cMutant : A Web Server and Compute Pipeline for Exploring the Effects of Amino Acid Substitutions via Rigidity Mutation Maps

Hunter Read, Kyle Daling, Connor Freitas, Filip Jagodzinski*
Western Washington University
Bellingham, WA, 98225, USA
filip.jagodzinski@wwu.edu

Abstract

Pharmaceutical companies rely on the ability to analyze the effects of protein mutations to develop medicines for treating a variety of diseases. Although mutagenesis experiments performed in a physical protein can provide insights about the role of a single amino acid, such experiments are laboriously difficult and may require months of wet lab work. Consequently conducting exhaustive mutagenesis screens which involve mutating all residues to all other amino acids, is impractical. To help guide such wet lab experiments, computational approaches are available, but most do not permit an exhaustive screening of all residues and their impact on a protein when mutated. For this work we have integrated into a compute pipeline and server our *in silico* mutation analysis method for quickly generating protein variants. We leverage a quick computational algorithm to assess the rigidity of the wild type and mutants, and use the results to infer which residues are most sensitive to an amino acid substitution. Our server and pipeline leverage concurrency principles permitting an exhaustive screening of all mutations for all residues in a protein in as little as a few minutes. We report here on the performance and utility of the pipeline, and present a case study to highlight the utility of Mutation Maps generated by our server, cMutant, available at `https://cmutant.cs.wwu.edu/`.

Introduction

Experimentalists mutate and analyze proteins to develop better medicine for treating a wide range of diseases [27]. Conducting mutation analyses in a physical protein can require months of wet-lab work, with the aim to provide information to help engineer pharmaceutical drugs targeting specific proteins [22].

A variety of computational approaches and *in silico* protein mutation analysis tools aim to provide a screen to help guide wet lab experimentalists where they might focus their attention for conducting mutagenesis experiments on physical proteins. The majority of most existing screening software tools permit exploring the effect of only a single mutation at one specific residue in a protein, while the few approaches that permit exhaustive *in silico* studies for a protein have a variety of limitations due to their dependencies on homology or energetics data that may not always be available.

In our previous work [7, 2], we have motivated the use of a fast combinatorial approach called rigidity analysis, in combination with our custom *in silico* mutation engine for generating mutant structure files, in assessing the effects of amino acid substitutions.

For this work, we present a compute pipeline and publicly available server, cMutant, that relies on concurrency principles to greatly reduce the runtime of performing an exhaustive mutation screen for all residues in a protein. We reduce the runtime of *in silico* mutation experiments from days to hours – and sometimes to minutes. We achieve such a speedup by executing our pipeline concurrently on multiple cores available on our server. To permit a user to perform a visual inspection of the effects of the exhaustive mutation experiments, we generate a mutation map which is presented via a graphical user interface, and which is stored in a database for future retrieval. The utility of our mutation maps we have demonstrated in our previous work [28].

Related Work

To help complement and inform wet lab work, various modeling and computational methods, including some available via web servers, are available. They strive to predict the effects of mutations. Early algorithms ranged from those that searched for best side-chain conformations as a measure of the impact of a mutation [6, 16, 25], to those that relied on heuristic energy functions [10, 19]. Yet others relied on large data sets of homologous proteins [30, 3, 31]. More recently, machine learning (ML) approaches have gained notoriety, with some having high prediction rates upwards of 80% [4, 14, 17, 20]. However, the energy-, homology- and ML-based approached have several limitations. Many of

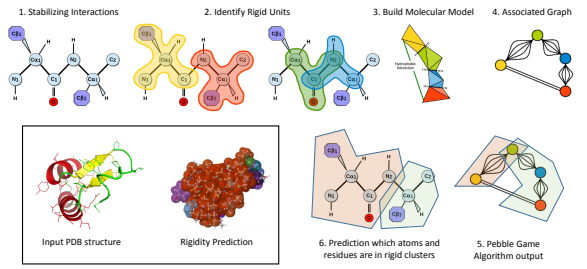

Figure 1: Rigidity analysis involves modeling a biomolecule as a mechanical model, which is analyzed using an efficient pebble game algorithms. The results are used to infer the rigid and flexible regions of a biomolecule.

them are dependent on large data sets [21, 32], some require costly energy calculations [5, 24, 26], and others still are dependent on free energy calculations as well as access to propensity tables [23], data which is not always available, or which is computationally costly to calculate.

