Chapter 5:

A Computational Library Design Protocol for Rapid Improvement of Protein Stability - FRESCO

Hein J. Wijma,a Maximilian J.L.J. Fürst,a Dick B. Janssen*a

aMolecular Enzymology Group, University of Groningen, Nijenborgh 4, 9747AG, Groningen, The Netherlands
*Corresponding author

Published in:

Abstract

The ability to stabilize enzymes and other proteins has wide-ranging applications. Most protocols for enhancing enzyme stability require multiple rounds of high-throughput screening of mutant libraries and provide only modest improvements of stability. Here we describe a computational library design protocol that can increase enzyme stability by 20-35°C with little experimental screening, typically fewer than 200 variants. This protocol, termed FRESCO, scans the entire protein structure to identify stabilizing disulfide bonds and point mutations, explores their effect by molecular dynamics simulations, and provides mutant libraries with variants that have a good chance (> 10%) to exhibit enhanced stability. After experimental verification, the most effective mutations are combined to produce highly robust enzymes.
Computational Protein Stabilization

Introduction

Thermostable enzymes are important for applications in research, analytics, diagnostics, and industry.\textsuperscript{1-4} For many enzyme classes no thermostable variants are available from nature. With most protein engineering techniques, the reported increases in apparent melting temperature ($T_m$) are in the range of 2–15 °C.\textsuperscript{2} These are small increases compared to the differences between naturally occurring thermostable enzymes ($T_m > 80$ °C) and mesostable enzymes ($T_m$ approximately 50 °C).\textsuperscript{5} To obtain larger stability improvements, the FRESCO workflow was developed. FRESCO uses the computational library design approach - (sets of) mutations are pre-screened \textit{in silico}. The result is a small high-quality library that can be experimentally screened in a short time. The results hitherto obtained with four enzymes showed promising $T_m$ improvements of 20 - 35 °C.\textsuperscript{6-9}

A challenge in thermostability engineering is related to the large size of most enzymes and their irreversible denaturation. Whereas small proteins often unfold reversibly in a single step (Figure 1A), larger ones mostly aggregate irreversibly following the initial unfolding of certain regions (Figure 1B).\textsuperscript{4} In case of reversible one-step unfolding, mutations at all positions are expected to have an effect on $T_m$ since interactions of all amino acids change in the unfolding step. For larger proteins, mutations outside the early unfolding region have a much smaller or a negligible effect\textsuperscript{4,10-11} and the spots where mutations can improve stability may be hard to find.

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure1.png}
\caption{Thermally induced inactivation of small and large proteins.\textsuperscript{12} A) Small proteins ($\leq$ 20 kDa), most commonly unfold in a single reversible step and the unfolded protein remains soluble. B) In large proteins, there is often a specific region (indicated with a red circle) that unfolds first. This partial unfolding often triggers irreversible aggregation.}
\end{figure}

The FRESCO workflow (Figure 2) addresses this challenge by \textit{in silico} screening for diverse types of potentially stabilizing mutations throughout the enzyme.\textsuperscript{6-7} Selecting a small subset of the FRESCO generated mutations bears
the risk to miss mutations stabilizing early unfolding regions. The most stabilizing mutations generated by FRESCO were found both in flexible and in rigid (low B-factor) stretches of the protein sequence.\textsuperscript{7,9} If the target protein is well expressed in an easy to transform host organism like \textit{Escherichia coli}, the complete FRESCO library can be experimentally screened in a few weeks and will produce enough stabilizing mutations to be combined into a highly robust final variant.

\begin{figure}
\centering
\includegraphics[width=0.8\textwidth]{protocol.png}
\caption{Framework for rapid enzyme stabilization by computational library design (FRESCO). The numbers refer to the sections in this protocol. The protocol differs slightly from the initial approach,\textsuperscript{7} in which chemically unreasonable mutations were filtered out prior to the MD simulations. The current protocol is faster as each mutant only needs to be inspected once.}
\end{figure}

Below, the entire FRESCO\textsuperscript{6-7} protocol is described in detail for experimentalists in a way that requires no prior experience with Unix(-like) systems, which are required for running the protocol. The protocol is implemented for the user-friendly Mac OS X operating system but can be modified to be used under Linux\textsuperscript{a}. Possibly the protocol can also be implemented under the Linux bash

