Data Analysis and Software Design to Assist Researchers in Choosing Effective Endogenous Genes for CRISPRa

Joely Nelson[1]

[1]Paul G Allen School of Computer Science Engineering at the University of Washington
joelyn@cs.washington.edu

Abstract

CRISPR activation (CRISPRa) is a tool used in synthetic biology to activate genes. However, there are stringent rules for where CRISPRa can effectively promote transcription. Many of these rules are either unknown or have not been compiled into a single model. In order to rectify this, I worked on two main software projects. (1) Mock Data Generation Model for FACS-seq CRISPRa was a project where synthetic data was generated in the style of a CRISPRa experiment. In the experiment, effectiveness would be collected for several guides with the goal of using this data to uncover more rules for effective CRISPRa. The intent of the software project was to create this synthetic data to better understand and explore the data output by the experiment before real data could be produced. (2) *P. putida* RNA Seq Activation Filtering was a project where the goal was to find effective candidates for CRISPRa in *Pseudomonas putida*'s endogenous genome. This work was done by taking in several *P. putida* related datasets and using known CRISPRa rules to generate a list of suitable candidates for further study. The software filtered the list down to 9% of the original genes. Without taking into account the outputs of genome scale model, 9% of genes input into the software were considered suitable for CRISPRa. When taking into account the model outputs, the software produced 5 candidates for further experimentation.

1 Introduction

Synthetic biology is an interdisciplinary area of research that aims to engineer and redesign biological components and systems. CRISPR-based technologies are being developed for a wide array of applications in synthetic biology. Many of these CRISPR technologies work by providing a mechanism for targeting specific sequences of DNA in a cell. A Cas9 protein is led to a specific DNA sequence by a guide RNA (gRNA) whose sequence base-pairs with nucleotides of the DNA target. The Cas9 protein then cleaves, or cuts, that DNA sequence at that location. [10]. By engineering and introducing gRNAs to the cell to direct Cas9, researchers can dictate the sequence where the Cas9 will bind. In recent years, researchers have been studying how to utilize nuclease deactivated Cas9 proteins (dCas9), which bind to the sequence without cleaving it, allowing it to find specific strands of DNA without making any edits to the genome [4]. By careful engineering of the gRNAs and allowing for an activator to bind to the gRNA, scientists have been able to use CRISPR to activate gene expression by initiating transcription in a process known as CRISPR activation (CRISPRa) [6].

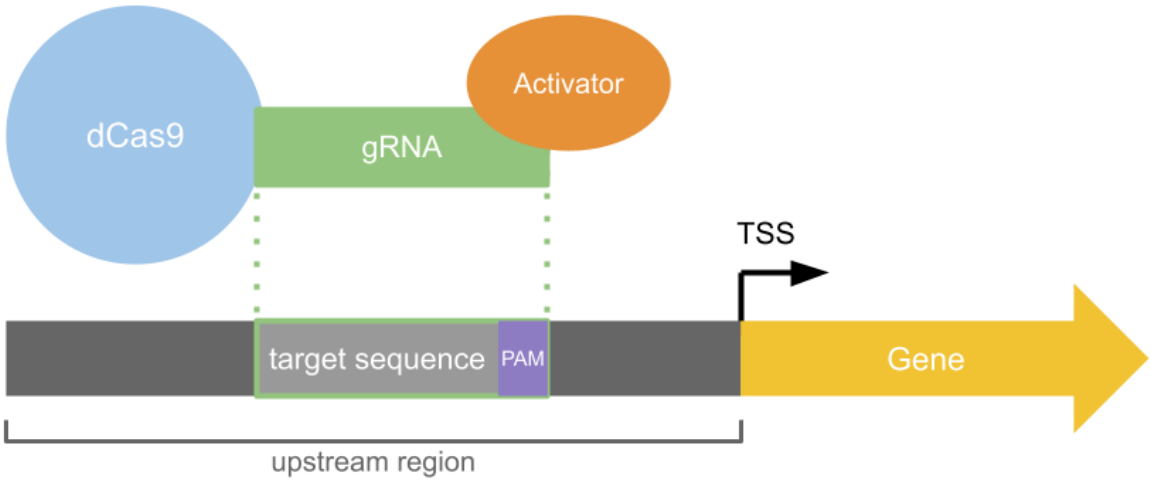

Figure 1: A model of how CRISPRa works. A dCas9, gRNA, and an activator come together to sit on a target sequence upstream of the gene and transcription start site (TSS) to promote transcription. The Protospacer-Adjacent-Motif (PAM) is the sequence immediately next to the target sequence, usually consisted of 3 nucleotides, and is required for dCas9 recognition.

However, there are limits to where CRISPRa can effectively initiate transcription in an organism's native, or endogenous, genome. Researchers can engineer gRNA sequences to target parts of the existing genome, but changing the sequence of the gRNA can have a dramatic impact on the effectiveness of the CRISPR technology [6] [2]. Prior work has uncovered some of these rules for engineering effective guides. Such rules include the strand of DNA to target, the baseline expression, the distance from the endogenous transcription start site, and the recognition sequences (i.e. protospacer adjacent motifs, or PAMs) that can be targeted. To date, these rules have not been collected and combined into a single model that can be applied to multiple genomes [6] [7]. One goal of my graduate

research work has been to use data analysis and software design to design and further the development of such a model. In addition, there are many rules yet to uncover. Other work I've done has involved exploring analysis techniques to uncover more rules.

My research has been focused on compiling data-driven approaches to CRISPRa engineering in order to aid with the CRISPR research. This has mainly been done by providing software to facilitate quick, easy, and reproducible CRISPRa data analysis. This work is meant to be part of the Learn and Design step in the Design-Build-Test-Learn cycle (DBTL). My software aims to learn from past experiments and suggest designs for future experiments [3].

I worked on two main software projects. (1) Mock Data Generation Model for FACS-seq CRISPRa was a project where synthetic data was generated in the style of a CRISPRa experiment where the effectiveness would be collected for several guides with the goal of using this data to uncover more rules for effective CRISPRa. The intent was to create this synthetic data to better understand and explore the expected data format before real data could be produced. (2) *P. putida* Gene Expression CRISPRa Filtering was a project where the goal was to find effective candidates for CRISPR in *Pseudomonas putida*'s endogenous genome. This work was done by taking in several *P. putida* related datasets and using known CRISPRa rules to generate a list of suitable candidates for further study. Given 5571 genes, this software initially filtered out 90% of input genes. After using outputs from a genome scale model, the software filters out an additional 90% of genes, resulting in 5 genes.