In our previous work, we developed several computational approaches for quickly generating large data sets of *in silico* mutants. Incipient experiments enabled mutating a residue to only one of Alanine, Glycine, or Serine [15], but more recently our mutation software has been expanded to permit *in silico* mutating a residue to all possible other amino acids [2].

To help reason about the effects of mutations, we take an approach that does not rely on propensity tables, costly energy calculations, nor is dependent on homology data. Instead we rely on a fast combinatorial approach for assessing the rigidity of a protein [9, 13]. In rigidity analysis, atoms and their chemical interactions are used to construct a mechanical model. A graph is constructed from the model, and pebble game algorithms [12] are used to analyze the rigidity of the associated graph. The results are used to infer the rigid and flexible regions of the protein (Figure 1).

Rigidity Distance

In this work cMutant compares the rigidity analysis results of the wild type (WT), non-mutated form of a protein, to the rigidity analysis results of the mutant. This builds on our previous work [1, 8], in which we developed and utilized a$RD_{WT\rightarrow mutant}$ rigidity distance metric to quantitatively assess the impact of mutating a residue to one of the other 19 naturally occurring amino acids:

$$RD_{WT\rightarrow mutant} : \sum_{i=1}^{i=LRC} i \times [WT_i - Mut_i]$$

where WT refers to Wild Type, Mut refers to mutant,

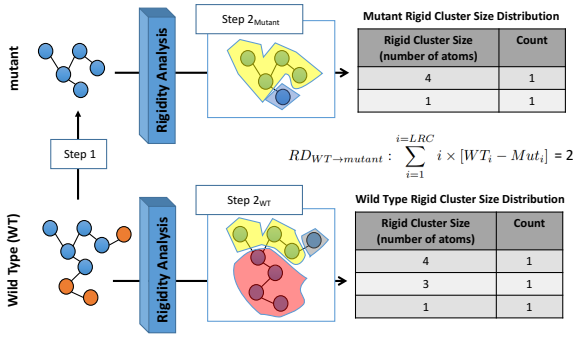

Figure 2: Comparing the rigid cluster distributions (sizes and counts) for the Wild Type and Mutant structures enables assessing quantitatively the effect of an amino acid substitution via the Rigidity Distance $RD_{WT\rightarrow mutant}$ metric.

LRC is the size of the Largest Rigid Cluster (in atoms). Each successive summation term of the $RD_{WT\rightarrow mutant}$ metric calculates the difference in the count of a specific cluster size, i, of the wild type and mutant, and weighs that difference by i (Sample in Figure 2).

Server & Software Design

Our contributions for this work includes a concurrent implementation of our mutation and analysis software and the auto-generation of mutation maps [28] to aid in the visual analysis of an exhaustive mutation screen. In this section we describe the server and compute pipeline, as well as the analysis methodology that culminates in a mutation map.

Overview

cMutant offers features that are not available via other tools and web services. Upon invocation, the server generates all mutant structures as asked-for by the user via the front-end. The infrastructure leverages principles from concurrency theory to vastly reduce the execution time needed for conducting exhaustive mutation experiments. cMutant offers a graphical user interface (GUI), that enables a user to view all mutations via a mutation map which permits a user to investigate individual point mutations and download specific results. The system design is summarized in Figure 3.

Back-End Infrastructure

The computational infrastructure integrates a variety of our in-house custom software, as well as off-the shelf

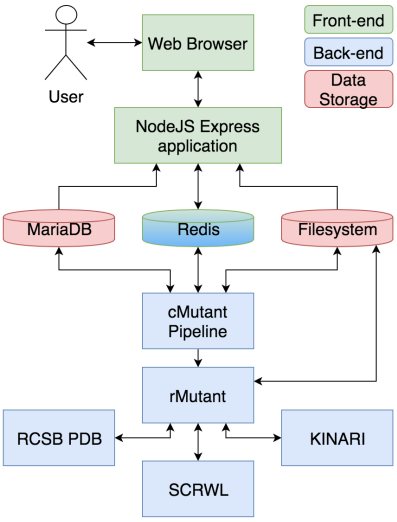

Figure 3: cMutant includes front-end (GUI) and back-end functionality, enabling a user to interface with our custom mutation and analysis software.

and freely available tools. These include KINARI [9] and ProMuteHT[2], along with SCWRL [18]. The pipeline is invoked when a user interacts with the GUI to specify the PDB ID (protein structure file), along with parameters designating which residues are to be mutated. Use of concurrency principles enabled by the threading capabilities of the multi-core server allows for each unique *in silico* point mutation to be invoked in a separate thread. The count of threads is limited by the number of available processors, and the output data files of each experiment are stored to files locally on the server for archiving and retrieval by the user.