\footnote{This protocol could be adapted for any Unix(-like) system on which one can install the required software. Typical challenges then are finding the correct graphic drivers that YASARA needs to run smooth molecular graphics, compiling Rosetta if the precompiled binaries do not work for that particular Linux distribution, and Linux distribution specific differences in the commands that need to be given. For example, \texttt{top -o cpu} does not work under Linux but \texttt{top} does. It is possible to run the visual inspection under Windows.}
shell that recently became available under Windows 10, or other Unix-like environments. The underlying algorithms are described elsewhere.\textsuperscript{7,13-15} The example that is described in the protocol, the stabilization of the enzyme limonene epoxide hydrolase (LEH),\textsuperscript{7} enables the users to verify that all the installed software works properly.

**Materials**

The FRESCO workflow consists of executables, scripts, and parameter files for running the three software packages that identify the stabilizing mutations. FoldX and Rosetta are employed to predict stabilizing point mutations.\textsuperscript{13-14} YASARA is used for molecular graphics, for designing disulfide bonds, and for MD simulations.\textsuperscript{15} The FRESCO specific software is made available via [https://groups.google.com/forum/#!forum/fresco-stabilization-of-proteins](https://groups.google.com/forum/#!forum/fresco-stabilization-of-proteins). Via this forum, it is also possible to ask questions about the protocol to other FRESCO users. A short introduction to using the command line (unixIntroduction.pdf, with short exercises for people without command line experience) and instructions for obtaining and installing the other software are provided there as well. The procedure described below assumes that all software has been installed as described in “installationInstructions.pdf”. YASARA Dynamics (YASARA View lacks the required functionalities), Rosetta, and FoldX require licenses. Below, differently colored layouts distinguish UNIX command line input and YASARA command line input.

**Hardware requirements**

The only part of FRESCO that needs a large amount of calculation power is the MD simulation of the mutants (see below), for which a computer cluster may be needed. The calculation time of these MD simulations increases roughly with the square of protein size but also depends on protein shape. Accordingly, it depends on the target protein whether or not a computer cluster is required\textsuperscript{b}. The MD simulations for LEH (32 kDa) took < 45 min per variant on a desktop computer (Intel Core i5, 4 cores, 3.2 GHz). Thus, testing 500 mutants by MD would take 10 days. On a computer cluster, this could be done in a few hours. To test the protocol on the LEH example as shown below, only a few selected MD simulations are required.

\textsuperscript{b} The increase of calculation time with the square of the protein length is due both to the increasing number of mutants that will be screened as well as the increase in computation time per MD simulation.
Methods

Setting up a suitable directory structure and preparing the target protein

In this section, defects to the pdb file, such as missing hydrogen atoms, are repaired. The resulting structure, which should be representative for the protein in solution, will be used for the rest of the procedure.

1. Create a design directory and subdirectories for each step of the procedure, e.g. in your home folder (which can be abbreviated with ~):

   ```
   mkdir ~/frescoLEH
   cd ~/frescoLEH
   mkdir disulfides foldx rosetta designsMD finalVariants
   ```

2. Obtain a pdb file of the protein of interest. Best are crystal structures of high resolution and with a low Rfree (< 0.25). Structures obtained through homology modeling are probably too inaccurate. For training purposes, download the LEH structure 1NWW.pdb.

3. Move the downloaded 1NWW.pdb to ~/frescoLEH. In this directory, type `yasara 1NWW.pdb&` (opens the process in the background, so the console can still be used while YASARA is running).

4. In YASARA, delete buffer, ligand and any other nonstructural molecules. For 1NWW, this can be done by typing `DelRes MES HPN` (MES and HPN are the names of the buffer and ligand molecules). For other proteins, possible ligands or buffer molecules have to be identified by visual inspection with YASARA. Usually, they are displayed in the amino acid sequence panel in the bottom of the YASARA window. Cofactors, such as heme or NADP, should not be deleted at this stage.

