Distribution Agreement

In presenting this thesis as a partial fulfillment of the requirements for a degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis in whole or in part in all forms of media, now or hereafter now, including display on the World Wide Web. I understand that I may select some access restrictions as part of the online submission of this thesis. I retain all ownership rights to the copyright of the thesis. I also retain the right to use in future works (such as articles or books) all or part of this thesis.

Nithin Bagal 04/12/2022

Using Machine Learning Algorithms to Predict Conditions for Protein Crystallization

by

Nithin Bagal

Dr. Katherine Davis
Adviser

Department of Chemistry

Dr. Katherine Davis

Adviser

Dr. Joel Bowman

Committee Member

Dr. James Kindt

Committee Member

2022

Using Machine Learning Algorithms to Predict Conditions for Protein Crystallization

By

Nithin Bagal

Dr. Katherine Davis

Adviser

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Department of Chemistry

2022

Abstract

Using Machine Learning Algorithms to Predict Conditions for Protein Crystallization

By Nithin Bagal

Radical *S*-adenosyl-methionine (rSAM) enzymes comprise a large, primarily uncharacterized metalloenzyme superfamily, important for the biosynthesis of a wide range of natural products across many living organisms. All constituent members rely on the reductive cleavage of SAM bound to a [4Fe-4S] cluster to generate a highly reactive 5'-deoxyadenosyl radical. This radical mechanism facilitates challenging chemistries and enables a diverse array of reactivities including methylation and isomerization of substrates. However, little is known about the structural basis for this impressive breadth of reaction outcomes. Protein crystallography is a powerful method for determining the 3-dimensional structure of proteins, but the logistical complications of this method, coupled with rSAM enzyme sensitivity to molecular oxygen complicate crystallization of enzymes in this superfamily. Correspondingly, structures are rare.

Using machine learning methods to predict the conditions in which a crystal will grow for a given protein would greatly increase the efficiency of protein crystallography and aid in developing a deeper understanding of the rSAM family. This goal was partitioned into two aims: (1) Developing a clear understanding of current protein crystallography methodology by crystallizing the rSAM enzyme YydG. (2) Developing machine learning algorithms to predict crystallization conditions for the rSAM enzyme SuiB. After aim one was met and sufficient expertise was gained with protein crystallography through working with YydG, three machine learning models were applied to crystallization data for SuiB. The models all performed higher than 50% accuracy, indicating that a computational approach to predicting crystallization conditions can improve the process of developing protein crystals.

Using Machine Learning Algorithms to Predict Conditions for Protein Crystallization

By

Nithin Bagal

Dr. Katherine Davis

Adviser

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Department of Chemistry

2022

Acknowledgements

I would like to acknowledge and thank my committee members, Dr. Katherine Davis, Dr. Joel Bowman, and Dr. James Kindt for providing the guidance and resources necessary to complete this project. I would also like to thank my graduate student mentor Tamra Blue for her assistance and guidance through this project, as well as the rest of the Davis Lab for providing such a great environment for me to learn and work in.

Table of Contents

Introduction:

Enzymes have the potential to conduct complicated reactions under homeostatic conditions in living cells. Reactions that would be incredibly challenging or not possible in a reaction vessel are routinely executed in the active sites of enzymes, often using mechanisms that would not be feasible outside of a living system.[1] Enzymes accomplish this task by orienting the substrates in the active site in a manner that lowers the activation energy necessary to the reaction to proceed.[2]

The radical *S*-adenosyl-methionine (rSAM) superfamily is an underexplored [Fe/S] containing metalloenzyme superfamily involved in the biosynthesis of a wide range of natural products.[3] These versatile enzymes are found across all domains of life and traditionally rely on the reductive cleavage of a SAM molecule bound to a [4Fe-4S] cluster as shown in Fig. 1 to generate a highly reactive 5'-deoxyadenosyl (5'-dA) radical as seen in Fig. 2. This [4Fe-4S] cluster is housed in what is dubbed, the radical SAM domain, bound to a canonical $CX_3CX\phi C$ motif where ϕ is an aromatic residue.[3] Enzymatic turnover is initiated when an iron atom of the [4Fe-4S] cluster chelates with the SAM molecule to form a deoxyadenosyl radical. This highly reactive radical facilitates challenging and diverse chemistries, including C-H functionalization and isomerization of their substrates.[4]

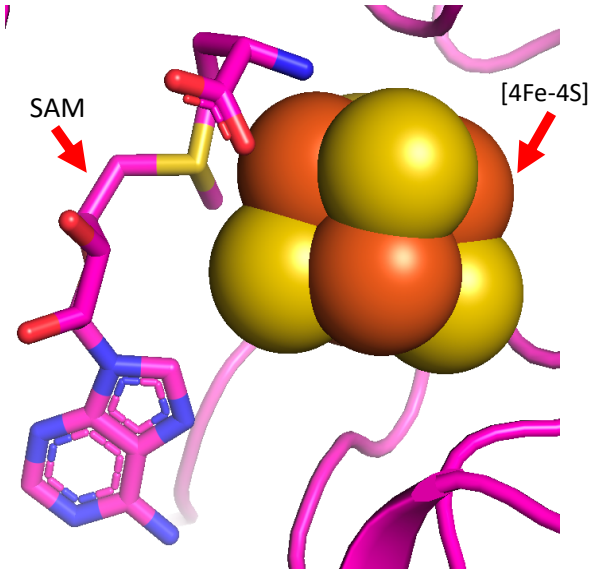

Figure 1. SAM and methionine bound to [4Fe-4S] cluster (PDB: 3IIZ) (Nicolet et. al, PNAS, 2009)

A subset of rSAM enzymes are involved in the biosynthesis of ribosomally-synthesized and post-translationally modified peptides (RiPPs).[4] RiPPs can be divided into a diverse set of molecular classes including, but not limited to, lanthipeptides, proteusins, linear azole containing peptides (LAPs), and cyanobactins.[5] These peptides once modified by their respective rSAM enzymes, go on to be used for

functions such as anti-microbial defense, anti-tumor, and anti-malarial activity making them promising targets for pharmaceutical investigation.[6] Therefore, understanding the mechanisms for the biosynthesis of these peptides is of paramount importance.

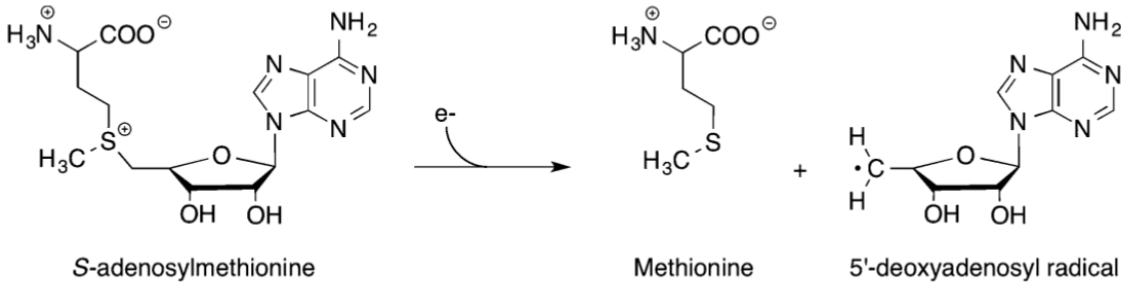

Figure 2. Cleavage of SAM into methionine and 5'-dA radical (Broderick et. al, FChem, 2014)

A big step in understanding the interaction between peptide substrates and tailoring enzymes in RiPP biosynthesis came from the identification of a potential RiPP Recognition Element (RRE).[7] This sequence of approximately 100 amino acids forms a winged helix-turn-helix motif (wHTH) as seen in Fig. 3 that is hypothesized to bind to the substrate and shuttle it into the active site of the enzyme. However, not all of the enzymes in the rSAM family contain this RRE which can exist as a distinct protein in the gene cluster, or fused to a larger protein domain.[8] This leaves open the question of how substrate selectivity is achieved.

