Computational Assembly of Polymorphic Amyloid Fibrils Reveals Stable Aggregates

Mohamed Raef Smaoui,[†] Frédéric Poitevin,[‡] Marc Delarue,[‡] Patrice Koehl,[§¶] Henri Orland,[||] and Jérôme Waldispühl[†]*

[†]School of Computer Science, McGill University, Montreal, Canada; [‡]Unit of Structural Dynamics of Macromolecules, Institut Pasteur, CNRS UMR 3528, Paris, France; [§]Department of Computer Science and [¶]Genome Center, University of California, Davis, California; and [||]Institut de Physique Théorique, CEA-Saclay, Gif/Yvette Cedex, France

Raef Smaoui et al.

Polymorphic Amyloid Fibrils

Submitted July 12, 2012, and accepted for publication December 10, 2012.

*Correspondence: jeromew@cs.mcgill.ca

Supporting Information

1 Materials and Methods

1.1 CreateFibril: Rigid Affine Transformations

Affine Transformations are represented by mathematical matrices which we use to act on the atom positions of proteins stored in PDB files. The translation matrix is capable of moving atoms, and when applied to a protein's atom positions it produces the effect of moving the protein in 3D space. We represent an atom in the following homogenous representation,

$$a = \begin{bmatrix} a_x \\ a_y \\ a_z \\ 1 \end{bmatrix} \tag{1}$$

where a_x, a_y, and a_z, are how far a lies in the direction of the x-axis, y-axis, and z-axis, respectively. To move the atom a by a distance d_x in the x-axis direction, d_y in the y-axis direction, and d_z in the z-axis direction, we apply the following translation matrix $T(d)$,

$$T(d) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

onto the homogenous representation of a in the following way

$$a' = T(d) * a = \begin{bmatrix} a_x + d_x \\ a_y + d_y \\ a_z + d_z \\ 1 \end{bmatrix}. \tag{3}$$

Ignoring the 1 at the very bottom of a', we observe that the point a' is in fact a translation of a distance of d_x in the x-axis direction, d_y in the y-axis direction, and d_z in the z-axis direction, as wanted. This way, we can move a whole protein in space a distance d by applying $T(d)$ onto all its atom positions. Furthermore, we apply $T(d)$ to copies of an amyloid protein and force them to aggregate side-by-side and simulate the assembly of fibrils.

The second type of Rigid Affine Transformation matrices we find useful is rotation matrices. We can rotate points in a 3D space around the x-axis, y-axis, or z-axis using standard affine transformation rotation matrices.

For example, to rotate the atom representation, a, about the x axis by θ degrees counterclockwise we multiply it by the following matrix,

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) & 0 \\ 0 & sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4}$$

Similarly, to rotate a about the y-axis by α degrees counterclockwise we multiply it by the following matrix,

$$R_y(\alpha) = \begin{bmatrix} cos(\alpha) & 0 & sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\alpha) & 0 & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

Also, to rotate a about the z-axis by β degrees counterclockwise we multiply it by the following matrix,

$$R_z(\beta) = \begin{bmatrix} cos(\beta) & -sin(\beta) & 0 & 0 \\ sin(\beta) & cos(\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

We are not constrained to rotations about the x, y, and z axes only. Correctly combining the three rotation matrices above allows to rotate about any arbitrary axis. The power of Affine Transformations lies in the fact that we can combine and chain many transformation operations on a single point, or atom in this case.

The amyloids that make up fibrils have been reported to aggregate together and rotate slightly with every monomer addition about a fibril axis. The translation and rotation matrices applied together enable us to model this fibril aggregation-rotation phenomenon. For our computational framework, we meticulously derive the fibril axis and transformation matrices that build our fibrils next.

1.2 CreateFibril: Fibril Axis Matrix

CreateFibril rotates aggregates around a fibril axis by θ degrees and moves them along the axis by d angstroms by computing a transformation matrix $F_{\theta,d}$ that applies rotation and translation matrices on the amyloid protein models.

