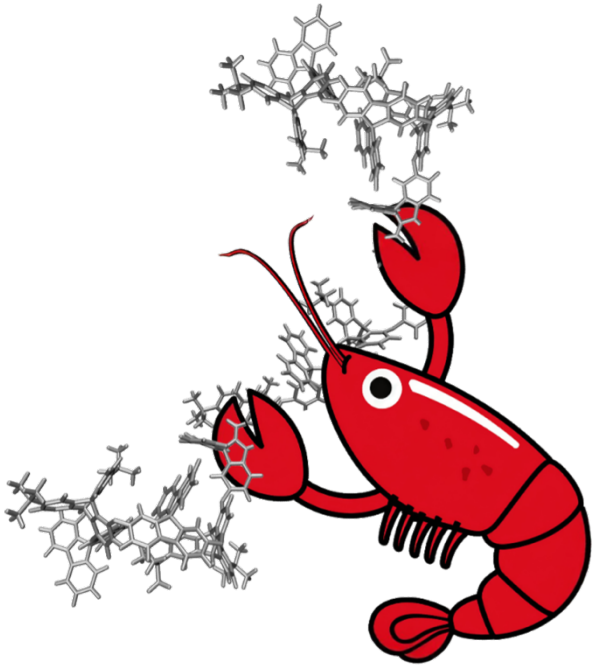
PolyPal

-User Manual-



Molly C. Warndorf, Timothy M. Swager, Alfredo Alexander-Katz
Massachusetts Institute of Technology

Table of Contents

1 Introduction

PolyPal is a Python package designed as a collaborative tool to assist with the creation, parameterization, and running of amorphous polymer simulations. The current version is designed to take quantum mechanically augmented force fields generated by Q-Force, create input files for Assemble!, and then generate files that can be directly submitted to GROMACS. Future versions will expand to different software.

2 Installation

The following software are required for using this workflow:

Python >= 3.7 (http://www.python.org/)
Q-Force >= 0.6.14 (https://qforce.readthedocs.io/en/latest/install.html)
Assemble! >= 1.0 (https://degiacom.github.io/assemble/)
GROMACS >= 4.X. (https://www.gromacs.org/)

Q-Force will require the use of a quantum chemistry program. It is currently compatible with:

Gaussian (https://gaussian.com/)
ORCA (https://www.faccts.de/orca/)
Q-Chem (https://www.q-chem.com/)
XTB (https://xtb-docs.readthedocs.io/en/latest/)

To install PolyPal:

```
pip install polypal
```

```
git clone https://github.com/warndorf/polypal.git
cd polypal
python setup.py install
```

Other recommended software for small molecule building, visualization, and data analysis:

MKTOP (https://github.com/aar2163/MKTOP)
Avogadro (https://avogadro.cc/)
PyMol (https://www.pymol.org/)
VMD (https://www.ks.uiuc.edu/Research/vmd/)
PoreBlazer (https://github.com/SarkisovGitHub/PoreBlazer)

3 Suggested environment setup

It is recommended to install all packages into an environment. The following commands will setup an environment with Q-Force, PolyPal, and GROMACS. Assemble! and your choice of a quantum chemistry software package will have to be installed separately.

```
conda create -n PolyPal
```

```
conda activate PolyPal
```

```
conda install python

pip install qforce

pip install polypal

conda install gromacs
```