The back-end infrastructure performs concurrent execution of KINARI and proMuteHT for quickly generating and processing of a large set of protein mutants (Table 1). cMutant is able to decrease exhaustive protein mutation run times by a factor equal to that of the number of cores available on the server, with additional speed-up obtained through allowing mutations to take advantage of the pipelining ability of the CPU architecture. Each experiment requires analyzing the wild type protein once, before any *in silico* protein mutations are performed. This first step is not run concurrently. Run-times were determined by clock time at initialization of the compute pipeline through execution of all intermediate steps, until pipeline termination resulting in a mutation map.

Front-End Infrastructure

The front-end GUI of cMutant includes an **Experiment (and Results)** section. There users specify experiment parameters, and view results as they become

Table 1: Run-times (minutes) for threaded (thread) and serial (ser) invocations of cMutant, and speedup ratios (sr) resulting from use of concurrency. # res=num. of residues; # muts=num. mutants generated.

PDB File	# res	# muts	thread	ser	sr
1PLW	5	100	0.65	4.37	6.72
1DPK	20	400	3.32	22.7	6.84
2LK0	30	600	7.35	45.3	6.16
1HN3	40	800	10.8	65.4	6.06
1YUG	50	1000	19.2	103	5.39
5NHQ	71	1420	36.9	190	5.15
1A1Z	83	1660	63.7	301	4.72
1HHP	99	1980	95.9	426	4.44

available. A **Retrieve Experiments** section permits retrieving data from past computation runs, as well as viewing the current server load.

The **Experiment** section offers a GUI (Figure 4) with options for a user to:

(1) specify a PDB ID for which a mutation screen is to be performed

(2) which residues to mutate (an **all** option is available for designating an exhaustive screen)

(3) specifying what each selected residue(s) should be mutated to, for which an **all** options is also available.

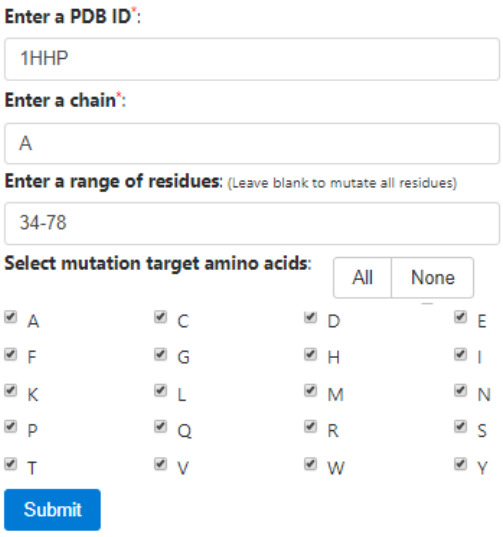

Figure 4: cMutant's GUI offers the option to specify an exhaustive mutation screen, or to mutate a subset of the residues (range of residues). Each selected residue can be mutated to all other possible amino acids, or a custom subset (mutation targets).

When an experiment is initiated, a user is provided with an alphanumeric experiment ID which can be used for later retrieval of the experiment data which is stored to a database.

Server-side technologies such as NodeJS and ExpressJS provide the functionality for transmitting data between the back-end software and GUI. As data is being generated by the multiple threads that are invoked, the results are displayed and updated in real time. Communication between the cMutant pipeline and GUI is accomplished by using the in-memory data structure store Redis.

The **Result** pane presents a Mutation Map, which is a heat map generated from the distance metric values (See section Rigidity Distance) computed for each residue that was mutated. A full explanation can be found in [28]. The color in each cell in a Mutation Map corresponds to a rigidity distance, which is a measure, based on the rigid clusters of the mutant and wild type. A user can mouse-over a specific cell in the Mutation Map to view the rigidity distance score for that residue, or to download the data for that specific *in silico* mutation. A rigidity distance far greater or far less than zero indicates that the mutant is structurally vastly different than the wild type, while a rigidity distance score near zero specifies that the wild type and mutant are structurally similar, as inferred using the rigidity cluster data. The magnitude of the rigidity distance can be used to indirectly infer the magnitude of the impact of an amino acid substitution.