The general idea behind constructing $F_{\theta,d}$ is to first move the protein a distance of d angstroms along the fibril axis by applying a translation matrix $T(d)$. To rotate the protein around the fibril axis, we move the fibril axis to the origin, rotate the axis so that it aligns with either the x, y, or z axis, apply a standard rotation transformation to it by θ degrees with respect to the chosen axis, and then return the fibril axis to its original position in space. This procedure involves a combination of translation and rotation matrices that we believe is worthy of explaining. Without loss of generality, given a point p lying in the -x, -z, +y direction for which the fibril axis passes through, we translate the system that contains our fibril axis and shifted protein atoms by $T(-d-p)$. This operation translates all the objects in our system towards the origin. The fibril axis now passes through the origin and all distances between the atoms and the fibril axis are preserved.

We will choose to align the fibril axis with the x-axis. To do so, we first need to rotate the fibril axis by ϕ degrees to intersect the x-y plane, and then rotate it by ψ degrees to align it with the x-axis. Therefore, we need to apply the rotation matrix $R_y(\phi)$ to the fibril axis to intersect the x-y plane followed by the rotation matrix $R_z(\psi)$ to rotate the fibril axis along the z-axis and align it with the x-axis.

We are now in a position to rotate our protein atoms around the fibril axis by θ degrees. We use the standard x-axis rotation transformation since the x-axis and the fibril axis are aligned at this point. We apply the rotation matrix $R_x(\theta)$ to the fibril axis and the protein. The fibril axis rotates around itself, while the protein atoms rotate around the fibril axis.

To conclude, we need to move our fibril and protein back to their initial positions in space. To undo all axis positioning rotations and translations we rotate the axis back by ψ degrees with $R_z(-\psi)$ followed by $-\phi$ degrees around the y-axis with $R_y(-\phi)$. We finally translate the axis and the protein back by $T(d+p)$. This puts the fibril axis exactly back where it was and achieves our desired rotation of the protein atoms by θ degrees around the fibril axis. Hence our fibril axis rotation matrix is,

$$F_{\theta,d} = O^{-1}(d,p,\phi,\psi) * R_x(\theta) * O(d,p,\phi,\psi) * T(d) \tag{7}$$

