Description & User Manual

Version 3.0

INDEX

What is FoldX?

Summary.

We have developed a computer algorithm, FoldX to provide a fast and quantitative estimation of the importance of the interactions contributing to the stability of proteins and protein complexes. The predictive power of FOLDEF has been tested on a very large set of point mutants (1088 mutants) spanning most of the structural environments found in proteins (see Figure 1). FoldX uses a full atomic description of the structure of the proteins. The different energy terms taken into account in FoldX have been weighted using empirical data obtained from protein engineering experiments.

The present energy function uses a minimum of computational resources and can therefore easily be used in protein design algorithms, and in the field of protein structure and folding pathways prediction where one requires a fast and accurate energy function. The FoldX is available via a web-interface at http://foldx.crg.es/.

The force Field

The FoldX energy function includes terms that have been found to be important for protein stability. The free energy of unfolding (ΔG) of a target protein is calculated using equation (1):

$$\Delta G = W_{vdw} \cdot \Delta G_{vdw} + W_{solvH} \cdot \Delta G_{solvH} + W_{solvP} \cdot \Delta G_{solvP} + \Delta G_{wb} + \Delta G_{hbond} + \Delta G_{el} + \Delta G_{Kon} + W_{mc} \cdot T \cdot \Delta S_{mc} + W_{sc} \cdot T \cdot \Delta S_{sc} \quad (1)$$

where ΔG_{vdw} is the sum of the van der Waals contributions of all atoms with respect to the same interactions with the solvent. ΔG_{solvH} and ΔG_{solvP} are the differences in solvation energy for apolar and polar groups respectively when these change from the unfolded to the folded state. ΔG_{hbond} is the free energy difference between the formation of an intra-molecular hydrogen bond compared to inter-molecular hydrogen-bond formation (with solvent). ΔG_{wb} is the extra stabilising free energy provided by a water molecule making more than one hydrogen bond to the protein (water bridges) that cannot be taken into account with non-explicit solvent approximations{Petukhov, 1999 #938}. ΔG_{el} is the electrostatic contribution of charged groups, including the helix dipole. ΔS_{mc} is the entropy cost of fixing the backbone in the folded state; this term is dependent on the intrinsic

tendency of a particular amino acid to adopt certain dihedral angles{Munoz, 1994 #939}. Finally ΔS_{sc} is the entropic cost of fixing a side chain in a particular conformation {Abagyan, 1994 #940}.

When we are working with oligomeric proteins or protein complexes, two extra terms are involved; ΔG_{kon}, which reflects the effect of electrostatic interactions on the association constant k_{on} (this applies only to the subunit binding energies){Vijayakumar, 1998 #941} and ΔS_{tr}, which is the loss of translational and rotational entropy that ensues on formation of the complex. The latter term cancels out when we are looking at the effect of point mutations on complexes.

The energy values of ΔG_{vdw}, ΔG_{solvH}, ΔG_{solvP} and ΔG_{hbond} attributed to each atom type have been derived from a set of experimental data, and ΔS_{mc} and ΔS_{mc} have been taken from theoretical estimates. The terms W_{vdw}, W_{solvH}, W_{solvP}, W_{mc} and W_{sc} correspond to the weighting factors applied to the raw energy terms. They are all 1, except for the van der Waals' contribution which is 0.33 (the van der Waals' contributions are derived from vapor to water energy transfer, while in the protein we are going from solvent to protein).

For a detailed explanation of the FoldX force field, see{Schymkowitz, 2005 #953; Schymkowitz, 2005 #962} and the FoldX webserver.

Petukhov M, Cregut D, Soares CM, Serrano L (1999) Local water bridges and protein conformational stability. Protein Sci 8:1982–1989

Munoz V, Serrano L (1994) Intrinsic secondary structure propensities of the amino acids, using statistical phi-psi matrices: comparison with experimental scales. Proteins 20:301–311

Abagyan R, Totrov M (1994) Biased probability Monte Carlo conformational searches and electrostatic calculations for peptides and proteins. J Mol Biol 235:983–1002

Vijayakumar M, Wong KY, Schreiber G, Fersht AR, Szabo A, Zhou HX (1998) Electrostatic enhancement of diffusioncontrolled protein-protein association: comparison of theory and experiment on barnase and barstar. J Mol Biol 278: 1015–1024

Schymkowitz J, Borg J, Stricher F, Nys R, Rousseau F, Serrano L (2005) The FoldX web server: an online force field. Nucleic Acids Res 33:W382–W388.

Schymkowitz, J. W. et al. Prediction of water and metal binding sites and their affinities by using the Fold-X force field. Proc Natl Acad Sci U S A 102, 10147-52 (2005).

Mutating Protein, DNA

FoldX can mutate the 20 natural aminocids, the phosphorylated versions of Ser, Thr and Tyr, the sulfated version of Try, methylated Lys, hydroxyl Proline and the standard 4 DNA bases, methylated adenosine and guanosine.

When mutating DNA the code automatically identifies the pairbase of the selected base and mutates both of them simultaneously to keep the base pairing. In the case of DNA we do not use rotamers but we allow the base to move with respect to the ribose.

FoldX consortium

FoldX was developped by the FoldX consortium centered around the laboratory of Luis Serrano at the European Molecular Biology Laboratory in Heidelberg. The following people have contributed to the development to different extents:

- Raphael Guerois,
- Jens Nielsen,
- Jesper Borg,
- Joost Schymkowitz
- Frederic Rousseau,
- Francois Stricher,
- Luis Serrano

The current FoldX release is a complete rewrite of the original code by Raphael Guerois, the new code is optimised for speed and applicability and some changes have been made in the energy calculations, but in philosophy this release is identical to the earlier code.

Intellectual property

Intellectual property issues up to version 2.6 are handled by EMBLEM. Only executables are available for download from our website, based on a material transfer agreement (mta) with EMBLEM. The source code is protected. Version 3.0 is handled by the CRG technology transfer.

Description of the latest additions and modifications in version 3.0

Francois Stricher, Tom Lenaerts, Joost Schymkowitz, Frederic Rousseau and Luis Serrano (2008). FoldX 3.0. "In preparation".

References

When referring to FoldX, please reference the following papers:

Guerois, R., J. E. Nielsen, et al. (2002). "Predicting changes in the stability of proteins and protein complexes: a study of more than 1000 mutations." J Mol Biol 320(2): 369-87.

Schymkowitz, J. W. et al. Prediction of water and metal binding sites and their affinities by using the Fold-X force field. Proc Natl Acad Sci U S A 102, 10147-52 (2005).

Other references where FoldX has been used are:

Kiel, C., L. Serrano, et al. (2004). "A detailed thermodynamic analysis of ras/effector complex interfaces." J Mol Biol 340(5): 1039-58.

Wohlgemuth, S. et al. Recognizing and defining true Ras binding domains I: biochemical analysis. J Mol Biol 348, 741-58 (2005).

Kiel, C. et al. Recognizing and defining true Ras binding domains II: in silico prediction based on homology modelling and energy calculations. J Mol Biol 348, 759-75 (2005)

Arnould, S., P. Chames, et al. (2006). "Engineering of large numbers of highly specific homing endonucleases that induce recombination on novel DNA targets." J Mol Biol **355**(3): 443-58

Schymkowitz, J. et al. The FoldX web server: an online force field. Nucleic Acids Res 33, W382-8 (2005)

Kempkens, O., E. Medina, et al. (2006). "Computer modelling in combination with in vitro studies reveals similar binding affinities of Drosophila Crumbs for the PDZ domains of Stardust and DmPar-6." Eur J Cell Biol.