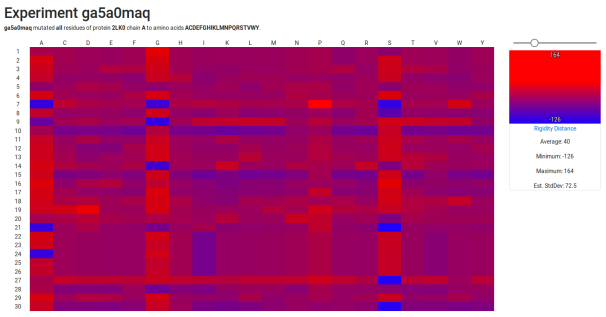

Figure 5: Mutation Maps : for each residue number (y-axis), a color at each target residue (x-axis) specifies the rigidity distance metric score for a mutation.

A sample Experiment results pane, for experiment `ga5a0maq`, is shown in Figure 5. That mutation map is for an exhaustive *in silico* mutation screen for the 30-residues PDB file 2LK0, which is the structure of a RanBP2-type zinc finger of RBM5. A dynamically updated color legend indicates that a red cell has a high rigidity distance, while a blue cell has a low rigidity distance score, and that the average, minimum, and maximum Rigidity Distance scores are 40, -126, and 164. Most telling in the Mutation Map for 2LK0 is that specific residues upon their *in silico* mutation to certain residues yield very low (highly negative), or very high (highly positive) rigidity distance scores. A very low rigidity distance score for a residue's mutation to a specific amino acid indicates that that mutation results in a mutant that has far more large rigid clusters than the WT. Such a mutation can be inferred to be stabilizing. The converse is true for very high positive rigidity scores. In the case of 2LK0, using the Mutation Map, the blue spots identify that residues 7, 9, 14, 21, 24, 27, and 30, have strong stabilizing effects on the protein as inferred using rigidity analysis.

Case Study, 1HHP

To assess the speed and usefulness of cMutant, we exhaustively *in silico* mutated all residues of PDB structure 1HHP, which is the monomeric form of the 99 amino acid HIV-1 Protease. A zoomed in portion (residues 15 to 40) of the Mutation Map for 1HHP is shown in Figure 6.

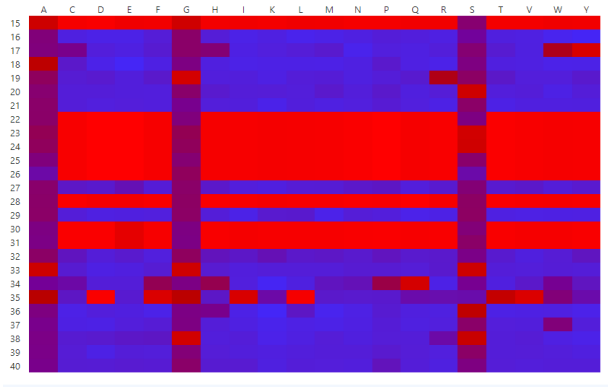

Figure 6: Zoomed in Mutation Map for 1HHP, residues 15-40. Residues 22-26, as well as 28, and 30 and 31 are especially sensitive to mutations as evidenced by the red Rigidity Distance scores for nearly all mutations performed at those residues.

Residues 24-26 of HIV-1 Protease constitute a catalytic triad, the active site of the protein, on which a host of wet lab experiments have been conducted and for which there is a lengthy literature [11, 29]. The residues near the active site of HIV-1 Protease are known to be critical to the protein's function, and indeed are highly resistant to mutations. Specific residues at those locations must be present in order for the protein to perform its catalytic function. As a first proof-of-concept result, we consider it encouraging that cMutant identified those residues near the active site as being least resistant to mutations, because *in silico*

mutations performed on them in nearly all cases highly disrupted the protein's structure. See [28] for a more detailed example of the utility and use, including a box plot analysis, of Mutation Maps.

