Evaluating accuracy of *MuMu*

A bioinformatical approach in protein stability prediction

A previous study; 'A bayesian multibody multinomial model as a rapid method of measuring protein stability', revealed that a bayesian multibody nominal model, MuMu, can be used as a high-throughput method for rapid prediction of protein stability. The study explored the correlation of predictions of protein stability change upon 95 mutations and 95 experimentally determined ΔΔG values from the protein 1STN. Here I examine the robustness of protein stability prediction by MuMu. I include 15 proteins with 458 mutational variations. I probe for underlying correlations based on physical properties of mutated amino acids and secondary structure of mutated regions. By statistical analysis I determine correlation coefficients and their errors on MuMu data compared to ΔΔG values. It is revealed that there is an overall decrease in correlation when more data is included. MuMu protein stability prediction does not correlate well with experimentally determined ΔΔG values, however there may be a reasonable trade-off in accuracy in turn of speed.

Introduction

Describing the stability of a protein is the science of protein stability prediction. It is a complex matter to describe and interpret stability. Computational methods involve models that seek to describe not only molecular interactions within but beyond proteins themselves, e.g. protein stability upon interaction with other bodies. However, computationally methods are not as precise as experimentally obtained information on protein stability, which is still used as a mean of comparison for computational models (Potapov, et. al. 2009). The stability of a protein is based on a protein being in a native folded state, N, and it being in a denatured unfolded state, U. The equilibrium, K_{eq}, between these states gives a measure of which state is most abundant (Park, C. et al., 2004). The balance of forces within protein can be describe in terms of Gibbs free energy. A change in Gibbs free energy between the two states of a protein can be related by the folding equilibrium: $\Delta G = -RT \ln(K_{eq})$ (Park, C. et al., 2004). The equilibrium can be obtained by different experimentally methods, e.g. absorbance, fluorescence, circular dichroism, NMR, activity assays, etc (Murphy, Kenneth P., 2001). Common to all methods is that they require time-consuming laboratory work. In contrast, computationally methods are generally fast. Thus, the complexity and imprecision of a computationally model may be a fair trade-off if the method is fast enough.

The common goal in protein stability prediction is to gain speed and improve accuracy. Essentially we seek to approach the energy of the ideal structure that is offered by X-ray crystallography (Qian, B. et al., 2007). For maximization of stability we try to search for a structure with minimal energy (Desjarlais & Clarke, 1998).

Prediction of stability of a protein gives the opportunity to predict the stability of a protein upon a mutation. Determining the stability of a protein before and after a mutation tells us if a mutation increases or

decreases stability of a given protein. The change in experimentally determined ΔGs upon mutation are given as:
$\Delta\Delta G = \Delta G_{WT} - \Delta G_{MUT}$. We may also describe computationally predicted stability changes via the same method:
$\Delta Stability = Stability_{WT} - Stability_{MUT}$.
Accuracy offered by experimentally determined Gibbs free energy can be used as a way of benchmarking computational models. The correlation of ΔΔG values and ΔStability values can provide us with a correlation coefficient that may be used as a benchmark for a computational model.
In a previous study: 'A bayesian multibody multinomial model as a rapid method of measuring protein stability' a Multibody Multinomial model, *MuMu*, was used as computationally method for determining stability of a protein, 1STN. 95 ΔΔG values were obtained from PROTHERM: "http:/ www.abren.net/protherm/". 95 mutations was introduced in the protein structure of 1STN and *MuMu* was used to determine 95 $-\log P$ values. These values were converted into $-(-\log P)$ and the change in stability was found by subtracting each $-(-\log P)$ of the mutated proteins predictions from the wild type proteins predictions, giving $\Delta\log P = \log P_{WT} - \log P_{MUT}$. A Pearson correlation analysis revealed a correlation between ΔΔG and $\Delta\log P$ of $r = 0.45$ and a hypothesis test rejected a null hypothesis of no correlation. $r = 0.45$ indicates a modest correlation (Taylor, R. 1990). However, the correlation is relatively high in terms of protein stability prediction.
All this was done with a python script that allowed for a fast high-throughput method for gaining stability information through the MuMu model.