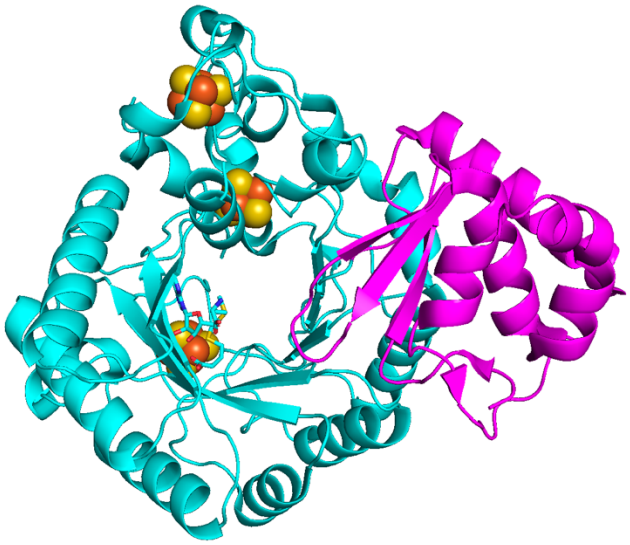

Figure 3. Fig 3. RRE region of SuiB from Streptoccocus Suis highlighted in magenta (PDB 5V1T) (Davis et. al, PNAS, 2017)

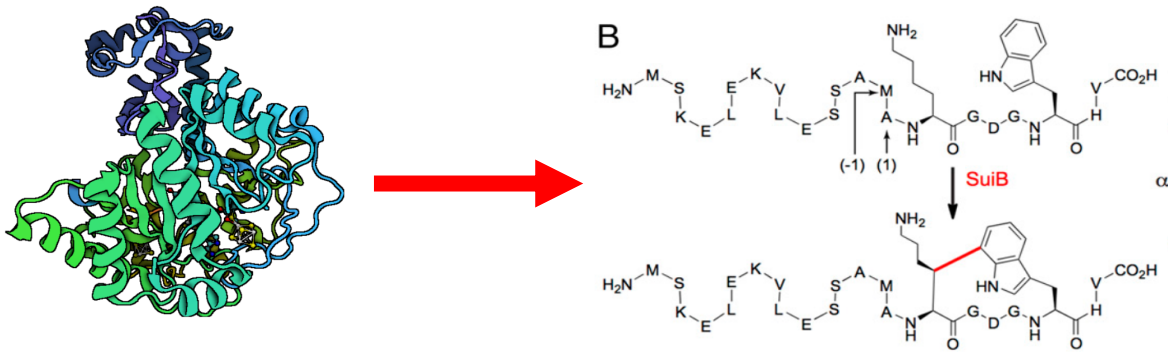

Figure 4. rSAM enzyme SuiB installs a Lysine-Tryptophan Crosslink in its substrate peptide SuiA. (PDB 5V1T) (Davis et. al, PNAS, 2017)

rSAM enzyme SuiB, native to *Streptoccocus Suis,* is one example of an enzyme in this class that contains an RRE, however substrate peptide SuiA was not shown to be bound to this domain its crystal structure.[3] SuiB performs the installation of a lysine-tryptophan crosslink on SuiA as shown above in Fig. 4.[9] After an electron the [4Fe-4S] cluster cleaves SAM into methionine and 5'-dA, the 5'-dA radical reacts with the lysine to form a lysyl radical. This radical then reacts with the tryptophan to install this crosslink. SuiA is a streptide peptide implicated in quorum sensing of *Streptoccocus Suis*.[3]

YydG is an example of an rSAM that does not possess the RRE, and therefore crystallization of this enzyme with its substrate peptide YydF could help elucidate binding mechanisms for the rSAM enzymes that do not have the RRE motif.

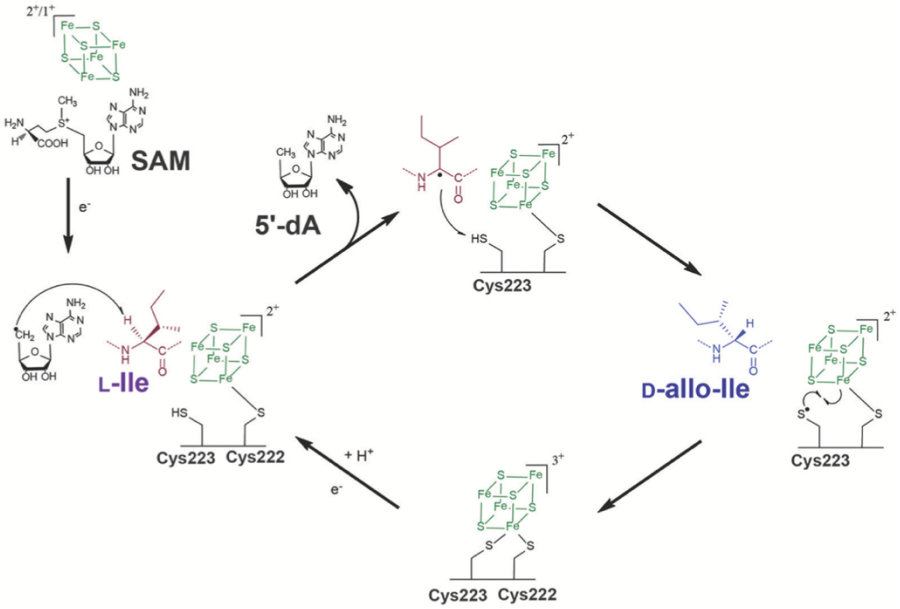

Figure 5. Epimerization of YydF substrate by YydG. Ile in YydF is converted from L form to D-allo-Ile. (Bendjia et. al, Nat. Chem, 2017)

YydG is a 37kDa peptidyl epimerase native to *Bacillus subtilis* that epimerizes an Isoleucine and Valine

residue in its substrate peptide YydF.[10] The mechanism for the epimerization of the Isoleucine is shown in Fig. 5.[11] The 5'-dA radical generated by the [4Fe-4S] cluster abstracts a hydrogen from the L-Ile in the YydF substrate. This now radical Ile abstracts a hydrogen from the Cys 223 residue in YydG, which inverts the stereochemistry of the alpha carbon. The radical sulfur on the Cys223 then binds to one of the iron atoms in the cluster, and the catalyst is regenerated by an electron and proton from solution. However, primary sequence analysis shows that the enzyme could contain up to three [4Fe-4S] clusters through the identification of three cysteine motifs. One of these motifs is the rSAM domain, shown in the primary sequence of YydG in Fig. 6, with the remaining motifs potentially being SPASM motifs. SPASM motifs are known to bind auxiliary iron sulfur clusters although their exact role in the catalytic activity of rSAMs is not known.[12]

```
  1 mynktvsinl dsrcnascdh ccfsssptst trmekeyire lvtefaknkt iqvisftgge
 61 vfldykflke lmeiikpyek qitlisngfw glskkkvqey fhdmnslnvi altisydeyh
121 apfvksssik nilehsrkyp didislnmav tkdkmsnhil eelgdsilgv kitkfpmisv
181 gaaktrikqe nihkfysled edslhcpgyd ivyhhdgeiy pccspaifet kitlreeynq
241 sfertvekln snlllfilrk egfkwflnil kennkieefd ipyefssicg vcgslfnsae
301 kinyfypyme kyynenfkv
```

Figure 6. Primary sequence YydG with rSAM domain boxed. (Yoshida et. al, DNA res., 1994)

YydG plays an important role in *Bacillus subtilis* by regulating the LiaRS protein.[13] The LiaRS protein is implicated in cellular stress detection specifically in relation to the cell wall.[14] LiaRS has been noted to be upregulated when bacteria are exposed to antibiotics that specifically target the cell wall such as vancomycin and bacitracin. Understanding the structure and mechanism of YydG could broaden our understanding of antibiotic resistance and the mechanisms by which bacterial cells evade antibiotic treatments.[11]

Elucidating the structure of enzymes in the rSAM family is a key step in understanding their function, especially for examining the basis for interaction with their substrates. However, the production of protein crystals is a labor-intensive process that requires large quantities of purified protein. Instead, there are many computational methods that have been applied to the field of structure determination. For

instance, a recent breakthrough in the field came in the form of AlphaFold, a neural network that bypasses any wet lab work to predict a structure of a protein given a primary sequence.[15] By relying heavily on the thermodynamics, kinetics and physics of protein folding, the model outputs a structure along with a metric score in each region of the protein as to the confidence of its prediction. While this technology is a massive leap forward in computational prediction, it is not quite ready to replace protein crystallography. AlphaFold struggles with several more challenging aspects of structure prediction such as prediction of multiple states of a protein, protein structure inside a multimer, and accurate predictions of intrinsically disordered regions (IDRs).[16]

Furthermore, there exists a field of computational structure prediction known as template-based methods, where an existing known structure is used to predict the structure of an unknown molecule if the homology between the two is high. However, the clear limitation of this method is that a solved structure for a protein with high homology to the unknown must exist first.[17] Current template-based models include RaptorX, which identifies highly homologous sequences in different proteins, and predicts from a set of common stable folds which shape that sequence is most likely to take.[18]

Template free methods currently exist as well. For example, the Rosetta server uses a monte-carlo method to repeatedly predict the structure of small sections (3-9 residues) of a polypeptide to ultimately arrive at a structure of a larger peptide.[19] However, due to the small scope at which this model performs calculations, the computational cost of this method steeply increases with larger macromolecules. Accuracy also starts to suffer with peptides longer than 150 residues, as the forcefields used to perform predictions become less able to accurately reflect interactions in these large polypeptides.[20]

All of the methods mentioned above apply computational chemistry to try to bypass the need for structure determination through protein crystallography. However, they are not currently viable options to replace the wet lab work necessary to solve a structure for a protein. For this reason, I plan to apply to

machine learning methods to improve the process of protein crystallography itself as opposed to trying to replace it.

Being able to predict which conditions will yield a protein crystal would greatly reduce the time and resources necessary to determine macromolecular structures. Not only would vast amounts of purified protein be saved, and therefore large quantities of reagent, but knowing what conditions will grow a protein crystal will also vastly decrease the waiting time for good morphology crystals, contributing to faster solving of 3-dimensional structures of rSAM enzymes.

