of a single protein-ligand complex to be rapidly post-processed to evaluate tens to hundreds of potential modifications involving multiple

sites on the parent ligand. Given this, the best results are achieved when SSFEP is used to evaluate small modifications to the parent ligand.

In a recent study [6], the ability of standard FEP and SSFEP to reproduce the experimental relative binding affinities of known ligands for two proteins, ACK1 and p38 MAP kinase, was tested. SSFEP was able to produce comparable results to full FEP while requiring a small fraction of the computational resources.

15.2 Running SSFEP from the SilcsBio GUI

Please see *SSFEP simulation from the GUI* in the *Graphical User Interface Quickstart* for instructions on running SSFEP from the SilcsBio GUI.

15.3 Running SSFEP from the command line interface

The following usage details are provided for completeness. **We strongly recommend using the SilcsBio GUI to set up, run, and analyze SSFEP calculations**.

To perform the SSFEP precomputation simulations, protein coordinates in PDB file format and parent ligand coordinates in Mol2 file format are required. The protein should have termini properly capped, missing loops built or the ends of the missing loops capped, standard atom and residue names, and sequential atom and residue numbering. Using these two files, run the following:

```
${SILCSBIODIR}/ssfep/1_setup_ssfep prot=<Protein PDB> lig=<Ligand Mol2/
↪SDF>
```

Warning: The setup program internally use the GROMACS utility `pdb2gmx`, which may have problems processing the protein PDB file. The most common `pdb2gmx` issue involves mismatches between the expected residue name/atom names in the input PDB and those defined in the CHARMM force-field.

To fix this problem: Run the `pdb2gmx` command manually from within the `1_setup` directory for a detailed error message. Please contact support@silcsbio.com for additional assistance.

Following completion of the setup, run 10 MD jobs:

```
${SILCSBIODIR}/ssfep/2_run_md_ssfep prot=<Protein PDB> lig=<Ligand␣
↪Mol2/SDF>
```

This command will submit 10 jobs to the pre-defined queue: 5 for the ligand in water and 5 for the ligand complexed with protein.

Once the precomputation simulations are completed, the `2_run_md/1_lig/[1-5]` and `2_run_md/2_prot_lig/[1-5]` directories will contain `*.1-10.whole.trr` trajectory files. If these files are not generated, then your simulations are either still running or have stopped due to a problem. Look into the log files within these directories for an explanation of the failure.

15.4 Ligand modifications

Follow the instructions in *Chemical group transformations* to create modifications to your parent ligand.

15.5 Evaluating binding affinity changes

Once `modifications.txt` has been prepared and the MD simulations involving the parent ligand are completed, run the following script to set up a $\Delta\Delta G$ calculation.

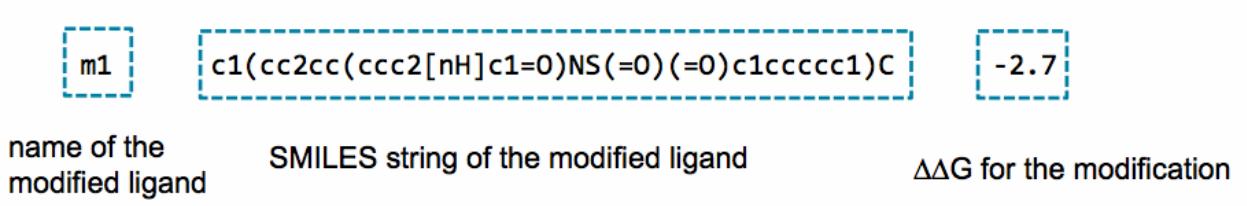
```
${SILCSBIODIR}/ssfep/3a_setup_modifications prot=<Protein PDB> lig=
↪<Ligand Mol2/SDF File> mod=modifications.txt
```

This will submit 10 jobs to evaluate all snapshots from the completed MD simulations of the parent ligand in order to calculate the change in free energy for every modification specified in your `modifications.txt`. Structures of these modifications in mol2 format are output as `3_analysis_<modified ligand name entry in modifications.txt>/mod_files/*.mol2`.

After these jobs complete, you may obtain $\Delta\Delta G$ for your full list of modifications using:

```
${SILCSBIODIR}/ssfep/3b_calc_ddG_ssfep mod=modifications.txt
```

Example output follows:



CHAPTER

SIXTEEN

CHEMICAL GROUP TRANSFORMATIONS

SILCS-MC ligand optimization and SSFEP both entail computations on derivatives of a parent ligand. Additionally, the user may wish to perform SILCS docking using a series of structures chemically related to a reference compound (i.e. parent ligand). Therefore, a task common to these use cases is that of performing chemical group transformations on a parent ligand to create a series of chemically-related compounds.

The SilcsBio GUI provides an intuitive way to handle this task directly in the context of SILCS-MC ligand optimization (see *Running SILCS-MC ligand optimization from the SilcsBio GUI*) and SSFEP (see *SSFEP simulation from the GUI*). You can also use the SilcsBio GUI to perform chemical transformations independently of any other task by choosing *Modify ligand for SILCS-MC/SSFEP* from the Home page. Output Mol2 files will be saved to a directory of your choosing on the machine where you are running the SilcsBio GUI.

It is also possible to perform chemical group transformations using the command line interface, as described below.

You will need to create a text file `modifications.txt` with instructions listing the desired modifications to your parent ligand.

Three types of modifications can be performed, as listed in the table.

Command	Modification
JOIN	Join a fragment (mol2) to the parent at a defined site
REPL	Replace an atom at a defined site on the parent with a fragment (mol2)
MUDE	Change an atom at a defined site on the parent to another atom-type

Table 16.1: Available modification types.

An example ligand modification input file is povided with your SilcsBio server software, `${SILCSBIODIR}/examples/ssfep/modification.txt`.

For details on how to modify and process this file to create Mol2 files containing modifications to your parent ligand, run the following command:

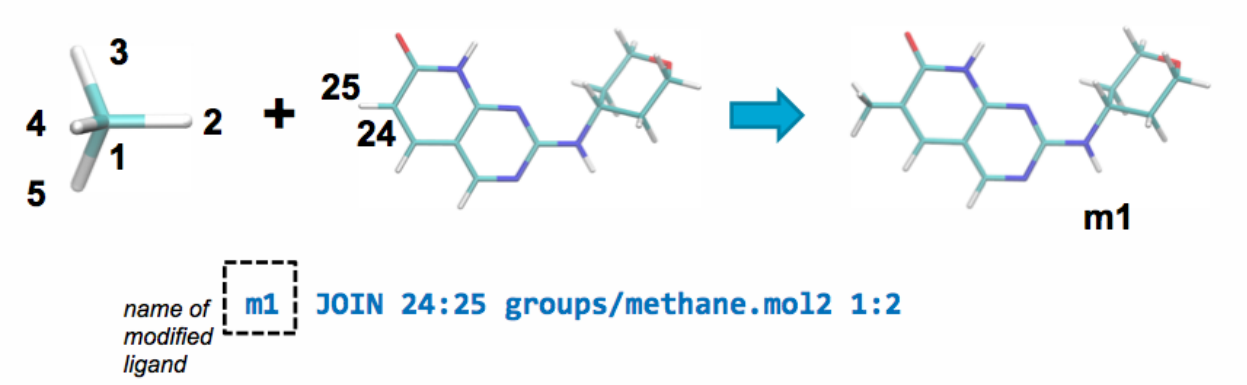
```
${SILCSBIODIR}/programs/chemmod -h
```