MUMU
The multibody multinomial model, MuMu, is a probabilistic model formulated by Johansson, K, et. al. 2013. It is a model that essentially determines the probability of a given amino acid, A_i, conditional on all other amino acids, $A_{\sim i}$, and the structure of the protein, E_i:

$$P(A \mid X) \approx \prod_i P(A_i \mid A_{\sim i}, X).$$

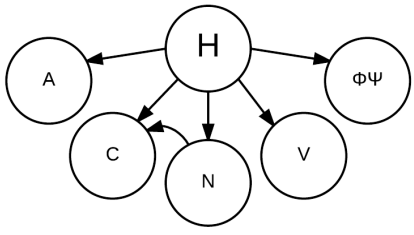
The conditions can be described as the given amino acids environment:

$$E_i = A_{\sim i}, X.$$

The probability of an amino acid is conditional on its environment, which we can describe as being:

$$P(A \mid X) = \prod_i P(A_i \mid E_i) = \prod_i P(A_i \mid C_i, N_i, V_i, \Phi_i, \Psi_i)$$

First shell of neighbors, C_i, the total neighbor count, N_i, the visible volume, V_i, and the backbone angles, Φ_i and Ψ_i. The MuMu model is a bayesian network that has a hidden parent, H:



From the bayesian network we see that we have a joint probability given a hidden parent, h:

$$P(A_i, E_i) = \sum_{h \in H} P(A_i \mid h) P(C_i \mid N_i, h) P(N_i \mid h) P(V_i \mid h) P(\Phi_i, \Psi_i \mid h) P(h)$$

We can expand, by the product rule, the probability of A_i given its environment E_i:

$$P(A_i \mid E_i) = \frac{P(A_i, E_i)}{P(E_i)}.$$

If we insert the joint probability with the hidden parent, we have the MuMu model:

$$P(A_i \mid E_i) = \frac{1}{P(E_i)} \sum_{h \in H} P(A_i \mid h) P(C_i \mid N_i, h) P(N_i \mid h) P(V_i \mid h) P(\Phi_i, \Psi_i \mid h) P(h)$$

It is important to notice, that the model has been trained on non-membrane proteins, proteins that are not disordered, small or protein complexes. MuMu is a part of the Phaistos package that is a framework for Monte Carlo simulations of proteins (Boomsma, W. et. al. 2013). MuMu calculates a probability for each amino acid and returns a negative logarithm to the probability, $-\log P$.

The predicted stability of a given protein is given as the sum of all probabilities of amino acids in the protein:

$$P = \prod_i P(A_i \mid E_i)$$

All values are given as logarithmic values, which means that the values can be summed by addition:

$$\log(P) = \sum_i \log P(A_i \mid E_i)$$

This can be used to give a measure of change in stability upon introduction of a mutation. This value is defined as $\Delta \log P$, which is the change in stability:

$$\Delta \log P = \log P_{WT} - \log P_{MUT}$$

Here I present an analysis of the robustness of MuMu as a accurate high-throughput methods. I expand the dataset to 458 mutations from 15 different proteins. The logarithmic probabilities of the mutated variations are compared to 458 ΔΔG values. I determine the correlation of all logarithmic probabilities and ΔΔG values. I a search for an underlying correlation I also group the data in groups of biochemical properties, secondary structure, and each protein individually.

Method

The previous python script in: 'A bayesian multibody multinomial model as a rapid method of measuring protein stability' was rewritten in order to gain flexibility. The script was made so that it could read .csv files with data that was copy-pasted from PROTHERM. This allowed for a large flow of data that potentially could contain an unlimited amount of proteins and mutations. All the proteins were found by searching for exposed proteins and having ΔΔG values. This resulted in a dataset of 458 mutations with matching ΔΔG values. This was far less than I anticipated as I wanted more than 1000 ΔΔG values. It seems, however that the database only contains relatively few proteins that are exposed. A search of exposed proteins gave a hit of 4000 mutational variations of proteins. Only about a fraction of these contained useable ΔΔG values. The script finds the wanted proteins from the protein data bank (http:// www.rcsb.org/) with help of the BIO.PDB package (T. Hamelryck, et. al. 2003).