My goal is to apply machine learning methods to predict which conditions in a sparse matrix screen will cause a protein to crystallize. By applying machine learning to the ever-growing repository of crystallization conditions, models can be trained to identify which conditions will yield a protein crystal for a given protein. Furthermore, these methods are easily accessible through python packages such as scikitlearn,[21] with extensive documentation on their use and industry practices for accuracy quantification.

Before being able to improve the process of protein crystallography of rSAM enzymes, it was necessary to work in wet lab conditions to try to crystallize an rSAM enzyme to understand the current methodology. Therefore, the overall goal of using machine learning to improve the process of protein crystallography was split into two aims as follows.

Aim 1: Become familiar with the current methodology of protein crystallography by crystallizing rSAM enzyme YydG

Aim 2: Use machine learning algorithms to predict protein crystallization conditions for rSAM enzyme SuiB.

Aim 1 Background:

One method for determining the structure of macromolecules such as proteins is through X-ray crystallography. When protein crystals are hit with a high intensity X-ray beam, the resulting diffraction

pattern can be used to calculate a 3-dimensional structure. After this structure is solved, it is deposited in the Protein Data Bank (PDB),[22] which is a repository of solved 3-dimensional structures of proteins. Generally speaking, the process of producing a protein crystal involves production of the protein of interest, usually through the expression of a plasmid vector containing the gene for the protein of interest inside a bacterial system. After lysis of this bacterial system, this protein is isolated through a series of chromatography columns. The amount of purification needed is unique to each protein. Methods can include, but are not limited to size exclusion chromatography, nickel-nta resin chromatography, and/or anion/cation exchange chromatography. Following the chromatography, sparse matrix screens are used to identify the conditions necessary to form a crystal. These screens comprise a combination of different crystallization reagents including but not limited to buffers, precipitating agents, and salts. Purified protein is mixed with a buffered solution that is hypertonic compared to an adjacent reservoir solution. After the purified protein is placed into each well of this sparse matrix screen, vapor diffusion that occurs from the protein sample into the reservoir solution in the ideal conditions should leave behind a protein crystal as the protein solution becomes more and more saturated. Ideal conditions will slowly move the protein crystal into a concentration that causes nucleation, as opposed to a hyper-saturated or hypo-saturated solution. After these conditions that produced a crystal are identified, they are further optimized by adjusting factors such as pH, and concentration of salts and/or precipitating agents to produce diffraction quality crystals with good morphology.[4]

Aim 1 Methods:

BL21 *Escherichia coli* cells containing two plasmids relevant to the full expression of YydG were provided by my graduate student mentor Tamra Blue. One kanamycin resistant plasmid was used for the expression of YydG along with a C-terminal 6x His tag, and another ampicillin resistant isc plasmid was used to express the machinery necessary for incorporation of the [4Fe-4S] cluster into the protein.[23] Both plasmids are shown in Fig. 7.

Expression was conducted in pseudo-anaerobic conditions due to [4Fe-4S] cluster sensitivity to oxygen using the following method.

1.5L of LB media was inoculated with 10 ml of an starter culture of the BL21 *E.coli* cells and supplemented with ampicillin (100 μg/ml) and kanamycin (50 μg/ml). This culture was left to shake at 240 RPM at 37 °C until ocular density of the culture reached 0.3. Ocular density was measured taking 500 μl samples of the culture at OD600 via a nanodrop. When an ocular density of 0.3 was reached, arabinose (0.1w/v) solution was added along with L-Cys (60 mg) and FeCl3 (200 mg) were added and the culture. Temperature was reduced to 18 °C while shaking. When OD ~0.6, IPTG (200 mg) was added to induce protein of interest plasmid. This was then allowed to grow overnight while shaking at 18 °C.

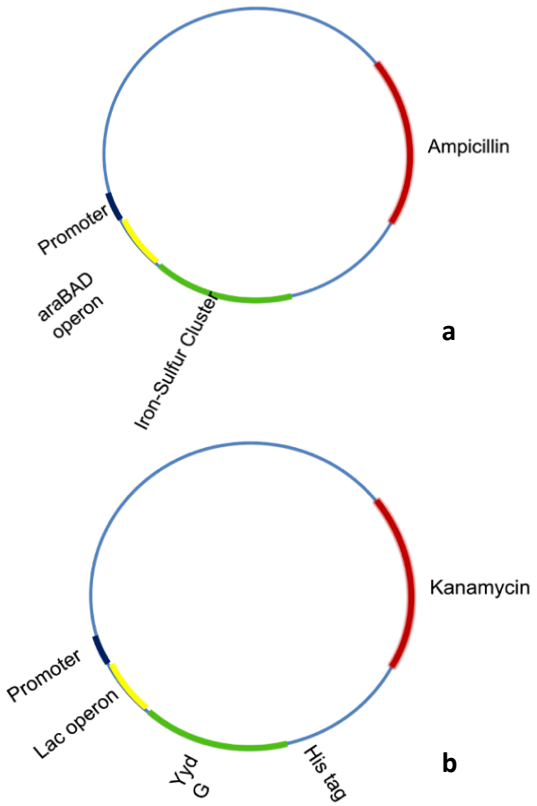

Figure 7. a) Plasmid used to express the machinery necessary to install the [4Fe-4S] cluster. b) Plasmid to express YydG

E.coli cells were then harvested by centrifuging the culture at 9000g for 30 mins. Cells were then prepared for nickel purification by sonication lysis inside an anaerobic glovebox to prevent exposure of the protein to oxygen. For chemical lysis, the pellet was immersed in a lysis buffer (40 mM Tris, 300 mM NaCl, 20 mM imidazole, 10 mM Beta mercaptoethanol (BME), lysozyme 1 μg/ml of buffer, DNAse at 1μl/ml of buffer, 1 mM PMSF, and 0.1% protease inhibitor cocktail). The pellet was mixed with a stir bar until resuspended, and then spun down at 30,000g for 30 mins.

The supernatant was collected for purification with nickel affinity resin. This resin was able to be used because of the His tag attached to the C terminus of the protein. The nickel resin was first equilibrated

with 5 column volumes of lysis buffer. The supernatant was then poured through and washed with a series of buffers containing progressively higher concentrations of imidazole, as higher concentrations would begin to displace the His tag from the resin and cause the protein to flow through. The buffers contained 40mM tris at pH 7.5, 10% glycerol, 300 mM NaCl, 10 mM BME, and concentrations of imidazole starting at 45mM and ending at 200 mM. The flow throughs from each one of these washes were collected, and samples were run on an SDS page gel to determine what fractions of washes contained relatively pure YydG.

After relevant washes were collected, they were concentrated using 10kD amicon filter tubes. Five mls of the concentrated YydG was then loaded onto a 5 ml superloop. This superloop was connected to a size exclusion column on an Akta go fast protein liquid chromatography machine. After fractions determined to contain YydG were obtained from the size exclusion, they were once again concentrated down using the amicon tubes mentioned above. The concentrated YydG was then buffer exchanged using a PD10 column into anion exchange buffer (40 mM tris HCl, 10 mM NaCl) and washed off the column with anion elution buffer (40 mM Tris HCl and 2 M NaCl). The collected fraction of YydG was analyzed by Bradford assay and concentrated to 12 mg/ml. This crystallography grade sample was used to set up sparse matrix screens with the plates mentioned earlier provided by Hampton Research.

1 μl of protein was placed mixed with 1 μl of reservoir solution on the pedestal of each well in the 96 well plate, with 50 μl of reservoir solution sitting in the reservoir. These plates were then sealed and left in an anaerobic glovebox for crystals to develop using the principle of vapor diffusion mentioned above.

Aim 1 Results:

By working with YydG, understanding of the conventional methodology of crystallization was gained. Potential for improvements that a computational approach to the sparse matrix screen also became clear after becoming familiar with the current protocol. Currently, at the time of submission of this

manuscript, several crystals of YydG have been sent to Argonne National Lab's X-ray beam with the goal of generating diffraction patterns that can be used to solve the structure of YydG.

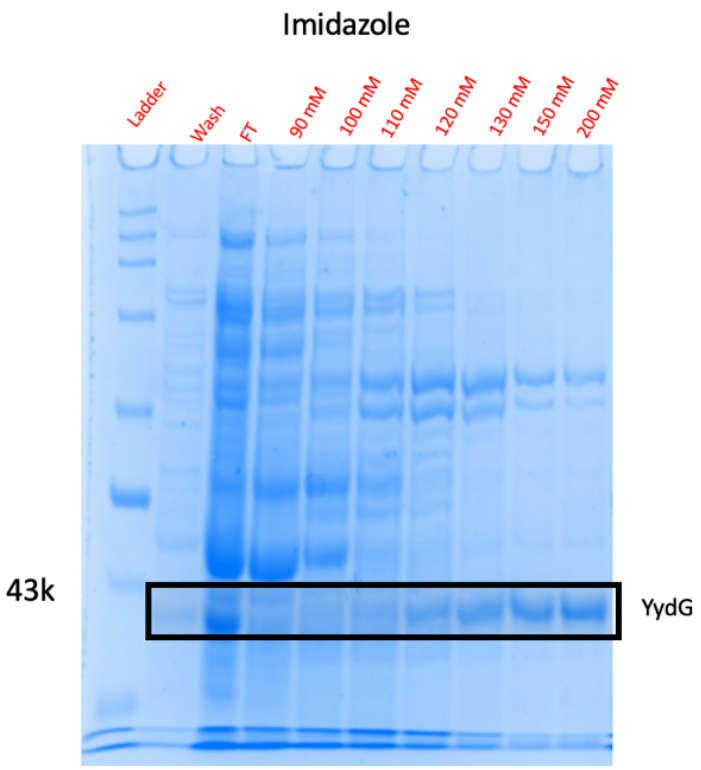

Figure 8. SDS-Page gel showing usable fractions of YydG being eluted at 120mM Imidazole

Working with this enzyme was challenging due to its high sensitivity to factors such as temperature and buffer conditions, and oxygen. For this reason, the expression and purification protocol was repeatedly optimized. For example, the expression protocol was modified to optimize the amount of YydG being expressed in the *E.coli* cells. The OD checkpoint that was used as a marker to begin expression of the plasmid producing YydG was changed from 0.8 to 0.6 to allow for the cells to spend more time growing the protein.