Your SilcsBio server software comes with a large number of fragments in Mol2 format that can be added/joined to your parent ligand. These fragments are found in:

```
${SILCSBIODIR}/data/groups
```

16.1 Modifications JOIN example

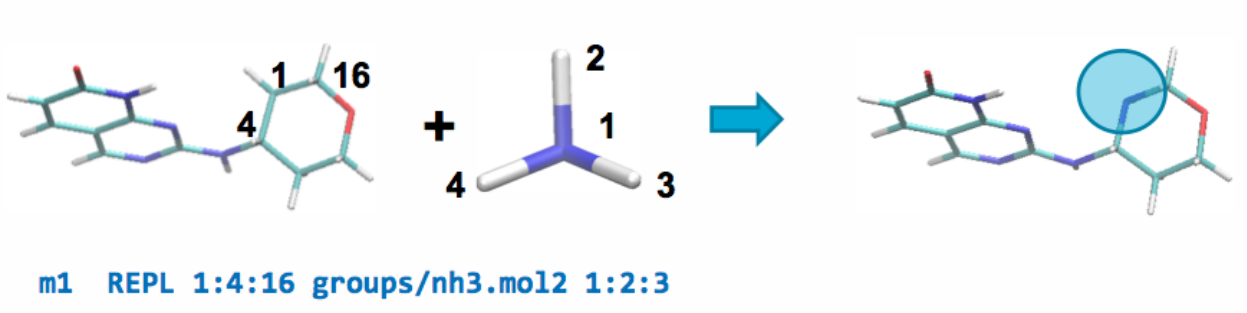
A **JOIN** operation can be performed between the parent and a `ch4.mol2` fragment by adding the following line to `modifications.txt`:



This line will join atom 24 in the parent ligand with atom 1 in `methane.mol2` and delete atoms 25 and 2 in the parent ligand and the joined fragment, respectively.

16.2 Modifications REPL example

A **REPL** operation can be performed between the parent and a `ch4.mol2` fragment by adding the following line to `modifications.txt`:



This line will replace atom 1 of the parent ligand with atom 1 of the fragment by aligning atoms 2

& 3 of the fragment with atoms 4 & 16 of the parent, respectively, and replacing the carbon in the ring with a nitrogen atom.

16.3 Modifications MUDE example

The same transformation in the previous section can also be achieved using a **MUDE** operation:

```
m2  MUDE MU 1:7 DE 21
```

This line will mutate atom 1 (atom index number) in the parent with nitrogen (atom index number 7) along with deleting the hydrogen (atom index number 21) attached to the parent carbon.

16.4 Ligand decoration

To evaluate multiple modifications to a single site on the parent ligand, use the following syntax:

```
m1  JOIN 24:25 <filename>
```

Replace `<filename>` with the name of a text file containing each of the multiple modifications, with one modifications per line. Examples of such text files are `list.txt` and `small_join_list.txt` located in the `${SILCSBIODIR}/data/` folder, and you can use these example files as the basis for your own custom file that includes modifications using Mol2 files from `${SILCSBIODIR}/data/groups` or your own custom Mol2 files. Be careful to pay attention to chemistry; if a modification in your text file is not suitable for a site, you can comment it out using `!` at the beginning of that line. Based on the above `JOIN` line, Mol2 output files from running `${SILCSBIODIR}/programs/chemmod` will all begin with the prefix `m1_`.

CHAPTER

SEVENTEEN

ATOM SELECTION IN THE SILCSBIO GUI

The SilcsBio GUI uses the same atom selection syntax as the NGL molecular structure viewer. This syntax is described at https://nglviewer.org/ngl/api/manual/selection-language.html and is reproduced below for convenience.

17.1 Keywords

- all, *
- sidechain
- sidechainAttached (not backbone or .CA or (PRO and .N))
- backbone
- protein
- nucleic
- rna
- dna
- hetero
- ligand ((not polymer or hetero) and not (water or ion))
- ion
- saccharide/sugar
- polymer
- water
- hydrogen
- helix
- sheet

- turn (not helix and not sheet)
- small (Gly or Ala or Ser)
- nucleophilic (Ser or Thr or Cys)
- hydrophobic (Ala or Val or Leu or Ile or Met or Pro or Phe or Trp)
- aromatic (Phe or Tyr or Trp or His)
- amid (Asn or Gln)
- acidic (Asp or Glu)
- basic (His or Lys or Arg)
- charged (Asp or Glu or His or Lys or Arg)
- polar (Asp or Cys or Gly or Glu or His or Lys or Arg or Asn or Gln or Ser or Thr or Tyr)
- nonpolar (Ala or Ile or Leu or Met or Phe or Pro or Val or Trp)
- cyclic (His or Phe or Pro or Trp or Tyr)
- aliphatic (Ala or Gly or Ile or Leu or Val)
- bonded (all atoms with at least one bond)
- ring (all atoms within rings)

17.2 Expressions

- residue number: 1, 2, 100
- residue number range: 3-40 (Note that around the dash - no spaces are allowed)
- chain name: :A
- atom name: .CA, .C, .N, . . .
- model: /0, /1, . . .
- residue name: ALA, GLU, SOL, DMPC, . . .
- numeric residue name: [032], [1AB], . . .
- list of residue names: [ALA,GLU,MET], [ARG,LYS], . . .
- element name: _H, _C, _O, . . .
- alternate location: %A, %B, . . . or % for non-alternate location atoms
- insertion code: ^A, ^B, . . . or ^ for residues with no insertion code

Some of these expressions can be combined (in this order) - residue number (range), insertion code, chain name, atom name, alternate location, model - like this

```
// select C-alpha atoms of residue 10 with insertion code A
// from chain F in model 0 at alternate location C
10^A:F.CA%C/0
```

which is the same as

```
10 and ^A and :F and .CA and %C and /0
```

Single expressions may be left out as long as the order (see above) is kept, for example:

```
:A/0 # select chain A from model 0
```

17.3 Atomindex

A list of atom indices can be given as a comma seperated list (no spaces in between) prefixed with the @ character.

```
@0,1,4,5,11,23,42
```

17.4 Logical operators (in order of binding strength)

- NOT
- AND
- OR

Additionally, parentheses () can be used for grouping:

```
:A and ( 1 or 10 or 100 ) # select residues 1, 10 and 100␣
↪from chain A
```

CHAPTER

EIGHTEEN

CGENFF: CHARMM GENERAL FORCE FIELD

18.1 Background

The CHARMM General Force Field (CGenFF) covers a wide range of chemical groups present in biomolecules and drug-like molecules, including a large number of heterocyclic scaffolds [3]. The parametrization philosophy behind the force field focuses on quality at the expense of transferability, with the implementation concentrating on an extensible force field.