MUTAGENESIS

The proteins sequences were parsed and matched with mutations from PROTHERM. These were then passed to SCWRL4, which can predict side-chains of proteins by using data from a library of rotamers (Krivov, G. G., 2009). SCWRL4

was chosen as it one of the fastest command-line tool programs in its field (Harder T, et. al. 2010). SCWRL4 outputs a .pdb file for each mutation.

STABILITY PREDICTION

The wild type protein along with mutated proteins were parsed by the python script, to MuMu in the Phaistos package. The logarithmic probabilities for each mutated .pdb file were summed and then subtracted from the sum of logarithmic probabilities from the wild type .pdb file. This gave a list of $\Delta\log(P(A|X))$ values. The values of the change in logarithmic probability and the ΔΔG values were parsed to a pandas data-frame that is found in the pandas python package, which is an open source library (http://pandas.pydata.org/). The data-frame allowed for easy manipulation of data.

STATISTICAL ANALYSIS

First, data from 'A bayesian multibody multinomial model as a rapid method of measuring protein stability' was successfully replicated by using the new python script. This was done as a test to see if it could produce the same result.

A complete dataset derived from the 15 proteins was made running the python script. Null values were filtered out of the data. 458 values of ΔΔG and 458 were the result after filtering the data. The data was manually assessed to make sure that there were no errors in its calculation.

pyROOT, a python module for analysis and plotting of data (http://root.cern.ch/drupal/content/pyroot), was used to produce a scatter plot with a fitted linear function (figure 1). The assumption for the fit being that no change in probability would mean no change in stability and thus no change in ΔΔG. This mean that the fit was without additional parameters. The $\Delta\log(P(A|X))$ was calculated so that a negative value would indicate increasing stability which is the same for ΔΔG values. The data was separated into 28 different groups, 9 of them based on the physical properties the mutated amino acids. 4 of the groups were based the secondary structure of the wildtype protein at the point of mutation, which were determined by the program DSSP (Joosten, R.P, 2011). 15 of the groups were each individual proteins (supplemental). Like the complete dataset, all these groups were plotted and fitted with a linear function (figure S2-15).

The correlation factor, rho, was calculated for all plots by using pyROOT. The correlation coefficient was transformed into the variable z, as the sample size is not very large:

$$z = \frac{1}{2}\ln(\frac{1+\hat{\rho}}{1-\hat{\rho}}).$$ (Barlow, R. J.)

As z has a more gaussian shape than rho, the standard derivation could be calculated for each correlation:

$$std = \frac{1}{\sqrt{N-3}}.$$ (Barlow, R. J.)

The correlation and its standard derivation is shown in tables (table 1-3).

Results

Figure 1 is a plot of the complete dataset of 458 points. From the fit there is a correlation of: $z = 0.24 \pm 0.05$. This is much lower than the original correlation than for 1STN: $z = 0.47 \pm 0.11$ (table 3). It is clear that an increase in data size has lowered the overall correlation. The individual correlation for each protein is seen in table 1.