I started by meeting with researchers to define and discuss the problems they wanted to solve and how software and data analysis could help aid their research. I worked with them to understand their project goals, input data, and determined different ways to model CRISPRa systems. After discussion, I then and explored datasets to determine their limitations and how to prepare them for analysis. I then wrote software in Python that would use these datasets to perform analysis.

In the following thesis paper, I begin by outlining the motivation for my projects. Next, I explore related work done in this field. Then I discuss the motivation, methods, and results for my two software projects: Mock Data Generation Model for Facs Seq CRISPRa and *P. putida* Gene Expression CRISPRa Filtering. Finally I conclude by talking about overall limitations of data-driven approaches for CRISPRa.

2 Motivation

2.1 Focus on Endogenous CRISPRa

One goal of synthetic biology is to control multi-gene biological circuitry in a way that response to designated stimuli can be programmed. With appropriate tools, several applications, such as chemical production, can be achieved. This is often achieved by engineering genetic regulatory networks. A genetic regulatory network is a collection of genes that regulate and interact with each other to govern the gene expression of proteins. Cur-

rently, there are few examples of dynamic synthetic regulatory networks capable of multi-gene regulation. A possible avenue of research to rectify this involves using CRISPRa. CRISPRa, in conjuction with CRISPR inhibition (CRISPRi), has emerged as a possible route for building gene regulatory networks, since they allow for programmable control at many genes simultaneously [18]. In addition, since the gRNAs in CRISPRa are entirley programmable by engineers this allows for significant more flexibility in which genes the dCas9 protein can activate than other technologies [1]. Using endogenous CRISPRa in addition to heterologous CRISPRa has the potential to take advantage of and improve already existing metabolic pathways to generate new products [12].

Taking advantage of cellular machinery and using CRISPRa technology in this way has the potential to revolutionize the field of biology. Multilayered CRISPRa networks can increase access to products such as renewable chemicals and life-saving drugs. By designing circuits which react to certain stimuli, biologists can create medical diagnostics which can detect molecules or stimuli [15] [18].

2.2 Application to All Genomes

CRISPR is a tool that can be applied to many different organisms. One of the goals of these projects was to create software that would work regardless of the organism. This is useful because different organisms have the potential to be useful for the creation of different products. For example *Escherichia coli* (*E. coli*) is a bacteria that is the most popular organism for designing cellular factories for the production of biofuels and bulk chemicals, like ethanol [5]. *P. putida* is another organism that can produce many natural products, such as fluorescent siderophore pyoverdine, which issued for applications such as plant growth promotion. *P. putida* has a high tolerance towards xenobiotics including antibiotics and organic solvents. This makes it an especially suitable organism for production processes in two-phase systems that utilize xenobiotics to regulate the system [16]. Recently, CRISPRa has been demonstrated in P. putida which enable endogenous genes activation and biochemical productions with distinct metabolic properly [13]. As such the software created for this work was aimed to take inputs any genome so that the software could have value beyond specific organisms.

3 Related Work

3.1 Known Rules for Effective CRISPRa

Based on prior work, there are several rules that have been found to be vital in determining what endogenous genes will be good choices for CRISPRa. These are as follows: the location and the type of PAM sequence, the gene's sigma factor, and the gene's baseline expression. Each of these features will be discussed in further detail in the following subsections.

3.1.1 Available PAM sites

CRISPR binding require two elements: 1) binding of Cas-protein to PAM, and 2) Watson-Crick base-pairing of gRNA spacer to target DNA [8]. The combination of a PAM's location and the nucleotide (nt) sequence of the PAM have been found to have a large impact the effectiveness of CRISPRa. [6]

The most effective PAMs are located at certain base pairs away from the gene's transcription start site (TSS). They exist in a 40 base window at 60 bases to 100 bases upstream from the TSS. They cannot be modeled with a simple linear distance function; these effective locations are often 2-4 base windows with a periodicity that corresponds to one helical turn of DNA. [6].

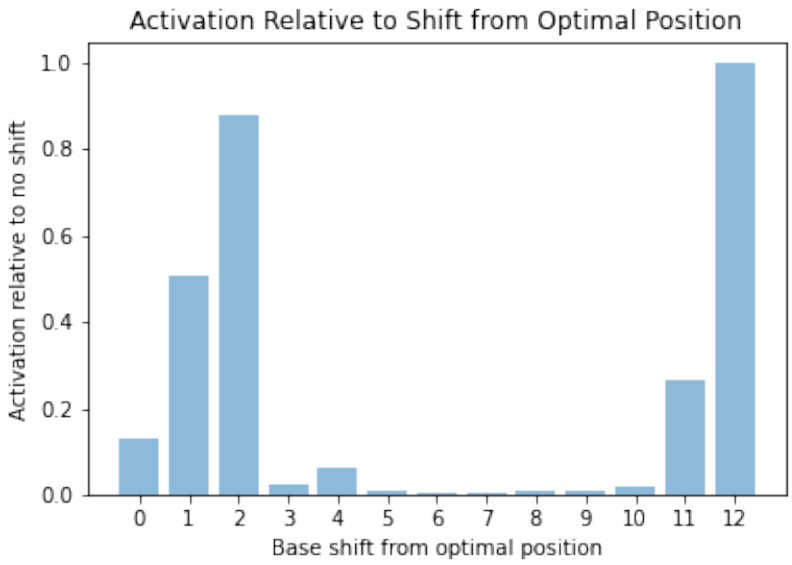

Figure 2: Activation relative to guides with PAMs at different locations relative to the optimal location from experimental data.

The 3 nucleotide sequence that makes up the PAM is also very important for the effectiveness of the binding. dCas9 prefers NGG PAMs (a PAM that consists of any nucleotide A, C, T, or G, followed by two Gs). Although this PAM site is the best for CRISPR with native, or wild-type Cas9, other sequences of PAM sites might also be suitable.

There are Cas9 variants which have an expanded list of suitable PAMs. One such is dxCas9-3.7 which has higher activation for many other sites besides NGG. These differences have been observed both in human cells [11] and *E. coli* cells [6], implying it is inherent to the CRISPR mechanism.