CGenFF uses the CHARMM additive potential energy function to calculate the energy as a function of the Cartesian coordinates of the system, as shown below.

$$\mathrm{Intramolecular(internal, bonded terms)}$$

$$\sum_{\text{bonds}} K_b(b-b_0)^2 + \sum_{\text{angles}} K_\theta(\theta-\theta_0)^2 + \sum_{\text{dihedrals}} K_\phi(1+\cos(n\phi-\delta))$$

$$+ \sum_{\text{improper}} K_\psi(\psi-\psi_0)^2 + \sum_{\text{Urey}-\text{Bradley}} K_{\mathrm{UB}}(r_{1,3}-r_{1,3;0})^2$$

$$\mathrm{Intermolecular(external, nonbonded terms)}$$

$$\sum_{\text{nonbonded}} \frac{q_i q_j}{4\pi D r_{ij}} + \epsilon_{ij}\left[\left(\frac{R_{\min,ij}}{r_{ij}}\right)^{12} - 2\left(\frac{R_{\min,ij}}{r_{ij}}\right)^{6}\right]$$

The bonded or intramolecular part of the potential energy function consists of terms for the bonds, valence angles, torsion or dihedral angles, improper dihedral angles, and a Urey-Bradley term, where b_0, θ_0, ϕ_0, and $r_{1,3;0}$, respectively, are the bond, angle, improper and Urey-Bradley equilibrium values, the K's are the corresponding force constants, and n and δ are the dihedral multiplicity and phase. The nonbonded or intermolecular portion consists of an electrostatic term, with q_i and q_j being the respective partial atomic charges on atoms i and j, and a van der Waals (vdW) term, which is treated by the Lennard-Jones (LJ) 6-12 potential in which ϵ_{ij} is the well depth, $R_{\min,ij}$ is the radius, and r_{ij} is the distance between i and j.

It is apparent that a simulation on any system of practical interest requires large numbers of parameters. To make the assignment of these parameters practical, force fields require atom types to be assigned to all the atoms in the system, with the parameters associated with combinations of atom types. For instance, the parameter list will contain K_ϕ, n, and δ values for the dihedral parameters

associated with all combinations of four atom types that occur in the molecules supported by the force field. Thus, the first step of assigning parameters for a chemical system is assigning atom types to that system.

CGenFF comes with an algorithm that can automatically assign atom types to a given molecule. The atom typer is rule-based and programmable, making it easy to implement complex atom typing rules and to update the atom typing scheme as the force field grows. Assignment of bonded parameters is based on substituting atom types in the definition of the desired parameter. A penalty is associated with every substitution and the existing parameter with the lowest total penalty is chosen as an approximation for the desired parameter. Charges are assigned using an extended bond-charge increment scheme that is able to capture inductive and mesomeric effects across up to 3 bonds. More details can be found in [4].

18.2 Usage

The CGenFF program accepts Mol2 format files. For correct atom typing, it is important that all hydrogens are present, the system has correct protonation and tautomeric states, and that the bond orders are correct.

Once the Mol2 file is ready, the CGenFF program can be executed:

```
${SILCSBIODIR}/programs/cgenff <mol2file>
```

This produces a stream file in standard output. The output can be redirected to a `.str` file. The stream file consists of a topology file and the extra parameter file combined in a single outpu. It is up to the user to whether to use the stream file directly or to split the file into two separate files.

As an example, the adrenalin molecule is parametrized in CGenFF. The example input and output files can be found in `${SILCSBIO}/examples/cgenff/adrenalin.mol2` and `${SILCSBIO}/examples/cgenff/adrenalin.str`, respectively.

The output stream file has two major sections.

```
* Toppar stream file generated by
* CHARMM General Force Field (CGenFF) program version 1.0.0
* For use with CGenFF version 3.0.1
*

read rtf card append
... topology section ...

END

read param card flex append
... parameter section ...
```

(continues on next page)

(continued from previous page)

```
END
RETURN
```

In CHARMM, asterisk and exclamation marks represents comments. If you wish to separate topology and parameter sections, you can take out the corresponding section after `read rtf` or `read param` to the `END` statement. Let's take a look at topology section:

```
RESI ADRN           0.000 ! param penalty= 113.000 ; charge penalty= ␣
↪16.607
GROUP             ! CHARGE   CH_PENALTY
ATOM O1     OG311  -0.649 !   11.757
ATOM O2     OG311  -0.530 !    0.000
ATOM O3     OG311  -0.530 !    0.000
ATOM N      NG311  -0.524 !   16.607
ATOM C1     CG311   0.127 !   16.171
```

Above is the beginning of the topology section. This section defines the atom names and types, as well as the atomic charges. Each entry is followed by a charge penalty score. The atomic partial charges are assigned based on chemical analogy to existing parametrizations in the force field. The penalty score becomes high when a good analogy is not found.

Below is an excerpt from the parameter section. When an exact parameter is not found in an existing parameter database, an entry is added based on an analogous parameter. Similar to atomic partial charges, each entry in the parameter section is also followed by a penalty score.

```
ANGLES
CG2R61 CG311  OG311    75.70    110.10 ! ADRN , from CG2R61 CG321␣
↪OG311, penalty= 4
CG311  CG321  NG311    43.70    112.20 ! ADRN , from CG331 CG321 NG311,
↪ penalty= 1.5
CG321  NG311  CG331    40.50    112.20    5.00   2.42170 ! ADRN , from␣
↪CG3AM1 NG311 CG3AM1, penalty= 35
```

High penalty scores indicate that a parameter may be targeted for further optimization. Typically, penalty scores greater than 50 warrant further optimization.

It is possible to request that the CGenFF program preserve atomic partial charges from the input Mol2 file instead of assigning these values. This is done using the `-z` flag:

```
${SILCSBIODIR}/programs/cgenff -z <mol2file>
```

Warning: We STRONGLY advise against using your own custom atomic partial charges. Nonbonded interactions in the force field depend upon both the Coulomb and the Lennard-Jones terms, and the parameter values for these terms have been carefully developed together.

Assigning custom atomic partial charges with the `-z` flag is likely to disrupt the balance in these nonbonded force field terms.