Future and On-Going Work

Future and-going work on cMutant involves three main avenues, including 1) improving the server's speed by leveraging additional concurrency principles, 2) adding additional front-end GUI features, and 3) assessing and improving the accuracy of the predictions doled up by the Mutation Map. In our most recent work, we have developed machine learning models capable of predicting at up to 80% accuracy the effect of mutations [8]. That predictive capability is being integrated into cMutant.

For improving the GUI, we are developing additional UI elements to allow the user to quickly access important trends and details of the results from a computation experiment run. In addition to the mutant and WT structure files, along with the rigidity data, for each cell in a Mutation map that can be currently downloaded, we aim to integrate a protein viewer visualization engine that will color code the 3-D surface of a protein to display rigidity metrics of those residues on the surface.

A current limitation of the server is that it is able to perform exhaustive mutation screens for single chain proteins only. Current work in our lab has culminated in an improved mutation engine, ProMuteHT, which is being integrated into the cMutant pipeline allowing it to reason about any protein in the PDB.

For further validation of the use of Mutation Maps beyond what we have reported previously [28], we are correlating our rigidity distance scores for point mutations against $\Delta\Delta G$ data attained from experiments on physical proteins, which gives empirical evidence of the effects of mutations. We are tallying Pearson Correlation coefficients, and aim to supplement the Mutation Map data with that information.

Conclusions

We have developed a compute pipeline and server, cMutant, for performing a rigidity-based mutation screen that exhaustively generates and analyzes all possible mutant structures with a single amino acid substitution. We achieve fast run-times by leveraging concurrency principles, and also generate a Mutation Map which aids in a visual analysis enabling identification of residues that are highly sensitive to mutations. We present a case study for HIV-1 Protease, and correlate our interpretation of the analysis of the Mutation Map with known biological properties of the protein's active site.

Acknowledgments

HR designed and deployed the database, and developed the pipeline enabling the rigidity and proMuteHT software to run concurrently. KD developed the web server code used for serving the GUI and transmitting data to it. CF designed the results page and mutation map functions. FJ supervised the work. All authors contributed to writing the manuscript.

References

[1] E. Andersson, R. Hsieh, H. Szeto, R. Farhoodi N. Haspel, and F. Jagodzinski. Assessing how multiple mutations affect protein stability using rigid cluster size distributions. In *Computational Advances in Bio and Medical Sciences (ICCABS), 2016 IEEE 6th International Conference on*, pages 1–6. IEEE, 2016.

[2] Erik Andersson and Filip Jagodzinski. ProMuteHT: A high throughput compute pipeline for generating protein mutants in silico. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 655–660. ACM, 2017.

[3] Jeffrey R Brender and Yang Zhang. Predicting the effect of mutations on protein-protein binding interactions through structure-based interface profiles. *PLoS Comput Biol*, 11(10):e1004494, 2015.

[4] J. Cheng, A. Randall, and P. Baldi. Prediction of protein stability changes for single-site mutations using support vector machines. *PROTEINS: Structure, Function, and Bioinformatics*, 62:1125–1132, 2006.

[5] Y Dehouck, JM Kwasigroch, M Gilis, and Rooman M. Popmusic 2.1: a web server for the estimation of protein stability changes upon mutation and sequence optimality. *BMC Bioinformatics*, 12, 2011.

[6] R.L. Jr. Dunbrack and M. Karplus. Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nature Structural Biology*, 1:334–340, 1994.

[7] I. Streinu F. Jagodzinski, J. Hardy. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics, Computational Biology*, 10:1242010–1242027, 2012.

[8] Roshanak Farhoodi, Max Shelbourne, Rebecca Hsieh, Nurit Haspel, Brian Hutchinson, and Filip Jagodzinski. Predicting the effect of point mutations on protein structural stability. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 247–252. ACM, 2017.

[9] N. Fox, F. Jagodzinski, and I. Streinu. KINARI-Lib: a C++ library for pebble game rigidity analysis of mechanical models. In *Minisymposium on Publicly Available Geometric/Topological Software, Chapel Hill, NC, USA*, June 2012.

[10] D. Gilis and M. Rooman. Predicting protein stability changes upon mutation using database-dervied potentials: Solvent accessibility determines the importance of local versus non-local interactions along the sequence. *Journal of Molecular Biology*, 272(2):276–290, 1997.