5. Use the YASARA commands `CleanAll` and `OptHydAll` to obtain reasonable protonation states for most residues. For each protein, one should carefully check by visual inspection that the protein structure is realistic.

---

For more information about the YASARA commands and their syntax, use the SearchDoc function within YASARA. For example `SearchDoc AddBond`.

For other proteins, common problems encountered are: unusual numbering of the amino acids in the pdb file, gaps in the protein sequence, unusual residues, unusual protonation states, cofactors that need to be manually adapted to ensure the simulated state is physically relevant, etc. Careful inspection and editing solves such problems.
6. Save the structure as a pdb file: `SavePdb OBJ 1, 1NWW_cleaned`.

**Running an MD simulation for Dynamic Disulfide Discovery**

To explore possible protein conformations for disulfide bond design, an MD simulation of the wild-type enzyme is carried out. The result will be a series of snapshots that provide samples of the possible protein conformations. These conformations are later used for designing disulfide bonds (Figure 2).

1. Within the disulfides directory, make a subdirectory: `mkdir disulfides/trajectoryMD`.
2. Enter this directory (`cd disulfides/trajectoryMD/`) and copy the cleaned pdb file to this directory: `cp ../../1NWW_cleaned.pdb .`.
3. To run the MD simulation, type (as a single command):

   ```bash
   yasara -txt ~/frescoSoft/FRESCO/MDSimulBackboneSamp1.mcr "MacroTarget = ‘1NWW_cleaned’" > LOG_MD&
   ```

   This will start the macro (.mcr) file that contains all necessary specifications for the simulation.
4. Verify that the MD simulation started. The command `ls –rlt` should reveal new files being created and `top -o cpu` should reveal the processes running (exit top with `q`). It should take several hours or even days before this simulation is finished. One can already start with the next two sections.

**Predicting stabilizing point mutations with FoldX**

This section starts the FoldX calculations, which predict the \( \Delta \Delta \text{G}_{\text{fold}} \) for individual mutations. These results will be used to select stabilizing point mutations (Figure 2).

1. In the frescoLEH directory (`cd ~/frescoLEH`), create a table file (.tab) that lists the protein residues that are allowed to mutate by executing the following command, type:

   ```bash
   yasara -txt ~/frescoSoft/FRESCO/FarEnoughZone.mcr "MacroTarget = ‘1NWW’" "AvoidResidue = ‘HPN’" "AvoidDistance = 5"
   ```

   For other proteins, replace “HPN” with the PDB abbreviation of either an active site ligand, or cofactor. The “AvoidDistance” is the minimal distance that residues should have from the “AvoidResidue” to be allowed to mutate. If the entire protein should be allowed to mutate, use ‘XXX’.
2. This should result in a new file (ls -rtl) named 1NWW_MoreThan5AngstromFromHPN.tab.

3. Go to the FoldX directory (cd foldx) and copy the rotabase.txt file (cp ~/frescoSoft/FoldX_2017/rotabase.txt .).

4. Set up the FoldX calculations using (the text in between <X> should be replaced, including the brackets themselves):

```
~/frescoSoft/FRESCO/DistributeFoldX Phase1 1NWW_cleaned 2 A B ../1NWW_MoreThan<X>AngstromFrom<AvoidedResidue>.tab 1000
~/frescoSoft/FoldX_2017/foldx<version>
```

A short explanation will appear on the command line describing what DistributeFoldX does. This explanation will also provide guidance when setting up the calculation for one’s own protein of interest. Write down the number of mutations that will be analyzed by FoldX.

5. Start the calculations by running the todolist file: ./todolist &.

6. Verify that no error messages appear and check whether the calculations are indeed running (top -o cpu). Type `tail */LOG` to verify that no problems were encountered. It may take a day for the calculations to finish. One can estimate how much time the calculations will take, based on the information provided by the command `ls -rtl Subdirectory*`.

Predicting stabilizing point mutations through Rosetta

This is essentially the same procedure as described for FoldX in the previous section.

1. Enter the Rosetta directory (cd ~/frescoLEH/rosetta) and copy the necessary parameter file FLAGrow3 into this directory (cp ~/frescoSoft/FRESCO/FLAGrow3 .).