Kiel, C. and L. Serrano (2006). "The ubiquitin domain superfold: structure-based sequence alignments and characterization of binding epitopes." J Mol Biol **355**(4): 821-44

Musi, V., B. Birdsall, et al. (2006). "New approaches to high-throughput structure characterization of SH3 complexes: the example of Myosin-3 and Myosin-5 SH3 domains from S. cerevisiae." Protein Sci **15**(4): 795-807.

van der Sloot, A.M., Tur, V., Szegezdi, E., Mullally, M.M., Cool, R.H., Samali, A., Serrano, L. and Quax, W.J. (2006) Designed tumor necrosis factor-related apoptosis-inducing ligand variants initiating apoptosis exclusively via the DR5 receptor. *Proc Natl Acad Sci U S A*, **103**, 8634-8639.

Kiel, C., Foglierini, M., Kuemmerer, N., Beltrao, P. & Serrano, L. A genome-wide Ras-effector interaction network. *J Mol Biol* **370**, 1020-32 (2007).

Kiel, C. & Serrano, L. Prediction of Ras-effector interactions using position energy matrices. *Bioinformatics* 23, 2226-30 (2007)

Kolsch, V., Seher, T., Fernandez-Ballester, G. J., Serrano, L. & Leptin, M. Control of Drosophila gastrulation by apical localization of adherens junctions and RhoGEF2. *Science* **315**, 384-6 (2007)

Pey, A. L., Stricher, F., Serrano, L. & Martinez, A. Predicted effects of missense mutations on native-state stability account for phenotypic outcome in phenylketonuria, a paradigm of misfolding diseases. *Am J Hum Genet* **81**, 1006-24 (2007)

Szczepek, M. et al. Structure-based redesign of the dimerization interface reduces the toxicity of zinc-finger nucleases. *Nat Biotechnol* **25**, 786-93 (2007)

Tokuriki, N., Stricher, F., Schymkowitz, J., Serrano, L. & Tawfik, D. S. The stability effects of protein mutations appear to be universally distributed. *J Mol Biol* **369**, 1318-32 (2007)

Tokuriki, N., F. Stricher, et al. (2008). "How protein stability and new functions trade off." *PLoS Comput Biol* **4**(2): e1000002.

Operation of the command line version

There are two general ways to run FoldX from the command line

1) Foldx -manual <pdb> <options> <commands>
2) Foldx -runfile <filename>

Option 1: –manual

The user specifies 1 pdbfile and associates 1 option file and 1 command file with it.

Option 2: –runfile

The user specifies 1 runfile, which contains 1 or more jobs. Each job can associate a list of pdbs with a command set and an option set.

XML-like format

In general the format is XML-like, i.e. labels surrounded by < > are followed by a value list (comma seperated) and terminated by a semicolon ";"

<label1>value1,value2,value3;

where there is no value specified # should replace the value list

<label1>#;

Option file

The option file looks like this:

```
<TITLE>FOLDX_optionfile;
<option1>value1;
<option2>value2;
 <option3>#;                    // default specifier
```

If an option label is not put in the list nor has value #, the default is set.

Command file

The command file looks like this:

<TITLE>FOLDX_commandfile;
//comment line
<command1>filename; // results written to filename
<command2>#; // results written to default file

When a command is omitted or commented out with // it is not executed.
When the value is #, the result from the command is written to a file called “command_pdbfile.txt”
When there is a value, the result from the command is written to the filename specified by value.

Run file

The runfile is a combination of command and option file and allows a crude form of scripting to FoldX. A run file looks like this:

```
<TITLE>FOLDX_runscript;
<JOBSTART>#;          // start of job
<PDBS>pdb1, pdb2;     // or # for no files
<BATCH>list1:         // if both PDBS and BATCH are specified, the combined list is used
<COMMANDS>#;          // start of commands
<command1>#;
…
<commandx>#;
<END>#;               // end of commands
<OPTIONS>#;           // start of options
<option1>
…
<option2>
<END>#;               // end of options
<JOBEND>#;            // end of job
other jobs…
<ENDFILE>#;           // end of file
```

Between the labels JOBSTART and JOBEND must be the description of a job, this contains (in this order):

- PDBS: a comma separated list of pdb files to analyse
- BATCH: a comma separated list of batch files: these are text files that contain names of pdb files (1 per line).
- Between COMMANDS and END: the list of commands, as in a command file, except that now if a result file is specified, the results for that command for all pdbs in the job are written to this file, otherwise individual files for each job are created.
- Between OPTIONS and END: the list of options, as in an option file

Several jobs can be specified in a single run file.

The file needs to be terminated by the label ENDFILE.

Pdbs specified in PDBS and via files using BATCH can be combined – 1 big list will be built.

Options

Parameter	Default	Description
Physico-chemical		
<Temperature>	298	Temperature (K)
<R>	$1.9859.10^{-3}$	The gas constant ($kcal.mol^{-1}.K^{-1}$)
<IonStrength>	0.05	Ionic strength of the solution (M)
<pH>	7.0	At the moment only used for pH effects on metal binding. Will not affect the protonation of charged groups in this version
Superimposition (works with <rmsd> command		
<rmsd_pdb>	true	If set to true you get the pdb output of the fitting protein, if set to false you just get the rmsd value.
<fit_atoms>:	BB	by default set to BB. This option represents the atoms involved in the fitting function: **BB** (or **#**) for backbone atoms, **CA** for Calpha, CB for Cbeta, **CA_CB** for both Calpha and Cbeta **N_CA_O** for N, Calpha and O (three of the four backbone atoms).
Energies.		
<VdWDesign>	2	When set to2, it considers rotamer penalizations due to internal clashes, maximum penalization for interresidue VanderWaals' clashes, a ceiling for the

		VanderWaals' clashes between two atoms of 5 Kcal/mol and strict H-bond geometry. When set to zero there is a weak rotamer penalization, there is a ceiling for the VanderWaals' clashes of 1 Kcal/mol and we use a relaxed H-bond geometry. This option should be set to 0, when doing an Ala scanning mutagenesis of a protein. Whenever doing design, or repairing a PDB, it should be set to 2.
Complexes.		
<complex_with_DNA>	false	When set to true it considers only two molecules in a protein-DNA complex, DNA + protein, considering that the two chains of DNA is a single molecule, and the same if there is more than one protein chain. This option should be used when determining the binding energy between DNA and a protein.
<full_main_entropy>	false	When set to true, it assigns the full main chain entropy cost for each residue, without correction by accessibility. This should be used when trying to build position matrices for a poly-Ala ligand.
<repair_Interface>	ALL	If set to ALL all residues in a complex are optimized when running the RepairPDB command. When set to NONE it will not optimize or move the residues at the interface of a complex. When set to ONLY then only the residues at the interface will be optimized. This option could be used to ONLY when working on a big complex and been interested only at the binding energy.

Mutagenesis.		
<fixed_cbeta>	true	This option forces the new side chain on top of the crystal CB. In this way we solve a typical problem when trying to place side chains that is that the X-ray CB and the standard rotamer CB do not coincide.
<do_cloud>	false	This option makes a file (cloud.pdb) that stores all rotamers within 1 Kcal/mol of the best rotamer. It should be used together with a command file that mutates a residue. It is automatically set to true with the commandline: <Do_Rotamer_cloud>.
<moveNeighbours>	true	This option when set to false prevents the neighbours of a residue targeted for mutation to move. It should not be used normally.
<numberOfRuns>	1	This option is used in combination with the command <BuildModel>. It tells the algorithm how many times it should do the specified mutations. Normally it should be set to 1. However, sometimes it can be set to higher numbers (typically 5) to see if the algorithm has achieved convergence, or in other words if the solution offered is the optimal or a trapped solution.
Water Bridges and Metals		
<water>	-IGNORE	If -CRYSTAL uses the X-ray waters bridging two protein atoms. If -PREDICT, waters that make 2 or more hydrogen bonds to the protein are predicted. If -COMPARE it compares the predicted water bridges with the X-ray ones.
<metal>	-IGNORE	If -CRYSTAL uses the X-ray waters bridging two

		protein atoms. If -PREDICT, binding metals defined in the command **<metalbinding_analysis>** are predicted. If OPTIMIZE it optimizes the position of the metals. If -COMPARE it compares the predicted metals with the X-ray ones.
<num_water_partners>	2	Minimum number of partners to consider a water bridge
Other Options		
<no_Cterm >	false	When set to true it eliminates the charge of the C-terminus group.
<no_Nterm>	false	When set to true it eliminates the charge of the N-terminus group.
Output		
<OutPDB>	true	When set to true it it ouputs the PDBs of the mutants or of the selected chains in a complex.
<pdb_hydrogens>	false	If true, hydrogen atoms are included in the pdb output
<pdb_waters>	false	If true, predicted waters are included in the pdb output
<pdb_dummys>	false	If true, dummy positions used for mimic the helix dipole effect, the free orbitals of acceptor groups and the H-bond acceptor nature of the center of aromatic rings are included.
<overwriteBatch>	true	If true, file outputs will overwrite files with the same name.