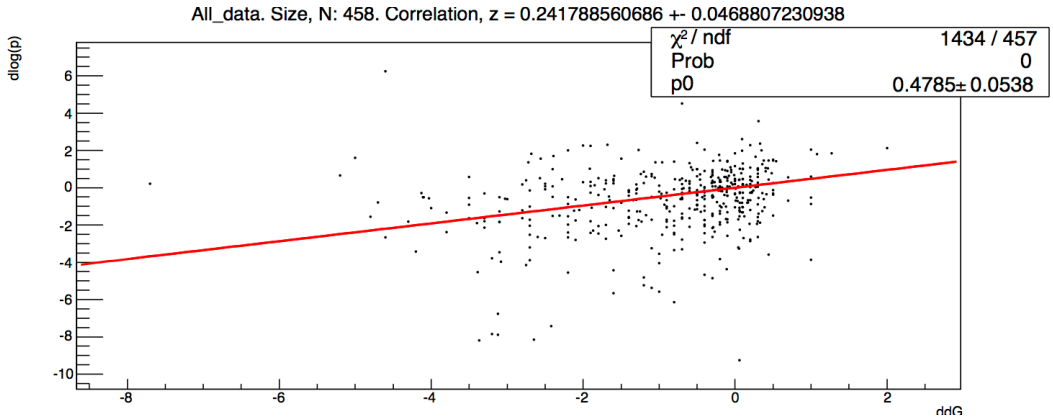


Figure 1. Scatter plot of ΔΔG values, x-axis, and Δlog(p) values, y-axis (complete data set). Data is fitted with a linear function.

Correlation	z	error	N
All data	0,24	0,05	458
1BNI	0,23	0,08	144
1BPI	0,03	0,25	19
1BVC	0,63	0,41	9
1C90	0,28	0,29	15
1CSP	0,17	0,25	19
1CYO	-0,32	0,41	9
1FTG	-0,1	0,41	9
1IGV	-	-	2
1PGA	0,64	0,41	9
1ROP	0,75	0,25	19
1STN	0,47	0,11	93
1VQB	-0,2	0,25	19
2LZM	0,29	0,12	71
2RN2	0,16	0,32	13
3SSI	1	0,45	8

Table 1. Correlation coefficients, z, their errors, and data-size, N. Based on groupings of each individual protein.

It is clear that 1STN contributes positively to the correlation. There are some proteins that have a very high correlation, e.g. 1BVC, 1PGA, and 1ROP. There are also some proteins that have a negative contribution to the correlation, e.g. 1CYO, 1FTG, 1VQB, and 1BPI. Common to them all, however, is a large error and most important, a small sample size. It is difficult to conclude anything from that small sample sizes, the largest being 19 points. The only proteins that have a sensible sample size is 2LZM, 1BNI, and 1STN with correlations of $z = 0.29 \pm 0.12$, $z = 0.23 \pm 0.08$, $z = 0.47 \pm 0.11$, respectively. It seems that MuMu is better at predicting stability of 1STN than the two others. The values are compared to correlations determined by Potapov, et. al. 2009 in table 2.

r	MuMu	EGAD	Fold X	Hunter	I-Mutant2.0	Rosetta
1STN	0.45	0.62	0.73	0.70	0.62	0.28
1BNI	0.23	0.17	0.73	0.55	0.83	0.42
2LZM	0.29	0.48	0.63	0.57	0.87	0.23

Table 2. Correlation coefficients, based on data from MuMu, EGAD, Fold X, Hunter, I-Mutant2.0, and Rosetta. Data from Potapov, et. al. 2009.

Table 2 shows that MuMu generally performs worse than other programs, excluding Rosetta.

Table 3 shows the correlations of groups based on physical properties of the mutated amino acids.

	z %	error on z %	N
All data (figure S1)	24	5	458
Mutated to charged amino acid (figure S2)	44	11	93
Mutated to uncharged amino acid (figure S3)	20	5	345
Mutated to aromatic amino acid (figure S4)	-20	24	20
Uncharged amino acid mutated to charged amino acid (figure S5)	25	8	178
Charged amino acid mutated to uncharged amino acid (figure S6)	65	15	50
Mutated to hydrophobic amino acid (figure S7)	22	6	271
Mutated to hydrophilic amino acid (figure S8)	15	10	98
Hydrophobic amino acid mutated to hydrophilic amino acid (figure S9)	43	17	38
Hydrophilic amino acid mutated to hydrophobic amino acid (figure S10)	40	12	70

Table 3. Correlation coefficients, z, their errors, and sample sizes, N, from MuMu data and ΔΔG values in groups based on physical properties at the point of mutated amino acids. The data is plotted in figure S1-S10.