2. Open this file FLAGrow3 using a plain text editor (open -e FLAGrow3) and adapt the Rosetta database location, behind “–database”, to match that in your own computer. Alternatively, for manual editing you might use Perl:

```
perl -pi -e "s,-database.*,-database ~/frescoSoft/rosetta_bin_<version >_bundle/main/database/,,g" FLAGrow3
```

---

*e Input files or commands often cause problems due to (small) abnormalities in the formatting or errors in spelling and punctuation. If Rosetta, FoldX, or YASARA do not work as expected, it is best to first examine whether the log files contain error messages. This can be done by entering `tail log` if the output of the failing program was redirected to a file called log.*
3. The Rosetta_ddg application is parameterized for implicit water and has not (yet) been programmed to accept multimeric proteins. Therefore, explicit water molecules have to be deleted. If the pdb file of the protein contains more than one chain, residues and chain IDs have to be renamed. For example if there are two chains –A and B having 400 amino acids each– residues of chain B have to be renumbered to 401-800, and Rosetta will accept this as a “monomeric” protein. Use YASARA to adapt the earlier cleaned pdb file: For LEH type: yasara ../1NWW_cleaned.pdb. Then, remove all the water molecules (DelRes HOH), remove an amino acid that occurs only in one of the LEH subunits (DelRes 4), rename subunit B to A (RenameMol B, A) and ensure the software forgets that the protein are two different chains with (JoinRes protein). Ensure consecutive residue numbering, without shifting the original positions in the first subunit, with RenumberRes protein, 5 and save the file in the current directory (SavePdb OBJ 1, 1NWW_forRosetta).

4. Set up the calculations by typing (use Tab-completion):

```bash
~/frescoSoft/FRESCO/DistributeRosettaddg Phase1 ../1NWW_MoreThan5AngstromFromHPN.tab 2 A 5 B 150 1NWW_forRosetta.pdb 4000 FLAGrow3 ~/frescoSoft/rosetta_bin_<version>_bundle/main/source/bin/ddg_monomer.macosclanreleas
```

5. Start the calculations (./.todolist&)

6. Again, verify that these calculations are running correctly (top -o cpu, ls -rI Subdirectory*, tail */LOG).

**Predicting stabilizing disulfide bonds through Dynamic Disulfide Discovery**

The snapshots created in the section “Running an MD simulation for Dynamic Disulfide Discovery” are now used to design disulfide bonds.

1. Verify the snapshot files exist in the disulfides directory (cd ~/frescoLEH/disulfides) with ls trajectoryMD/*pdb. The files have names ending with for example 1000ps.pdb, where ps stands for picoseconds.

2. In the disulfides directory, make a new subdirectory mkdir all_designs and copy both the cleaned pdb file (cp ../1NWW_cleaned.pdb all_designs/) and
the snapshots from the MD trajectory in there `cp trajectoryMD/*ps.pdb all_designs/`.

3. Go into this new directory (`cd all_designs/`)

4. If desired, the minimum number of residues spanning between a disulfide bond can be increased by editing

   ~/frescoSoft/FRESCO/DisulfideDiscovery.mcr. However, this is unnecessary for thermostability engineering.\textsuperscript{16}

5. Type `chmod +x ~/frescoSoft/FRESCO/commandRunningDisulfideDesign` and

   `~/frescoSoft/FRESCO/commandRunningDisulfideDesign` to generate a todolist file. Inspect the todolist: `less todolist`.

6. Start the calculations with `./todolist&`. Verify with `top` that YASARA started. The calculations may take several hours to finish on a desktop computer.

7. Type `tail LOG` to verify no errors were encountered. Type

   `ls disulfideBonds_1NW8_cleaned__1NW8_MoreThan<X>AngstromFrom<avoidResidue>/pdb | wc -l`

   This command counts the number of pdb files in the directory with disulfide bonds. For the LEH example, this should be 27 once the calculation has finished. This includes multiple conformations of the same disulfide bond.