18.3 GROMACS-readable parameters

To get parameters in GROMACS format, execute the following command.

```
${SILCSBIODIR}/cgenff/cgenff_to_gmx mol=<mol2file>
```

This command produces three files `<mol2file>_gmx.top`, `<mol2file>_gmx.pdb` & `<mol2file>_charmm.str`. `<mol2file>_gmx.top/pdb` contain the GROMACS readable parameters and the coordinate information. `<mol2file>_charmm.str` contain CHARMM readable parameters.

18.4 CGenFF Parameter Optimizer

The CGenFF program performs atom typing and assignment of parameters and charges by analogy in a fully-automated fashion. Atom typing is done by a deterministic programmable decision tree. Assignment of bonded parameters is based on substituting atom types in the definition of the desired parameter. A penalty is associated with every substitution and the existing parameter in the current version of the force field with lowest total penalty is chosen as an approximation for the desired parameter. The penalty score is returned to the user as an indication of the approximation needed to obtain the desired parameter. For example, dihedral parameters with higher penalties (typically >50) may be candidates for further optimization. When using the Optimizer, the atom typing and parameter assignment is performed internally such that the user only needs to supply the Mol2 file of the molecule of interest.

The threshold for selecting parameters for optimization is based on multiple criteria, including user preference. The penalty score is determined by the analogy of the atom types associated with the new parameter to those of parameters already in the CGenFF force field. Parameters with high penalties may actually be of satisfactory accuracy while parameters with low penalties may be of poor accuracy. Accordingly, the user should make an initial evaluation of the treatment of a given molecule by CGenFF and then identify the parameters that may contribute to possible inaccuracies in the conformational properties of the molecule or its interactions with the environment. Dihedral parameters will typically have the largest impact on the conformational properties of a molecule and, therefore, should be considered for optimization.

CGenFF Parameter Optimizer allows for automatic optimization of rotatable dihedrals. Once the user specifies the dihedral to be optimized, the Optimizer coordinates the generation of quantum mechanical (QM) target data followed by the fitting of force field parameters to these target data. The Optimizer will first spawn Psi4 QM jobs. It will then collate the resulting QM dihedral scan

data. Finally, it will fit force field parameters to these data using the least-square fitting procedure LSFitPar and output the new, optimized force field parameters. If multiple dihedrals are to be optimized, then each dihedral is fitted separately targeting independent QM scans on each dihedral. During the fitting, the initial multiplicities are those assigned by the CGenFF program. Once the initial fit is complete and the RMS error determined, the program automatically refits the data using a multiplicity of 1, then multiplicities of 1 and 2, 1, 2 and 3 and 1, 2, 3 and 6. If improvements in the RMSE are obtained beyond a cutoff (default is 10% of the RMSE) with the additional multiplicities, then those parameters are selected.

The Optimizer accepts molecules in Mol2 file format, with the following requirements: hydrogens must be explicitly defined, ionizable groups must have correct protonation states, and bond-order between atoms must be correctly defined.

To run the program, set the following environment variables,

```
export SILCSBIODIR=<silcsbio>
export PSI4DIR=<psi4>
export GMXDIR=<gromacs/bin>
```

The Psi4 package can be obtained from the Psi4 download page. For the easiest installation, use the Psi4 Binary Installer, run the command:

```
./Psi4conda-latest-py36-Linux-x86_64.sh
```

and following the step-by-step guide.

After Psi4 has been installed, set the `PSI4DIR` variable correctly and proceed with using the optimizer. To make sure `PSI4DIR` is correctly set, check if `$PSI4DIR/bin/psi4` is accessible.

Note that downloading the Psi4 source code and compiling the program locally with the appropriate switches can lead to a significant gain in performance over the downloaded binary version.

18.4.1 Usage

```
${SILCSBIODIR}/cgenff-optimizer/optimize_cgenff_par mol=<mol2file>
```

The second line of the Mol2 file contains a string naming the molecule. Certain molecular modeling programs will put the string "*" on this line instead of a descriptive name for the molecule. In such cases, please replace the string with an alphanumeric string, else the Optimizer will fail.

Along with the required argument of `mol=<mol2file>`, the following additional parameters can also be set:

1. Penalty score :

```
penalty=<score; default 50>
```

By default, the program identifies rotatable dihedrals with penalties greater than 50 for optimization. This can be changed by adjusting the `penalty` argument, to increase/decrease the list of dihedrals that will be optimized.

2. Dihedral step size :

```
dihedral_step_size=<step size; default 15>
```

Each dihedral identified for optimization is rotated through 360 degrees with a step-size of 15 degrees to generate QM target data. This step-size can be changed by adjusting the `dihedral_step_size` argument to increase or decrease the resolution of the dihedral scan.

3. Number of cores used for QM optimization:

```
nproc=<number of core; default all available cores on␣
↪workstation/8 cores on HPC>
```

At each scan point for each dihedral, a constrained QM optimization is performed. Psi4 can make use of multiple CPU cores to run this optimization more quickly. When the Optimizer is installed on a high-performance computing cluster (HPC), separate jobs are submitted for each of the identified dihedrals simultaneously, and each job requests `nproc` cores. When the Optimizer is installed on a workstation, then each scan point for each dihedral is run one after the other using all available cores on the workstation. This default behavior can be changed using the `nproc` argument.

4. Molecular-orbital theory:

```
mo_theory=<MP2/HF; default MP2>
```

Constrained QM optimization for each dihedral at each scan point is performed using Møller–Plesset perturbation theory (MP2, specifically the Psi4 DF-MP2/6-31G(d) model chemistry). If the user wants to instead use Hartree-Fock (HF), then it can be set using `mo_theory`.

5. Energy difference cutoff to eliminate rotamers from QM optimization and subsequent fitting:

```
dg_cutoff=<energy_difference; default 100.>
```

The optimizer initially generates all the rotamers for each dihedral being optimized as required for the potential energy surface (PES) and performs a CGenFF energy optimization on each restrained rotamer. If the relative energy of each rotameter is greater than the `dg_cutoff` value (kcal/mo), that rotatamer is omitted from the QM optimization and subsequent parameter fitting. This is performed to eliminate rotamers with steric clashes that can be problematic for the QM optimization as well as lead to the parameter optimizer attempting to fit to high energy regions of the PES that are typically not sampled in MD simulations.

6. Specification of compute node for job submission:

```
hpc=<true/false; default true>
```

Installation of the CGenFF optimizer specifies the computer on which the jobs will be run, which is typically a local HPC cluster such that the default is `hpc=true`. However, if the user wants to run the job on their local workstation then `hpc=false` may be specified. Alternatively, if the default is false, `hpc=true` can be specified to run the job on the HPC cluster. This requires that the appropriate cluster be specified in the default installation.

7. Running jobs sequentially or in parallel when `hpc=true`.

```
jobtype=<seq/par; default seq>
```

When the QM jobs are submitted to an HPC queue they will run sequentially using the specified number of processors (`nproc`). However, if adequate compute resources are available it is possible to have all the QM jobs run simultaneously in parallel by specifying `jobtype=par` with each job using the specified number of processors (`nproc`). Care should be taken when using this option as the number of individual QM jobs can be quite large (n x 24 jobs where n is the number of dihedral parameters being optimized). However, use of the option can lead to the QM jobs finishing rapidly.