[11] Viktor Hornak, Asim Okur, Robert C Rizzo, and Carlos Simmerling. Hiv-1 protease flaps spontaneously open and reclose in molecular dynamics simulations. *Proceedings of the National Academy of Sciences of the United States of America*, 103(4):915–920, 2006.

[12] D.J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *Journal of Computational Physics*, 137:346–365, 1997.

[13] D.J. Jacobs, A.J. Rader, M.F. Thorpe, and L.A. Kuhn. Protein flexibility predictions using graph theory. *Proteins 44*, pages 150–165, 2001.

[14] F. Jagodzinski, B. Akbal-Delibas, and N. Haspel. An evolutionary conservation & rigidity analysis machine learning approach for detecting critical protein residues. In *CSBW (Computational Structural Bioinformatics Workshop), in proc. of ACM-BCB (ACM International conference on Bioinformatics and Computational Biology)*, pages 780–786, September 2013.

[15] F. Jagodzinski, J. Hardy, and I. Streinu. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics and Computational Biology*, 10(3), 2012.

[16] J Janin and S Wodak. Conformation of amino acid side-chains in proteins. *J Mol Biol*, 125(3):357–386, Nov 1978.

[17] Lei Jia, Ramya Yarlagadda, and Charles C Reed. Structure based thermostability prediction models for protein single point mutations with machine learning tools. *PloS one*, 10(9):e0138022, 2015.

[18] Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack. Improved prediction of protein side-chain conformations with scwrl4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, 2009.

[19] C. Lee and M. Levitt. Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core. *Nature*, 352:448–451, 1991.

[20] Yunqi Li and Jianwen Fang. Prots-rf: a robust model for predicting mutation-induced protein stability changes. *PloS one*, 7(10):e47247, 2012.

[21] Majid Masso and Iosif I Vaisman. Auto-mute: web-based tools for predicting stability changes in proteins due to single amino acid replacements. *Protein Engineering Design and Selection*, 23(8):683–687, 2010.

[22] Tatiana Maximova, Ryan Moffatt, Buyong Ma, Ruth Nussinov, and Amarda Shehu. Principles and overview of sampling methods for modeling macromolecular structure and dynamics. *PLoS computational biology*, 12(4):e1004619, 2016.

[23] Caitlyn L McCafferty and Yuri V Sergeev. In silico mapping of protein unfolding mutations for inherited disease. *Scientific Reports*, 6:37298, 2016.

[24] Vijaya Parthiban, M. Michael Gromiha, and Dietmar Schomburg. Cupsat: prediction of protein stability upon point mutations. *Nucleic Acids Research*, 34(suppl 2):W239–W242, 2006.

[25] J.W. Ponder and F.M. Richards. Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes. *Journal Molecular Biology*, 193:775–791, 1987.

[26] Lijun Quan, Qiang Lv, and Yang Zhang. Strum: structure-based prediction of protein stability changes upon single-point mutation. *Bioinformatics*, 32(19):2936–2946, 2016.

[27] Boris Reva, Yevgeniy Antipin, and Chris Sander. Predicting the functional impact of protein mutations: application to cancer genomics. *Nucleic acids research*, 39(17):e118–e118, 2011.

[28] Michael Siderius and Filip Jagodzinski. Mutation sensitivity maps: Identifying residue substitutions that impact protein structure via a rigidity analysis in silico mutation approach. *Journal of Computational Biology*, 25(1):89–102, 2018.

[29] Sergio Filipe Sousa, Bruno Tamames, Pedro Alexandrino Fernandes, and Maria Joao Ramos. Detailed atomistic analysis of the hiv-1 protease interface. *The Journal of Physical Chemistry B*, 115(21):7045–7057, 2011.

[30] C.M. Topham, N. Srinivasan, and T. Blundell. Prediction of the stability of protein mutants based on structural environment-dependent amino acid substitutions and propensity tables. *Protein Engineering*, 10(1):7–21, 1997.

[31] C.L. Worth, R. Preissner, and L. Blundell. Sdm-a server for predicting effects of mutations on protein stability and malfunction. *Nucleic Acids Research*, 39(Web Server Issue):W215–W222, 2011.

[32] Hongyi Zhou and Yaoqi Zhou. Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein science*, 11(11):2714–2726, 2002.