3.1.2 Sigma Factor

A sigma factor is a specific protein needed to initiate transcription in a bacteria. Sigma factor levels are regulated by the gene in response to external factors and cell state to

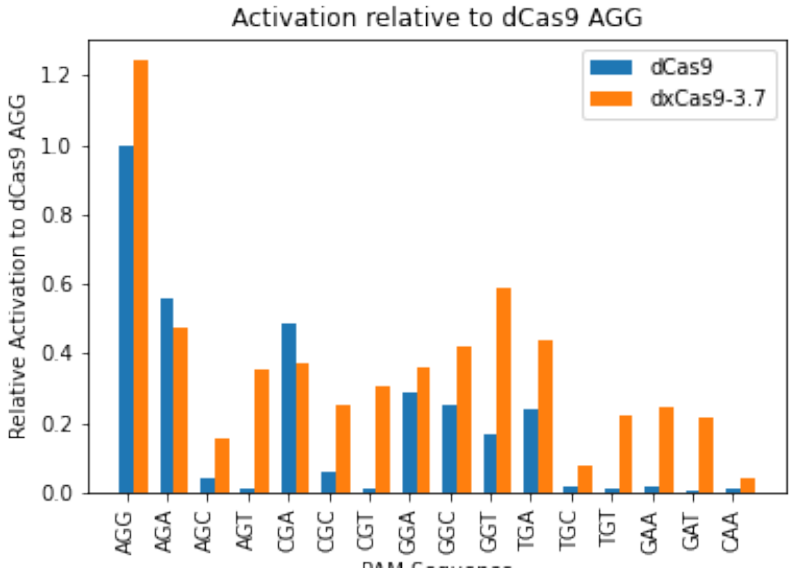

Figure 3: 16 tested PAMs and their mean activation based on the dCas9 variant from experimental data.

control gene expression. [9] It is believed that these different sigma factors could play a role in how effective the genes could be for CRISPRa.

We lacked experimental data that would inform researchers predicted levels of effectiveness based on the sigma factor, but the researchers I discussed with believed that sigma factors were important to be shown alongside the genes chosen.

3.1.3 Baseline Expression

CRISPRa is sensitive to promoter strength, which can be measured by observing the baseline expression of a given gene. If a promoter is too weak, it will be difficult to activate and acquire high fold change. This is because its so weak to the point that an activator cannot make an impact. For a strong promoter, fold-change due to activation is minimal because the activation level will be capped by another limiting factor. This has been observed in both eukaryotic and prokaryotic systems [6]. The range of an acceptable baseline is not known and must be inferred from existing data.

3.2 Prior Scoring Function Work

In prior work, I was directly involved in cleaning *E. coli* datasets and scoring sequences based on many of the rules above. To clean the data, upstream sequences were extracted from *E. coli* sequence datasets. Next, a scoring function was applied which took the PAM locations and sequences into account. This work filtered over 2,000 *E. coli* genes down to 25. Researchers used these scores and additional information to chose 7 genes for future experimentation. 3 of those 7 genes had desirable activation [6].

Using this scoring function to get 3/7 genes with effective activation is a good start, but it implies there are other missing rules. This motivates the first project: Mock Data Generation Model for Facs Seq CRISPRa. In this project we aimed to use the genes found in the prior work to test many more guides and learn from the resulting experiment. The project aimed to generate synthetic data that should look similar to the data that would result from that experiment.

Although this scoring function incorporated many of the known rules that effect CRISPRa, it did not incorporate baseline expression. In addition, prior work was specific to *E. coli*. Work in this thesis aimed to have the process generalized to more organisms. As such, for the second project, *P. putida* Gene Expression CRISPRa Filtering, I aimed to write code that filtered out any genes not within the range of acceptable baseline expression. This methodology depends on systematic input, RNA-seq, which is generalizable to any organism upon preliminary characterization

4 Mock Data Generation Model for FASeq CRISPRa

Mock Data Generation Model for FACS Seq CRISPRa was a project where synthetic data was generated in the style of a CRISPRa experiment where the effectiveness would be collected for several guides with the goal of using this data to uncover more rules for effective CRISPRa. The intent was to create this simulated data to better understand and explore the expected data format before real data could be produced.

In the experiment various different guides with different sequences would be tested. Their fluorescence would be taken as a measure of effectiveness. The sequences and the fluorescence would be analyzed in order to get an idea of what particular features were affecting the effectiveness of CRISPRa. The experiment would consist of testing over 1700 guide variants, many from the same library but having a slightly different sequence. Each guide variant would have approximately 15,000 replicates.

The data would then be sequenced and binned in a process known as fluorescence-activated cell sorting (FACS). This technique would allow for sorting of various cells based on the amount of fluorescent (which indicated CRISPRa effectiveness) detected by flow cytometry. [14] Afterwards, the populations could be sequenced with would inform which guide was present in that cell.

It was somewhat ambiguous what data would be produced by the experiment. I aimed to create mock data that would mimic what would be produced by the experiment. Having this synthetic data available for researchers could help with the design analysis pipelines without having to wait for the experiment to be carried out. This was something that was deemed import, as due to the COVID-19 pandemic, the researchers planning to carry out this experiment were unable to. In the meantime, we planned to create software pipelines that would be ready to complete analysis the moment the experiment was finished.