Commands

VERY IMPORTANT: YOU SHOULD ALWAYS HAVE IN THE SAME FOLDER WHERE YOU ARE RUNNING THE CODE A FILE CALLED rotabase.txt THAT IS DOWNLOADED WITH THE CODE

All commands are encapsulated by the < and > signs. They should always finish with a semicolon; In the following lines Output_file indicates the name of the file you want to contain the output of the selected command (ie results_dG). If you do not want to use a name but just the name of the pdb you are running type # instead after the command.

ie <COMMAND>Output_file; or <COMMAND>#;

.

Repair PDB structures

<Optimize>This function does a quick optimization of the structure, moving all side chains slightly to eliminate small VanderWaals' clashes.

Example: < Optimize >Output_file;

<RepairPDB> Here we identify those residues which have bad torsion angles, or VanderWaals' clashes, or total energy, and we repair them.
The way it operates is the following:

a) First it looks for all Asn, Gln and His residues and flips them by 180 degrees. This is done to prevent incorrect rotamer assignment in the structure due to the fact that the electron density of Asn and Gln carboxamide groups is almost symmetrical and the correct placement can only be discerned by calculating the interactions with the surrounding atoms. The same applies to His.
b) We do a small optimization of the side chains to eliminate small VanderWaals' clashes. This way we will prevent moving side chains in the final step.

c) We identify the residues that have very bad energies and we mutate them and their neighbours to themselves exploring different rotamer combinations to find new energy minima.

This command admits several specifications:

<RepairPDB>Output_file,`Mutate: XXX,YYY;`

<RepairPDB>Output_file,`Fixed: XXX,YYY;`

Where XXX is a list of residues with the standard FoldX format i.e. LC43 (residue, chain, number). You can put as many as you want as long as you put a comma as a spacer.

Residues put after Mutate will be mutated to themselves before the repair of the PDB. This is normally done to eliminate distorted residues which can be obtained sometimes after molecular dynamics simulation. Residues after Fixed: will not be moved while doing the repair of the PDB.

Water-Bridge analysis

<crystalwater_analysis> This command reads a PDB and outputs a file called with the name given in the command (i.e <crystalwater_analysis>waters.pdb;); IMPORTANT NEVER GIVE THE SAME NAME OF THE PDB YOU ARE READING OR YOU WILL OVERWRITE IT) in which the crystallographic water bridges (those waters making 2 or more bonds to the protein) are printed (see Option <num_water_partners> to decide the threshold). It will also produce an extra file with the name of the first pdb followed by "Compare_water.txt", in which a comparison between the predicted and crystal water bridges is made. In this file we use FAILED when the predicted water bridge does not have a crystal counterpart (please think that this is OK, since we do not consider neighbour crystal units) and PREDICTED when the distance is less than 1 Å (the first number is the distance). At the end of the file the ratio between predicted water bridges with a counterpart in the X-ray and total number of X-ray water bridges is shown.

Example

<crystalwater_analysis>Output_file;

Metalbinding analysis

<metalbinding_analysis> This commands is used to predict the binding energy of different metals to proteins and DNA. It is used with together with the option <metal>.

Example

< **metalbinding_analysis** >Output_file,MetalName; Where MetalName is the name of the metal we want to predict, i.e. CA for calcium.

Depending of the option qualifier <metal>, it could determine the interaction energy of crystal metals (<metal>CRYSTAL;), optimize the crystal metals and then predict the binding energy(<metal>OPTIMIZE;), or *de novo* predict binding sites and interaction energies of the selected ion (<metal>PREDICT;).

Outputs:

All <metal>options will produce an output file with the name of the file giving in the command (Output_file) plus some other files specific for each option.

PREDICT

This option will produce an output if a high affinity metal binding site is predicted.

Output_file The file will contain lines with the pdb name and the identity of the metal. In case that the predicted metal is close to an existing crystal one it will also print the coordinates of the crystal metal, the sum of the interactions with the protein, the coordinates of the predicted metal and the sum of its interactions with the protein and the decomposed binding energy of the predicted metal;

"predict_output.txt" This file will contain lines with the pdb name, the identity of the metal and the binding energy.

OPTIMIZE

Output_file The file will contain lines with the pdb name, the identity of the metal, the coordinates of the crystal metal, the sum of the interactions with the protein, the coordinates of the optimized metal and the sum of its interactions with the protein and the decomposed binding energy of the optimized metal;

CRYSTAL

Output_file The file will contain lines with the pdb name, the identity of the metal and the decomposed binding energy in kcal/mol (positive numbers means high affinity).

Superimposition

<rmsd> It works like this: you always compare/fit proteins to the "original one" loaded in FoldX. The "fitting proteins" are given by the arguments of the <rmsd> command. Moreover you can precise which residues you want to use for the fit. One important thing is that the program can only fit two sets of residues of the same length.

Examples

<rmsd>#,<PDBS>,toto.pdb, tata.pdb**;** in this case, you will fit the fit_atoms of toto and tata on the "original pdb". For this you will use all residues, so the proteins should be of the same size.

If not, or if you want to fit on a chain or on a small set of residues, you have to use a batch file to provide all the details:

<rmsd>#,<BATCH>, rmsd_batch.txt**;** rmsd_batch.txt is, of course, the name of the batch file.

The batch file should be like this:

<TITLE>FOLDX_rmsd_batch_file;
<ORIGIN>RES,DA14,QA16,KA18;
<EXTRACT_CHAIN>C; //This one is optional
<CUT_PASTE_ORIGIN> RES,DA14,EA16; //This one is optional
<CUT_PASTE_TARGET>toto.pdb, RES,EB15,KB17;//This is optional but it should be here if you define <CUT_PASTE_ORIGIN>
<CUT_PASTE_TARGET>tata.pdb, RES,EA15,KA17;//This is optional but it should be here if you define <CUT_PASTE_ORIGIN>
<PDB>toto.pdb,RES,DB14,QB16,KB18;
<PDB>tata.pdb,RES,EA14,MA16,KA18;

The <EXTRACT_CHAIN> line indicates that you want to extract a chain of the molecule you are using to superimpose your target and copy it on the superimposed target.

The <CUT_PASTE_A> line indicates that the user wants to cut a peptide segment of defined length and after superimposition replace the equivalent segment of the other protein with it. This is useful when you want to make chimaeras. FoldX will examine the RMSD of the joining ends and give a warning if the joining is not possible. The format is as shown above. Where RESA and RESB are tab markers to tell the code the residues at the beginning and at the end of the selected fragments. You could use fragments of different size, but the beginning and end of the fragments should superimpose with less than 0.3 Å RMSD.

If the chain is not defined in the PDB, then you can put a space, i.e. D 14,Q 16, K18. You can put as many residues as you want, but the minimum should be 3.

You can precise after each pdb name (or after <ORIGIN> for the "original pdb") on which residues you want to do the fit:

#; for all residues;
CHAIN,X; for chain X;
RES,AC13,DA14**;** for residues Alanine 13 of chain C and Aspartate 14 of chain A.

You have to give the same identifier for <ORIGIN> and <PDB>, i.e., if you put CHAIN for the "original pdb" you have to fit the pdbs with a CHAIN identifier also.