where O^{-1} and O are defined as,

$$O^{-1}(d,p,\phi,\psi) = T(d+p) * R_y(-\phi) * R_z(-\psi) \tag{8}$$

$$O(d, p, \phi, \psi) = R_z(\psi) * R_y(\phi) * T(-d - p). \tag{9}$$

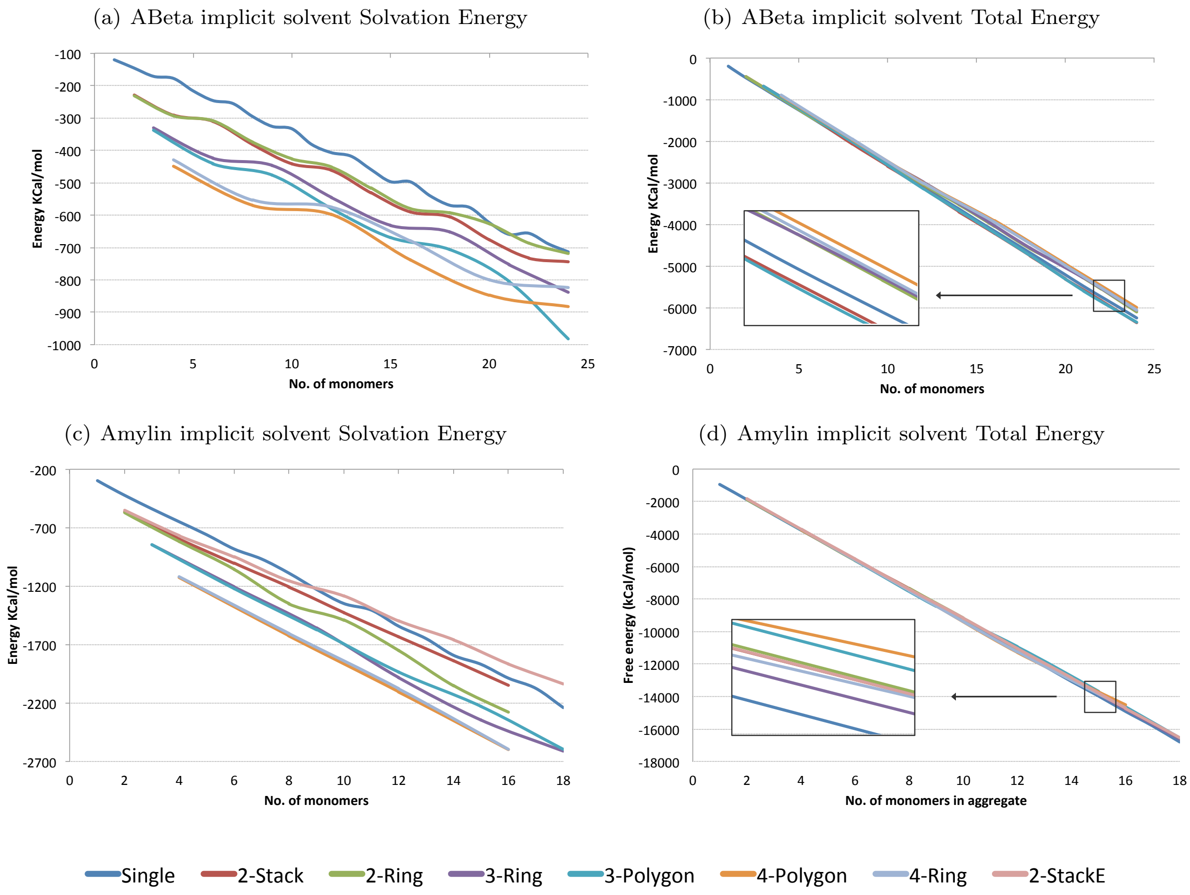
The last piece of the puzzle is to explain how to obtain the correct values for ϕ and ψ. The ϕ angle is the angle needed to rotate the fibril axis to intersect the x-y plane, i.e., the angle between the fibril axis and the x-y plane. To find ϕ, we set v to be the projection of the fibril axis on to the x-z plane (v is the fibril axis with the y component set to 0) and we set w to be a vector on the x axis (w is the fibril axis with the y and z component set to 0) and make use of the following dot product property to find ϕ,

$$\phi = \cos^{-1} \frac{v \cdot w}{\|v\|\|w\|}. \tag{10}$$

With this ϕ value, we can rotate our fibril axis to fall into the x-y plane and denote it as f. If we set vector l to be the projection of the fibril axis in the x-y plane (l is f with the y component set to 0) then we can find ψ by,

$$\psi = \cos^{-1} \frac{l \cdot f}{\|l\|\|f\|}. \tag{11}$$

$F_{\theta,d}$ is dynamically calculated in CreateFibril whenever a rotation by θ and translation by d is needed. Supplying CreateFibril with values for d and θ creates fibrils that elongate at a distance d apart and rotate an angle of θ degrees about the fibril axis.