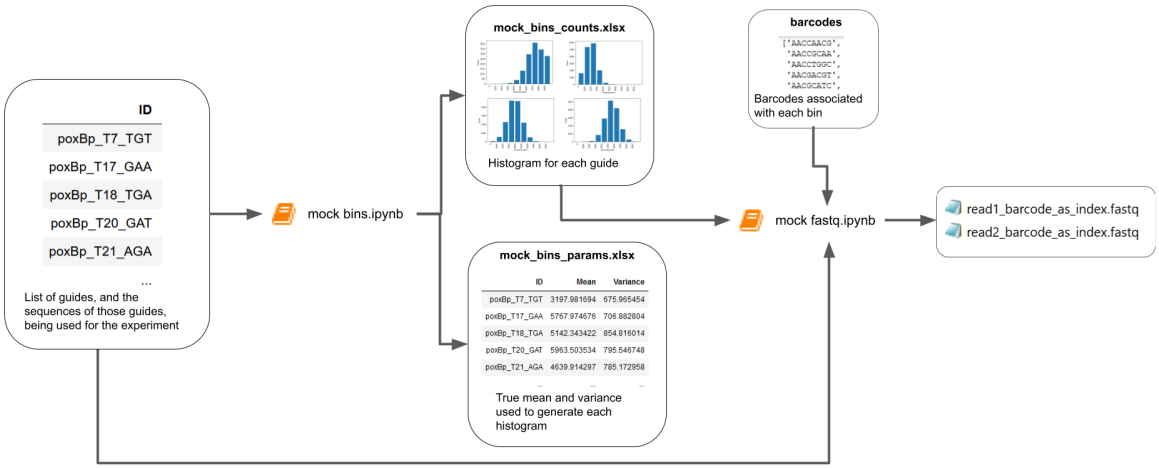

Figure 4: Graphical representation of the mock FACS data generation pipeline

4.1 Mock Bins Generation

The first step in the pipeline is a program called *mock_bins.ipynb* which will generate "mock bins", a binned histogram of the fluorescence expected to be output by the FACS process for the various guides used in the experiment.

The distribution in which the fluorescence for a particular guide i (F_i) is generated from is assumed to be a Gaussian distribution, where, $F_i : -N(\mu_i, \sigma_i^2)$. Each μ_i and σ_i were randomly generated from a uniform distribution, where the distributions are defaulted as follows: $\mu_i : -U(2750, 7000)$ and $\sigma_i : -U(600, 900)$. The user can modify the max and min values for each uniform distribution. The uniform values were chosen to create distributions similar to real fluorescent distributions similar to the type seen in the proposed experiment. (See Fig. 5 for an example of what is generated by this step).

Let *reads* be a number that represents the number of different cells for this guide that will be sorted by the cell sorter. Based on the planned experiment, this was set to 15,000 during the run time of the program. For each guide, for each cell, a fluorescence value will be output. Therefore there will be as output a total of $reads \cdot num\_guides$ rows in the final output.

The cell sorter is not able to determine the exact fluorescence produced by a given cell, instead it will be able to determine if that cell's fluorescence falls within a certain range. For example, if in actuality a cell had a fluorescence of 100 units, the cell sorter may only be able to bin it within 50 units - 100 units. The amount of resolution we expect the cell sorter to distinguish between is determined by *n_bins*

The program requires one file as input called *Guide IDs*. This file contains a list of

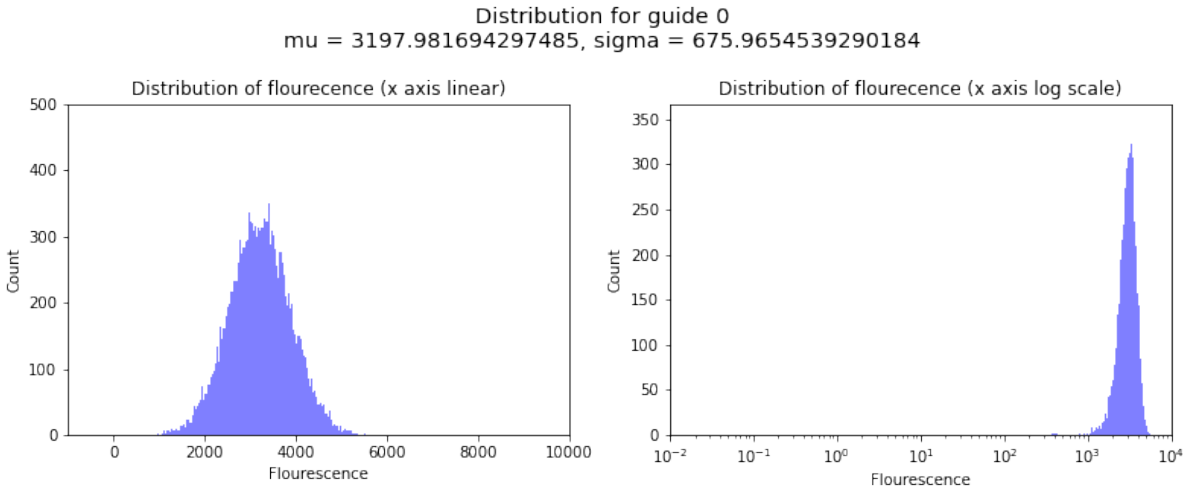

Figure 5: An example of mock fluorescent histograms generated during the execution of mock_bins.ipynb for a particular guide before they are "binned" by the cell sorter. The left image shows a version where the x axis is a linear scale, the right image shows a version where the x axis is on a log scale.

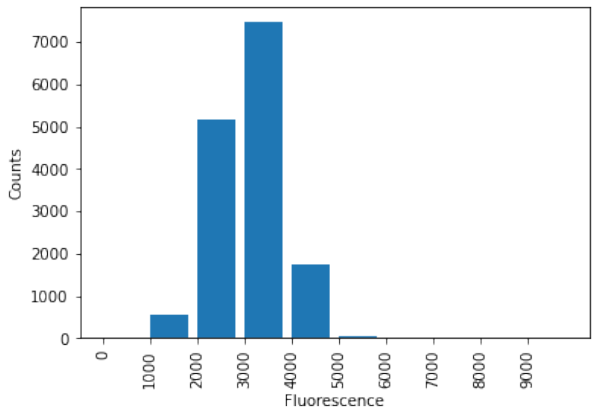

Figure 6: The output binned result generated by mock_bins.ipynb of guide the same guide seen in Figure 5 with 1500 reads and 10 n_bins.

all unique guide IDs being used in the experiment. These must be unique for each guide. This notebook will output a file representing the binned histograms for each guide, called *mock_bins_counts*, as well a file containing the true mean and variance for each guide.

4.2 Mock fastq Generation

After the mock bins are generated, the next step is to convert this data into a form mimicking what would be output by the Illumina sequencer. This format is fastq files. This program takes in 3 inputs: (1) *gene names and sequences* contains a list of unique IDs for each guide variant and the associated sequence. (2) *barcodes* contains all sequence barcodes, where each barcode corresponds to a bin that each cell was binned to. Barcodes are short sequences, unique to each cell sorter bin, that would be added via PCR. (3) *mock_bins_counts* the file that has the mock bins generated from the last step.