4 Useful links

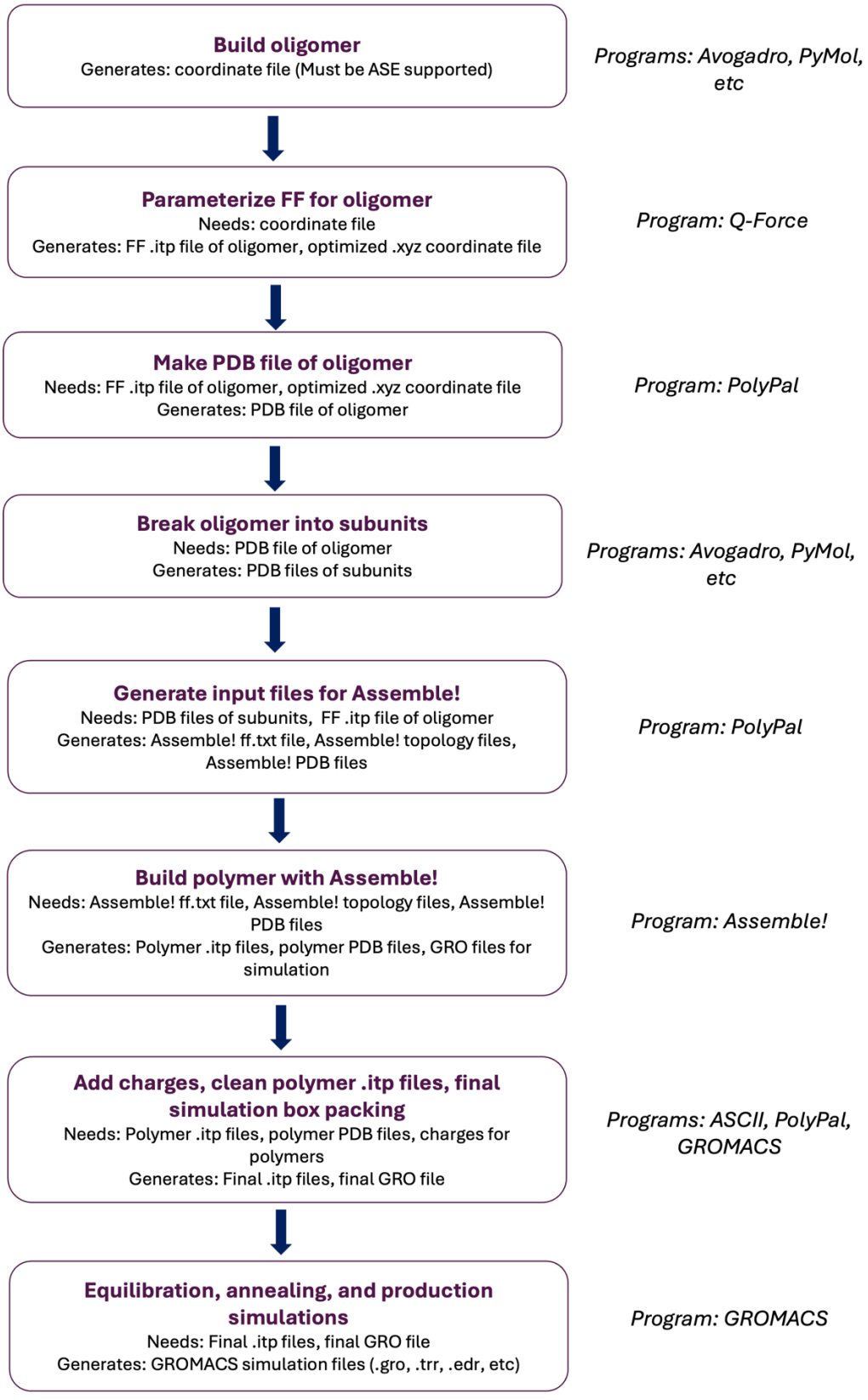
Documentation for Q-Force:

Usage: https://qforce.readthedocs.io/en/latest/usage.html
Options: https://qforce.readthedocs.io/en/latest/options.html

Documentation for Assemble!:

GitHub with manual: https://github.com/degiacom/assemble

5 General Workflow



6 Tutorial

Step 1: Build oligomer

Use a molecular building software such as Avogadro, PyMol, ChemDraw etc to make a coordinate file of an oligomer of the polymer of interest. Coordinate file format must ASE supported such as XYZ, PDB, GRO, etc.

Oligomers should be large enough so force field parameterization can capture important dynamics, such as rotation around monomer units or stereocenters, but not too large where DFT calculation become cost-prohibitive. Depending on the polymer dimers, trimers, and tetramers have been reliably large enough for parametrization.

In the tutorial, Avogadro was used to make a trimer of polyethersulfone (PES), and saved as an PDB file (Figure 1).

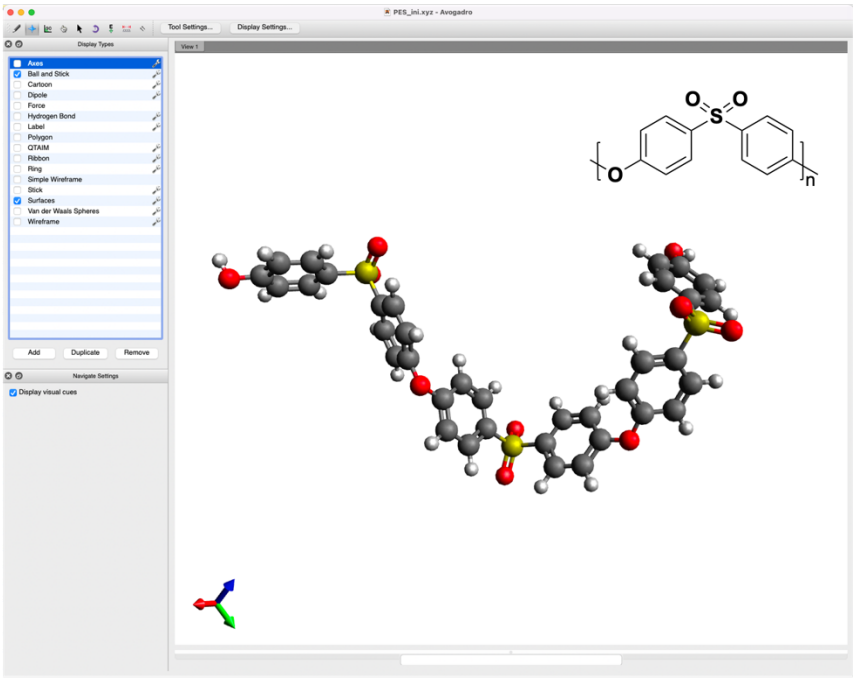


Figure 1. Oligomer of PES (three repeat units)

Step 2: Parameterize FF for oligomer

In this tutorial, ORCA will be used as the quantum chemistry software. A settings file for Q-Force is provided in the Tutorial folder. Some default settings have been changed, but users are encouraged to change settings if different methods are preferred. For more Q-Force setting options please see: https://qforce.readthedocs.io/en/latest/options.html