Charged amino acid mutated to uncharged (peptide) amino acid (figure S6) and mutations from anything to a charged amino acid (figure S2) have relatively high correlation (table 3), in terms of structure prediction. It would seem that they have a high contribution to the correlation of the data. A manual inspection of the data reveals that the data does seem somewhat random and it is not clear if that is the case without collecting more data. The correlation coefficient for mutations of polar amino acids to hydrophobic amino acids is $z = 0.40 \pm 0.12$ (table 3). It does look like there is a linear tendency (figure S10). MuMu may be better at predicting stability when there are mutations from hydrophilic to hydrophobic. Mutations that lead to aromatic amino acids have an negative overall influence on the correlation. However it is inconclusive if this is the case due to the low sample size of 19 and a large spread in the data (figure S4).

Table 4 shows the correlation of groups categorized on secondary structure.

Correlation	z	error	N
Helix (figure S12)	0.23	0.08	153
Turn (figures S13)	0.21	0.11	86
Strands (figures S14)	0.38	0.12	72
Non (figure S15)	0.33	0.11	87

Table 4. Correlation coefficients, z, their errors, and sample sizes, N, from MuMu data and ΔΔG values in groups based on secondary structure at the point of mutated amino acids. The data is plotted in figure S12-S15.

Table 4 shows a higher correlation of MuMu predicted values in extended strands (residues in isolated beta-bridges)(figure S14). For the Helix and Turns there is a relatively low correlation (figure S12-S13). The Non correlation is for amino acids in areas with secondary structure that could not be determined (figure S15). It seems that MuMu is better at Non-defined regions than regions of Helixes or Turns.

Conclusion

It seems that the MuMu data is gaussian distributed (figure S16), however the ΔΔG values does not follow a gaussian shape (figure S17), it seems that the mutations are mostly favorable for protein stability with negative ΔΔG values. A recurring thing is the need for a larger sample size. This is a problem especially when looking at the subsets. There is an indication that MuMu might be better at predicting stability of proteins with mutations of hydrophobic

amino acids to hydrophilic. The fit for the entire dataset is not good. It may be that there is a need for optimization of linear regression. The analysis, however, does not lead to any obvious underlying correlation that could be used to optimize the general fit. Though seems that MuMu is better at predicting areas with turns in the secondary structure.

It was shown that MuMu could be used in a high-throughput method for determining stability on an unlimited amount of proteins and mutations.

The predictions does not correlate well enough with ΔΔG values in order to call MuMu an accurate method for protein stability prediction. It may be that, in the given protein, there are some underlying protein dynamics that are not suitable for MuMu prediction.

Perhaps more light could be shed on the fits by doing additional analyze, e.g. it would be interesting to see if there is a correlation with MuMu data and the Accessible Surface Area, ASA, of the proteins.

The main advantage of MuMu is its speed. MuMu is able to do stability predictions of approximately 31000 residues/minute in comparison we have:

I-mutan2.0: 6-8 residues/minute.

Hunter: 0.2-0.4 residues/minute.

PoPMuSiC: 4300-9000 residues/minutes.

MuMu is 3-7 times faster than PoPMuSiC. This effectively makes MuMu one of the fastest protein stability prediction tools. What MuMu lacks in accuracy it has in speed, which makes MuMu a good tool for analyzing large amounts of data. Due to its speed it could be used as a way of analyzing complete protein structures for suitable mutations for stabilization. MuMu could introduce all possible mutations, calculate a probability for each and by a cut-off value it would be able to recommend suitable mutations based on the users preferences. Thus, it may well be a tool that can be used in protein design applications.

Learning

All the data in this project was produced by using python together with a combination of programs. I only just learned python last year. Due to this project I am already quite confident in my python skills, which have evolved during this project. I have furthered my understanding in the probabilistic model of MuMu and the general understanding of a Bayesian network and Monte Carlo simulations. I have been able to apply statistical analysis on the data in order to get meaningful comparative data. I believe that this project have pushed me towards being a better bioinformatician.