Each entry in a FASTQ file consists of 4 lines:

- Sequence identifier, which contains the following elements:
  - @
  - instrument ID
  - run number on instrument
  - flowcell ID
  - lane number
  - tile number
  - x coordinate of cluster
  - y coordinate of cluster
  - UMI sequences for Read 1 and Read 2
  - **read number** (1)
  - Y if the read is filtered, N otherwise
  - control number
  - **index** (2)
- **sequence** (3)
- Quality score identifier line (consisting only of a +)
- Quality Score

All items in the mock fastq are placeholders, except for (1) **read** number is 1 or 2 depending on if it is read1 or read2 (2) **index** consists of the barcode index associated with the bin (3) the **sequence** of the guide variant.

First for each guide variant from the *mock_bins_counts* file, its associated read1 and read2 are generated. read1 is 39 bases taken from the sequence and begins with TAGG. read2 is 36 bases taken from the reverse complement of the sequence and begins with CCTA.

For each of the bins from the cell sorter in *mock_bins_counts*, the program assigns a barcode to each bin.

Below this is detailed in psuedocode:

```
For each guide variant from the mock_bins_counts file:
    seq = this guide's sequence
    rev = the guide's reverse compliment
    seq_i = index in seq where TAGG is
    rev_i = index in rev where CCTA is
    read1 = seq[seq_i : seq_i + 39]
    read2 = rev[rev_i - 32: rev_i + 4]
    For each bin:
        set the barcode to the barcode sequence associated with this bin
        let n = number of entries in this bin
        output n entries with read number=1, index=barcode, and sequence=read1
        output n entries with read number=2, index=barcode, and sequence=read2
```

4.2.1 Example

Take poxBp_T7_TGT. It has a sequence of:

```
CTGAAGTCAGCCCCATACGATATAAGTTGTTACTAGATTGACAGCTAGCTCAGTCCTAGGTATAATACTA
GTTAACGGTTAAATAGCCCGATGTTTTAGAGCTAGAAATAGCAAGTTAAAATAAGGCTAGTCCGTTATCA
ACTTGAAAAAGT
```

It's read1 is calculated as:

```
TAGGTATAATACTAGTTAACGGTTAAATAGCCCGATGTT
```

It's read2 as:

```
ATCGGGCTATTTAACCGTTAACTAGTATTATACCTA
```

The second bin, bin 1, labeled 1000, is associated with AACCGCAA. This is from the unique barcode file.

Based on the binning file, poxBp_T7_TGT has 32 reads in the second bin. So I generated 32 records for read1 and 32 records for read2.

The 32 records in read1 look like this:

```
@SIM:1:FCX:1:14:6329:1045:1:N:0:AACCGCAA
TAGGTATAATACTAGTTAACGGTTAAATAGCCCGATGTT
+
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

and the 32 records in read2 look like this:

```
@SIM:1:FCX:1:14:6329:1045:2:N:0:AACCGCAA
ATCGGGCTATTTAACCGTTAACTAGTATTATACCTA
+
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

4.3 Results

This process was repeated for all guide variants and each of the bins to get a total of 25980000 records per file. Each file generated was over 3 GBs of data, but could be processed into a program and a smaller data format with Biopythons SeqIO in the order of minutes. This implies that processing the data with Biopython could be completed in a reasonable amount of time for researchers.

4.4 Discussion & Future Work

The mock data for the pipeline will be validated once experimental results are generated and compared to the data generated by the pipeline. Some subset of the experiment will need to be performed and compared to the mock data. Statistical analysis would be needed to check the similarities between the synthetic and real-world data to understand the differences, and what changes need to be made to improve the statistical model. After this validation, researchers can be more confident that the synthetic data can be used as a stand in for real data when it comes to developing the framework for analysis pipelines.

5 *P. putida* Gene Expression CRISPRa Filtering

The goal of this particular project was to find effective candidates for CRISPRa/i in the *Pseudomonas putida* genome. *P. putida* has unique metabolism that CRISPRa could take advantage of to convert various renewable materials into desired compounds with high precision and efficiency [16]. *P. putida* CRISPRa has been demonstrated, but the rules to suggest effective CRISPRa are currently unknown [13]. In order to utilize *P. putida* in this way, we aimed to discover which endogenous genes were good candidate for CRISPRa. This work was done by taking in several *P. putida* related datasets and using known CRISPRa rules to generate a list of suitable candidates for further study.

5.1 Data Sources

1. **RNA-Seq Data**. RNA Sequencing is a sequencing technique that uses next-generation sequencing to measure the quantity of RNA in a biological sample in order to measure gene expression. In this paper, RNA-Seq is used to refer to a specific set of RNA-Seq data that was captured as part of an experiment to measure the baseline gene expression of various genes in *P. putida*. This data was provided by Joshua Elmore.

   This data consists of 14 csvs. Each csv has 4 columns

   - **locus_tag**. An identifier that represents the gene. By convention, it begins with "PP" followed by an underscore and then a 4 letter sequence. For example "PP_0001"
   - **Name**. A description of the gene's product. If it is unknown what the gene product is, it will be labeled as "hypothetical protein CDS", where CDS stands for Coding Sequence.
   - **Raw Read**. The raw data read for this particular gene.
   - **FPKM**. Short for Fragments Per Kilobase of transcript per Million mapped reads. Basically, this is the total reads for that sample, divided by 1 million, divided by the length of the gene. This was used in the pipeline to represent baseline gene expression.

   Each csv was also performed using a particular nitrogen source, either ammonium (NH_4) or nitrate (NO_3).

2. **List of Activated Genes**. This data is a list of experimental data produced by experiments from the Carothers lab. 11 genes were tested to see if they were suitable candidates for activation based on their baseline expression. 4 of those genes were found to have suitable activation: PP_1776, PP_1992, PP_0786, PP_3668

3. **TSS Primary**. This dataset contains a list of genes considered to be the primary TSS location. Each gene can be under control of different promoters, thus having different TSS. The primary promoters have been predicted to be the most important one for the control of that particular gene.