OPLS Lennard-Jones parameters can be provided to Q-Force in an ext_lj file. OPLS atom-types can were generated with the program MKTOP (Figure 2).

```
perl mktop.pl -i PES.pdb -o PES_mktop.top -ff opls -conect no
```

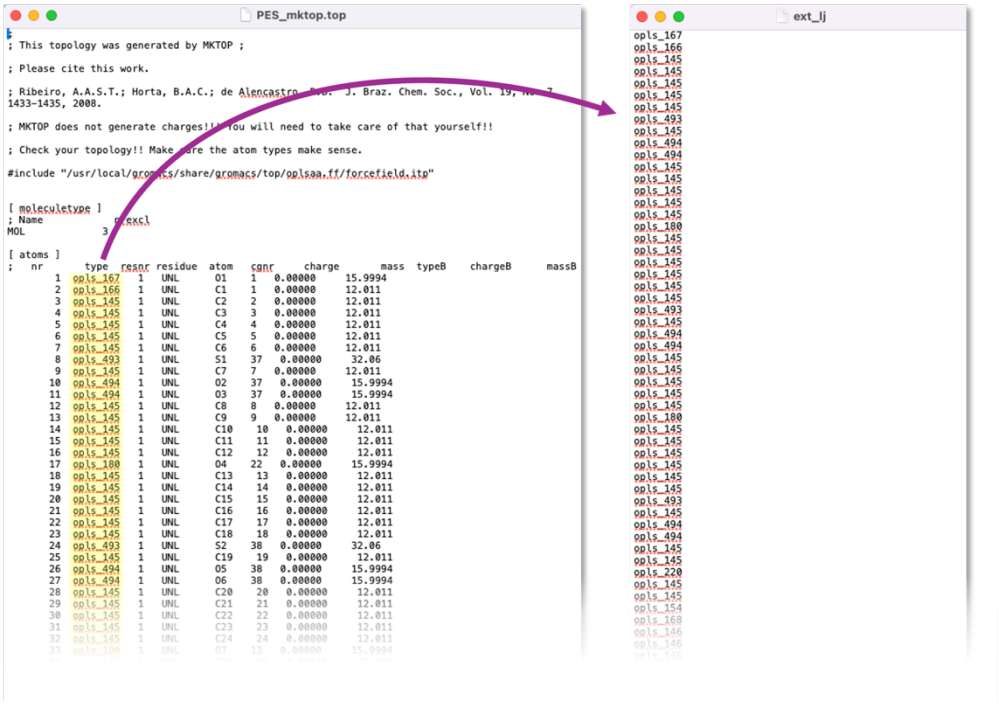


Figure 2. Atom types extracted from MKTOP output and put into a file names ext_lj

Generate QM input files using Q-Force:

```
qforce PES.pdb -o settings.ini
```

Run QM calculation

Ex: `orca PES_hessian.inp > PES_hessian.out`

After QM calculation finishes, generate flexible dihedral calculation inputs:

```
qforce PES
```

Run all flexible dihedral calculations. For the trimer of PES, Q-Force found three.

Run Q-Force again once all dihedral calculation information is available. Make sure the ext_lj file is in the correct folder.

```
qforce PES
```

Step 3: Make a new PDB file of the oligomer

Make a PDB file that will contain atom names and optimized geometry from QM calculation.

```
polypal pdb -xyz -x PES_opt.xyz -q PES_qforce.itp
```

This will generate PES_opt.pdb.