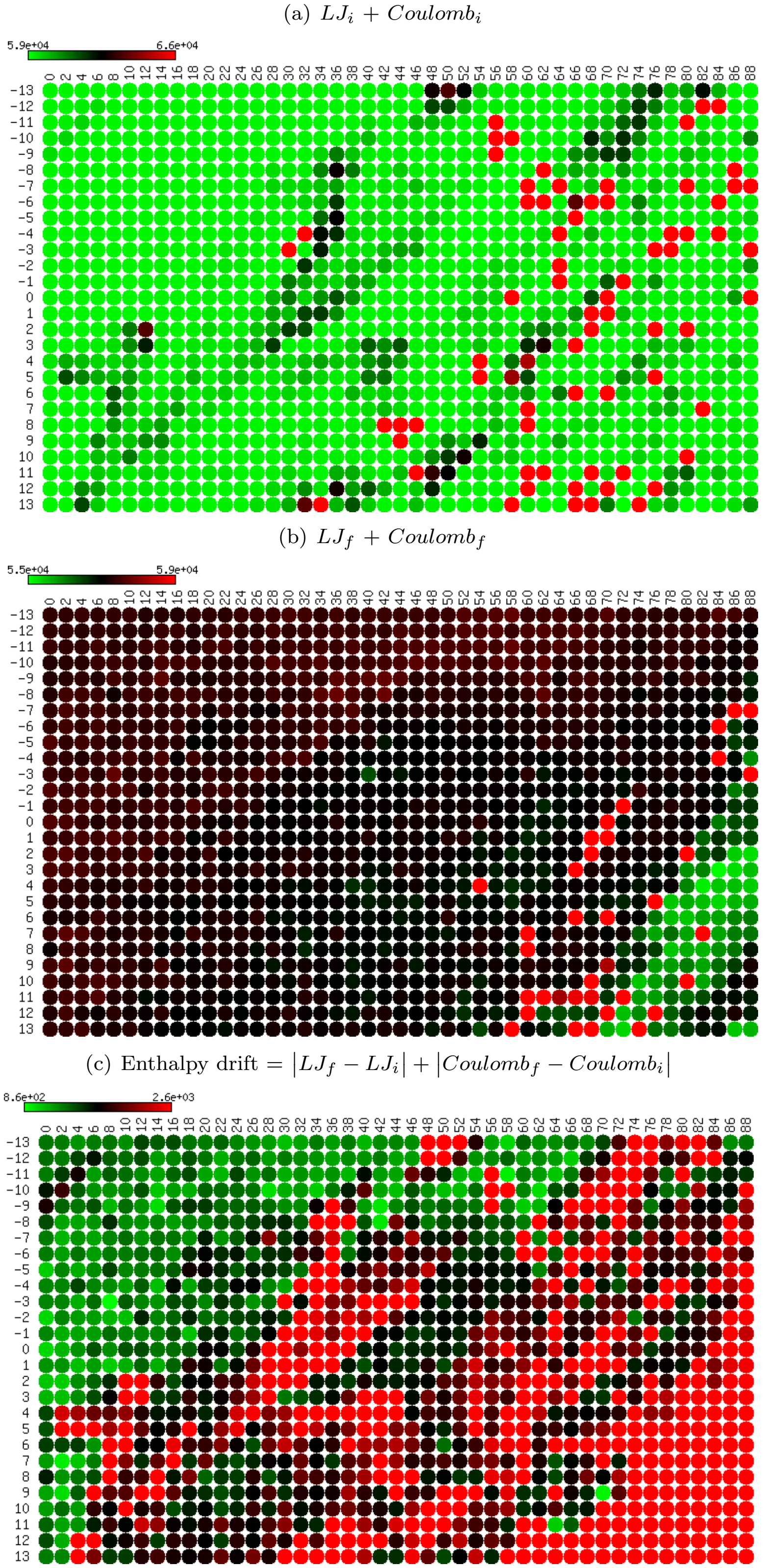
(a) ABeta implicit solvent Solvation Energy

(b) ABeta implicit solvent Total Energy

(c) Amylin implicit solvent Solvation Energy

(d) Amylin implicit solvent Total Energy

Figure S1: Energies of ABeta and Amylin fibrils as they aggregate. (a) and (c) Solvation energy by implicit solvent, (b) and (d) Free energy by implicit solvent.