Literature

ARTICLES

Taylor, R. (1990). Interpretation of the Correlation Coefficient: A Basic Review. Journal of Diagnostic Medical Sonography, 6(1), 35–39. doi: 10.1177/875647939000600106

Hamelryck, T., & Manderick, B. (2003). PDB file parser and structure class implemented in Python. Bioinformatics Applications Note, Vol. 19. no. 17, 2308-2310. doi: 10.1093/bioinformatics/btg299

Boomsma, W., Frellsen, J., Harder, T., Bottaro, S., Johansson, K. E., Tian, P., ... Hamelryck, T. (2013). PHAISTOS: a framework for Markov chain Monte Carlo simulation and inference of protein structure. Journal of Computational Chemistry, 34(19), 1697–705. doi:10.1002/jcc.23292

Potapov, V., Cohen, M., & Schreiber, G. (2009). Assessing computational methods for predicting protein stability upon mutation: good on average but not in the details. Protein Engineering, Design & Selection : PEDS, 22(9), 553–60. doi:10.1093/protein/gzp030

Park, C., & Marqusee, S. (2004). Analysis of the stability of multimeric proteins by effective ○, 2553–2558. doi:10.1110/ps.04811004.1

Roland L. Dunbrack Jr, Chase, F., & Avenue, B. (2002). Rotamer libraries in the 21 st century, 431–440.

Voigt, C. a, Gordon, D. B., & Mayo, S. L. (2000). Trading accuracy for speed: A quantitative comparison of search algorithms in protein sequence design. Journal of Molecular Biology, 299(3), 789–803. doi:10.1006/jmbi.2000.3758

Qian, B., Raman, S., Das, R., Bradley, P., McCoy, A. J., Read, R. J., & Baker, D. (2007). High-resolution structure prediction and the crystallographic phase problem. Nature, 450(7167), 259–64. doi:10.1038/nature06249

Krivov, G. G., Shapovalov, M. V, & Dunbrack, R. L. (2009). Improved prediction of protein side-chain conformations with SCWRL4. Proteins, 77(4), 778–95. doi:10.1002/prot.22488

Harder, T., Boomsma, W., Paluszewski, M., Frellsen, J., Johansson, K. E., & Hamelryck, T. (2010). Beyond rotamers: a generative, probabilistic model of side chains in proteins. BMC Bioinformatics, 11, 306. doi: 10.1186/1471-2105-11-306

Johansson, K. E., & Hamelryck, T. (2013). A simple probabilistic model of multibody interactions in proteins. Proteins, 81(8), 1340–50. doi:10.1002/prot.24277

Meeker, a K., Garcia-Moreno, B., & Shortle, D. (1996). Contributions of the ionizable amino acids to the stability of staphylococcal nuclease. Biochemistry, 35(20), 6443–9. doi:10.1021/bi960171

Robbie P. Joosten, Tim A.H. te Beek, Elmar Krieger, Maarten L. Hekkelman, Rob W.W. Hooft, Reinhard Schneider, Chris Sander, and Gert Vriend. A series of PDB related databases for everyday needs. Nucleic Acids Research 2011 January; 39(Database issue): D411-D419. doi: 10.1093/nar/gkq1105 PMCID: PMC3013697

BOOKS

Roger J. Barlow. Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences.

Murphy, Kenneth P. Protein Structure, Stability, and Folding. Totowa, NJ: Human, 2001.

LINKS & PROGRAMS

Python: https://www.python.org/

DSSP: http://swift.cmbi.ru.nl/gv/dssp/

SCWRL4: http://dunbrack.fccc.edu/scwrl4/

SPECIAL THANKS

My supervisor Thomas Hamelryck for supervising (now properly spelled).

Old but still applicable;

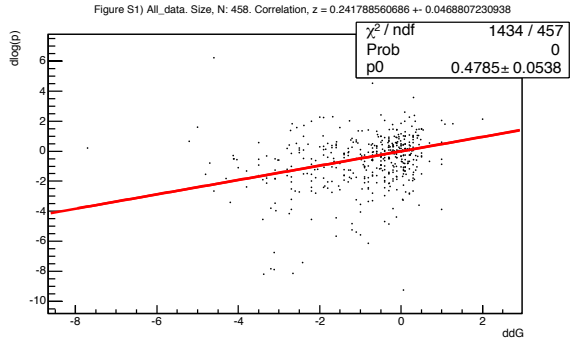
Kristoffer Johansson for providing articles and guidance.
Wouter Boomsma for hours spent setting me up with Phaistos.