Stability Analysis

<Stability> Produces the Gibbs energy of folding decomposed into the different terms used by FOLD-X.

Example

<Stability>Output_file; Will produce a file with the name of the “Output_file” you have typed.

<EnergyDifference> Is intended for comparing the stability of mutant and wild type structures. The pdb files should be put in a list as follows:

Wt_a

_mutanta1

_mutanta2

wt_b

_mutantb1

_mutantb2

The underscore indicates mutants, they are compared with the last wild type that was encountered and the difference energy is output to a file.

Example

<EnergyDifference>Output_file; Will produce a file with the name of the "Output_file" you have typed.

<MetalBindingEnergy>

This option calculates the interaction energies of metals bound to the protein, as long as they are in the FOLD-X database. Please take care that the metal it is not coordinated to a molecule that is not yet recognized by FOLD-X.

Example

<MetalBindingEnergy>Output_file; Will produce a file with the name of the "Output_file" you have typed.

Output Commands

<SequenceOnly>

It prints the sequence information of the protein in a column format: AA, Chain, Pdb Number.

Example

<SequenceOnly>Output_file; Will produce a file with the name of the “Output_file” you have typed.

<Dihedrals>

Prints all dihedral angles of the molecule

Example

<Dihedrals>Output_file; Will produce a file with the name of the “Output_file” you have typed.

<SequenceDetail>

Prints all possible relevant information in terms of energetics and solvent accessibility at the amino acid level.

Example

<SequenceDetail>Output_file; Will produce a file with the name of the “Output_file” you have typed.

<QualityAssessment>

Checks that the dihedral angles are reasonable and produces a file with the ones which are not.

Example

<QualityAssessment>Output_file; Will produce a file with the name of the “Output_file” you have typed.

<PrintNetworks>

Prints all interaction networks and residue details of a protein into different files:

"H_bonds_"+ Output_file; H_bonds made by the protein atoms

"volumetric_bonds_"+ Output_file; Related to accesibility of the atoms

“charge_bonds_"+ Output_file; Electrostatic contributions of the different protein atoms.

“contact_bonds_"+ Output_file; Contacts made by the protein atoms.

"disulfide_bonds_"+ Output_file; Disulfide bonds if present

"partcov_bonds_"+ Output_file; Interactions made by protein atoms and metals

"water_bonds_"+ Output_file; Interactions made by protein atoms and water molecules

"distances_"+ Output_file; Distances between different protein atoms

Example

<PrintNetworks> Output_file;.

<PDBfile> Produces a FOLD-X pdb file.

Example

<PrintNetworks>Output_file; Will produce a file with the name of the "Output_file" you have typed.

<DNAContact> Prints the amino acids contacting the DNA and the nucleotides contacting the protein in a protein-DNA complex

Example

<DNAContact>Output_file; Will produce a file with the name of the "Output_file" you have typed.

Complex-related Commands

<AnalyseComplex>

It determines the interaction energy between 2 molecules or 2 groups of molecules. The way it operates is by unfolding the selected targets and determining the stability of the remaining molecules and then subtracting the sum of the individual energies from the global energy. In case there is a metal bound between the two molecules it will assign it to the one which makes the stronger interactions with the metal.

The user can select which side chain or group of side chains he/she wants to use to determine the interaction energy with the rest of the protein. The format to do this is

Example

<AnalyseComplex>#,AB; where # is the name of the output file (in this case the pdb name, or it can be specified by the user) and AB in this case means that we want to determine the interaction energy of chains AB of the pdb molecule with the rest of the molecule. The user can put from 0 to any number of letters. 0 means that by default the program will calculate all possible interactions between all molecules in the complex.

There is an option that can be used with this command: <complex_with_DNA>, When set to true, automatically will divide the PDB in 2 groups: DNA and Protein and will calculate the interaction between the two.

The output file contains the standard columns corresponding to the different energy terms (all of them reflecting changes in the respective energies upon binding) plus a column at the beginning showing intrachain clashes of the residues of the selected molecules or group of molecules. This term is important when designing protein complex interfaces since we could have a residue that has a very good interaction with the neighbour chain, but is in a very strained conformation with respect to its own chain. Thus that conformation is not realistic.

Another output file with the name of the pdb followed by "Interface_Residues.txt" will contain all the residues involved in the interface between the selected chains in a format that can be directly used later to do any kind of mutagenesis scanning (i.e. AC99,DC85; were the first letter is the amino acid or DNA base, the second letter is the chain and the number the PDB number).

<complex_alascan> This works the same way than <AnalyseComplex>, but in this case the program just mutates every residue on the interface of the selected molecules to Ala and produces the ΔΔG of the mutation.

Protein Mutagenesis

<BasicMutate> Performs fast mutations without moving the neighbour atoms. It should only be used for truncation mutations.

. Example

<BasicMutate>Output_file; Will produce a file with the name of the "Output_file" you have typed.

<BuildModel> This is the workhorse of FoldX mutation engine. This command ensures that whenever you are mutating a protein you always move the same neighbours in the WT and in the mutant producing for each mutant a corresponding PDB for its WT (each mutation will move different neighbours and therefore you need different WT references).
The format should always be **<BuildModel>** Output_file,**"File";**
Where File is a text file and has the following three flavours and names:

<BuildModel>Output_file, mutant_file.txt; The mutant_file.txt should be in the same folder we are running FoldX. It should contain the whole sequence (or part of it) of the WT sequence in one-letter code, followed by the sequences of the mutants in one letter code we want to make. The code will automatically identify the sequence in the pdb and mutate the residues that are different. You need to be careful to put a sequence long enough so that it does not find more than one target, unless you are working with homopolymers and you want to do the same mutations in each subunit.
Example of mutant_file.txt
GQETSYIEDNSNQN
GQETSAILWNSNAN
GRETSAILWNSNAN

<BuildModel>Output_file, individual_list.txt; In this case the file individual_file.txt contains the mutations we want to make separated by a comma, in the classical FoldX way (WT residue, chain, residue number, mutant residue).
Example of individual_list.txt
FA39L,FB39L;
LA41F,LB41F;
KA42I,KB42I;
GA46S,GB46S;

AA47V,AB47V;

LA48S,LB48S;

LA52S,LB52S;

<BuildModel>Output_file,DNA_scan; This specification of BuildModel will look for DNA bases and will mutate each base of the DNA and its pairbase to the other 4 bases while moving the neigbours.

- In the above three specifications of BuildModel it is possible to add an extra specification to prevent certain residues to move. The format should be like:

<BuildModel>Output_file, individual_list.txt, fixed_residues;
<BuildModel>Output_file,DNA_scan, fixed_residues;
<BuildModel>Output_file, mutant_file.txt, fixed_residues;

fixed_residues is a text file where the residues we do not want to move should be detailed as in the individual_list.txt
Example of fixed_residues
WA39,EA40,KA41;

The residues selected in this file will not be moved upon mutation of neighbour residues.

- BuildModel can be used in combination with the option: <numberOfRuns>. This option tells the algorithm how many times it should do the specified mutations. Normally it should be set to 1. However, sometimes it can be set to higher numbers (typically 5) to see if the algorithm has achieved convergence, or in other words if the solution offered is the optimal or a trapped solution. When it is larger than 1 the algorithm will do the same mutations but changing the rotamer set used and the order of the moves. In this way other alternative solutions could be explored. For most of the mutations this will not result in significant differences from one run to the other. However, in some cases, like mutations involving Arg neighbours with a large degree of freedom, significant differences could be found.

Output

-PDBs

Remember to activate the option OutPDB to true. You will have a PDB for the WT reference for each mutant and for the corresponding mutant numerated 0 to …..X.

For example if your PDB file is named 1aay.pdb, your output PDBS will be named:

WT_1aay_0.pdb for the WT reference and 1aay_0.pdb for the first mutant. For the second one it will be WT_1aay_1.pdb for the WT reference and 1aay_1.pdb, etc….

-File Outputs

Output_file. This file will have the name of the mutation, the standard deviation of the different runs results and the average energy of the different runs.

