Sparse dictionary learning identifies genetic pleiotropy in a genome-scale Perturb-seq screen

William Colgan

***Abstract*— Understanding the function of genes in the human genome is a fundamental goal for molecular biology. CRISPR-Cas9 has provided a powerful tool for characterizing gene function by studying the phenotypic consequences of genetic knockouts. Perturb-seq, a technology combining CRISPR-Cas9 perturbations with single-cell RNA sequencing, enables high-dimensional phenotyping via the measurement of gene expression changes. In this paper, I present a sparse dictionary learning approach using dual graph regularized k-SVD to factorize gene knockout by gene transcriptional response matrices obtained from Perturb-seq experiments. I validate my method using synthetic data and then apply it to a genome-scale Perturb-seq dataset. I determine the number of distinct transcriptional programs in this dataset and analyze their alignment with transcription factor targets. My factorization reveals numerous examples of genetic pleiotropy and provides insights into the biological functions of protein complexes, including the Mediator complex's role in the regulation transcription by both RNA polymerase I and II. My approach demonstrates the potential of sparse dictionary learning for characterizing transcriptional programs and deconstructing gene functions in Perturb-seq data.**

I. Introduction

The comprehensive understanding of the function of every gene in the human genome remains a central objective of molecular biology research. The development of the CRISPR-Cas9 technology was a significant milestone towards achieving this objective, providing researchers with a powerful tool to disrupt any gene in the genome [1]. The CRISPR-Cas9 system employs a guide RNA to direct the Cas9 endonuclease to the specific DNA segment coding for the targeted gene. With a vast library of guide RNAs, a cell population that harbors every possible gene knockout can be created, thereby allowing the evaluation of the phenotypic consequences of each gene deletion [2]. For instance, the relative abundance of gene knockouts can be quantified following multiple rounds of cell division to determine the impact of each gene on cellular proliferation.

Genome-scale CRISPR-Cas9 fitness screens have become common in biology research with many cell types under diverse conditions being screened. Among the most substantial initiatives employing this technique is the Cancer Dependency Map (DepMap) project. This project aims to screen hundreds of cancer cell lines to systematically map cancer vulnerabilities [3]. While the primary objective is to identify novel therapeutic targets, the dense cell type by gene effect matrix derived from screens facilitates the characterization of gene function. This is primarily because genes with analogous functions often exhibit similar fitness profiles across different screened cell lines [4].

This observed similarity in fitness profiles is attributed to the variation in sensitivity among different cell lines to the disruption of distinct biological processes. For instance, certain cell lines demonstrate heightened sensitivity to disruptions in the MAPK pathway. Consequently, all genes within the MAPK pathway tend to present a more pronounced fitness effect in these particular cell lines. Several methods have been developed to learn gene function from the DepMap data [5], including Webster, which uses sparse dictionary learning to factorize the fitness matrix into a dictionary of biological functions and a loading matrix with the contribution of each function to the fitness profile of each gene [6].

While CRISPR-Cas9 fitness screens are a powerful tool for determining gene function, especially when conducted in many cell types, they are fundamentally limited by the fact that only a single phenotype is measured for each gene: its impact on fitness [2]. A substantial proportion of genes in the genome display no discernible effect on fitness following knockout, rendering their function indeterminable through this method [7]. Thus, more nuanced phenotypic characterization of CRISPR-Cas9 screens is required to gain a comprehensive understanding of gene function.

Recent technological advancements have facilitated richer phenotypic characterizations by enabling the imaging of cells with specific perturbations or measurement of their RNA transcriptome [8], [9]. One such advancement is Perturb-seq, a revolutionary technology that merges pooled CRISPR-Cas9 perturbations with single-cell RNA sequencing (scRNA-seq). This technology allows for the measurement of differential gene expression in response to gene knockouts, thereby offering a more detailed depiction of the perturbation’s effect which can be used determine gene function [9].

In 2022, a landmark study by Replogle and colleagues utilized Perturb-seq to generate the first genome-scale transcriptional response dataset, comprising approximately 10,000 gene knockouts in a cancer cell line [10]. To identify sets of genes with similar functions, the authors clustered the perturbations based on transcriptional response similarity. This analysis revealed 38 distinct transcriptional programs, providing a new way to characterize the function of numerous poorly understood genes. For example, they were able to

MIT 18.051 matrix methods final project submission. William Colgan worked alone on this project and is solely responsible for the contents of this paper.

classify C7orf26 as a new member of the integrator complex since its knockout resulted in a similar transcriptional response to the knockout of other genes in this complex.