Step 4: Break oligomer into subunits

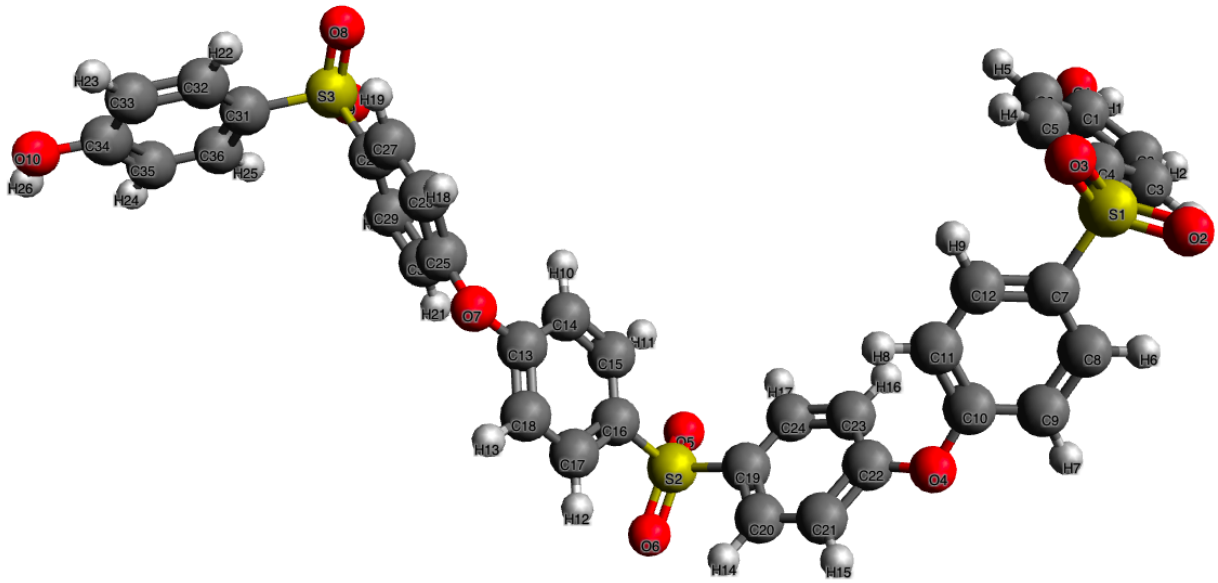
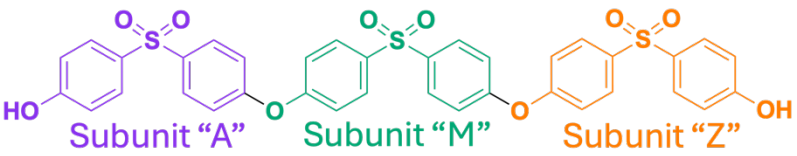
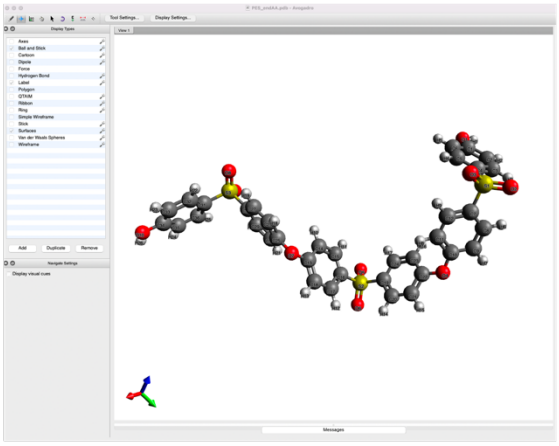


Figure 3. Trimer of PES with atom names

The PES trimer (Figure 3) was broken into three subunits: an A-terminal monomer, a middle monomer, and a Z-terminal monomer. The PES_opt.pdb was duplicated and Avogadro was used to remove atoms not part of the subunit (Figure 4). Be careful not to mess up the atom names (Figure 3). PolyPal uses these to create the topology files for Assemble!.

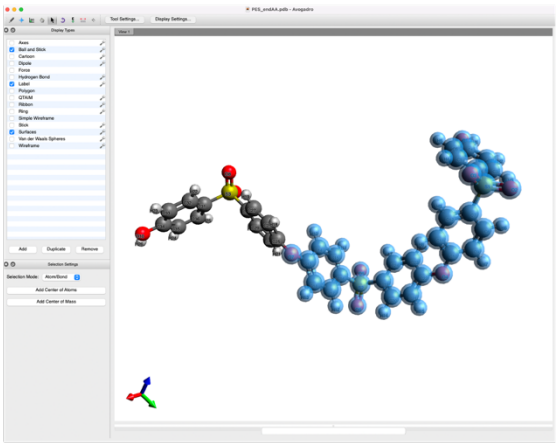
```
cp PES_opt.pdb PES_endAA.pdb
```



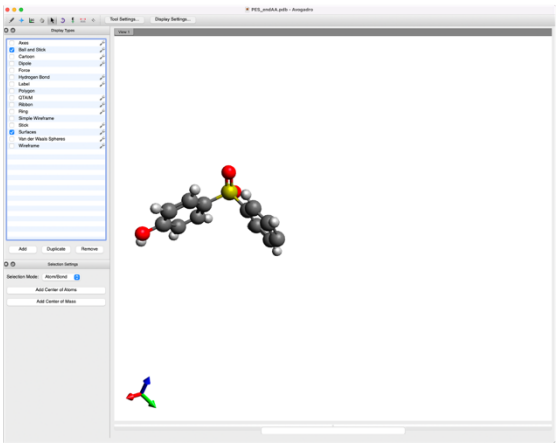


Information from PES_opt.pdb copied to new file PES_endAA.pdb

PES_endAA.pdb opened Avogadro



endAA subunit isolated and unnecessary atoms deleted



PES_endAA.pdb saved

Figure 4. Breaking the PES trimer into subunits using Avogadro

Confirm that the PDB of the subunit still contains the original atom names from the oligomer PDB file (Figure 5).