PdbList_" Output_file". In this file you will have the names of all the mutants you have made, as well as of the corresponding WT references.

Dif_"Output_file" will have the difference in energy between the WT reference and the corresponding mutant.

Raw_"Output_file" contains the full energy decomposition for the WT references and the mutants.

<PositionScan> mutates one amino acid to the other 20, or to selected ones and repairs the neighbour residues. The way it operates is: first it mutates the selected position to Ala and annotates all neighbour residues; then it mutates the WT residue to itself, and then the neighbours to themselves followed each time by the WT residue to itself. In this way we ensure that when we mutate we will not move any residue that has not been moved in the WT reference. Once this is done, the new WT reference is mutated at the selected position to the target amino acids. To prevent problems, neighbour side chains are only optimized when creating the WT reference after self-mutation, but not when making the individual mutants unless we select a new rotamer for the neighbour.

The format for specifying mutants is LC43a (residue, chain, number, mutation), were at mutation you can have:

a 20 amino acids

d 24 amino acids (Includes phosphorylated Tyr, Ser and Thr and hydroxyl Proline)

h {"GLY", "ALA", "LEU", "VAL", "ILE","THR", "SER", "CYS", "MET", "LYS", "TRP", "TYR", "PHE", "HIS"};

c {"ARG", "LYS", "GLU", "ASP", "HIS"};

p {"ARG", "LYS", "GLU", "ASP", "HIS", "TRP", "TYR","THR", "SER", "GLN", "ASN"};

n……4 bases, it mutates any base to the other three and itself.

or any amino acid in one letter-code, i.e. LC43G;

You can put as many as you want as long as you put a comma as a spacer.

Example

<PositionScan>Output_file,LC43a,KC55h,EC60p,WC70Y; In this case FoldX will independently mutate Leu43 of chain C to the 20 standard amino acids. Lys55 of chain C to the hydrophobic aminoacids. Glu 60 of chain C to the polar amino acids and Trp70 of chain C to Tyr.

The output will be a file for each position called:

`"energies_"+position+"_"+pdb_name.txt` where position is the number in the pdb sequence and pdb_name the name of the pdb you are mutating.

Inside this file you will have first the energy of the new WT reference before mutation followed by those of the mutants. You should not be surprised if you find that mutating a residue to itself result in better energy: this is because the neighbourhood of the position was not optimized.

If you are working with a complex that has 2 subunits, you will also get a file called:

`"binding_energies_"+position+"_"+protein_name.txt`

where you will have the interaction energies for the reference WT and the mutants.

- Important. If you are working with a polyalanine ligand for example that you have created by deleting the sidechains found in a crystal structure you can activate the option `<read_dihedrals>true;` Then the command will look for a file called Dihedrals. In this file you should have the rotamer angles of the original side chains. The code will read them and ensure that when an Ala residue is mutated to the original X-ray side chain the original rotamer is considered.

Example:

```
PL1001,      35.7697,-34.4197;
```

Where this is a Pro in chain L at pdb number 1001 with Chi1 and Chi2 dihedral angles of 35.77 and -34.42, respectively.

<**DNAScan**> This function mutates every DNA base to itself and the other three in a systematic fashion producing a PDB for each mutation.

It will also give you three outputs:

-File Outputs

ListDNAMutants_"Output_file". In this file you will have the names of all the mutants you have made, as well as those of the corresponding WT references.

Dif_"Output_file" This will be created if the pdb file is a complex between protein and DNA. Then it will write the difference in binding energy between the protein and DNA for the WT reference and the corresponding DNA mutants.

<alascan> Mutates every single protein residue to Ala without moving the neighbours and produces a file with the resulting energies.

Example: <alascan> Output_file;

Full Side Chain Reconstruction

<**ReconstructSideChains**>// reconstructs all the side chains of a PDB. First it mutates all non-Gly residues to Ala. Then it mutates the Ala residues to the WT sequence while moving all neighbour residues with bad energies.

Example: <ReconstructSideChains>Output_file;

-runfile example

The symbol // is used to comment the line.

<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list2.txt;
<COMMANDS>FOLDX_commandfile;
//<RepairPDB>#;
//<AnalyseComplex>out_complex.txt,AB;
//<Do_Rotamer_cloud>#;
//<PositionScan>#,GB236a,GB237a,PB238a;
//<PrintNetworks>#;
//<ReconstructSideChains>#;
//<BasicInfo>#;
<BuildModel>#,individual_list.txt;
//<DNAscan>#;
//<Stability>result_DG.txt;
//<ExtractResidueCloud>#;
//<crystalwater_analysis>waters.pdb;
//<SequenceOnly>#;;
//<SequenceDetail>#;
//<Dihedrals>#;
//<MetalBindingEnergy>results_ions.txt,H2O;
//<EnergyDifference>DDresults.txt;
//<Energy_sequence>enerseq.txt;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;

<IonStrength>0.050;

<water>-IGNORE;

<metal>-CRYSTAL;

<VdWDesign>2;

<OutPDB>false;

<numberOfRuns>2;

<END>#;

<JOBEND>#;

<ENDFILE>#;

Protein, DNA and metals recognized by FoldX.

Three letter code	One Letter code	Mutable	Description
Protein			
GLY	G	yes	standard aa
ALA	A	yes	standard aa
LEU	L	yes	standard aa
VAL	V	yes	standard aa
ILE	I	yes	standard aa
PRO	P	yes	standard aa
ARG	R	yes	standard aa
THR	T	yes	standard aa
SER	S	yes	standard aa
CYS	C	yes	standard aa
MET	M	yes	standard aa
LYS	K	yes	standard aa
GLU	E	yes	standard aa
GLN	Q	yes	standard aa
ASP	D	yes	standard aa
ASN	N	yes	standard aa
TRP	W	yes	standard aa
TYR	Y	yes	standard aa
PHE	F	yes	standard aa
HIS	H	yes	standard aa (it is considered to be 50% charged which could be the case when working around pH 7.0)
PTR	y	yes	Phoshoporylated Thr
TPO	p	yes	Phoshoporylated Tyr
SEP	s	yes	Phoshoporylated Ser
HYP	h	yes	HydroxiProline
TYS	z	yes	Sulfo-tyrosine
MLZ	k	yes	Monomethylated Lys
MLY	m	yes	Dimethylated Lys
M3L	l	yes	Trimethylated Lys
H1S	o	yes	Charged ND1 His group
H2S	e	yes	Charged NE2 His group
H3S	f	yes	Neutral His

DNA

A	a	yes	standard adenosine
G	g	yes	standard guanosine
C	c	yes	standard cytosine
T	t	yes	standard thymidine
6MA	b	yes	6-methylated adenosine
5CM	d	yes	5-methylated cytosine

Nucleotides

ATP	-	No
ADP	-	No
GTP	-	No
GDP	-	No

Metals

CA	-	No
MG	-	No
MN	-	No
NA	-	No
ZN	-	No
FE	-	No
CU	-	No
CO	-	No
K	-	No

Waters

HOH	-	No

PDB format

FoldX reads standard PDB format:

```
ATOM      1  N   LEU A 235      43.211  46.266  -8.575  1.00 81.99
```

It can also read some non-standard nomenclature for residues and amino acids:

CYS atom E or SE is converted to the standard SG

Selenometionine (MSE) is converted to standard metionine (MET) and the atom E or SE is converted into the standard SD.

GNP is converted into GTP and the N3B atom into 3B

C-terminal OT1 is converted into standard O and T2 into standard OXT

H20 is converted into standard HOH

In some PDBs non-standard amino acids or atoms of non-standard amino acids are defined as HETATM. FoldX automatically corrects them to ATOM. The opposite applies to metals which sometimes they are defined as ATOMS and are automatically defined as HETATM.