4. **Genome Scale Model Outputs**. This file contained predictions of effective CRISPRa/i genes from a genome-scale model produced by our collaborator Hector Garcia Martin [17]. This genome-scale model can be thought of as another method to filter activatable genes Although this model determines in theory which genes would be best, it does not take into account if we can activate those genes in practice, which is why the additional analysis of this project is needed.

locus_tag
PP_1776
PP_4812
PP_3839
PP_1992
PP_0786
PP_1972
PP_3668
PP_5046
PP_1231
PP_4701
PP_3161

Table 1: 11 genes that were tested to see if they were suitable candidates for CRISPRa. Highlighted genes (PP_1776, PP_1992, PP_0786, PP_3668) were found to be suitable for activation.

5.2 Pipeline

1. **Normalization**. First the FPKM of the RNASeq data was normalized by the FPKM of a reference gene. This gene was chosen to be npII after suggestion from researchers. Every other gene's FPKM was divided by the FPKM of this reference gene.

2. **Averaging Across Results**. There were 4 different experimental conditions in the RNASeq data, each with a number of replicates: (1) JE1657 NH4, (2) JE1657 NO3, (3) JE2212 NH4, (4) JE2212 NO3. Each had 3-4 replicates. For each gene in each of these replicates, they were averaged, resulting in 4 combined datasets total.

   This experiment was performed under two conditions where the nitrogen source differed: it was either ammonium (NH4) or nitrate (NO3). The user can specify if they want to filter using both datasets or just one in the parameters part of the code. The experiment with ammonium will be closer to what we plan to perform, so the user can specify to use only this dataset in the filtering.

3. **Filtering By Primary TSS**. Each gene can be under control of different promoters, thus having different TSS. The primary promoters have been predicted to be the most important one for the control of that particular gene.

   When using the software, the user has the option to filter by the primary TSS or not. If this option is chosen, genes not in the Primary TSS dataset are filtered out.

   Filtering by the primary TSS significantly reduces the amount of genes outputted. At the beginning, there were a total of 5570 genes in the RNA Seq dataset. There are 1105 genes in the TSS Primary dataset. Without filtering by the TSS Primary Dataset, the filtering by just activation resulted in 1103 genes. Filtering by the TSS Primary Dataset and the activation yields 320.

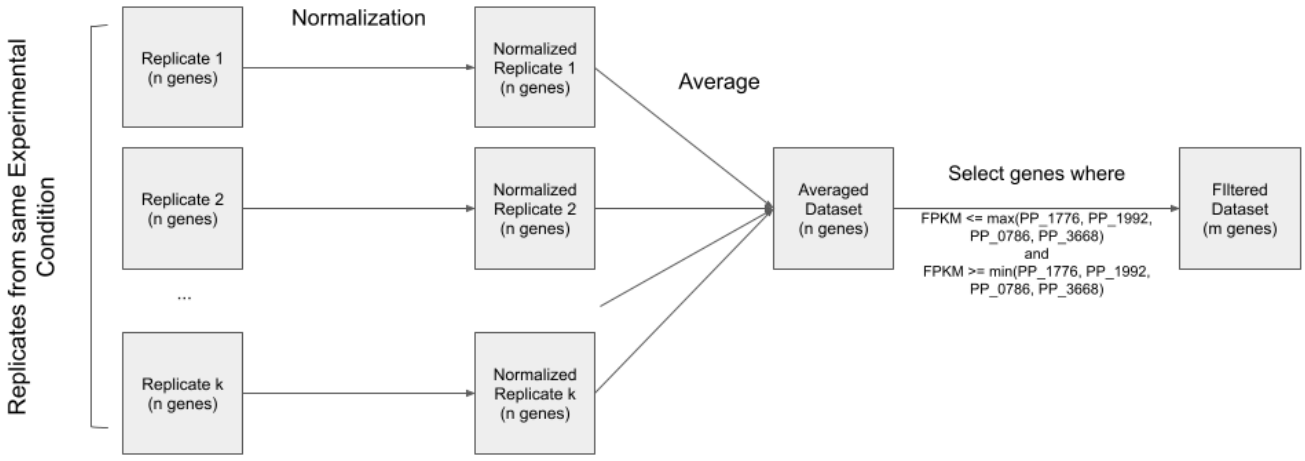

Figure 7: Filtering RNA Seq Data for each experimental condition. Includes the steps in the pipeline from 1 - 4(a). k is the number of experimental conditions, n is the number of unique genes, and m is some number where m ¡= n

4. **Filtering by Activation**. The next thing to do is to filter by genes that had activation within the FPKM range of genes that had activation within a desirable range in the experimental data: PP_1776, PP_1992, PP_0786, PP_3668.

   For each dataset, for each of the genes in the activation list, collect a list of FPKMs. Let the maximum activation be called max, and the minimum activation be called min. Remove all genes from the dataset that do not fall within this range. This is also written in psuedocode below:

```
For each dataset d:
    FPKM_list = empty list
    For each gene in the activation list:
        act = FPKM of that gene in d
        add act into FPKM_list
        max = maximum FPKM in FPKM_list
        min = minimum FPKM in FPKM_list
    For each gene in d:
        act = FPKM of gene in d
        if act > max or act < min:
            remove gene from d

For each dataset d:
    For each gene in d:
        if gene is in all other datasets:
            output gene
```

Several different methods were tested to see the best way to filter by these genes. These methods are detailed more in the experimentation section under filtering by activation.

5. **Interesction with Targets**. Finally, the intersection of genes in the set produced by the RNASeq filtering and the set of targets from the genome scale model were output.

5.3 Filtering by Activation Experimentation

Several different methods were tried order to filter genes that had activation within the FPKM range of the 4 genes deemed to have activation within desirable range.

The first step was to determine how each dataset should be filtered.

- **Filter by max and min for each dataset**. For each dataset, for each of the genes in the activation list, collect a list of FPKMs. Let the maximum activation be called max, and the minimum activation be called min. Because of how it is found, there will be a different max and min for each dataset. Remove all genes from the dataset that do not fall within this range.

- **Filter by a global highest max and lowest min**. For each dataset, for each of the genes in the activation list, collect a list of FPKMS. Combine this list of FPKMs across all datasets. Take the highest max and lowest min seen. Each dataset will filter by the same max and min. Remove all genes from each dataset that do not fall within this range.

- **Filter by a global lowest max and highest min**. For each dataset, for each of the genes in the activation list, collect a list of FPKMS. Combine this list of FPKMs across all datasets. Take the lowest max and highest min seen. Each dataset will filter by the same max and min. Remove all genes from each dataset that do not fall within this range.