8. Use the appropriate script to create an overview of all the disulfide bonds (`~/frescoSoft/FRESCO/OverviewDisulfides`) and inspect the result (`less BestEnergyUniqueDisulfideBonds.tab`). The UniqueDisulfides should now contain the pdb files of the disulfide bonds structures with the best energy, as well as their templates.

**Selecting computationally designed variants for MD simulation**

In this section, mutations that are predicted to be stabilizing by FoldX and Rosetta are identified. The predicted 3D structures of the mutants, and those of the disulfide bond mutants, are collected to carry out MD simulations (see below).

1. Go to the all_designs directory and copy the UniqueDisulfides folder to designsMD (`cp -r UniqueDisulfides ../designsMD/`). Also copy the list with structures there:

   `cp BestEnergyUniqueDisulfideBonds.tab ../designsMD`
2. Go to the Rosetta folder (cd ../../rosetta). Select all mutations that are predicted to have a more than 5 kJ mol\(^{-1}\) improvement of \(\Delta \Delta \text{G}^\text{predicted}\): 

```shell
~/frescoSoft/FRESCO/DistributeRosettaAddG Phase2 ../1NWW_MoreThan5AngstromFromHPN.tab 2 A 5 B 150 1NWW_forRosetta 4000 -5
```

Use the resulting command line output to verify that indeed all targeted mutations were screened.

3. In the FoldX folder (cd ../foldx), type:

```shell
~/frescoSoft/FRESCO/DistributeFoldX Phase2 1NWW_cleaned 2 A B ../1NWW_MoreThan5AngstromFromHPN.tab 1000 -5
```

Again, verify that the planned number of mutations has indeed been screened.

4. Make a list of all the mutations that are predicted to be stabilizing, by either FoldX or Rosetta, by entering (using Tab-completion):

```shell
cat ../rosetta/MutationsEnergies_BelowCutOff.tab > list_SelectedMutations.tab && tail -n +2 MutationsEnergies_BelowCutOff.tab >> list_SelectedMutations.tab
```

5. Before doing MD simulations, re-add the water molecules (and possibly the cofactors) to the pdb files of the designs. Do this by running the HydrateDesigns script:

```shell
yasara -txt ~/frescoSoft/FRESCO/HydrateDesigns.mcr > log_conversion &
```

A few lines of this short script need to be altered if any other protein than LEH is targeted, as indicated in the script itself.

6. Look at the resulting directory (ls -rlt NamedPdbFiles/). Verify that there are indeed pdb files in the generated subdirectories. With `top` and `tail -f log_conversion` one can check whether YASARA has already finished.

7. For a selected target protein, use YASARA to open one of the pdb files with waters added to verify that the structure is realistic, e.g. with all cofactors present\(^g\).

8. Once finished, copy the subdirectory with pdb files of the hydrated structures to the designsMD folder (cp -r NamedPdbFiles ../designsMD/;)

---

\(^f\) The cutoff of \(-5 \text{ kJ mol}^{-1}\) can be made less strict (e.g. \(-2.5 \text{ kJ mol}^{-1}\)) to increase the number of stabilizing mutations that can be discovered.

\(^g\) The HydrateDesigns.mcr script ought to put cofactors back together with the crystallographic water molecules. This script has been tested for several cofactors but may fail for others. If cofactors or covalent bonds are missing, or other errors occur, the user needs to adapt the HydrateDesigns.mcr script\(^c\) and rerun it.
there should not be a `/ behind NamedPdbFiles) as well as the list of selected mutations (cp list_SelectedMutations.tab ../designsMD/).

**MD simulations of mutants**

For each of the mutants, MD simulations are carried out. This is done to predict their flexibility.

1. Go to the MD directory `cd ~/frescoLEH/designsMD`. One should see (`ls`) two directories named UniqueDisulfides and NamedPdbFiles.

2. Run the script to set up the MD simulations `~/frescoSoft/FRESCO/commandRunningMDsimulations` (if one is targeting another protein than the LEH example, this file should be modified using a text editor according to the instructions in the file itself). After that, run the resulting todolist `./todolist&`. Verify that YASARA is running with `top –o cpu`.