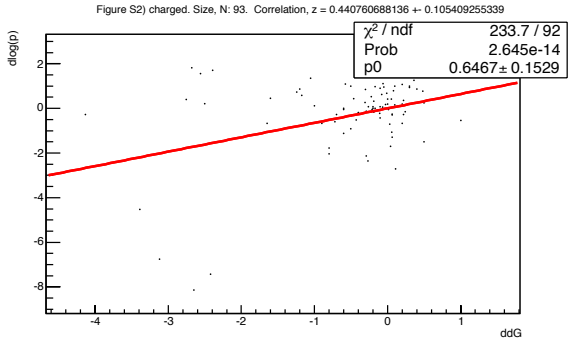
APPENDIX

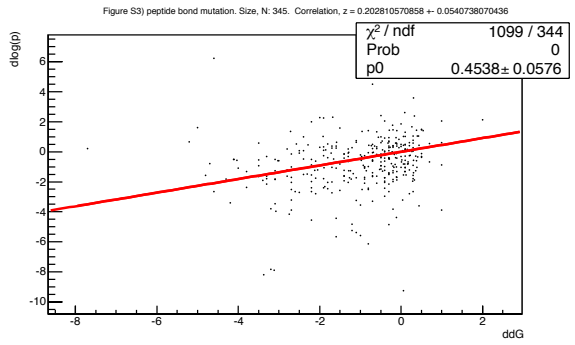
Supplemental figures are found on the following pages. Except supplemental figures of individual proteins, they are handed in as individual image files. All scripts associated this project are delivered as external files. Generated data is available on request.

Supplemental

FIGURE S1-S4







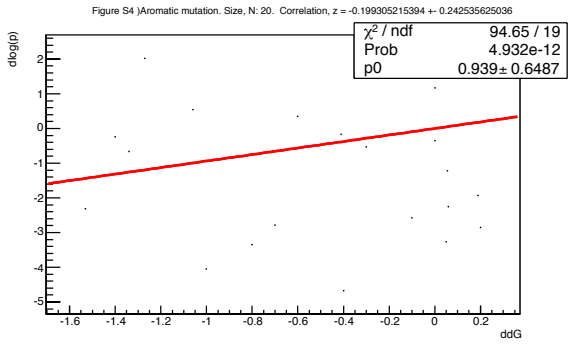
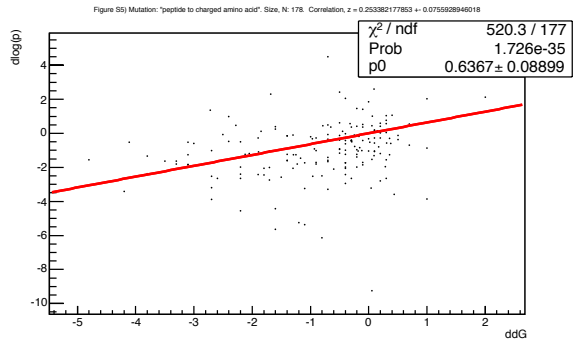
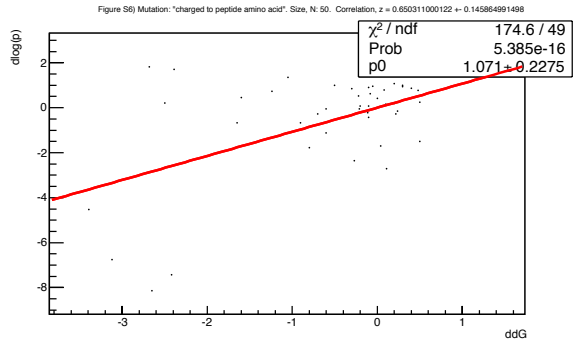
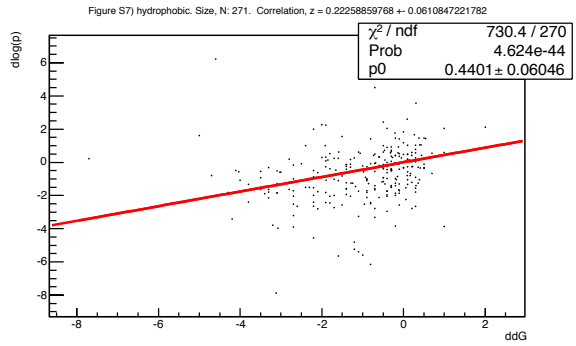