- **Filter by a global average max and average min**. For each dataset, for each of the genes in the activation list, collect a list of FPKMS. Find the maximum, add it to a list called max list. Find the minimum, add it to a list called min list. Let the max be the average of the max list and the min be the average of the min list. Each dataset will filter by the same max and min. Remove all genes from each dataset that do not fall within this range.

- **Datasets combined**. For each dataset, for each gene, take the average FPKM. Now there is one dataset. For each of the genes in the activation list, collect a list of FPKMs. Take the max and min. Remove all genes from the dataset that do not fall within this range.

The second step was to determine which genes should be output based on the filtering.

- **Gene must be in at least one dataset**. In order for a gene to be considered a good

candidate, it must have ”survived” the filtering process for at least one dataset. This is a weaker criteria, and so therefore will output

- **Gene must be in all dataset**. In order for a gene to be considered a good candidate, it must have ”survived” the filtering process for all datasets. This is a stronger criteria, and so therefore will output less genes. less genes.

Each method was tried on the 4 datasets. The best method was evaluated by looking at the final set of activatable genes and using the following criteria:

- It should have as many of the 4 activated genes as possible
- It should have as few of the 7 non-activated genes as possible.

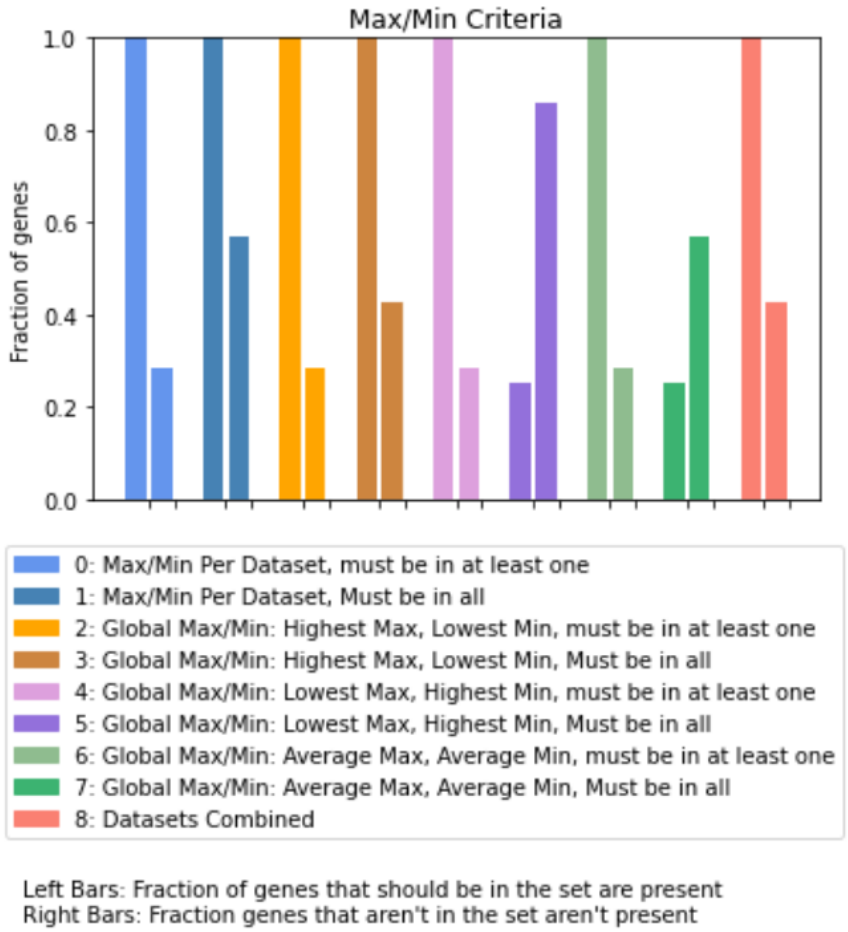

Figure 8: Results of the filtering by activation experiment

It was found that the method where a max/min is found per dataset and that gene must be in all datasets yielded the best results (See Fig 8). It screened out 0% of the genes known to be good activation candidates, and filtered out 60% of the genes that were known to not

be suitable activation candidates.

5.4 Results

Without taking into account the outputs of genome scale model, the software reduced 5571 genes down to 503 genes. 9% of genes input into the software were considered suitable for CRISPRa.

For CRISPRa, there were 15 genes provided by the genome scale model:

`PP_5078, PP_4297, PP_2329, PP_2082, PP_1830, tyrB, PP_5079, PP_4169, PP_1075, PP_4965, PP_4715, PP_2168.`

Using the pipeline, 5 of those genes were found to be potential candidates for CRISPRa:

`PP_2082, PP_1830, PP_1075, PP_4965, PP_1972`

The pipeline completes its run-time in under one minute, a reasonable length for this type of analysis.

5.5 Discussion & Future Work

This work is meant to be part of the Learn and Design stages of Design-Build-Test-Learn cycle to discover effective CRISPRa targets in *P. putida*. The Design-Build-Test-Learn cycle is meant to be an iterative cycle, where the process is repeated, where experiments are designed, built, tested, and learned from. Hopefully during each iteration more knowledge is gained and the experiments perform with more desirable results. The overall approach will be validated once experimental results are compared with the model-derived predictions. The genes output by the model would need to be built and tested then evaluated for their CRISPRa effectiveness. We could then take the number of the genes output by the software that were actually viable as a measure of the program's accuracy. Afterwards we could consider redesigning the software based on these results.

Another large limitation of this project is the validity of the filtering analysis method. Basing how to do filtering based on a technique that retains as many of the 4 activated genes as possible but as few of the 7 non-activated genes as possible is based on a model, but not based on statistics. Future work should explore different methods of filtering.

6 Conclusion

This work has detailed two software projects. The first was the Mock Data Generation Model for Facs Seq CRISPRa project. In this project synthetic data was generated in the style of a CRISPRa experiment where the effectiveness would be collected for several guides with the goal of using this data to uncover more rules for effective CRISPRa. The intent was to create this synthetic data to better understand and explore the expected data format before real data could be produced. The second project was *P. putida* Gene Expression CRISPRa Filtering. In this project we aimed to find effective candidates for CRISPR

in *Pseudomonas putida*'s endogenous genome. This work was done by taking in several *P. putida* related datasets and using known CRISPRa rules to generate a list of suitable candidates for further study. Given 5571 genes, this software initially filtered out 90% of input genes. After using the 15 outputs from a genome scale model, which are relevant to the metabolic engineering project. The software filters out an additional 30% of genes, resulting in 5 genes.