With respect to purification, the nickel affinity purification was changed from a wash to elution purification to a gradient elution. The gradient elution method allowed us to isolate more YydG off of the nickel column, as well as collect more clean fractions of the protein. As can be seen in the Fig. 8, the band containing YydG starts being eluted off the column with imidazole concentrations as low as 120 mM. However, it can also be seen from the gel that further purification is necessary as with the elution bands, there are bands above YydG that are also present in the sample. These proteins could be proteins expressed by the first plasmid in the cell meant to express the machinery necessary to create the [4Fe-4S] cluster.

After these relevant elutions of YydG were collected size exclusion chromatography greatly increased the purity of the sample, but the collected fractions were still not of crystallization grade purity as can be seen in Fig 10. Size exclusion chromatography separates out molecules by trapping smaller molecules in longer channels of an agarose matrix. Larger molecules bypass these channels and are eluted first.[24] For collected fractions A-E, there is still some of the unwanted protein sitting above YydG in the gel.

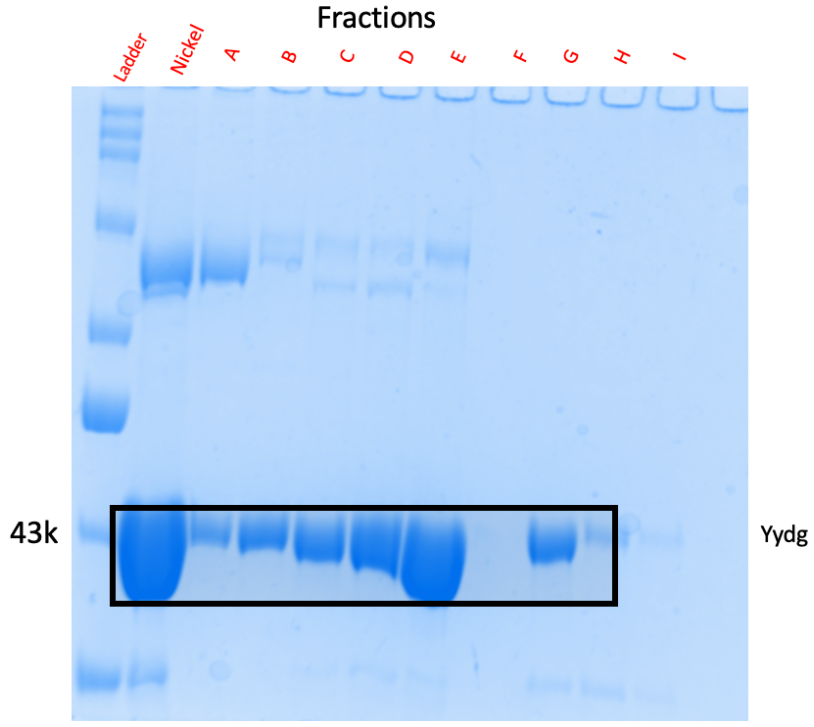

Figure 9. SDS-Page gel showing results of size exclusion chromatography. There are still some impurities in collected fractions.

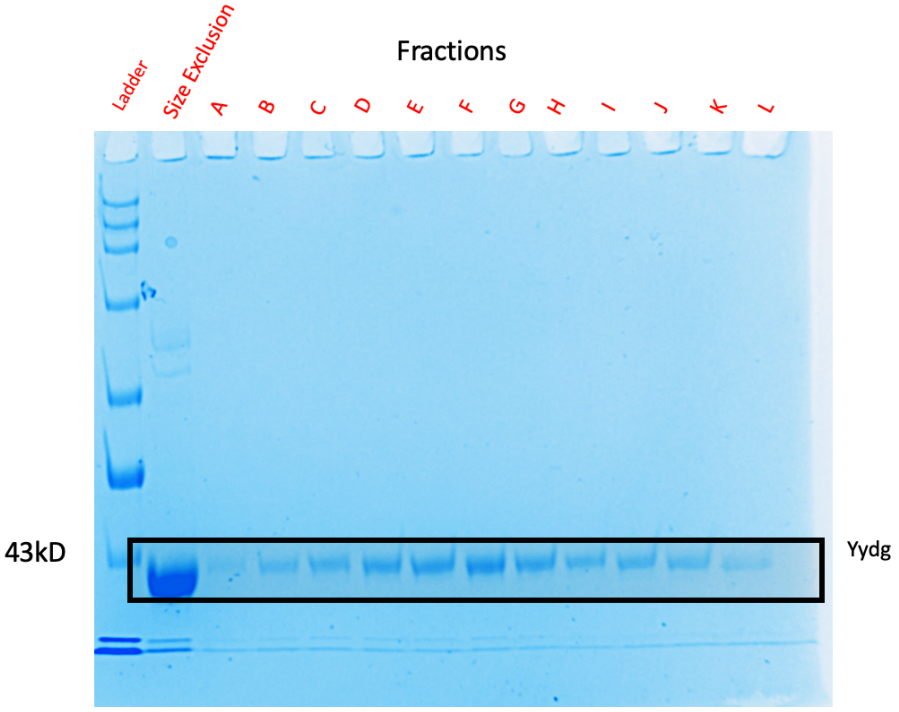

Figure 10. SDS-Page gel showing collected fractions from anion exchange chromatography. YydG fractions reached crystallization grade purity.

These collected fractions reached crystallization grade purity after being purified with anion exchange as can be seen in Fig 10. Anion exchange separates out molecules by their charge. More negative molecules bind to the column with tighter affinity, and they are eluted with a buffer that contains a more negative charge than the molecules bound to the column.[25] The lane containing the fractions collected from size exclusion has faint bands of the same size that was seen throughout the nickel affinity and size exclusion gels, but all successive bands collected from the anion exchange column show that the only thing in the fraction is YydG. This sample was then used to create crystals using a sparse matrix screen. The same plates mentioned above provided by Hampton Research were used for the sparse matrix screens (Index HT, Crystal Screen HR2-130, and PEGRx HT).

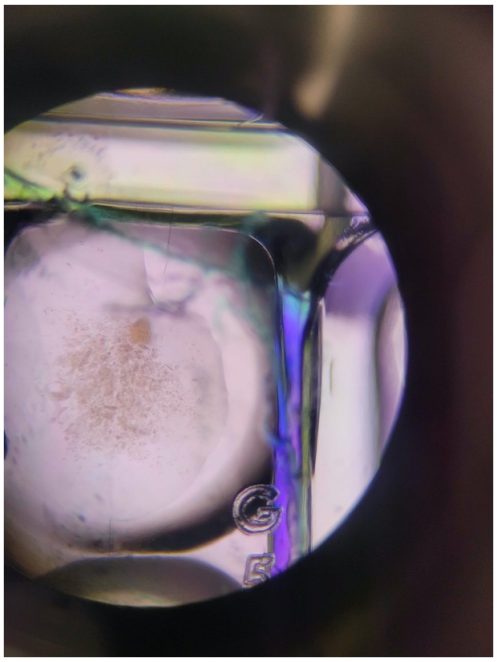

Figure 11. Pseudo crystal of YydG.

Shown in Fig. 11 is a well of a sparse matrix screen containing a pseudo crystal of YydG. It is possible that this well could yield a crystal of YydG due to the aggregation of the protein sample that is beginning to occur. If the correct amount of vapor diffusion occurs, the sample could reach the saturation needed for nucleation to form an ordered lattice.

Aim 2 Background:

Machine learning is the process of a computer continually refining a decision-making process to achieve higher accuracy on a prediction. Machine learning can also be split into regression and classification problems.[26] Regression in a machine learning context means using algorithms to interpolate or extrapolate predictions for values based on a training data set. Classification in machine learning is training an algorithm to make a discrete prediction that will fit into certain categories. Applied to the sparse matrix screen, it can be restated as how can a computer learn what conditions are necessary to form a protein

crystal for a given protein. This is therefore a classification problem as the predictions are meant to be "no crystal" or "crystal". The process of machine learning can be divided into several stages as follows.[27]

1. Data Collection – gathering relevant datapoints for use
2. Data processing – scaling data, excluding unusable datapoints
3. Model Training – "learning" part of the process where models first see data
4. Model Testing – using trained models to output predictions
5. Model Improvement – changing parameters of dataset or of model to obtain higher accuracy

Detailed sparse matrix conditions with notes on the crystal formation of SuiB provided by Dr. Katherine Davis were used for training and testing.