8. Setting the RMS energy difference to not redo the least-squares fitting using different multiplicities.

```
rmse_cutoff=<RMS energy cutoff to determine if additional␣
↪multiplicities should be tested; default=0.0>
```

The initial least-squares fit applies the multiplicities in the original CGenFF parameters assigned to the dihedral being optimized. From the fit the RMS difference between the target QM and optimized MM parameters is calculated. If that values is greater than `rmse_cutoff` then least-squares fitting is performed with additional multiplicities. rmse_cutoff is set to zero by default to force the additional multiplicities to be included in the least-squares fitting. Note that the fitting may be repeated using the calculated QM data as described in the "Practical Considerations" section below.

18.4.2 Example Usecase

The following demonstrates usage of the CGenFF Parameter Optimizer. The input file `ethene_co_isoxazole.mol2` is available in `${SILCSBIODIR}/examples/cgenff/`. When running the Optimizer it is suggested that a subdirectory be created for each molecule and the job run from within that subdirectory.

```
${SILCSBIODIR}/cgenff-optimizer/optimize_cgenff_par mol=ethene_co_
↪isoxaole.mol2
```

The resulting output is a CHARMM-compatible CGenFF stream file named `ethene_co_isoxazole_optimized.str`. Since this molecule does not contain any dihedral parameter with penalty greater than 50, running the above command results in:

```
Nothing to optimize based on the penalty score threshold of  50
```

The user can look through the output stream file to identify dihedral parameters that may benefit from optimization:

```
DIHEDRALS
CG2DC3 CG2DC1 CG2O5  CG2R51     1.4000  2   180.00 ! NONAME.* , from␣
 ↪CG2DC3 CG2DC1 CG2O5 OG2D3, penalty= 26.5
HGA4   CG2DC1 CG2O5  CG2R51     0.0000  2   180.00 ! NONAME.* , from␣
 ↪HGA4 CG2DC1 CG2O5 OG2D3, penalty= 26.5
CG2DC1 CG2O5  CG2R51 CG2R51     1.5850  2   180.00 ! NONAME.* , from␣
 ↪CG2O3 CG2O5 CG2R61 CG2R61, penalty= 46.5
CG2DC1 CG2O5  CG2R51 OG2R50     1.5850  2   180.00 ! NONAME.* , from␣
 ↪CG2O3 CG2O5 CG2R61 CG2R61, penalty= 49
OG2D3  CG2O5  CG2R51 CG2R51     1.5850  2   180.00 ! NONAME.* , from␣
 ↪OG2D3 CG2O5 CG2R61 CG2R61, penalty= 33.5
OG2D3  CG2O5  CG2R51 OG2R50     1.5850  2   180.00 ! NONAME.* , from␣
 ↪OG2D3 CG2O5 CG2R61 CG2R61, penalty= 13
CG2O5  CG2R51 CG2R51 CG2R52     0.0000  2   180.00 ! NONAME.* , from␣
 ↪CG2O1 CG2R51 CG2R51 CG2R52, penalty= 3
CG2O5  CG2R51 CG2R51 HGR51      1.0000  2   180.00 ! NONAME.* , from␣
 ↪CG2R51 CG2R51 CG2R51 HGR51, penalty= 16.5
CG2O5  CG2R51 OG2R50 NG2R50     8.5000  2   180.00 ! NONAME.* , from␣
 ↪CG2R51 CG2R51 OG2R50 NG2R50, penalty= 16.5
```

If the user now wants to fit `CG2DC1 CG2O5 CG2R51 OG2R50`, which has a penalty score of 49, the program can be re-run with a lowered penalty threshold of 48:

```
${SILCSBIODIR}/cgenff-optimizer/optimize_cgenff_par mol=ethene_co_
 ↪isoxazole.mol2 penalty=48
```

This will result in:

```
###############################################
for dihedral = CG2DC1 CG2O5 CG2R51 OG2R50, parameter penalty = 176.
 ↪000000;
###############################################

Will be optimizing the parameters of the above listed dihedral(s)
```

Note that all parameters associated with dihedrals about a rotatable bond with a penalty score exceeding the penalty threshold will be optimized. Dihedrals that are considered rigid based on being located in rings, being double bonds and so on, are excluded from fitting. This selection

process can lead to a situation in which two or more sets of dihedral parameters associated with the same rotatable bond are being fit independently. While each of those fits may appear to be successful, when those parameters are combined and applied to the molecules of interest, the conformational energy for rotation about that bond will likely be inaccurate. Such a situation should be avoided by setting the penalty tolerance to be higher that the penalty values for those parameters and one of the dihedrals selected for optimization as described in the following section.

If the user wants to fit the parameters for one or more specific dihedrals if none are selected based on the penalty score, or in addition to the automatically selected dihedrals, the user can manually supply the four atoms defined each specific dihedral on prompt:

```
${SILCSBIODIR}/cgenff=optimizer/optimize_cgenff_par mol=ethene_co_
↪isoxazole.mol2 penalty=50
```

will result in :

```
CHARMM General Force Field (CGenFF) program version 2.1.0
released the 27th of October 2016
Copyright (C) 2017 SilcBio LLC
and University of Maryland, School of Pharmacy. All Rights Reserved.

Now processing molecule sulf ...

##############################################
##############################################

Nothing to optimize based on the penalty score threshold of  50

Do you want to perform optimization with anymore dihedrals? [Y/N;␣
↪default N]
```

To add to the list, type `Y`, and then supply dihedral(s) with the format `AtomType1 AtomType2 AtomType3 AtomType4`. The program then summarizes the updated-list:

```
List updated; will be attempting optimization with the following␣
↪dihedrals

##########################################
CG2DC1 CG2O5 CG2R51 OG2R50
##########################################
```