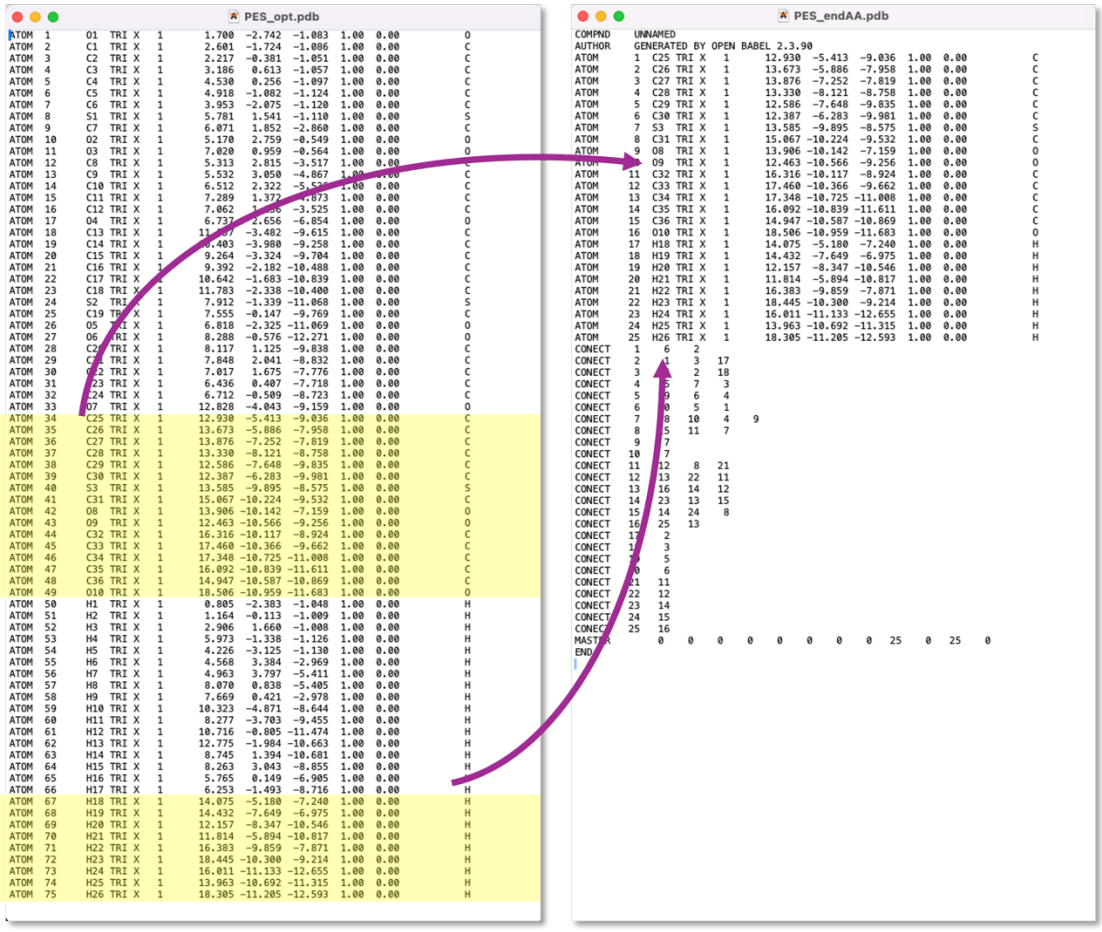


Figure 5. The atom names of the "A" subunit match from the trimer to the new subunit PDB file

This process was repeated for the middle monomer unit to generate PES_M.pdb and PES_endZZ.pdb

Step 5: Generate input files for Assemble!

Assemble! requires a PDB file of the monomer coordinates, a monomer topology file, and a force field file.

To generate the force field file:

```
polypal ff -q PES_qforce.itp -lj ext_lj
```

For more information on designating hook atoms, please see the Assemble! manual. For PES the hook atoms were selected according to Figure 6.

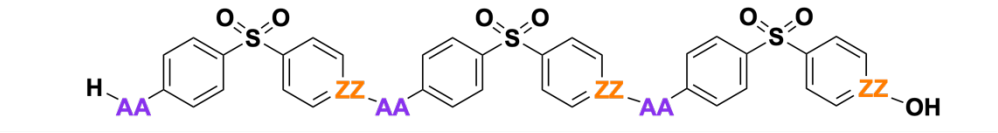


Figure 6. Depiction of how atom hooks were determined for the PES trimer

Users will have to know which atom names correspond to the "AA" and "ZZ" hook atoms (Figure 7). PolyPal can be used to modify PDB files with hook atoms and generate topologies.