Three different machine learning models were used to generate predictions and to make comparisons between the different methods to determine the most accurate one. The first method used was a Random Forest Algorithm, the second was a Neural Network, and the last method was a Support Vector Machine. All three methods were coded using the scikitlearn package provided by Python (3.8.13).[21]

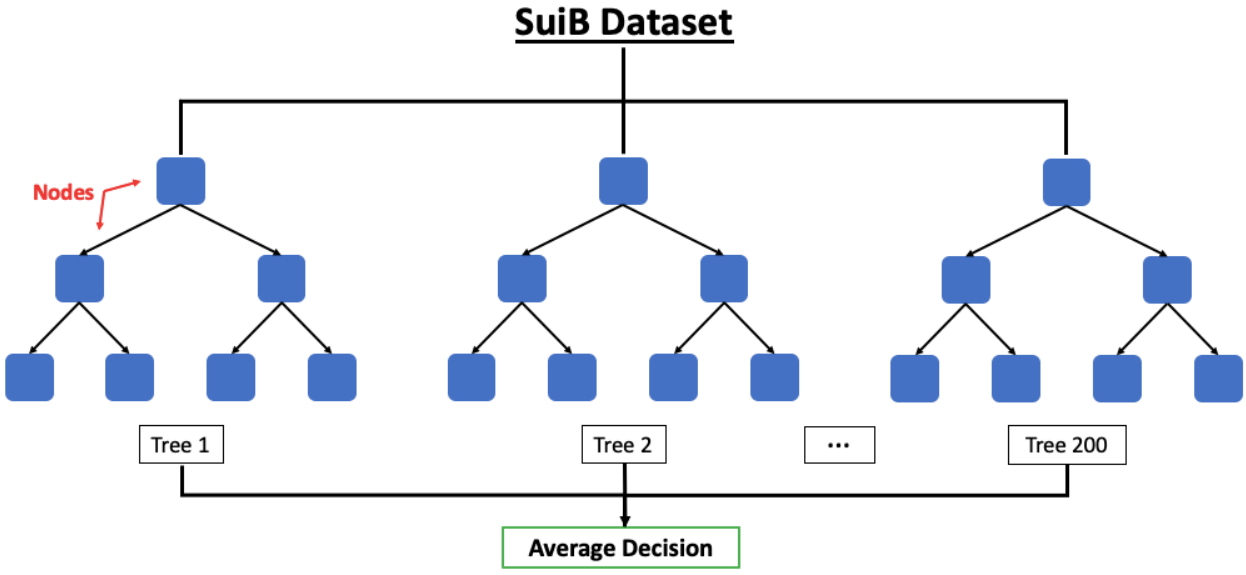

Figure 12. A random forest algorithm illustrated as an average decision of many decision trees. Each tree has nodes that lead down a different path to an output decision

Random Forest Algorithms are based on a different machine learning algorithm, decision trees. Decision trees are a data structure in which each parameter forms a node with a cutoff boundary.[28] The "learning" that occurs is the restructuring of the tree to organize important parameters higher in the tree. Traversing the height of the tree should produce a prediction after each point is assessed at the nodes along its path on the tree. Random Forest algorithms are an ensemble of decision trees, shown in Fig. 12, designed to reduce the variability or overfitting to data that can be common with only decision trees. By reducing overfitting, the random forest model should therefore produce more accurate predictions.[29]

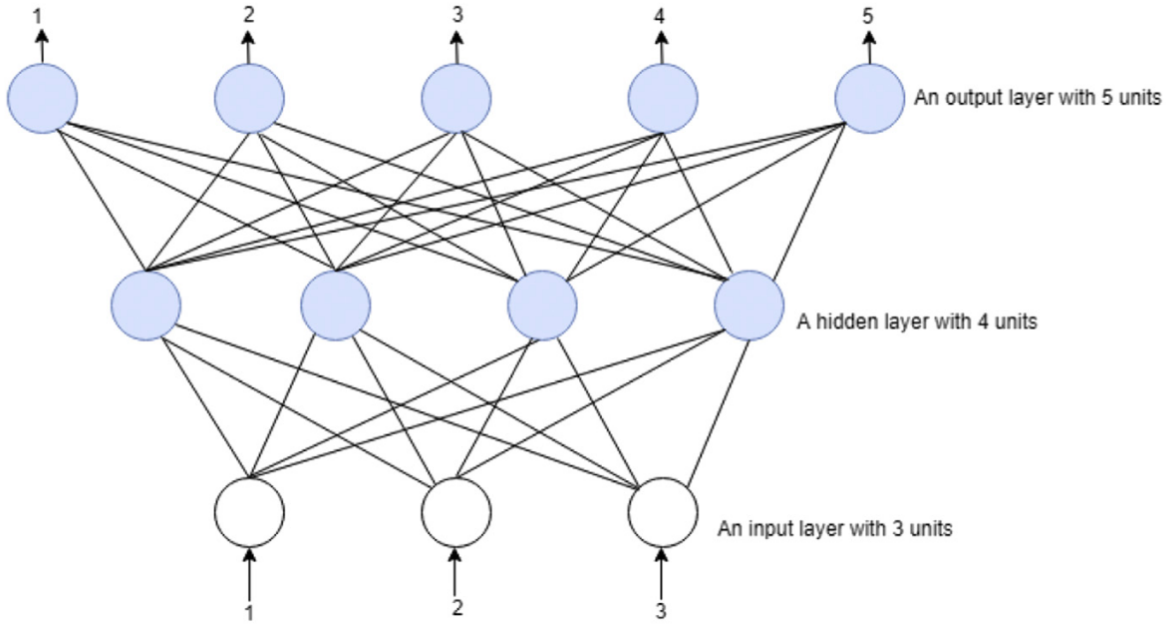

Figure 13. A neural network has layers of nodes that weight parameters to arrive at an output decision. (From Abidouin et. al, Heliyon 2018)

The second method used was a feed forward neural network. Neural networks are built on the single unit "perceptron" meant to simulate the decision-making process used by real neurons. A full neural network has layers of these perceptrons, the simplest containing the input layer, one or more "hidden" layers, and finally an output layer as shown above in Fig. 13. Each node creates a linear formula of the sum of the input parameters multiplied by its weight or importance to the prediction, added to the bias of each parameter. Therefore, for each parameter present in the model, each node is factoring each parameter with

an equation in the form of y = mx+b. The learning process involved the minimization of the cost function by continuously changing the weights.[30]

Finally, the last method used for predicting crystallization conditions of SuiB was the Support Vector Machine. This model operates by creating a decision boundary shown in Fig. 14, which should partition the data into negative or positive conditions. Each parameter in the dataset adds another dimension in space for which a boundary must be drawn.[31]

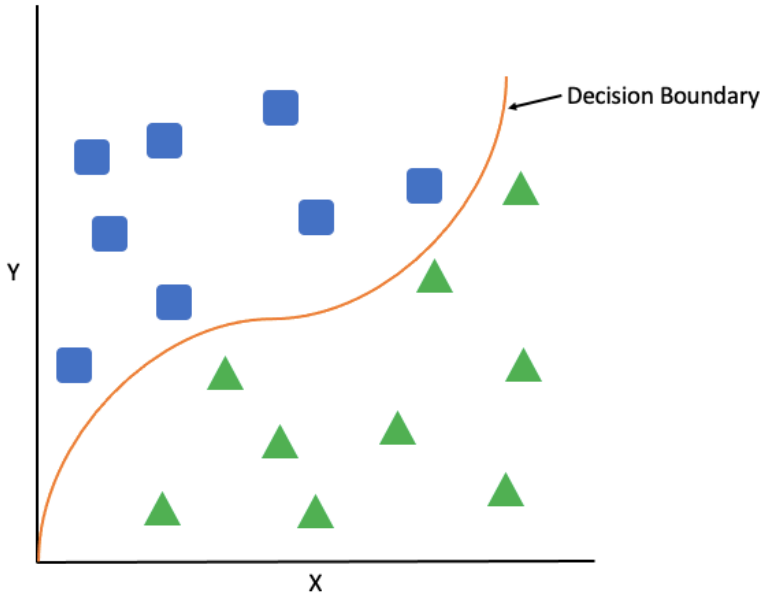

Figure 14. A simplified diagram of a support vector machine. The learning process involves creating a decision boundary to partition data into separate categories

Aim 2 Methods:

The dataset used to train the three algorithms was the scoring sheets of three sparse matrix screens used by Dr. Katherine Davis during her crystallization of rSAM enzyme SuiB.[3] The three screens used were Index HT, Crystal Screen HR2-130, and PEGRx HT. All of these screens were provided by Hampton Research. Each screen was a 96 well plate, meaning that there were 288 total conditions that SuiB was tested against for crystallization. After this dataset was curated, feature extraction was performed manually for each well in the three plates, to create 13 parameters per condition. These parameters are illustrated in Fig. 15:

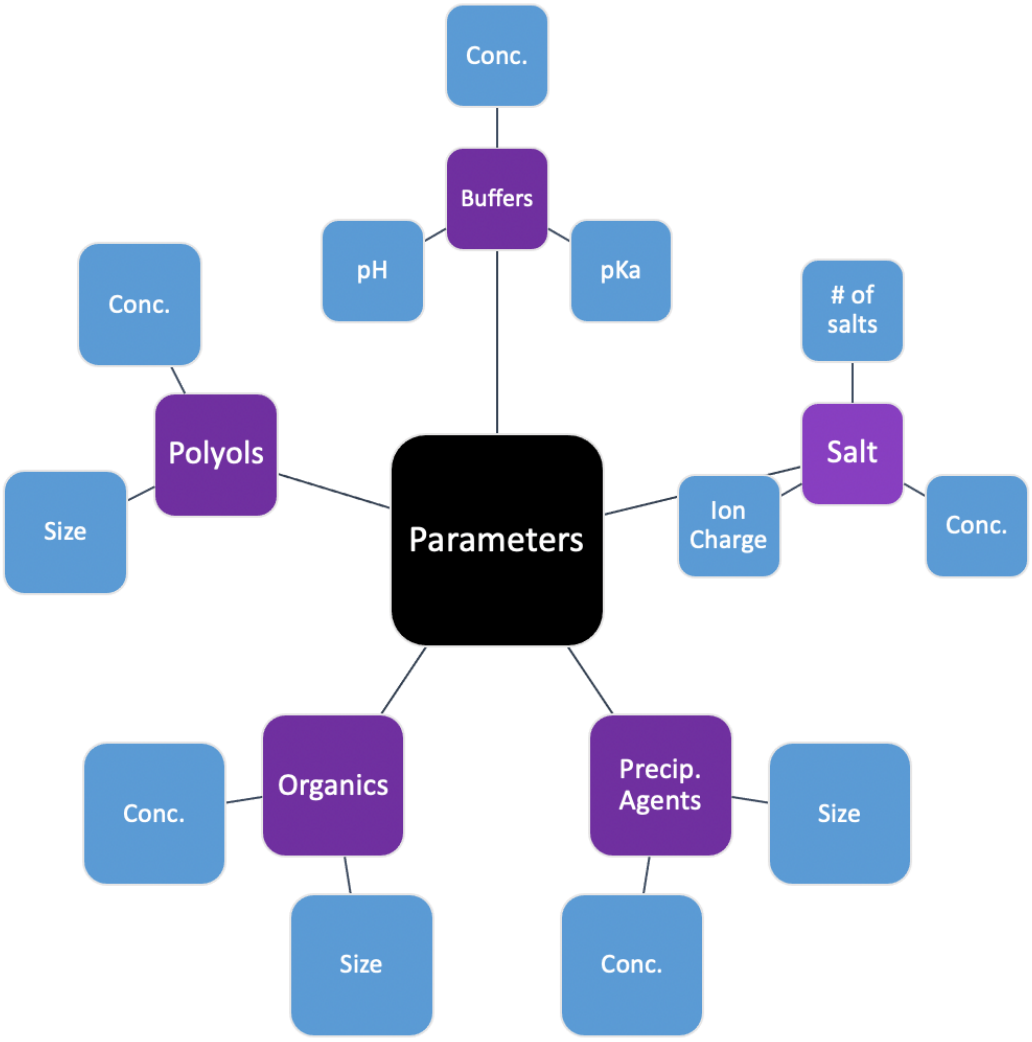

Figure 15. A concept map containing each category of parameter in the SuiB crystallization dataset

The dependent variable of this data was a classification of the presence of a crystal in the well. This was a binary classification. Out of the 288 points, 21 points were excluded due to their inability to fit into the model. For example, several conditions in the sparse matrix conditions use a specially developed crystallization reagent called Tacsimate.[32] This reagent is a titrated mixture of 7 organic acids in a ratio that is proprietary to Hampton research. It was therefore difficult to express this buffering reagent's pKa, so conditions containing this reagent were excluded.

The final dataset after cleaning therefore contained 267 data points, with 215 conditions having no crystal, and 52 having a crystal. This dataset was then ready to be used for training and testing machine learning models.

```
crystal = pd.read_csv('SuiB Final Set.csv', sep=',')
```

```
crystal.head()
```

	Buffer Con	pKa	pH (main)	Salt Conc	ion charge	Salt 2 conc	ion charge 2	Precipitating Agent Conc	Precipitating agent size	Organic Concentration	Organic Size	Polyol Concentration	Polyol Size	Crystal
0	0.1	3.10	3.5	2.0	2	0.0	0	0.0	0	0	0	0	0	0
1	0.1	4.54	4.5	2.0	2	0.0	0	0.0	0	0	0	0	0	0
2	0.1	6.46	5.5	2.0	2	0.0	0	0.0	0	0	0	0	0	0
3	0.1	6.46	6.5	2.0	2	0.0	0	0.0	0	0	0	0	0	1
4	0.1	7.50	7.5	2.0	2	0.0	0	0.0	0	0	0	0	0	0

Figure 16. First several rows of data in the SuiB dataset after being loaded into a Jupyter Notebook

Data was imported into python through a csv file. Fig. 16 shows the first several rows of the set. In the crystal column, 0 represents no crystal in the given condition, with 1 being presence of a crystal. The data was then split into the independent variables, and the one dependent variable.

According to the standard practice for machine learning methods, the data was then split into a 25% testing, 75% training set.[27] Using the standard scaler provided by scikitlearn, all the independent variables were then scaled using the code in Fig. 17. The standard scaler provided by python automatically scales all variables relative to themselves using a formula used to calculate a standard score for a given datapoint. The formula is as follows. $Z = (x-\mu)/s$, where z is the standard score of the datapoint, x is the true value of the point, μ is the mean, and s is the standard deviation of the independent variable.[21] Looking at this formula, it becomes clear that this formula will generate a gaussian distribution of the independent variable, centering the mean value at 0.

Scaling data is necessary before use on any machine learning algorithms because if variance of one independent variable is significantly higher than another, it will start to dominate the learning process, resulting in inaccurate weighting of that parameter in the model.[33] This problem is especially relevant with the parameters being used in this crystallization problem.

```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

Figure 17. Code to scale data with standard scaler from scikitlearn

For example, in the buffer concentration parameter, the range is very small with most of the datapoints having a concentration of 0.1 M. However, with parameters like PEG size, there is a massive range, from 0 when there is no PEG in the well, all the way up to PEG molecules with 10,000 monomers. Not scaling the data would cause parameters like PEG to start to be inaccurately weighted as much more important to crystallization than buffer concentration.

```
x = 0
weight = 0
store = 0
rfc = RandomForestClassifier(n_estimators = 200)
while x < 100:
    rfc.fit(X_train, y_train)
    pred_rfc = rfc.predict(X_test)
    report = classification_report(y_test, pred_rfc)
    store = classification_report_weight(report)
    if store > weight:
        weight = store
        joblib.dump(rfc, "./random_forest.joblib")

    x = x + 1
```

Figure 18. Code to run methods in a loop 100 times, saving the highest accuracy run

After cleaning the data, splitting it into training and testing, and scaling the parameters, the data was then ready to be used for the three machine learning models discussed previously. For the three methods

used, each one was run in a loop 100 times using the code in Fig. 18, and the set of weights that produced the highest weighted accuracy was saved in a joblib file.

First, a random forest algorithm with 200 decision trees was used to predict crystallization After the highest accuracy model was saved, it was reloaded and quantified using the classification report method provided by scikitlearn.[21] This process was repeated for both the support vector machine and neural network.

After initial runs of the machine learning methods were conducted, the methods were then run in a loop 100 times, with the highest weighted average accuracy model saved to a joblib file, using the joblib dump method. These files were then reloaded, and their accuracies quantified using the classification report.

For trial 2, the quantile transform method was used to rescale the entire dataset. Quantile transform non-linearly maps all the data to a gaussian distribution.[34] A non-linear scaler was desired because linear scalers such as the standard scaler assume a linear relationship between in the increase or decrease of parameters and the dependent variable. However, this may not be the case in a crystallization problem. For example, a factor of 5 increase in one parameter may cause more or less than a 5x increase in the chance of a crystal forming. Through non-linear mapping, the quantile scaler essentially squeezes any outliers closer into the dataset. After rescaling the data, the three methods were rerun in a loop 100 times, with the highest accuracy model being saved to a joblib file, and reloaded to be quantified by a classification report.

Aim 2 Results:

Trial 1:

For all machine learning methods, a classification report provided by scikitlearn was used to determine performance. As shown in Fig. 19, the matrix provides a score for precision, recall, F1-score and support. To understand the performance of the model, it is important to first understand what these terms are referring to.

Performance: The number of correct results divided by the number of results returned

Recall: The number of correct results returned divided by the total number of correct results

Fl-score: The harmonic mean of precision and recall (2*(precision*recall)/(precision+recall))

Support: Value calculated from the confusion matrix that identifies the number of points in each condition.