The last column of the PDB file `<mol>_cgenff_atomtypes.pdb` created by the Optimizer identifies the atom type for each atom in the system which may be inspected to select specific parameters for optimization. For the current example, from the information in `ethene_co_isoxazole_cgenff_atomtypes.pdb`, atom 5 has the atom type `CG2R51`:

```
ATOM      1 C    NONA    1       0.759   0.680   1.188  1.00  0.00
 ↪ CG2R52
ATOM      2 H    NONA    1       1.645   1.222   1.497  1.00  0.00
 ↪  HGR52
ATOM      3 C    NONA    1      -0.353   1.220   0.530  1.00  0.00
 ↪ CG2R51
ATOM      4 H    NONA    1      -0.508   2.244   0.227  1.00  0.00
 ↪  HGR51
ATOM      5 C    NONA    1      -1.205   0.177   0.349  1.00  0.00
 ↪ CG2R51
ATOM      6 C    NONA    1      -2.542   0.163  -0.296  1.00  0.00
 ↪  CG2O5
ATOM      7 C    NONA    1      -3.305  -1.129  -0.378  1.00  0.00
 ↪ CG2DC1
ATOM      8 H    NONA    1      -2.855  -2.041   0.043  1.00  0.00
 ↪   HGA4
ATOM      9 C2   NONA    1      -4.530  -1.255  -0.944  1.00  0.00
 ↪ CG2DC3
ATOM     10 H21  NONA    1      -5.050  -2.226  -0.980  1.00  0.00
 ↪   HGA5
ATOM     11 H22  NONA    1      -5.045  -0.383  -1.387  1.00  0.00
 ↪   HGA5
ATOM     12 O1   NONA    1      -3.016   1.204  -0.756  1.00  0.00
 ↪  OG2D3
ATOM     13 O    NONA    1      -0.636  -0.949   0.875  1.00  0.00
 ↪ OG2R50
ATOM     14 N    NONA    1       0.613  -0.617   1.406  1.00  0.00
 ↪ NG2R50
```

Following the identification of the dihedrals to the fitted, the Optimizer performs a complete molecular mechanics (MM) minimization of the molecule using the default CGenFF force field parameters. Next, each identified dihedral is independently rotated through 360 degrees with the specified step-size. Separate Psi4 jobs for constrained QM optimization around the specified dihedral are then performed either serially, one-by-one when running on a workstation, or in parallel when running on a HPC cluster.

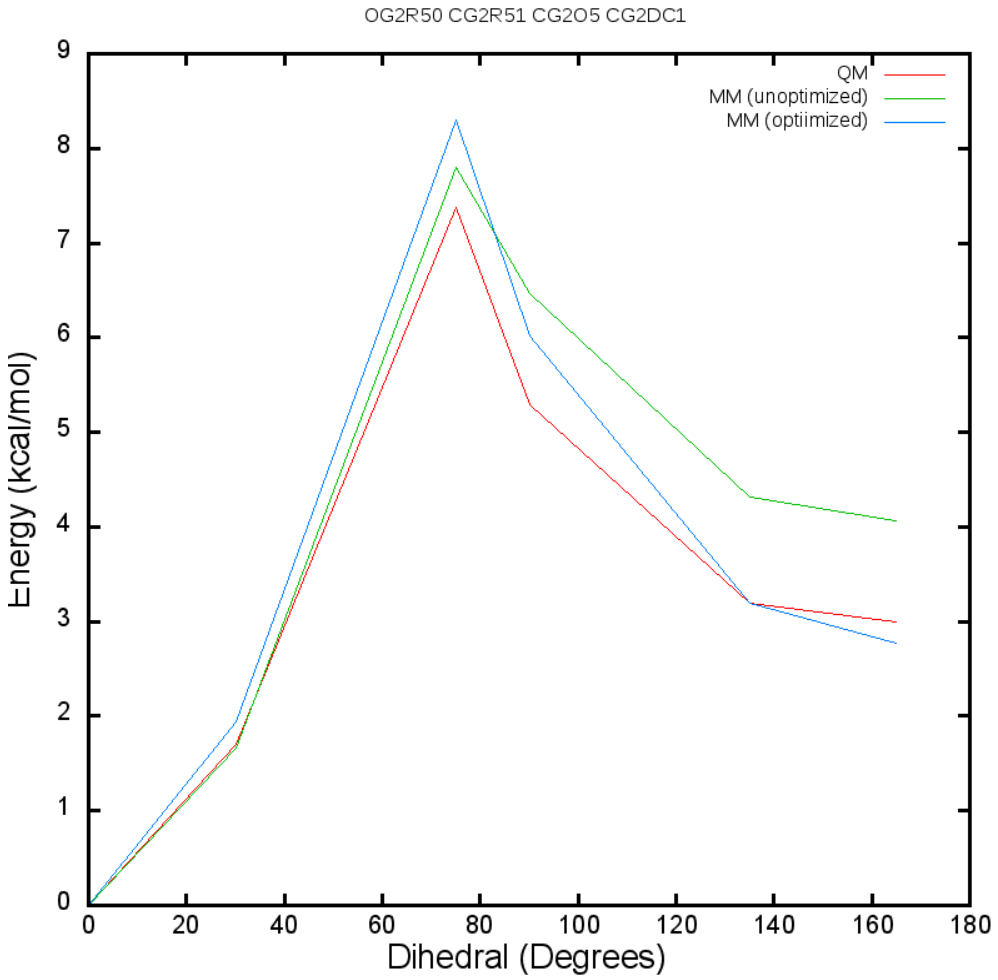
The program waits on the QM jobs to finish. After the completion of the QM jobs, the MM energy is computed for each scan point. Both the QM and MM energies are then used by the lsfitpar program to fit the dihedral parameter. Updated parameters are output as `<mol>_optimized.str`.

Graphs of the dihedral scan energies are output in the files files `qm_mm_*.png`. If the computing system where you are running the Optimizer does not contain gnuplot or gnuplot linked against the png library, you may copy the `4plot` directory to a computing system that does and run the gnuplot files there.

Note that the most robust test of the optimized parameters involves testing them directly in a

molecular modeling package with the new parameters added to the CGenFF parameter file (see below).

The following is a graph produced by the Optimizer for the `ethene_co_isoxazole` example and demonstrates the better agreement with the QM data following dihedral parameter optimization:



The default CGenFF parameter for dihedral `CG2DC1 CG2O5 CG2R51 OG2R50` was

```
CG2DC1 CG2O5  CG2R51 OG2R50     1.5850  2   180.00 ! RMSE = 0.468999
```

and the replacement parameter produced by the Optimizer is

```
CG2DC1 CG2O5  CG2R51 OG2R50     0.5728  2     0.00 ! RMSE = 0.111323
```

18.4.3 Performing dihedrals optimization in the background

The above example runs the optimization on the command line requiring the terminal in which the job is submitted to remain open. To run the optimization in the background, the 'nohup' command may be used as in the following example. Note that a small script would be required to identify specific dihedrals for optimization.

```
nohup $SILCSBIODIR/cgenff-optimizer/optimize_cgenff_par.sh mol=ethene_
↪co_isoxazole.mol2 > cgenff_opt_run1.txt &
```

18.4.4 Including new dihedral parameters with CGenFF

Currently, the optimized parameters generated by the CGenFF optimizer are NOT automatically added to the CGenFF parameter database. Accordingly, the user needs to manually add the parameters to the file using a standard editor (e.g., vi or emacs).

```
$SILCSBIODIR/data/par_all36_cgenff.prm
```

In the editor go the the DIHEDRALS section of the parameter file and then manually add the new parameters into the file. The new parameters are in filename_optimized.str found in the directory in which the optimization was run. "filename" corresponds to the molecule filehame.mol2 used to initiate the optimization. The optimized dihedral are at the end of the DIHEDRALS section of that file and are indicated by the comment containing the RMSE values obtained from the fitting of each dihedral parameter as follows.

```
CG2R51   CG2R51   NG311    CG331        3.8499  2   180.00 ! RMSE = 0.
↪58618
CG2R51   CG2R51   NG311    CG331        1.5145  3   180.00 ! RMSE = 0.
↪58618
CG3C52   CG2R51   NG311    CG331        2.1370  2   180.00 ! RMSE = 0.
↪245345
CG3C52   CG2R51   NG311    CG331        1.4058  3   180.00 ! RMSE = 0.
↪245345
```

Those parameters may be cut and paste directly into par_all36_cgenff.prm. Note that the file is generally protected and system administrator assistance is required to update this file. In the case of replacing dihedral parameters that are already in par_all36_cgenff.prm, the original parameters should be commented with a ! and the new parameters added.