Although we have tried to implement as many code lines to read through non-standard PDBs there are still some files that cannot be run by FoldX and make the code crash or misread the residue. An example is phosphorylated residues where the phosphate group is separated from the residue and defined as HETATM at the end of the PDB. Other cases involve molecules in which the numbering of residues suddenly decreases instead of steadily increase etc…

In those cases in which FoldX crashes the user should carefully examine the PDB to find non-standard features.

Protein design and stability calculations for dummies.

The force field will automatically produce side chains if they are missing or incomplete in the X-tal structure.

Repairing the PDB

Whatever the promises there are no fully automated protein design algorithms where you press a button and the algorithm will offer you a 100% reliable solution. Always the user should examine the answer that the code provides and see if it makes sense. There are many good reasons for this. The first one is that no force field is fully reliable, even sophisticated MD programs like AMBER or CHARM are empirical. Thus there will always be an uncertainty factor. The second one is that X-ray structures are models based on electron density of molecules, sometimes refined by using molecular dynamics force fields, densely packing against each other and in many cases obtained under very low temperature, extreme pHs and high salt. Thus, there are going to be errors in the structure produced during the refinement which will result in non-standard angles or distances, biased conformations arising from the non-physiological conditions under which the structure is determined, and in some cases average rotamers representing two or more true rotamers in equilibrium. This is why in FoldX the RepairPDB command should always be run prior to any energy calculation. However, one should be careful when repairing a PDB since if the structure was partly incorrect at certain places so that some atoms are too close then the repair function that at the moment moves only the side chains could change the rotamer of the residues to decrease the VanderWaals' clash. Thus it is always useful to examine the repaired structure and see if the side chains that have changed rotamer make sense (remember you can always fix some side chains during the repair). It also makes some sense to run the <PrintNetworks> command on the repaired PDB to identify residues or atoms that could still be on a high energy conformation.

Stability studies

One of the applications of algorithms like FoldX is the study of the effects of mutations on protein stability. Normally there are two types of studies, polyala scanning or analysis of SNP (single nucleotide polymorphisms).

In the first case the user will mutate residues to Ala or Gly to find out which ones contribute the most to stability or binding, or to destroy protein function without disrupting its structure (other conservative mutations are Thr<->Val, Asn<->Asp, Gln<->Glu, Tyr->Phe, Ile->Val). Normally these kinds of mutations do not involve major protein reorganization and are the ones which are quite accurately predicted. These mutations can be analyzed in many cases using the simple **<BasicMutate>** or **<alascan>** commands**.**

For non-conservative mutations like the ones frequently observed in SNPs, we should use the **<BuildModel>** command that moves the neighbour sidechains. These types of mutants, specially when done in a tightly packed region of a protein, are less accurate since normally they involve a reorganization of the protein backbone and therefore require exploration of backbone alternative conformations.

When looking at the energy output of the mutants and comparing with the wild-type the user should look at the VanderWaals' clashes contribution of the energy decomposition. Sometimes as discussed above the wild-type structure is incorrect at one position and as a result a side chain is too close to others. Mutation of that side chain to Ala could result in apparent protein stabilization due to the removal of the clashes. Although there could be a certain degree of strain in a protein structure, experience tells us that it not should be above 0.5 Kcal/mol per atom. Thus if the mutation to a smaller residue results in an improvement of the VanderWaal's clash energy above 0.5 for the mutant the user is advised to remove the difference in VanderWaals' clashes between the mutant and the wild-type from the wild-type energy.

Protein Design

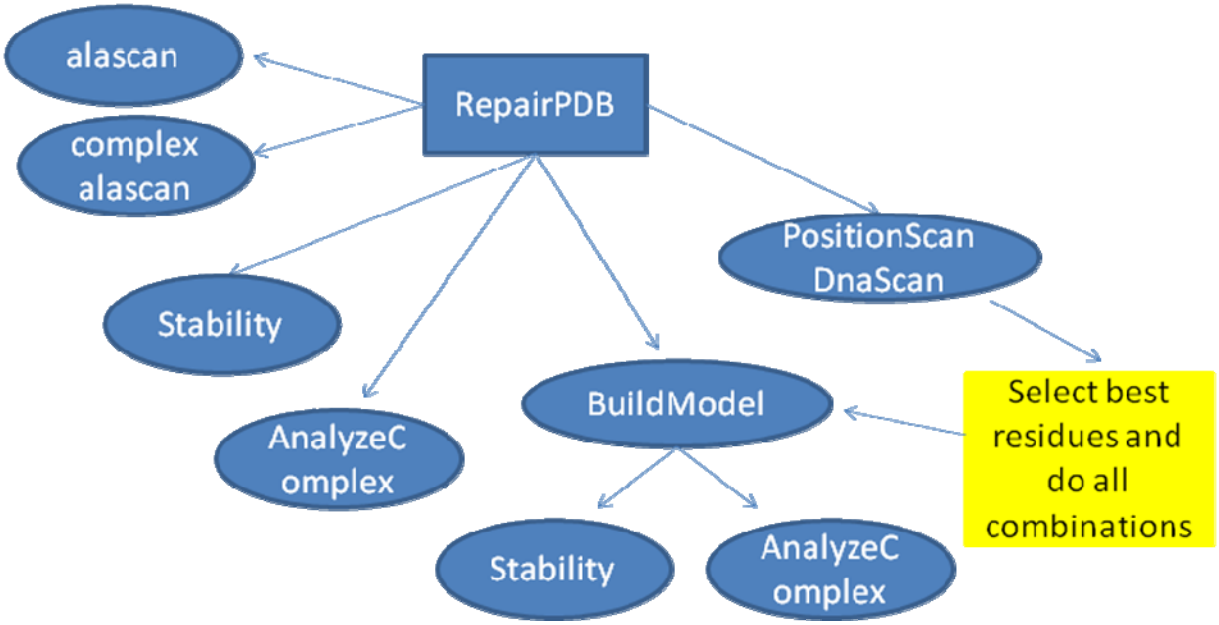
One of the major applications of algorithms like FoldX is protein design. Here the user will want to modify the stability of a protein, decrease its aggregation propensity, change the specificity of an interaction or/and change the affinity. A prerequisite before doing protein design is to have a good structure, this means a crystal resolution below 2.2 Å. This does not mean that resolutions below 2.8 Å cannot be considered, it just says that the results will be less reliable. The same applies to NMR structures that should not be used if there are crystal structures available.

The user is advised not to do too many mutations at a time in the same area, but rather play with one to three residues in different places. The reason is that as we said above Protein Design

algorithms have errors and the more residues you mutate the larger is the chance that something will go wrong.

Also never trust the output of the algorithm without looking at the resulting structure, the human eye is always going to be better than the computer in identifying wrong outputs. Look always for polar residues non-satisfied buried, cavities created during the design run, weird side chain geometries, on the limit H-bonds. Those could be an indication that the designed protein is not as good as the algorithm tells you.

EXAMPLES



Flow Chart of the use of the different FoldX commands

In the following lines we will illustrate the use of FoldX with some selected examples.
Let's put in a folder a pdb file (for example 1shg.pdb), the FoldX executable and the file called rotabase.txt. In the following examples unless stated we will always use 1shg.pdb.

Then we should create a run.txt file with the commands and options to be used. The pdb name should be stored in a file that we can call for example list.txt. Once this is done, the first thing is to repair the PDB. For that the following run file could be used

<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<RepairPDB>#;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>true;
<pdb_hydrogens>false;
<END>#;
<JOBEND>#;
<ENDFILE>#;

Since we are repairing the pdb we consider the crystal water and metal molecules. The <VdWDesign> is set to two to ensure maximum VanderWaals' clash penalty.

The output of this command will be:

- a file called RepairPDB_1shg.pdb which will contain the coordinates of the repaired structure.