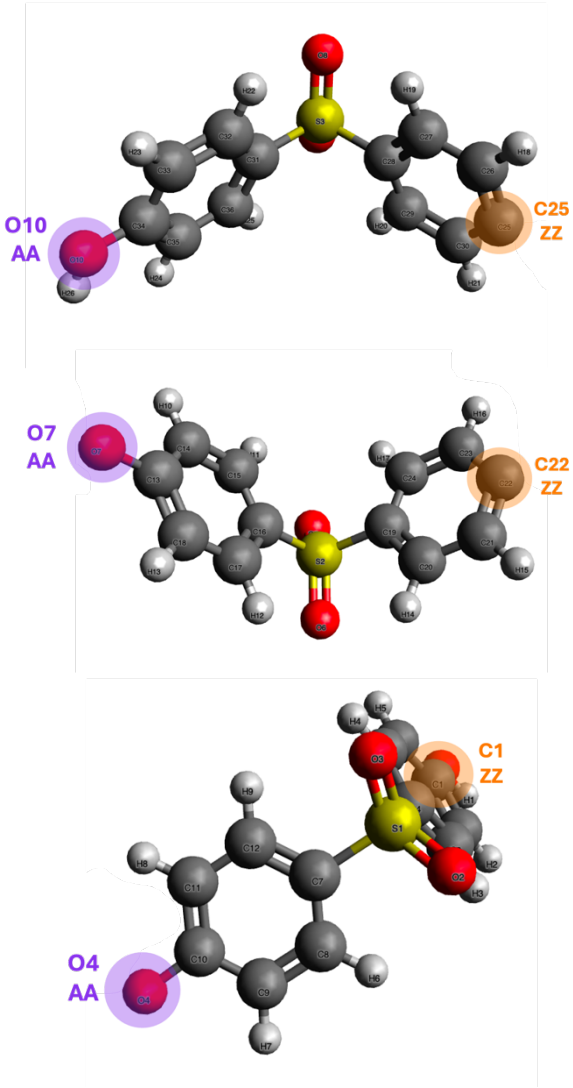


Figure 7. Atom names of hook atoms for each of the three subunits (Top: Subunit "A", middle: Subunit "M", bottom: Subunit "Z")

Once atoms are decided, PolyPal is used to change the atoms to “AA” and “ZZ” (Figure 8):

```
polypal top -p PES_endAA.itp -q PES_qforce.itp
```

```
------------------------------------------------------------------------

    @@@@   @@@  @    @   @  @@@@    @@@@@  @
    @   @ @   @ @     @ @   @   @   @      @ @
    @@@@  @   @ @      @    @@@@   @@@@@@@ @
    @     @   @ @      @    @      @      @ @
    @      @@@  @@@@@  @    @      @      @ @@@@@

Molly Warndorf, Tim Swager, Alfredo Alexander-Katz
Massachusetts Institute of Technology – 2024
------------------------------------------------------------------------

Processing PES_endAA.pdb

Enter AA name:O10
Enter ZZ name:C25

Topolgy for Assemble! written to PES_endAA_top
PDB file for Assemble! written to PES_endAA_assemble.pdb
```

Enter atom names that need to be changed

Figure 8. Correct use of `polypal top` where atom names are entered by user

Repeat for other two subunits. Confirm that name changes are correct (Figure 9).

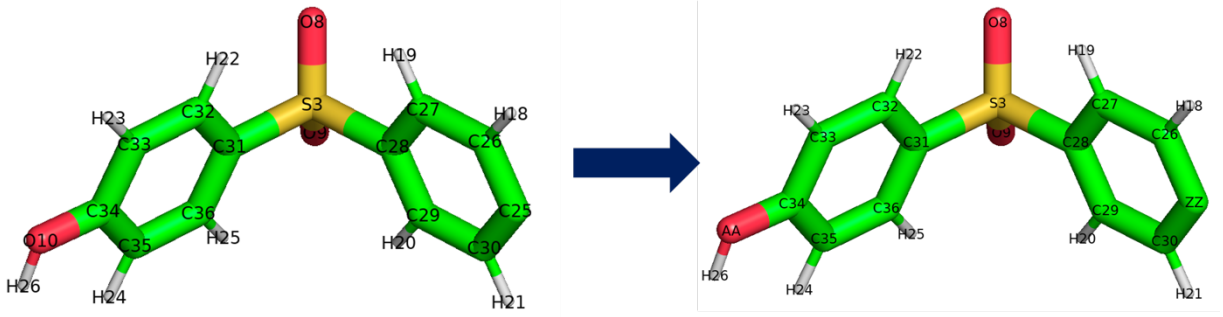


Figure 9. Subunit "A" where atoms were correctly changed to the hook atoms "AA" and "ZZ"

Add bond, angle, and dihedral hook information to the topology files. The assignment file (PES_qforce_assignments.txt) and PDB files can be used to assist with assigning force field information from the ff.txt names (Figure 10).

For example, the middle subunit (PES_M_top) was assigned the bond, angle, and dihedral information to adequately characterize the bond formation between the AA and ZZ hook atoms:

```
[ bonds ]
   ZZ     +AA         b54
[ angles ]
  -ZZ    AA     C13    a51
   C21    ZZ     +AA    a45
[ dihedrals ]
   C18    C13    AA     -ZZ    f12
   C14    C13    AA     -ZZ    f13
   C24    C23    ZZ     +AA    d89
   H15    C21    ZZ     +AA    d87
```

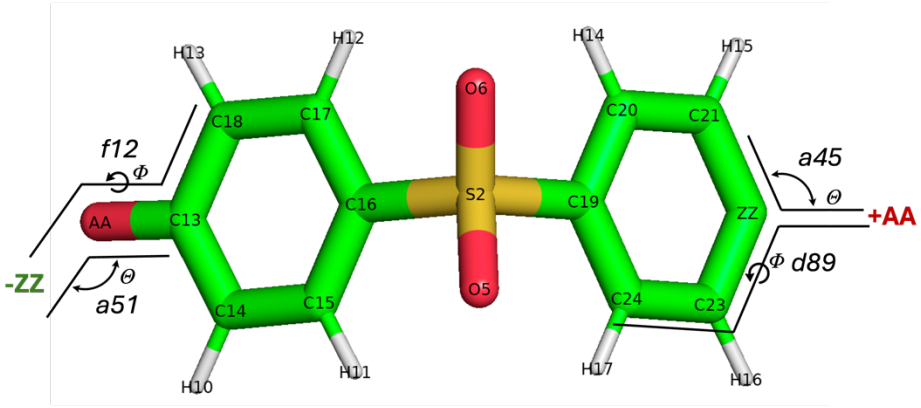


Figure 10. Depiction of some example hook parameters added to the topology file for Subunit "M"

Users must manually add this information, but a version of PolyPal that can assist with this step is currently in progress.