18.4.5 Additional output information

Upon completion of the optimization the data generated is organized and placed in the subdirectory "raw_data" in which there is an additional subdirectory "psi4_calc" that contains files from the QM calculations.

- `optimize_cgenff_par.log`

  Summary of the dihedrals being optimized and the optimization process including the RMSE data and final parameters.

- `dih_params_to_optimize*.txt`

  Dihedral parameters initially selected and updated list targeted for optimization.

- `params*.{inp,out}`

  Inputs and outputs of MM minimizations to calculate the potential energy surfaces. Includes information on highly unfavorable conformations that were not included in the final parameter fitting.

- `params_aft_qm.out`

  Outputs of MM minimizations using the optimized parameters to calculate the potential energy surface.

- `lsqfit_op_0*.{inp,out}`

  Inputs and outputs for the parameter least-squares fitting using the original multiplicity along with fits using alternate multiplicities (`grep "Final RMSE" *.out` to see RMSE for all runs).

- `mm*.prm`

  Final parameters from the fits with the different multiplicities.

- `*.dat`

  Files containing dihedral angles and energies for fitting. Note that the `mm_init_ener_0.dat` is the MM PES with the targeted dihedral parameters set to zero.

18.4.6 Practical considerations

The use of the CGenFF penalty scores to select dihedrals for optimization in conjunction with the selection of only rotatable bonds is designed for an automated approach to dihedral parameter optimization. However, this may often lead to multiple dihedrals being selected for optimization, which is computationally demanding, or no dihedrals being selected based on low or zero penalty scores, where zero indicates that the specific parameter is already in the CGenFF force field and, by default, such dihedrals will not be listed as available for optimization. However, in cases where the user is concerned about the accuracy of a particular rotatable bond even when it is already in the CGenFF parameter set (e.g., a rotatable bond involved in the link between two ring systems that are part of a lead compound), then the user may specifically identify the atom types defining that dihedral parameter(s) and input them individually. While the penalty scores represent a useful metric by which to judge the quality of a dihedral parameter it should be noted that the CGenFF penalty scores are based on analogy to known parameters. Accordingly, a parameter with a high penalty may yield acceptable conformational energies, while a parameter already in CGenFF (i.e.,

penalty = 0) may not yield an acceptable energy surface due to, for example, the terminal rings about the associated bond creating a significantly different context than that in which the dihedral parameter was initially optimized.

An important consideration is molecular size. As the fitting procedure requires QM calculations the size of the molecule under study significantly impacts the speed of the overall fitting procedure. To limit the computational demand while achieving the desired accuracy in the parameters it is suggested that compounds be subdivided into model compounds extracted from that full compound that contain two terminal ring systems along with the linker between those rings. The number of substituents on each ring should be limited to those deemed essential to represent the chemical character of the rings while minimizing the number of substituents to limit computational costs. For example a compound with three ring systems with two linkers would be subdivied into two model compounds with two rings each, with one of the rings common to both model compounds. Once the two individual CGenFF-optimizer runs are finished, the new parameters may be combined and added to your the CGenFF parameter files.

During parameter fitting, the program initially uses the dihedral parameter multiplicities assigned by the CGenFF program. Once the RMSE is returned, the program then does additional fitting with a multiplicity of 1, then multiplicities of 1 and 2, 1, 2 and 3 and 1, 2, 3 and 6. The current implementation outputs the results from all the multiplicities tested. As the lowest RMSE will generally be with multiplicities 1,2,3,6 the user may select a set of parameters with different multiplicities. Those parameters are in the files `raw_data/mm*.prm`. Note that the multiplicities of 4 and 5 are omitted as these are rarely used for rotatable bonds even though, in same cases, the CGenFF program will assign dihedral parameters with these multiplicities and the program will initially include a dihedral with a multiplicity of 4 or 5 in the initial fitting.

If the level of agreement using the final parameters is not satisfactory it suggests hysteresis in the energy surface. To overcome this the user may consider focusing the fitting on an 'appropriate' part of the energy surface by trimming down some data-points on the QM & MM energies to get a better fit to the low energy region of the energy surface. To achieve this consider running `${SILCSBIODIR}/cgenff-optimizer/refit_par_with_subsurface dih_qm_<dihno>.dat dih_mm_<dihno>.dat` (Under development).

Upon completion of the fitting a subdirectory 'raw_data' is created that contains an extensive number of data file with the various QM energies, MM energies and additional information. In addition, the subdirectory 'raw_data/psi4_calc' contains the inputs and outputs from the psi4 QM optimizations. These files may be used for additional fitting or analysis.

In cases where the least-squares fitting is not completed successfully, possibly due to all the QM jobs not finishing, fitting alone may be performed if all the QM data is directly available in subdirectory `psi4_calc` (not `raw_data/psi4_calc`). For example, if a subset of the QM jobs did not finished the job input scripts in `psi4_calc` (e.g., `psi4_inp_xxxx_1_23.inp`) maybe submitted directly to complete those jobs followed by resubmission of the cgenff-optimizer command from the parent directory.

BIBLIOGRAPHY

[1] Wenbo Yu, Sunhwan Jo, Sirish Kaushik Lakkaraju, David J Weber, and Alexander D MacKerell Jr. Exploring protein-protein interactions using the Site-Identification by Ligand Competitive Saturation methodology. *Proteins*, 87(4):289–301, April 2019.

[2] Sunhwan Jo, Amy Xu, Joseph E Curtis, Sandeep Somani, and Alexander D MacKerell Jr. Computational characterization of antibody-excipient interactions for rational excipient selection using the site identification by ligand competitive saturation-biologics approach. *Mol. Pharm.*, 17(11):4323–4333, November 2020.

[3] K Vanommeslaeghe, E Hatcher, C Acharya, S Kundu, S Zhong, J Shim, E Darian, O Guvench, P Lopes, I Vorobyov, and Alexander D MacKerell Jr. CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields. *J. Comput. Chem.*, 31(4):671–690, March 2010.