```
reloaded_rfc = joblib.load("./random_forest.joblib")
reloaded_rfc.fit(X_train, y_train)
pred_rfc2 = reloaded_rfc.predict(X_test)
print(classification_report(y_test, pred_rfc2))
```

```
              precision    recall  f1-score   support

           0       0.81      0.96      0.88        53
           1       0.50      0.14      0.22        14

    accuracy                           0.79        67
   macro avg       0.65      0.55      0.55        67
weighted avg       0.74      0.79      0.74        67
```

Figure 19. Classification report for first trial of random forest algorithm. Weighted average is higher than 50%, but crystallization recall is low.

The confusion matrix is a matrix comprised of true positives, false positives, true negatives, and false negatives.[21]

Fig. 19 shows the first trial of the random forest algorithm run in a loop 100 times. The most accurate version was reloaded and quantified. We can see immediately that the model has a high preference for predicting conditions that did not crystallize for SuiB, with 81% precision, and a high level of recall. 96% of all conditions that did not crystallize were returned correctly. However, this means that the model is more likely to incorrectly consider a condition that did yield a crystal for SuiB to not crystallize. This can

also be seen by the low recall for the crystallization conditions. Of the crystallization conditions the model was supposed to predict, it returned 14% of them.

```
reloaded_svc = joblib.load("./SVC.joblib")
reloaded_svc.fit(X_train, y_train)
pred_rfc3 = reloaded_svc.predict(X_test)
print(classification_report(y_test, pred_rfc3))
```

	precision	recall	f1-score	support
0	0.79	1.00	0.88	53
1	0.00	0.00	0.00	14
accuracy			0.79	67
macro avg	0.40	0.50	0.44	67
weighted avg	0.63	0.79	0.70	67

Figure 20. Classification Report for Support Vector Machine, illustrating poor prediction for crystallization.

The classification report for the looped trial of the Support Vector Machine told quite a different story. As seen in Fig. 20, the model has essentially predicted that none of the conditions in the testing set would crystallize. One reason that the support vector machine model could be performing poorly on this problem is the high ratio between the parameters to the number of datapoints (13:267). The addition of each parameter increases the dimensions in space that the model has to draw a decision boundary. Therefore, this model has to attempt to create a decision boundary in a 13-dimensional space, a very difficult task for such a small dataset.

The neural network results, once again run in a loop 100 times are displayed in Fig. 21. They show a similar overall weighted average accuracy to the random forest, but the accuracy for crystallization conditions is notably higher, with 75% of conditions predicted to be crystallization being correct. However, this model still misses most of the conditions that are crystallization conditions, returning 21% of the total number of crystallization conditions as crystallization conditions.

```
reloaded_nn = joblib.load("./neuralnetwork.joblib")
reloaded_nn.fit(X_train, y_train)
pred_rfc4 = reloaded_nn.predict(X_test)
print(classification_report(y_test, pred_rfc4))
```

```
              precision    recall  f1-score   support

           0       0.83      0.98      0.90        53
           1       0.75      0.21      0.33        14

    accuracy                           0.82        67
   macro avg       0.79      0.60      0.61        67
weighted avg       0.81      0.82      0.78        67
```

Figure 21. Classification report for first trial of neural network. Results are similar to Random Forest with higher crystallization recall

Conclusions from Trial 1:

Looking at the results from trial 1 we can conclude that all three models are better at predicting conditions that will not crystallize compared to ones that will. While it is valuable to know what conditions do not crystallize, for a practical application of this model on a protein that has not yet been crystallized, it is more important to have the model identify conditions that will crystallize. Therefore, in terms of utility the neural network was the most useful model considering that its precision and recall for the crystallization conditions was the highest, despite the fact that its overall accuracy was slightly lower than the random forest.

To further improve the accuracy of the models, alternative scaling methods for the data were investigated. For the first trial of the three methods, the scikitlearn standard scaler was used as mentioned above. However, this scaler assumes that increases or decreases in parameters and the relationship to crystallization is linear. For example, with a parameter like PEG size, a molecule 5 times larger than another PEG molecule will have 5 times more "importance" in the model. However, this linear relationship between

parameters may not be an accurate representation of the relationship the parameters have to crystallization. A 5x increase in the size of PEG may cause more than 5x more crystallization or vice versa.

Trial 2:

```
reloaded_rfc = joblib.load("./random_forestQ.joblib")
reloaded_rfc.fit(X_train, y_train)
pred_rfc3 = reloaded_rfc.predict(X_test)
print(classification_report(y_test, pred_rfc3))
```

```
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        53
           1       1.00      0.29      0.44        14

    accuracy                           0.85        67
   macro avg       0.92      0.64      0.68        67
weighted avg       0.87      0.85      0.82        67
```

Figure 22. Classification report for the random forest after quantile scaling. There was a significant jump in crystallization recall

After rescaling the data using a quantile transform and taking the highest accuracy model after running it in a loop 100 times, the classification report above was generated. The statistics in Fig. 22 show a marked improvement in accuracy. First, the recall for the non-crystallization conditions has reached 100 percent, meaning every condition that was not a crystallization condition was accurately returned. The biggest jump in accuracy for the random forest compared to the standard scaler came in the crystallization prediction. 100% of the conditions that were returned to be crystallization conditions were accurate. This rescaling also resulted in a more than 100% jump in recall.

```
reloaded_svc = joblib.load("./SVCQ.joblib")
reloaded_svc.fit(X_train, y_train)
pred_rfc5 = reloaded_svc.predict(X_test)
print(classification_report(y_test, pred_rfc5))
```

```
              precision    recall  f1-score   support

           0       0.79      1.00      0.88        53
           1       0.00      0.00      0.00        14

    accuracy                           0.79        67
   macro avg       0.40      0.50      0.44        67
weighted avg       0.63      0.79      0.70        67
```

Figure 23. Classification report for the Support Vector Machine after quantile scaling. The results were unchanged to the previous scaling

Interestingly, the rescaling had absolutely no impact on the accuracy of the support vector machine as shown in Fig. 23. Due to the constraints of the model described above, it is possible that something like rescaling the dataset would not be enough to overcome the challenges of using this model on a small dataset with many parameters.

```
reloaded_nn = joblib.load("./neuralnetworkQ.joblib")
reloaded_nn.fit(X_train, y_train)
pred_rfc6 = reloaded_nn.predict(X_test)
print(classification_report(y_test, pred_rfc6))
```

```
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        53
           1       1.00      0.29      0.44        14