3. For the LEH example, it will probably take several hours for this step to finish, as only a few selected designs will be subjected to an MD simulation. For any other protein than LEH, after a few MD simulations have finished on a desktop computer do a visual inspection (see below) and verify there are still no problems with the protein structure (such as missing cofactors). After careful inspection of a few structures, the remaining MD simulations can be done without risking to waste a large amount of CPU time.

4. Also determine how much time it takes for the MD simulations of a single mutant to finish (`ls -rlt */*/yob`). If the pace of MD simulations is too slow for all selected variants to finish in a reasonable time, obtain an account at a computer cluster and carry out the calculations there.

**Visual inspection**

Those mutations that are computationally predicted to be stabilizing will often have one or more identifiable biophysical errors due to simplifications in the energy functions and incomplete conformational sampling.\(^7,17\) With visual

\(^h\) To log in to a cluster, `ssh` can be used after one obtains a user name. Only YASARA will need to be installed at the cluster. The most useful command to transfer a large number of files to and from a computer (cluster) is `rsync -avu <origin> <destiny>`. To start the calculations, cluster specific scripts will be needed that can normally be obtained via the cluster’s website.
inspection, such variants are eliminated\textsuperscript{1}. This further improves the quality of the library that will be screened experimentally, and thus reduces the screening effort that is required.

1. Copy the YASARA plugin file in the appropriate folder:
   \texttt{cp ~/frescoSoft/FRESCO/MutantInspectPlugin.py ~/frescoSoft/YASARA.app/yasara/plg/}

2. Enter the MD directory (\texttt{cd ~/frescoLEH/designsMD}) and run YASARA with \texttt{yasara 1NWW\_cleaned.pdb&}. In the menu bar, go to Analyze>FRESCO>Prepare Excel file from Mutations list. Wait until the file is created, open it and copy/paste the text into a blank sheet of your favorite spreadsheet application. To this list of mutations, the user should add his or her own observations and a final judgment whether to keep or discard the mutation. Start the visual inspection in YASARA by clicking Analyze>FRESCO>Start Inspection of Mutants. The plugin will load the mutations showing the static structures of wild-type and mutant in a panel called main and the MD simulations of mutant and wild-type in two other panels\textsuperscript{j}.

3. Optionally, set YASARA to stereo vision (\texttt{Stereo CrossEyed} or \texttt{Stereo Parallel}).\textsuperscript{k}

   Carry out visual inspections for all mutations in the sequence of step 4 to 10\textsuperscript{l}. Variants can be eliminated as soon as they fail an inspection step. Usually, about 40–50\% of the mutations will be eliminated both during inspection for biophysical credibility (step 4 to 6) and during inspection for conserved

\textsuperscript{1} Visual inspection is a standard step in computational design and molecular modeling and is therefore often not mentioned in the materials and methods sections of publications.

\textsuperscript{j} The plugin automatically finds the files used for the visual inspection but it requires the above provided standard file and folder names to function properly. In a directory called designsMD there have to be two subdirectories: NamedPdbFiles and UniqueDisulfides. The pdb and yob files in subdirectories of these directories should bear a name of the type <anything>\_cleaned<name of the mutations><furtherExtensions>.

\textsuperscript{k} Cross-eyed or parallel stereo needs to be learned by the user. This will take a few hours. See the YASARA website for the available other forms of stereo. Some users prefer to manually rotate the structures for 3D depth perception.

\textsuperscript{l} Experienced protein designers can inspect more than 120 variants per day while beginners should aim at 30-50 variants per day. The fastest method for inspecting is to follow the described sequence of steps, in which mutants are initially eliminated based on common and fast to analyze problems.
rigidity (step 7 to 10). Thus, about 25–35% of the mutations usually survive the visual inspection. Some information for practicing visual inspection skills on example mutations of LEH are provided\textsuperscript{m}.

Figure 3. Examples of the most common structural errors encountered amongst top-ranked point mutations. Both mutations belong to the example set of LEH\textsuperscript{m}. The visualization is as provided by the YASARA FRESCO plugin. A) Introduction of a hydrophobic residue that is solvent exposed. F48 is surrounded by water molecules. B) A mutation that results in an unsatisfied H-bond donor (the backbone amide) and an unsatisfied H-bond acceptor (the water oxygen). In the native structure, S12 makes an H-bond to the backbone amide while there is room for an additional water molecule to make an H-bond to the now unsatisfied water.