[4] Kenno Vanommeslaeghe and Alexander D MacKerell Jr. Automation of the CHARMM General Force Field (CGenFF) I: bond perception and atom typing. *J. Chem. Inf. Model.*, 52(12):3144–3154, December 2012.

[5] E Prabhu Raman, Kenno Vanommeslaeghe, and Alexander D MacKerell Jr. Site-specific fragment identification guided by Single-Step Free Energy Perturbation calculations. *J. Chem. Theory Comput.*, 8(10):3513–3525, October 2012.

[6] E Prabhu Raman, Sirish Kaushik Lakkaraju, Rajiah Aldrin Denny, and Alexander D MacKerell Jr. Estimation of relative free energies of binding using pre-computed ensembles based on the Single-Step Free Energy Perturbation and the Site Identification by Ligand Competitive Saturation approaches. *J. Comput. Chem.*, October 2016.

[7] Olgun Guvench and Alexander D MacKerell Jr. Computational fragment-based binding site identification by ligand competitive saturation. *PLoS Comput. Biol.*, 5(7):e1000435–12, July 2009.

[8] E Prabhu Raman, Wenbo Yu, Olgun Guvench, and Alexander D MacKerell Jr. Reproducing crystal binding modes of ligand functional groups using Site Identification by Ligand Competitive Saturation (SILCS) simulations. *J. Chem. Inf. Model.*, 51(4):877–896, April 2011.

[9] E Prabhu Raman, Wenbo Yu, Sirish K Lakkaraju, and Alexander D MacKerell Jr. Inclusion of multiple fragment types in the Site Identification by Ligand Competitive Saturation (SILCS) approach. *J. Chem. Inf. Model.*, 53(12):3384–3398, December 2013.

[10] Sirish Kaushik Lakkaraju, E Prabhu Raman, Wenbo Yu, and Alexander D MacKerell Jr. Sampling of organic solutes in aqueous and heterogeneous environments using oscillating excess chemical potentials in grand canonical-like Monte Carlo-molecular dynamics simulations. *J. Chem. Theory Comput.*, 10(6):2281–2290, June 2014.

[11] Wenbo Yu, Sirish Kaushik Lakkaraju, E Prabhu Raman, Lei Fang, and Alexander D MacKerell Jr. Pharmacophore modeling using Site Identification by Ligand Competitive Saturation (SILCS) with multiple probe molecules. *J. Chem. Inf. Model.*, 55(2):407–420, February 2015.

[12] Sirish Kaushik Lakkaraju, Wenbo Yu, E Prabhu Raman, Alena V Hershfeld, Lei Fang, Deepak A Deshpande, and Alexander D MacKerell Jr. Mapping functional group free energy patterns at protein occluded sites: nuclear receptors and G-protein coupled receptors. *J. Chem. Inf. Model.*, 55(3):700–708, March 2015.

[13] Alexander D MacKerell Jr, Sunhwan Jo, Sirish Kaushik Lakkaraju, Christoffer Lind, and Wenbo Yu. Identification and characterization of fragment binding sites for allosteric ligand design using the Site Identification by Ligand Competitive Saturation Hotspots approach (SILCS-Hotspots). *Biochim. Biophys. Acta. Gen. Subj.*, 1864(4):129519–129519, April 2020.

[14] Abhishek A Kognole, Anthony Hazel, and Alexander D MacKerell Jr. SILCS-RNA: Towards a structure-based drug design approach for targeting RNAs with small molecules. *J. Chem. Theory Comput.*, ():, July 2022.

[15] You Xu, Kenno Vanommeslaeghe, Alexey Aleksandrov, Alexander D MacKerell Jr, and Lennart Nilsson. Additive CHARMM force field for naturally occurring modified ribonucleotides. *J. Comput. Chem.*, 37(10):896–912, April 2016.

[16] Samo Lesnik, Milan Hodoscek, Urban Bren, Christoph Stein, and Ana-Nicoleta Bondar. Potential energy function for fentanyl-based opiod pain killers. *J. Chem. Inf. Model*, 60(7):3566–3576, July 2020.

[17] Oskar Klaja, James A Frank, and Ana-Nicoleta Bondar. Potential energy function for a photoswitchable lipid molecule. *J. Comput. Chem.*, 41(27):2336–2351, October 2020.

[18] Anastasia Croitoru, Sang-Jun Park, Anmol Kumar, Jumin Lee, Wonpil Im, Alexander D MacKerell Jr, and Alexey Aleksandrov. Additive CHARMM36 force field for nonstandard amino acids. *J. Chem. Theory Comput.*, 17(6):3554–3570, June 2021.

[19] Yang Zhang and Jeffrey Skolnick. SPICKER: a clustering approach to identify near-native protein folds. *J. Comput. Chem.*, 25(6):865–871, April 2004.

[20] Marc O'Reilly, Anne Cleasby, Thomas G Davies, Richard J Hall, R Frederick Ludlow, Christopher W Murray, Dominic Tisi, and Harren Jhoti. Crystallographic screening using ultra-low-molecular-weight ligands to guide drug design. *Drug Discov. Today*, 24(5):1081–1086, May 2019.

[21] Richard D Taylor, Malcolm MacCoss, and Alastair D G Lawson. Rings in drugs. *J. Med. Chem.*, 57(14):5845–5859, July 2014.

[22] Vincent D Ustach, Sirish Kaushik Lakkaraju, Sunhwan Jo, Wenbo Yu, Wenjuan Jiang, and Alexander D MacKerell Jr. Optimization and evaluation of Site-Identification by Ligand Competitive Saturation (SILCS) as a tool for target-based ligand optimization. *J. Chem. Inf. Model.*, 59(6):3018–3035, April 2019.

[23] Geoffrey A Heinzl, Weiliang Huang, Wenbo Yu, Bennett J. Giardina, Yue Zhou, Alexander D MacKerell Jr, Angela Wilks, and Fengtian Xue. Iminoguanidines as allosteric inhibitors of the iron-regulated heme oxygenase (HemO) of Pseudomonas aeruginosa. *J. Med. Chem.*, 59(14):6929–6942, June 2016.

[24] Maryanna E Lanning, Wenbo Yu, Jeremy L Yap, Jay Chauhan, Lijia Chen, Ellis Whiting, Lakshmi S Pidugu, Tyler Atkinson, Hala Bailey, Willy Li, Braden M Roth, Lauren Hynicka, Kirsty Chesko, Eric A Toth, Paul Shapiro, Alexander D MacKerell Jr, Paul T Wilder, and Steven Fletcher. Structure-based design of N-substituted 1-hydroxy-4-sulfamoyl-2-naphthoates as selective inhibitors of the Mcl-1 oncoprotein. *Eur. J. Med. Chem.*, 113:273–292, May 2016.