- a file called RepairPDB_1shg.fxout with the record of how the energy has changed along the repairing process:

```
FoldX 2.8 (2008)
by the FoldX Consortium
Jesper Borg, Frederic Rousseau, Joost Schymkowitz,
Luis Serrano and Francois Stricher
-----------------------------------------------------
PDB file analysed: 1shg.pdb
Output type: RepairPDB
1shg.pdb
Starting Structure     22.68   -37.82  -13.18  -62.33  -3.03   83.79   -79.51  12.41   34.18   80.95
       0.00    0.00    0.00    7.21    26.20   0.00    0.00    0.00    0.00    0.00    0.00
Now Optimizing Residues.
LYS 6   22.62   -37.82  -13.18  -62.33  -3.01   83.75   -79.52  12.41   34.16   80.95   0.00    0.00
       0.00    7.22    26.20   0.00    0.00    0.00    0.00    0.00    0.00
GLU 7   22.10   -37.82  -13.18  -62.32  -2.99   83.72   -79.50  11.89   34.14   80.95   0.00    0.00
       0.00    7.22    26.20   0.00    0.00    0.00    0.00    0.00    0.00
…      0.00    6.44    26.16   0.00    0.00    0.00    0.00    0.00    0.00
…
…
..
LYS 60  16.68   -38.02  -13.51  -62.50  -3.00   83.30   -80.01  8.84    34.21   80.92   0.00    0.00
       0.00    6.44    26.16   0.00    0.00    0.00    0.00    0.00    0.00
LEU 61  16.67   -38.02  -13.51  -62.50  -3.00   83.30   -80.01  8.84    34.21   80.92   0.00    0.00
       0.00    6.44    26.16   0.00    0.00    0.00    0.00    0.00    0.00
```

```
ASP 62  16.67   -38.02  -13.51  -62.50  -3.00   83.30   -80.01  8.84    34.21   80.92   0.00    0.00
        0.00    6.44    26.16   0.00    0.00    0.00    0.00    0.00    0.00
Now moving residues with bad energies
LYS 6   14.74   -39.37  -16.53  -63.05  -2.92   84.41   -80.70  9.07    37.21   80.87   0.00    0.00
        0.00    5.76    26.16   0.00    0.00    0.00    0.00    0.00    0.00
GLU 7   12.76   -40.23  -17.42  -62.88  -3.03   83.92   -80.62  8.40    37.92   80.92   0.00    0.00
        0.00    5.76    26.17   0.00    0.00    0.00    0.00    0.00    0.00
…
..
..
LYS 60  -0.41   -41.33  -21.31  -63.82  -3.66   83.49   -82.73  5.46    40.55   80.19   0.00    0.00
        0.00    2.74    26.30   0.00    0.00    0.00    0.00    0.00    0.00
ASP 62  -0.54   -41.36  -21.31  -63.91  -3.75   83.50   -82.85  5.46    40.69   80.22   0.00    0.00
        0.00    2.78    26.30   0.00    0.00    0.00    0.00    0.00    0.00
```

This is only to have an overview of how the energy of the molecule improves along the optimization. There are several numbers for every residue optimized (please be aware that not all residues will be optimized, only the ones with bad energies). The important one is the first one that is the global energy. The other numbers represent the energy decomposition of the global energy.

After repairing the PDB the user is advised to compare the original and repaired structure to check if a residue that the user knows should not move has been displaced. If this is the case the residue could be fixed prior to the repair. For this the following run.txt should be used:

```
<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<RepairPDB>#,Fixed:K 39;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>true;
<pdb_hydrogens>false;
<END>#;
<JOBEND>#;
<ENDFILE>#;
```

In this way Lys39 will not be moved (please take note of the space between the K and the 39, this is because in 1shg.pdb there is no name for the chain).

Stability

Once the PDB structure is repaired the user can start working with it. We could run for example the <Stability> command to look at the energy of our molecule, provided that we change the pdb name in the list.txt file to RepairPDB_1shg.pdb.

```
<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<Stability>Stability.txt;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<END>#;
<JOBEND>#;
<ENDFILE>#;
```

If we use the above runscript the output will be a file called Stability.txt:

```
FoldX 2.8 (2008)
by the FoldX Consortium
Jesper Borg, Frederic Rousseau, Joost Schymkowitz,
Luis Serrano and Francois Stricher
-----------------------------------------------------

PDB file analysed: RepairPDB_1shg.pdb
Output type: Conformational Stability - Free Energy in kcal/mol
       total energy   Backbone Hbond Sidechain Hbond       Van der Waals  Electrostatics Solvation
Polar   Solvation Hydrophobic  Van de Waals clashes   entropy side chain     entropy     main     chain
       sloop_entropy  mloop_entropy  cis_bond       torsional clash        backbone clash
       helix dipole   water bridge   disulfide      electrostatic kon      partial covalent bonds
RepairPDB_1shg.pdb     -0.55   -41.36  -21.31  -63.92  -3.75   83.50   -82.86  5.46    40.69   80.22
       0.00    0.00    0.00    2.78    26.30   0.00    0.00    0.00    0.00    0.00    0.00    57
```

This file will have different headers and then rows (one for each PDB run) with the energy decomposition in Kcal/mol:

```
total energy              This is the predicted overall stability of your protein
Backbone Hbond            This the contribution of backbone Hbonds
Sidechain Hbond           This the contribution of sidechain-sidechain and sidechain-backbone  Hbonds
Van der Waals             Contribution of the VanderWaals'
Electrostatics            Electrostatic interactions
Solvation Polar           Penalization for burying polar groups
Solvation Hydrophobic     Contribution of hydrophobic groups
Van de Waals clashes      Energy penalization due to VanderWaals' clashes (interresidue)
entropy side chain        Entropy cost of fixing the side chain
entropy main chain        Entropy cost of fixing the main chain
sloop_entropy             ONLY FOR ADVANCED USERS
mloop_entropy             ONLY FOR ADVANCED USERS
cis_bond                  Cost of having a cis peptide bond
torsional clash           VanderWaals' torsional clashes (intraresidue)
backbone clash            Backbone-backbone  VanderWaals.   These  are  not  considered  in  the  total
energy.
helix dipole              Electrostatic contribution of the helix dipole
water bridge              Contribution of water bridges
disulfide                 Contributiuon of disulphide bonds
electrostatic kon         Electrostatic interaction between molecules in the precomplex
partial covalent bonds    Interactions with bound metals
```

Complex Analysis

If instead of having a simple molecule we could have a protein complex then we could analyze the interaction energy of the two molecules. An example could be 1abo.pdb. This pdb is a SH3 (chain A) with a ligand bound (chainB).

```
<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<AnalyseComplex>#,A;     //A means that we want the energy of interaction of chain A with the
rest
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<END>#;
<JOBEND>#;
<ENDFILE>#;
```

The output will be a file called AnalyseComplex_1abo2.txt that will look

```
FoldX 6.0 (2004) - foldx_beta_2004_07_27
```

```
by the FoldX Consortium @ EMBL-Heidelberg.de
Jesper Borg, Joost Schymkowitz, Francois Stricher,
Frederic Rousseau, Raphael Guerois, Luis Serrano
-------------------------------------------------------

PDB file analysed: 1abo2.pdb
Output type: Complex analysis

                          total energy   Backbone Hbond Sidechain Hbond       Van    der    Waals
                    Electrostatics      Solvation Polar       Solvation Hydrophobic Van      de
Waals clashes       entropy side chain  entropy main chain    sloop_entropy  mloop_entropy
                    cis_bond    torsional clash       backbone clash helix dipole   water
bridge              disulfide   electrostatic kon     partial covalent bonds
1abo2.pdb           Interaction between groups 0 and 1 =      Intraclashes=  0.456296       +
Interaction =       -5.37369    -1.275  -1.275  -5.46727      -0.235183      5.90492 -7.83785
                    0.0576435   1.17312 2.51838 0      0      0      1.38769 0.566374      0
                    0    0      -0.325154      0      2.384  69.
```

This file will have the same terms that the stability one, although in this case all energy terms are not absolute values but binding values. At the beginning of the row with the energy terms you will find some extra columns:

```
1abo2.pdb           Interaction between groups 0 and 1 =      Intraclashes=  0.456296       +
Interaction =       -5.37369
```

The first one is the name of the pdb. The second one tells you which groups are you considering for determining the interaction (if you do not define any chain and imagine your pdb has three chains you will get three energy rows one for groups 1 and 2, another for 2 and 3 and another for 1 and 3). The third column is very important `Intraclashes=  0.456296`. This column tells what are the VanderWaals' clashes of residues at the interface of a complex with their own molecule. You could make a mutation that will increase binding energy but also will result in an awkward local conformation with a high VanderWaals' energy with residues of its own molecule. Logically this is not good. Therefore when doing mutations to improve affinity you should always check that this value does not increase compared with the wild-type. Finally the third column that corresponds to total energy `Interaction =  -5.37369`, is the actual binding energy.