FIGURE S5-S8







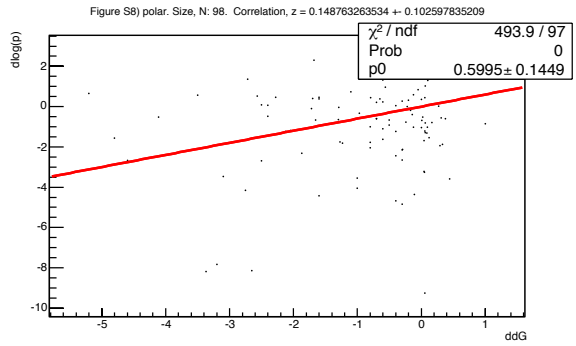
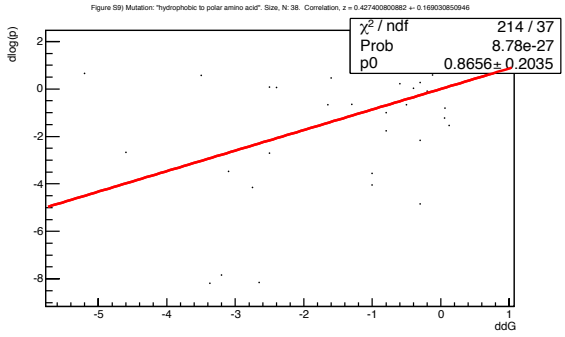
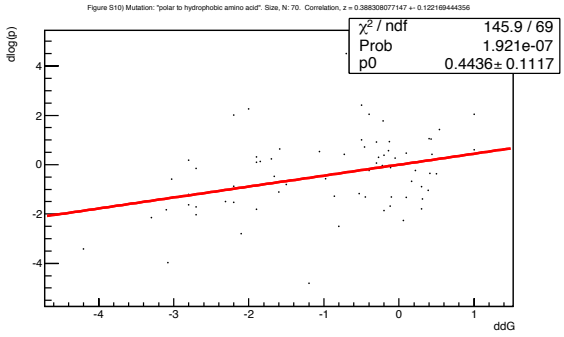
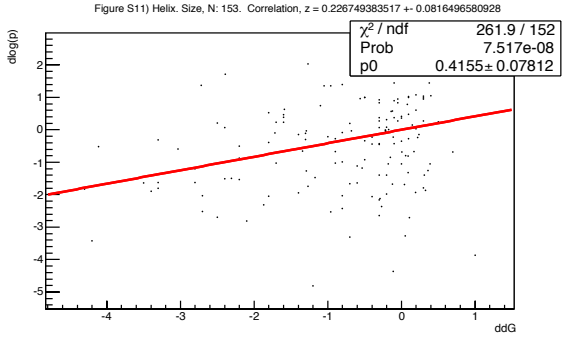


FIGURE S9-S13







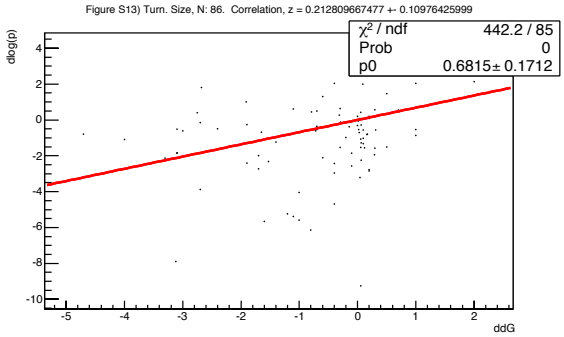


FIGURE S14-S19