The overall approach will be validated once experimental results are compared with the model-derived prediction. For the first project, Mock Data Generation Model for FACS-seq CRISPRa, some subset of the experiment would need to be performed and compared to the mock data. Statistical analysis would be needed to check the similarities between the synthetic and real-world data to understand the differences. For the second project, *P. putida* Gene Expression CRISPRa Filtering, the genes output by the filtering model would need to be evaluated for their CRISPRa effectiveness. We would then be able to take the number of the genes output by the software that were actually viable as a measure of the program's accuracy. For both projects, we should consider redesigning the software based on comparisons to the real-world findings.

Currently the code is available internally to the Carothers Research group which is available to any lab member. This is currently on the Carothers Github repository. The first project is available as its own repository which under the title facs_seq_crispra. The second project is currently available internally under the CRISPRa_Endogenous repository in the RNAseq Activation Filtering folder. Due to both projects use of unpublished data, the repositories are private. We expect that the code and datasets will be published once the experimental portion has been completed.

While working on this project, I learned the unfortunate truth that it's nearly impossible to create a pipeline that would work for any organism. Elements of the pipeline are often dependent on available data, which may not exist or might look very different than data from the organism used to prototype the pipeline. This is especially true when it comes to the second project, *P. putida* Gene Expression CRISPRa Filtering, the code is very specific to the data for the project and would require modifications to work with other types of data. As such, the pipeline requires a user to do significant overhead work when it comes to data generation, collection, and cleaning.

Even with these limitations, future work has the potential to further revolutionize the field of synthetic biology. Scientists have the potential to use data driven approaches to analyze mass amount of data that seems unfeasible to parse by the human eye to uncover the rules that drive CRISPRa. These rules will be able to guide synthetic biologists to create effective metabolic pathways, paving the way for the efficient production of many important metabolites.

References

[1] W. A. L. Antonia A. Dominguez and L. S. Qi. Beyond editing: repurposing crispr–cas9 for precision genome regulation and interrogation. `https://www.`

nature.com/articles/nrm.2015.2, 2015.

[2] M. M. Boettcher M. Choosing the right tool for the job: Rnai, talen, or crispr. https://doi.org/10.1038/s41467-019-11479-0, 2016.

[3] J. A. R. C. e. a. Carbonell, P. An automated design-build-test-learn pipeline for enhanced microbial production of fine chemicals. https://www.nature.com/articles/s42003-018-0076-9, 2018.

[4] N. N. Chou Khai Soong Karlson, Siti Nurfadhlina Mohd-Noor and B. C. Tan. Crispr/dcas9-based systems: Mechanisms and applications in plant sciences. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8540305/, 2021.

[5] H. Dong, Y. Cui, and D. Zhang. Crispr/cas technologies and their applications in escherichia coli. https://www.frontiersin.org/article/10.3389/fbioe.2021.762676, 2021.

[6] D. C. K. C. e. a. Fontana, J. Effective crispra-mediated control of gene expression in bacteria must overcome strict target site requirements. https://doi.org/10.1038/s41467-020-15454-y, 2020.

[7] J. Fontana, D. Sparkman-Yager, J. G. Zalatan, and J. M. Carothers. Challenges and opportunities with crispr activation in bacteria for data-driven metabolic engineering. https://www.sciencedirect.com/science/article/pii/S0958166920300604, 2020. Analytical Biotechnology.

[8] M.-E. H. e. a. Gleditzsch D, Pausch P. Pam identification by crispr-cas effector complexes: diversified mechanisms and structures. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6546366/, 2019.

[9] T. M. Gruber and C. A. Gross. Multiple sigma subunits and the partitioning of bacterial transcription space. https://www.annualreviews.org/doi/pdf/10.1146/annurev.micro.57.030502.090913, 2003. PMID: 14527287.

[10] F. Hille and E. Charpentier. Crispr-cas: biology, mechanisms and relevance. https://royalsocietypublishing.org/doi/10.1098/rstb.2015.0496, 2016.

[11] M. S. G.-M. e. a. Hu, J. Evolved cas9 variants with broad pam compatibility and high dna specificity. https://doi.org/10.1038/nature26155, 2018.

[12] J. D. K. Jens Nielsen. Engineering cellular metabolism. https://doi.org/10.1016/j.cell.2016.02.004, 2016.

[13] C. Kiattisewee, C. Dong, J. Fontana, W. Sugianto, P. Peralta-Yahya, J. M. Carothers, and J. G. Zalatan. Portable bacterial crispr transcriptional activation enables metabolic engineering in pseudomonas putida. *Metabolic Engineering*, 66:283–295, 2021.

[14] L. X. . J. V. E. . P. . N. . Liao X, Makris M. Fluorescence-activated cell sorting for

purification of plasmacytoid dendritic cells from the mouse bone marrow. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5226086/, 2020.

[15] W. X. . W. B. Liu, Y. Engineered crispra enables programmable eukaryote-like gene activation in bacteria. https://doi.org/10.1038/s41467-019-11479-0, 2019.

[16] A. Loeschcke and S. Thies. Pseudomonas putida-a versatile host for the production of natural products. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4495716/, 2014.

[17] S. Roy, T. Radivojevic, M. Forrer, J. M. Marti, V. Jonnalagadda, T. Backman, W. Morrell, H. Plahar, J. Kim, N. Hillson, and H. Garcia Martin. Multiomics data collection, visualization, and utilization for guiding metabolic engineering. https://www.frontiersin.org/article/10.3389/fbioe.2021.612893, 2021.

[18] B. I. Tickman, D. A. Burbano, V. P. Chavali, C. Kiattisewee, J. Fontana, A. Khakimzhan, V. Noireaux, J. G. Zalatan, and J. M. Carothers. Multi-layer crispra/i circuits for dynamic genetic programs in cell-free and bacterial systems. https://www.sciencedirect.com/science/article/pii/S2405471221004191, 2021.