4. Eliminate mutations that result in unusual solvent exposure of hydrophobic side chains. Inspect the structure of wild type and mutant around the mutated residue to see whether the introduced (hydrophobic) side chain atoms becomes unusually water exposed (Figure 3A). The inspection can be done both for the static structure and for the structures from the MD simulation. Visually inspect how many water molecules can contact hydrophobic atoms in the side-chain and evaluate whether this is still normal. In case of doubt, make a comparison by looking at the same type of residue elsewhere in the enzyme (e.g., for phenylalanines, type \texttt{ShowRes Phe, ColorAtom Res Phe element C, red, ShowRes res with distance < 5 from res Phe}). For trained eyes, the identification of this common problem is very fast, leading to elimination of mutants within seconds.

\textsuperscript{m} For the LEH example, Q7M, E68L, A48F, and S111M introduce highly surface-exposed hydrophobic side chains. Mutations S12M, T22D, and G129S introduce unsatisfied H-bond donors or acceptors while E49P, Y96W, and R9P cause local flexibility, which is larger than that of the template structure. All other variants, both those that solve structural problems (T85V) and those that merely lack clear biophysical errors (E45K, E124D), should be selected for experimental testing.
5. *If the number of unsatisfied H-bond donors/acceptors increases due to the mutation, eliminate the mutant.* This is the second most common reason for elimination. Count the number of unsatisfied H-bond donors and acceptors around the mutation (Figures 3B, 4). H-bond acceptors and donors that are only involved in one three-centered H-bond interaction (Figure 4C) are counted as half unsaturated.18

![Figure 4](image)

*Figure 4.* Schematic examples of saturated, unsaturated, and partially unsaturated H-bond networks. A) All H-bond donors and acceptors are saturated. B) The hydroxyl oxygen, a good H-bond acceptor, is unsaturated. C) Both carbonyl oxygens are half-unsaturated since they share a single H-bond donor, forming a 3-centre H-bond.

6. *Verify that the mutations do not violate other biophysical criteria.* With most proteins, for one or a few positions almost all substitutions are predicted to be stabilizing, which probably reflects a systematic error in the energy calculations. In such cases, only accept the mutations if the wild-type protein features structural problems (unsatisfied H-bonds, cavities) that are repaired by the proposed mutations. Further, no prolines should be introduced in an α-helix. In case of a disulfide bond mutation, eliminate the proposed mutations if these create a large cavity in the protein interior.

7. *Make the most different MD structures invisible for both wild type and mutant.* It is often found that one of the MD simulations samples a different conformation than all the others (Figure 5) and thus behaves as an outlier. If the results of these MD simulations were evaluated in an identical manner, the differences between mutant and wild type would be randomly

---

18 The algorithms used in molecular modeling are poor at assessing whether an H-bond is made. They typically use some kind of distance cutoff and use surface accessibility to predict whether water can form an H-bond. For this reason, also distrust the H-bonds as displayed by YASARA. There could be additional H-bonds that are not visualized. Visual inspection is needed to eliminate cases where the computer overestimates the feasibility of water H-bonds or fails to identify the three-center H-bonds, which are energetically unfavorable.

18 The calculated energy of the wild-type structure is subtracted from those of the mutants to predict ΔΔGFold. If almost all mutations are predicted to be stabilizing at a particular position, this suggests an error in the energy calculation of the wild-type structure.
exaggerated. To prevent this, always remove the most different structures. Click on the visibility button in the HUD display one by one for the structures while watching the screen. The picture will change most when hiding the most different structure.

**Figure 5.** Example of identifying an outlier. From the averaged structures of 5 independent MD simulations, the structure that differs most from the other 4 is removed. For clarity, only the part of the protein with the largest differences is shown.

8. *Eliminate the mutant if the introduced side chain is unusually flexible.* Flexibility depends on the nature of the side chain. High flexibility would, for example, be normal for a lysine but not for a tryptophan. When in doubt, compare with similar wild-type residues (for example, type `ShowRes Trp`).