If you want to determine the interaction between DNA and a protein you could use the following run.txt

```
<TITLE>FOLDX_runscript;
<JOBSTART>#;
```

<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<AnalyseComplex>#;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<complex_with_DNA> true;
<END>#;
<JOBEND>#;
<ENDFILE>#;

By setting <complex_with_DNA> to true the code will only consider two molecules, the DNA and the protein independently of the number of chains. This way the energy output will be the interaction energy of the DNA with the protein. "Please be aware that FoldX at the moment does not consider entropy costs for DNA and therefore the binding energy could be over evaluated.

PositionScan

When you are trying to improve the stability of a protein or of a complex it is not easy to decide which mutations you will make. One possibility is to ask FoldX to scan all positions you select (even the whole protein, although this will take long) and run the <PositionScan> command. This way FoldX will mutate every selected position to groups of residues, (or bases in the case of DNA). Remember that you could decide which groups to consider i.e. all natural 20 amino acids, natural plus phosphorylated, polar etc….(see COMMANDS chapter). A typical run.txt will look like:

<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<PositionScan>#,K 39a,W 41d;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;

<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<complex_with_DNA> true;
<END>#;
<JOBEND>#;
<ENDFILE>#;

This will mutate K39 to the 20 amino acids and W41 to the 20 plus the phosphorylated versions of Ser, Thr and Tyr and hydroxyproline.

The output files will be pdb files with a name made by the residue and position mutated (i.e. TRP41) followed by the pdbname TRP41_1shg.pdb, ARG41_1shg.pdb etc…., and an energy file with the name energies_41_1shg.txt. The energy file will have the total energy and all energy terms for the WT reference and all mutants (beware that if you are doing the self-mutation the energy could be lower than that of the original WT because of further optimization). In this case there are no headings for the energies but they correspond to the following columns:

```
total energy   Backbone Hbond Sidechain Hbond       Van der Waals  Electrostatics Solvation     Polar
       Solvation Hydrophobic  Van de Waals clashes  entropy side chain     entropy     main     chain
       sloop_entropy  mloop_entropy  cis_bond       torsional clash        backbone clash
       helix dipole   water bridge   disulfide      electrostatic kon      partial covalent bonds
```

Once you have run PositionScan you can select the best ones (or all combinations) and then make all the mutations at the same time. If the positions are far away you could consider them independent and just select the best residues, if they are close you should select a subset of good mutations and do all combinations.

BuildModel

This is the normal mutagenesis engine you will use in FoldX. This will allow you to do multiple mutations at the same time. There are three ways to run it and we will show examples of the three run.txt files.

-mutant_file.txt

-<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<BuildModel>#,mutant_file.txt;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<complex_with_DNA> true;
<END>#;
<JOBEND>#;
<ENDFILE>#;

In this case the mutant_file.txt should look something like this

KELVLA
KDLVLA
KQLALA

The first line should be the protein or DNA sequence in one letter code (you do not need to write the whole sequence just the region you want to mutate and it should be long enough to be unique). The following lines you should write a sequence of the same length with the mutations you want to make.

You will get a pdb file for each mutant (1shg_0.pdb, 1shg_1.pdb) and its corresponding WT reference (WT_1shg_0.pdb, WT_1shg_1.pdb) and three text files:

-Dif_

```
 FoldX 2.8 (2008)
by the FoldX Consortium
Jesper Borg, Frederic Rousseau, Joost Schymkowitz,
Luis Serrano and Francois Stricher
-----------------------------------------------------

PDB file analysed: 1shg.pdb
Output type: BuildModel
Pdb   total energy      Backbone Hbond    Sidechain Hbond   Van der Waals
      Electrostatics    Solvation Polar   Solvation Hydrophobic   Van de Waals
clashes     entropy side chain      entropy main chain      sloop_entropy
      mloop_entropy     cis_bond    torsional clash   backbone clash    helix
```

```
dipole      water bridge      disulfide   electrostatic kon partial covalent
bonds
1shg_1.pdb  -0.34 0.03  0.05  -0.08 -0.08 0.21  -0.11 0.01  -0.41 0.02  0.00
      0.00  0.00  0.03  0.01  0.00  0.00  0.00  0.00  0.00  0.00
```

Here you have the difference in energy between the mutation and the corresponding WT-Reference (Positive numbers mean less stability).

-PdbList_

```
1shg_1.pdb
WT_1shg_1.pdb
```

Here you have the list of pdbs you have created. You could use this list later with other commands to calculate binding, stability etc…

-Raw_

```
FoldX 2.8 (2008)
by the FoldX Consortium
Jesper Borg, Frederic Rousseau, Joost Schymkowitz,
Luis Serrano and Francois Stricher
-----------------------------------------------------

PDB file analysed: 1shg.pdb
Output type: BuildModel
Pdb   total energy      Backbone Hbond    Sidechain Hbond   Van der Waals
      Electrostatics    Solvation Polar   Solvation Hydrophobic   Van de Waals
clashes     entropy side chain      entropy main chain      sloop_entropy
      mloop_entropy     cis_bond    torsional clash   backbone clash    helix
dipole      water bridge      disulfide   electrostatic kon partial covalent
bonds
1shg_1.pdb  18.45 -38.63      -13.92      -62.21      -2.76 82.97 -79.70
      11.17 33.99 80.95 0.00  0.00  0.00  6.59  26.21 0.00  0.00  0.00  0.00
      0.00  0.00
WT_1shg_1.pdb     18.79 -38.66      -13.97      -62.13      -2.68 82.76 -79.59
      11.17 34.40 80.93 0.00  0.00  0.00  6.56  26.20 0.00  0.00  0.00  0.00
      0.00  0.00
```

In this file you have the stability for each PDB.

-individual_list.txt

-<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<BuildModel>#,individual_list.txt;
<END>#;
<OPTIONS>FOLDX_optionfile;

<Temperature>298;
<R>#;
<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<complex_with_DNA> true;
<END>#;
<JOBEND>#;

<ENDFILE>#;

In this case the individual_list.txt should look something like this

K6 L,E 7M,W 41A;
K6 Q,E 7R,W 41V;

Where in each row you specify the list of mutations you want to make in one letter code (the space between the aa letter and the pdb number is for the chain that in the case of 1shg.pdb is not defined, in other pdbs where it is defined you should put it). The code will make all mutants in a row and produce a pdb for each row (1shg_0.pdb, 1shg_1.pdb) and its corresponding WT references (WT_1shg_0.pdb, WT_1shg_1.pdb). You will also get the same files described above.

-DNA_scan.txt

This one should be used if you have a DNA structure or a complex between DNA and a protein and you want to see the effect of mutating every single base to the other three.

-<TITLE>FOLDX_runscript;
<JOBSTART>#;
<PDBS>#;
<BATCH>list.txt;
<COMMANDS>FOLDX_commandfile;
<BuildModel>#,DNA_scan.txt;
<END>#;
<OPTIONS>FOLDX_optionfile;
<Temperature>298;
<R>#;

<pH>7;
<IonStrength>0.050;
<water>-CRYSTAL;
<metal>-CRYSTAL;
<VdWDesign>2;
<OutPDB>false;
<pdb_hydrogens>false;
<complex_with_DNA> true;
<END>#;
<JOBEND>#;

<ENDFILE>#;

The output files will be exactly the same as in the previous cases.