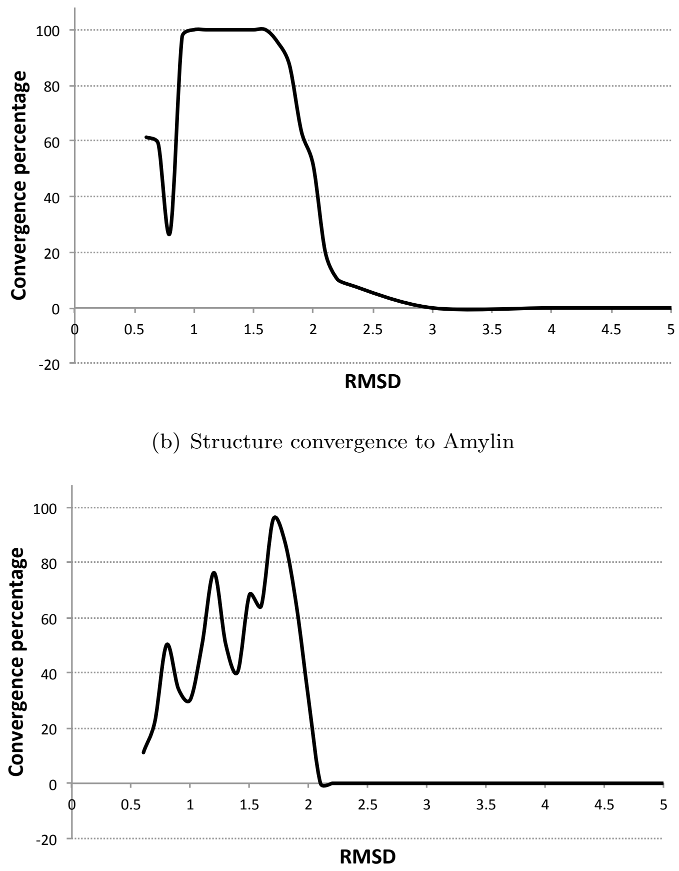


(a) $LJ_i + Coulomb_i$

(b) $LJ_f + Coulomb_f$

(c) Enthalpy drift $= |LJ_f - LJ_i| + |Coulomb_f - Coulomb_i|$

Figure S2: Heatmap representation of the stability landscape and the enthalpy drift of Abeta wrapped 2-Stack fibrils. Exploration covered crossing angles between 0 and 88 degrees and fibril rotation angles (main axis) between -13 and 13 degrees. (a) The energies of structures initially built by CreateFibril assembled through rigid affine transformations. (b) The energy plot of each structure after a run of Energy minimization. (c) The energy difference, or enthalpy drift, determines the initial structural stability of fibrils built by CreateFibril in (a). Parameters that produce low drifts are considered to build fixed point structures that are believed to live in local minimum neighbourhood. Energies in KJ/mol.



(a) Structure convergence to Abeta

(b) Structure convergence to Amylin

Figure S3: Convergence rate of Abeta and Amylin tweaked structures. Thousands of structures were created and tweaked for every RMSD decimal of convergence. MD simulations were performed on tweaked structures with the following parameters: CHARMM force field, SPC model, TIP3P box with a minimum of 15 $\AA$ from any edge of the box to any atom, and 20000 integration steps per minimization run following a steepest gradient decent algorithm.