You should now have for PES:

- Three pdb files for the subunits (NAME_assemble.pdb)
- Three top files for the subunits (NAME_top)
- One file for the force field (_ff.txt)

Polymers can now be made with Assemble!

Step 6: Build polymer(s) with Assemble!

Assemble! can be used with their GUI or in a terminal.

To create a polymer with eight repeat units, the GUI was used. The PDB files, topology files, and the force field was loaded in.

A polymer was made (see 8mer_only.log in 8mer_only directory for more details on the files loaded for Assemble!) with Assemble!.

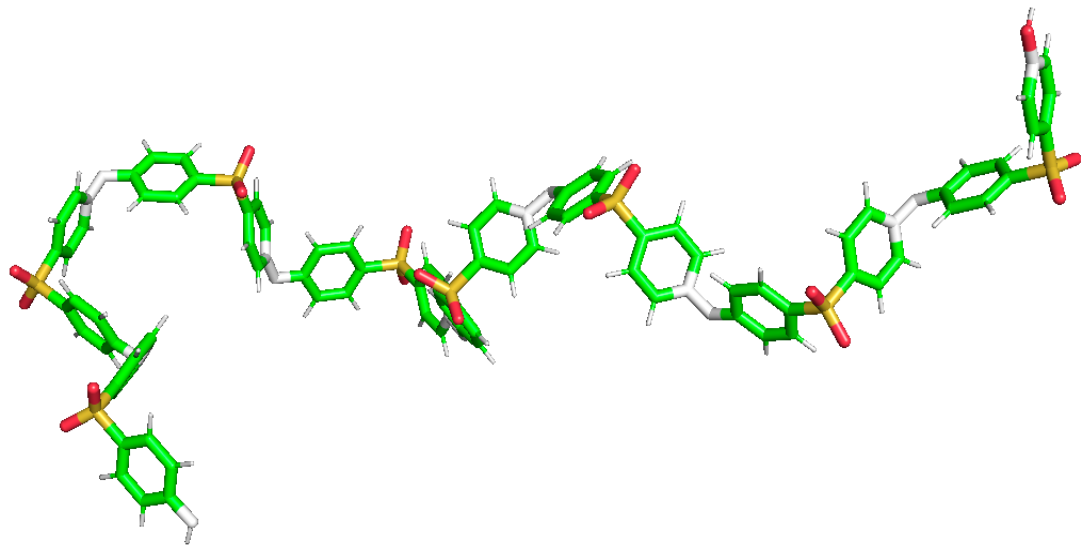


Figure 11. A polymer with eight repeat units made with Assemble!. Atoms "AA" and "ZZ" are still present (white atoms).

All output from Assemble! will have the atom names AA and ZZ still (Figure 11). In Step 7, users have the option to use PolyPal to clean up the files before simulating.

Step 7: Add charges, clean polymer .itp files, final simulation box packing

Using this workflow, polymer .itp files from Assemble! do not contain partial charges. PolyPal can assist with adding these charges.

The Atomic Charge Calculator II (https://acc2.ncbr.muni.cz/) can take PDB files and calculate partial charges for them.

To use ACCII and PolyPal for adding charges, first convert the Assemble! PDB files to a PDB file that ACCII can read by changing the atom names (Figure 12):

```
polypal gmx -p 8mer.pdb
```

```
--------------------------------------------------------------------------

    @@@@   @@@  @    @   @  @@@@    @@@@@  @
    @   @ @   @ @     @ @   @   @  @      @ @
    @@@@  @   @ @      @    @@@@   @@@@@@@ @
    @     @   @ @      @    @      @      @ @
    @      @@@  @@@@@  @    @      @      @ @@@@@

Molly Warndorf, Tim Swager, Alfredo Alexander-Katz
Massachusetts Institute of Technology — 2024
--------------------------------------------------------------------------

Atom 'AA' found. Please enter single letter corresponding to atom element (ex: C for carbon): O
Atom 'ZZ' found. Please enter single letter corresponding to atom element: C

AA changed to "O0" and ZZ changed to "C0" in 8mer.pdb, and saved to 8mer_final.pdb
Please ensure GROMACS can read these atoms correctly, and add any atom names to vdwraddi.dat if it cannot
```

Figure 12. Correct use of `polypal gmx` where "AA" and "ZZ" are changed back to their original elements

This will create 8mer_final.pdb, which was uploaded to ACCII and partial charges were computed with the QEq method and Rappe 1991 parameters. After downloading the charges, and unzipping the charge folder (8mer_charges) into the directory with 8mer_final.pdb in it, PolyPal can be used to add the charges to the 8mer.itp file and clean up the .itp file (Figure 13).

```
polypal gmx -f 8mer.itp -c 8mer_charges
```