    accuracy                           0.85        67
   macro avg       0.92      0.64      0.68        67
weighted avg       0.87      0.85      0.82        67
```

Figure 24. Classification report for the neural network after quantile scaling. Results were identical to Random Forest

The neural network performance after the quantile scaling interestingly produced the exact same results as the random forest with every score in the classification report being identical as shown above in Fig. 24. This was still an improvement for the neural network as well, with all the conditions predicted to be a crystallization condition being accurate, as well as all the negative conditions being returned accurately.

Conclusions for Trial 2:

The quantile rescaling of the data markedly improved the performance for the models that were already providing some level of accuracy, while leaving the support vector machine performance the same. However, since the accuracy of the random forest and neural network is jumping so much, we can say that the quantile scaling is a more accurate way to represent the relationship between the parameters of the dataset and the outcome of crystallization. Furthermore, with respect to the application of this model to an uncrystallized protein, having the models predicting crystallization with high specificity (precision of 100%) is more useful than a model that is still incorrectly prediction non crystallization conditions as crystallization conditions because when preparing to crystallize a protein because after purification has been completed, the researcher will be ready to select conditions that are likely to yield a crystal.

With respect to the identical accuracies of the neural network and the random forest, this is most likely happening because after the quantile transformation, the models are arriving at the same conclusions after their respective learning processes are complete. These identical statistics are encouraging that the parameterization of the dataset, and the quantile scaling is providing an accurate representation of the forces behind crystallization, and one that is reproducible between the two machine learning models.

Future Directions:

Aim 1:

As mentioned above, at the time of submission of the manuscript, crystals of YydG have been sent for beamtime to generate diffraction patterns to solve the structure of the enzyme. Along with my graduate

student mentor Tamra Blue, we are currently in the process of optimizing conditions that yielded crystals of YydG for various factors including pH and $LiSO_4$ concentration.

If the crystals at the beamtime yield a high enough quality diffraction pattern that electron density can be generated to a high resolution (<4 Ångstroms), the structure will be solved using Phenix and Coot, and validated using MolProbity.

Aim 2:

Before improving the accuracy of the models, the first thing to investigate would be to see if the models ran in trial 2 that yielded identical results were indeed returning the exact same conditions as crystallization conditions. If this were the case, it would then be interesting to see the weights assigned to the different parameters and see if these two independent models were truly coming to fully identical conclusions. If they were, it would be possible to make a conjecture about which parameters were more important than others in the crystallization of SuiB.

The first step to take for improving the accuracy of the models would be to try to reduce the number of parameters. To do this, parameters must be consolidated without any loss of information. Not every parameter in the dataset can be consolidated, but for example all the information about the buffer can be reduced to a single parameter. Currently, the dataset holds information on the buffer concentration, pH and pKa. These three parameters can be consolidated into a charge parameter using the Henderson Hasselbach equation, turning three parameters into one without any loss of information. Similarly, a weighted parameter for ionic charge can be created, replacing the concentration and charge parameter for each salt that currently exists in the dataset. This consolidation should improve all the methods, but it should especially help the performance of the support vector machine, as cutting the number of parameters will reduce the large dimensions in which the model must operate.

After exploring consolidation of the dataset, to apply this problem to a real protein that has not been crystallized, aspect of the protein itself must be parametrized. Currently, the model is only looking at one

protein, SuiB. Therefore, parameterization of protein characteristics was unnecessary since for every datapoint, these would be the same. However, when looking at other proteins, these parameters would change. Adding other crystallization data for other proteins would be important to increase the training set for the model. After the data has been added, the addition of parameters such as protein size, pI, and surface charge (calculated using predictive servers such as Phyre) would prepare the model for use to predict crystallization conditions for a protein that has not been crystallized.

Finally, after updating this model, wet lab verification of the predictions would solidify the utility of this method to change the process of developing a protein crystal. The use of this method could drastically save the time and resources necessary to develop diffraction quality crystals, and by improving protein crystallography, lead to vastly more protein structures being solved at a much faster rate.

Current limitations of the model include difficulty accessing more data to include in the model. The dataset provided to me was collected by Dr. Katherine Davis, but unfortunately most crystallization data is not published, and if the data is published, negative conditions are not included. However, negative conditions are important to include in a model for accurate training. Furthermore, when applying this model to proteins that have not yet been crystallized, there could be confounding variables in the process such as the purity of the protein sample when it is used for sparse matrix screens. A protein may not crystallize in the same conditions if the purity of the two samples are not the same.

However, to solve a structure of a protein, only one condition yielding good morphology is necessary. With the neural network and random forest algorithm providing 100% precision on the SuiB crystallization conditions, it has been shown that machine learning algorithms can distinguish between conditions that will not yield a protein crystal and conditions that will. This method can therefore be used as a preliminary screening protocol to at least narrow down conditions to try when crystallizing a protein, making the current methodology more efficient. For example, if 80% of conditions are eliminated by the model, only 20% of the purified protein needs to be used to test conditions in a sparse matrix. This would

allow for more protein sample for optimization, as well as trying even more conditions to grow protein crystals with less protein expression and purification.

References

1. Cooper, G., The Cell: A Molecular Approach 2nd edition Boston University. *Sunderland (MA): Sinauer Associates.[Google Scholar]* **2000**.
2. Blow, D., So do we understand how enzymes work? *Structure* **2000,** *8*, R77-R81.
3. Davis, K. M.; Schramma, K. R.; Hansen, W. A.; Bacik, J. P.; Khare, S. D.; Seyedsayamdost, M. R.; Ando, N., Structures of the peptide-modifying radical SAM enzyme SuiB elucidate the basis of substrate recognition. *Proc. Natl. Acad. Sci. U.S.A.* **2017,** *114*, 10420-10425.
4. Broderick, J. B.; Duffus, B. R.; Duschene, K. S.; Shepard, E. M., Radical S-adenosylmethionine enzymes. *Chemical reviews* **2014,** *114*, 4229-4317.
5. Arnison, P. G.; Bibb, M. J.; Bierbaum, G.; Bowers, A. A.; Bugni, T. S.; Bulaj, G.; Camarero, J. A.; Campopiano, D. J.; Challis, G. L.; Clardy, J., Ribosomally synthesized and post-translationally modified peptide natural products: overview and recommendations for a universal nomenclature. *Nat. Prod. Rep.* **2013,** *30*, 108-160.
6. Sivonen, K.; Leikoski, N.; Fewer, D. P.; Jokela, J., Cyanobactins—ribosomal cyclic peptides produced by cyanobacteria. *Appl. Microbiol. Biotechnol.* **2010,** *86*, 1213-1225.
7. Burkhart, B. J.; Hudson, G. A.; Dunbar, K. L.; Mitchell, D. A., A prevalent peptide-binding domain guides ribosomal natural product biosynthesis. *Nat. Chem. Biol.* **2015,** *11*, 564-570.
8. Kloosterman, A. M.; Shelton, K. E.; van Wezel, G. P.; Medema, M. H.; Mitchell, D. A., RRE-Finder: a genome-mining tool for class-independent RiPP discovery. *Msystems* **2020,** *5*, e00267-20.
9. Schramma, K. R.; Seyedsayamdost, M. R., Lysine-tryptophan-crosslinked peptides produced by radical SAM enzymes in pathogenic streptococci. *ACS Chem. Biol.* **2017,** *12*, 922-927.
10. Giger, S., Engineering of Epimerases towards Pharmaceutical Applications.
11. Benjdia, A.; Guillot, A.; Ruffié, P.; Leprince, J.; Berteau, O., Post-translational modification of ribosomally synthesized peptides by a radical SAM epimerase in Bacillus subtilis. *Nat. Chem.* **2017,** *9*, 698-707.
12. Grell, T. A.; Goldman, P. J.; Drennan, C. L., SPASM and twitch domains in S-adenosylmethionine (SAM) radical enzymes. *J. Biol. Chem.* **2015,** *290*, 3964-3971.
13. Butcher, B. G.; Lin, Y.-P.; Helmann, J. D., The yydFGHIJ operon of Bacillus subtilis encodes a peptide that induces the LiaRS two-component system. *J. Bacteriol.* **2007,** *189*, 8616-8625.
14. Popp, P. F.; Benjdia, A.; Strahl, H.; Berteau, O.; Mascher, T., The Epipeptide YydF Intrinsically Triggers the Cell Envelope Stress Response of Bacillus subtilis and Causes Severe Membrane Perturbations. *Front. Microbiol.* **2020,** *11*.
15. Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A., Highly accurate protein structure prediction with AlphaFold. *Nature* **2021,** *596*, 583-589.
16. Perrakis, A.; Sixma, T. K., AI revolutions in biology: The joys and perils of AlphaFold. *EMBO Rep.* **2021,** *22*, e54046.
17. Fiser, A.; Šali, A., Modeller: generation and refinement of homology-based protein structure models. In *Methods in enzymology*, Elsevier: **2003**; Vol. 374, pp 461-491.
18. Källberg, M.; Wang, H.; Wang, S.; Peng, J.; Wang, Z.; Lu, H.; Xu, J., Template-based protein structure modeling using the RaptorX web server. *Nat. Protoc.* **2012,** *7*, 1511-1522.
19. Rohl, C. A.; Strauss, C. E.; Misura, K. M.; Baker, D., Protein structure prediction using Rosetta. In *Meth. Enzymol.*, Elsevier: **2004**; Vol. 383, pp 66-93.
20. Deng, H.; Jia, Y.; Zhang, Y., Protein structure prediction. *Int. J. Mod. Phys. B.* **2018,** *32*, 1840009.
21. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V., Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011,** *12*, 2825-2830.

22. Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E., The protein data bank. *Nucleic Acids Res.* **2000,** *28*, 235-242.
23. Tokumoto, U.; Takahashi, Y., Genetic analysis of the isc operon in Escherichia coli involved in the biogenesis of cellular iron-sulfur proteins. *J. Biochem.* **2001,** *130*, 63-71.
24. Barth, H. G.; Jackson, C.; Boyes, B. E., Size exclusion chromatography. *Anal. Chem.* **1994,** *66*, 595-620.
25. Auler, L. M.; Silva, C. R.; Collins, K. E.; Collins, C. H., New stationary phase for anion-exchange chromatography. *J. Chromatogr. A.* **2005,** *1073*, 147-153.
26. Mahesh, B., Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]* **2020,** *9*, 381-386.
27. Blockeel, H.; Vanschoren, J. In *Experiment databases: Towards an improved experimental methodology in machine learning*, European Conference on Principles of Data Mining and Knowledge Discovery, Springer: **2007;** pp 6-17.
28. Kingsford, C.; Salzberg, S. L., What are decision trees? *Nat. Biotechnol.* **2008,** *26*, 1011-1013.
29. Breiman, L., Random forests. *Mach. Learn.* **2001,** *45*, 5-32.
30. Abiodun, O. I.; Jantan, A.; Omolara, A. E.; Dada, K. V.; Mohamed, N. A.; Arshad, H., State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018,** *4*, e00938.
31. Noble, W. S., What is a support vector machine? *Nat. Biotechnol.* **2006,** *24*, 1565-1567.
32. McPherson, A.; Cudney, B., Searching for silver bullets: an alternative strategy for crystallizing macromolecules. *J. Struct. Biol.* **2006,** *156*, 387-406.
33. Ahsan, M. M.; Mahmud, M.; Saha, P. K.; Gupta, K. D.; Siddique, Z., Effect of data scaling methods on machine learning algorithms and model performance. *Technologies* **2021,** *9*, 52.
34. Peng, B.; Yu, R. K.; DeHoff, K. L.; Amos, C. I. In *Normalizing a large number of quantitative traits using empirical normal quantile transformation*, BMC Proceedings, BioMed Central: **2007**; pp 1-5.