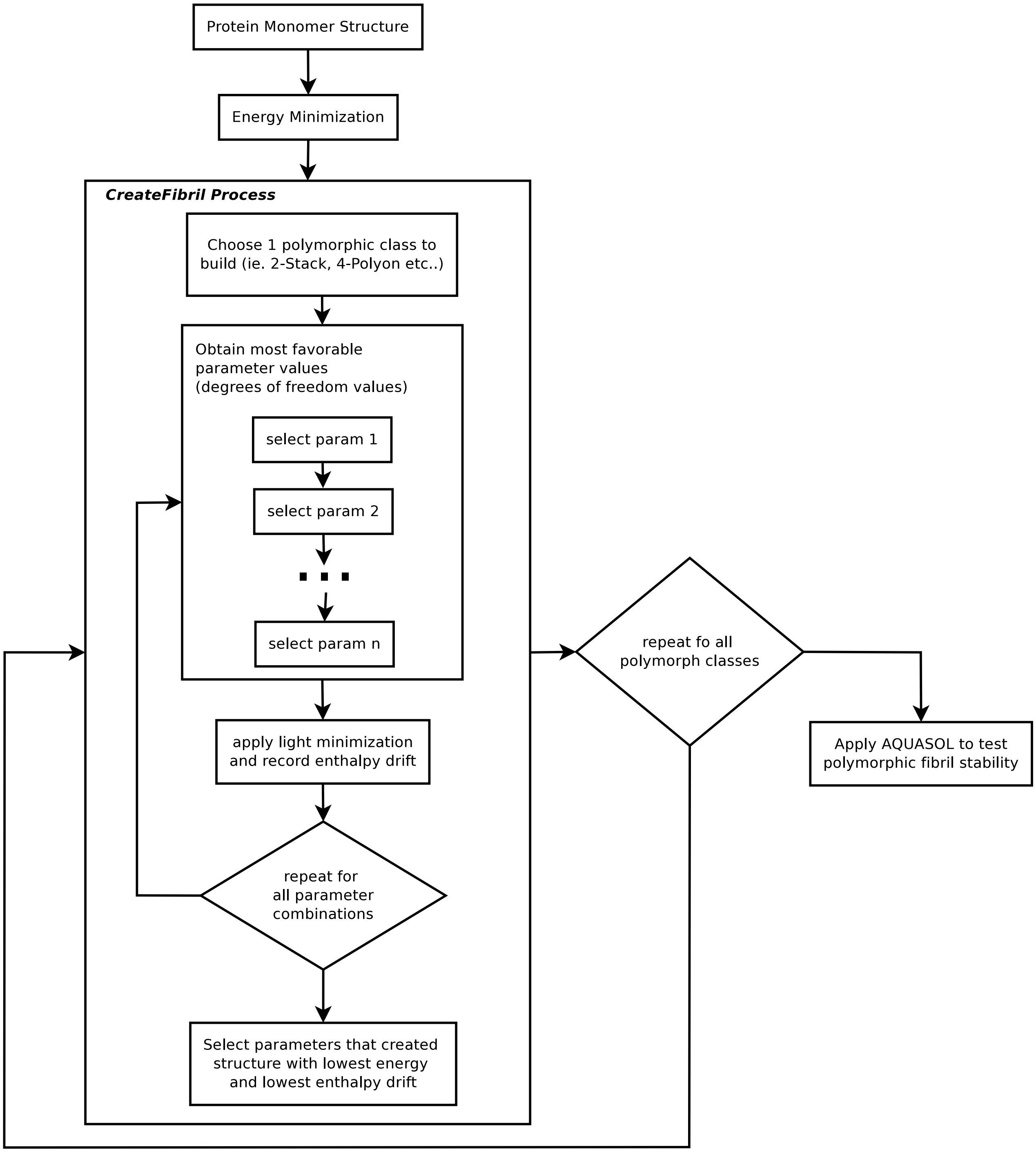

Protein Monomer Structure
Energy Minimization
CreateFibril Process
Choose 1 polymorphic class to build (ie. 2-Stack, 4-Polyon etc..)
Obtain most favorable parameter values (degrees of freedom values)
select param 1
select param 2
. . .
select param n
apply light minimization and record enthalpy drift
repeat for all parameter combinations
Select parameters that created structure with lowest energy and lowest enthalpy drift
repeat fo all polymorph classes
Apply AQUASOL to test polymorphic fibril stability

Figure S4: CreateFibril procedures and pipeline.