8mer_final.itp

```
; generated with Assemble.py, by Matteo Degiacomi and Valentina Erastova, 2014
; sequence: AMMMMMMZ

[ moleculetype ]
8mer      3

 [ atoms ]
     1 opls_145      1 TRI  C0      1    0.02540   12.01100
     2 opls_145      1 TRI  C26     2    0.01577   12.01100
     3 opls_145      1 TRI  C27     3    0.01897   12.01100
     4 opls_145      1 TRI  C28     4    0.02818   12.01100
     5 opls_145      1 TRI  C29     5    0.02017   12.01100
     6 opls_145      1 TRI  C30     6    0.02490   12.01100
     7 opls_493      1 TRI  S3      7   -0.13656   32.06000
     8 opls_145      1 TRI  C31     8    0.02879   12.01100
     9 opls_494      1 TRI  O8      9   -0.23516   15.99940
    10 opls_494      1 TRI  O9     10   -0.23509   15.99940
    11 opls_145      1 TRI  C32    11    0.02026   12.01100
    12 opls_145      1 TRI  C33    12    0.01942   12.01100
    13 opls_220      1 TRI  C34    13    0.02830   12.01100
    14 opls_145      1 TRI  C35    14    0.01839   12.01100
    15 opls_145      1 TRI  C36    15    0.02010   12.01100
    16 opls_154      1 TRI  O0     16   -0.24939   15.99940
    17 opls_146      1 TRI  H18    17    0.07484    1.00800
    18 opls_146      1 TRI  H19    18    0.07868    1.00800
    19 opls_146      1 TRI  H20    19    0.07909    1.00800
    20 opls_146      1 TRI  H21    20    0.08272    1.00800
    21 opls_146      1 TRI  H22    21    0.07921    1.00800
    22 opls_146      1 TRI  H23    22    0.07778    1.00800
    23 opls_146      1 TRI  H24    23    0.07648    1.00800
    24 opls_146      1 TRI  H25    24    0.07923    1.00800
```

Figure 13. Charges added in (7th column) from ACCII using `polypal gmx`

The PDB file generated by PolyPal (8mer_final.pdb) can also be used by GROMACS to pack a GRO system with.

```
gmx insert-molecules -ci 8mer_final.pdb -box 10 10 10 -nmol 50 -o ini.gro
```

Finally, the GROMACS topology file should be updated to reflect how many polymers are in the box, and that all ITP file names are updated (Figure 14).

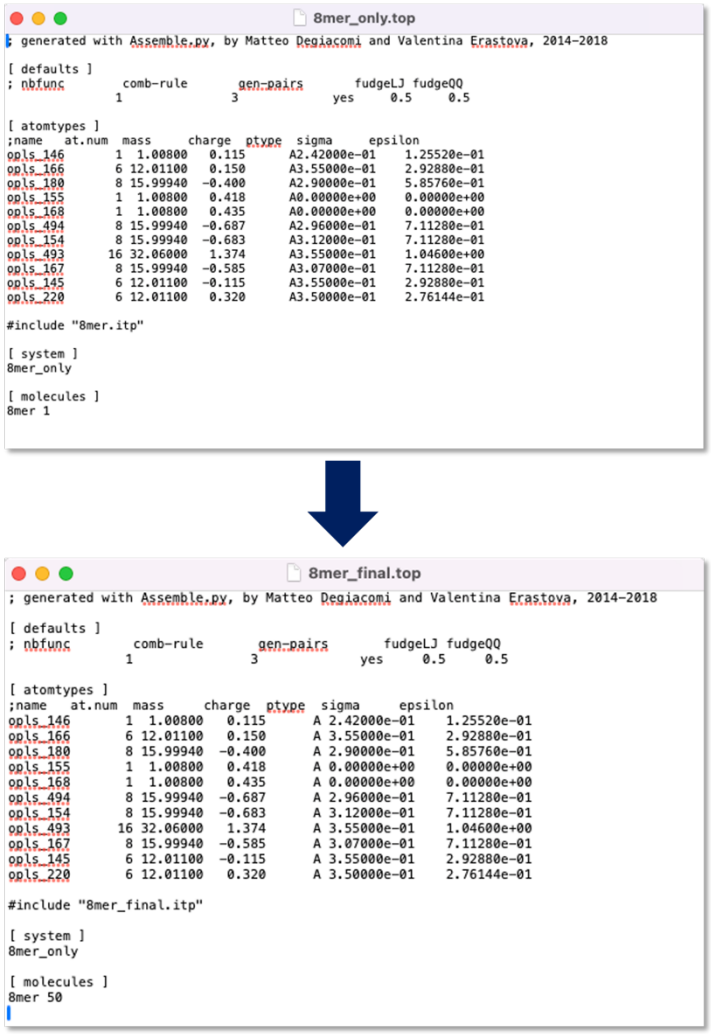


Figure 14. Gromacs TOP file updated with correct columns for [atomtypes], new ITP file name, and correct number of molecules

Step 8: Equilibration, annealing, and production simulations

Polymers can now be simulated! Example mdp files have been provided in the tutorial according to the equilibration annealing, and production simulations outlined in the manuscript.