9. *Eliminate the mutation if the backbone at the mutation site, or in the flanking regions, becomes significantly more flexible* (Figure 6).

10. *Eliminate the mutant if the overall structure becomes significantly more flexible.* This rarely occurs by introduction of single point mutations.

**Figure 6.** Example of a mutation that is predicted to increase local backbone flexibility. R9P is one of the LEH example mutations. Parts of the backbone that show significant increase of flexibility are marked with red. MD-averaged structures (see caption Figure 5) of the wild-type are shown in sea green while the corresponding structures of the mutant are shown in orange. The mutated residue is in magenta.
Experimental verification of the selected variants

The variants that survive visual inspection should be screened experimentally. The protocols for genetic engineering and thermostability assays are widely used and are therefore only briefly summarized here. Genetic engineering can be done rapidly and inexpensively using 15 µL scale QuikChange reactions (Agilent Technologies) in 96-well plates. The reactions should be very reliable, we find mostly only a single clone needs to be sequenced. The $T_m$ of the variants can be determined with the Thermofluor method after a small-scale purification (from 1-5 mL of culture). The mutants with improved thermostability should also be tested for preserved catalytic activity. Additional details have been described elsewhere.6-7

Combination of stabilizing mutations to a hyperstable final variant

1. Enter the finalCombinations directory: cd frescoLEH/finalCombinations.
2. Combine all compatible stabilizing mutations that do not decrease catalytic activity. Predict the structure of the proposed final variant(s) using the ~/frescoSoft/FRESCO/CombineMutations.mcr. This script contains instructions for how to generate a table file listing the mutations that should be combined. The protein structure as generated in the first section of this manual should be used as a template.
3. The generated pdb file(s), which already contain(s) the crystal waters, should then be used as starting point(s) for MD simulation as described above.
4. The resulting structures should be inspected as described above.
5. If the combination fails these inspection steps, identify possible (combinations of) mutations that cause problems and repeat step 2 to 4 while omitting these mutations.
6. Prepare the final variant(s) using consecutive QuikChange reactions. Determine the $T_m$ of all intermediate mutants as well. This allows experimental identification of incompatible mutations.

Acknowledgements

This research was supported by the European Union seventh framework project Kyrobio (KBBE-2011-5, 289646), by the European Union Horizon 2020 program (project LEIT-BIO-2014-1, 635734) by NWO (Netherlands Organization for Scientific Research) through an ECHO grant, and by the Dutch Ministry of Economic Affairs through BE-Basic (www.be-basic.org).
References

4 Eijsink, VG; Bjørk, A; Gåseidnes, S; Sirevåg, R; Synstad, B; van den Burg, B; Vriend, G. Rational Engineering of Enzyme Stability. *J. Biotechnol.* 2004 (113) 105-120.
6 Floor, RJ; Wijma, HJ; Colpa, DI; Ramos-Silva, A; Jekel, PA; Szymanski, W; Feringa, BL; Marrink, SJ; Janssen, DB. Computational Library Design for Increasing Haloalkane Dehalogenase Stability. *ChemBioChem* 2014 (15) 1660-72.
8 Wu, B; Wijma, HJ; Song, L; Rozeboom, HT; Poloni, C; Tian, Y; Arif, MI; Nuijens, T; Quaedflieg, PJ; Szymanski, W. Versatile Peptide C-Terminal Functionalization Via a Computationally Engineered Peptide Amidase. *ACS Catal.* 2016 (6) 5405-5414.
12 Wijma, HJ. In Silico Screening of Enzyme Variants by Molecular Dynamics Simulation. In: *Understanding Enzymes*; Pan Stanford; 2016; pp 829-858.
16 van Beek, HL; Wijma, HJ; Fromont, L; Janssen, DB; Fraaije, MW. Stabilization of Cyclohexanone Monooxygenase by a Computationally Designed Disulfide Bond Spanning Only One Residue. *FEBS Open Bio* 2014 (4) 168-174.
18 Feldblum, ES; Arkin, IT. Strength of a Bifurcated H Bond. *Proceedings of the National Academy of Sciences* 2014 (111) 4085-4090.