Despite the impressive results in Replogle et al.'s study, their clustering approach has limitations. One of these limitations is the assumption that each gene has a singular biological function. This assumption conflicts with the principle of genetic pleiotropy, which posits that genes can have multiple functions and their disruption can affect various cellular processes [11]. Therefore, an approach is required that accounts for the pleiotropic nature of genes, modeling each transcriptional response as the combination of a small set of transcriptional programs.

For this project I took inspiration from the CRISPR-Cas9 fitness literature to develop a model to factorize Replogle et al.'s gene knockout by gene transcriptional response matrix. Specifically, I employ the sparse dictionary learning approach using dual-graph-regularized k-SVD that was proposed Pan et al. for Webster [6]. This factorization deconstructs the Perturb-seq matrix into a dictionary of atoms which I refer to as transcriptional programs, since they are a basis of the gene response space, and a sparse loading matrix, which defines which transcriptional programs are used to reconstruct the transcriptional response to each gene perturbation. This model implicitly learns pleiotropy since transcriptional programs are shared between perturbations and each perturbation response is modeled as weighted a combination of a few transcriptional programs.

In Section II, I give a broad overview of sparse dictionary learning and the algorithms that have been developed to solve this problem. In Section III, I provide details about my methods including my implementation of dual-graph-regularized k-SVD. In Section IV, I present my results including validation with a synthetic dataset, comparison to PCA, and a detailed analysis of the factorization of perturbations in the RNA polymerase II and Mediator complexes. I conclude with Section V, discussing my results and the application of my method to other transcriptional response datasets.

II. Sparse Dictionary Learning

Sparse dictionary learning is a powerful technique in the field of signal processing and machine learning that aims to represent data signals as sparse linear combinations of atoms from a dictionary [12]. It has gained significant attention due to its ability to capture the underlying structure and redundancy in high-dimensional data, enabling efficient data representation, denoising, and compression.

In many real-world applications, signals can be expressed using only a small number of basis elements from a given dictionary. Mathematically, for a signal vector $\mathbf{y} \in \mathbb{R}^N$, the sparse representation can be formulated as:

$$\mathbf{y} \approx \mathbf{D}\mathbf{x}. \tag{1}$$

where $\mathbf{D} \in \mathbb{R}^{N \times K}$ is the dictionary matrix composed of K atoms (or columns), and $\mathbf{x} \in \mathbb{R}^K$ represents the sparse coefficient vector. The goal of sparse dictionary learning is to find both the optimal dictionary **D** and the sparse coefficient vector **x** that best represent the input signal vector **y**.

The dictionary **D** can be either undercomplete with $N < K$ or overcomplete with $N > K$. Atoms $\mathbf{d}_1, \ldots \mathbf{d}_n$ are not required require to be orthogonal. Dictionaries can be designed by either selecting one from a prespecified set of transforms are adapting the dictionary as part of the training procedure [12].

Prespecified dictionaries are predefined dictionaries designed to capture specific signal properties or structures. These dictionaries are often carefully crafted based on prior knowledge or domain expertise. Examples of prespecified dictionaries include wavelet dictionaries, Fourier dictionaries, and curvelet dictionaries [13], [14]. While prespecified dictionaries offer interpretability and computational efficiency, they may not always be optimally suited for a specific dataset or signal class.

In contrast, learning adapted dictionaries aims to automatically learn the dictionary from the given dataset itself. This approach allows the dictionary to be tailored to the specific characteristics and structures of the signals in the dataset. By learning the dictionary from data, it becomes possible to capture the most discriminative and relevant atoms that provide an accurate representation of the signals. Adaptive dictionary learning methods have shown remarkable success in various applications, including image denoising [15].

A. Sparse Coding

A key step in dictionary learning is sparse coding, the process of computing the representation vector **x** from the signal **y** and the dictionary **D**. The problem of finding the best sparse representation of a signal is known to be NP-hard, so approximate "pursuit" algorithms are generally used [16]. While inexact, these algorithms typically yield satisfactory results in practice.

Matching Pursuit (MP) is one widely used approach for approximate sparse coding. MP iteratively selects dictionary elements that have the highest inner product with the residual signal and updates the corresponding coefficients [17]. At each iteration, the atom with the highest inner product is added to the representation and its contribution is subtracted from the residual. This process continues until a predefined stopping criterion is met, such as a set number of non-zero coefficients or a desired level of approximation error.

Orthogonal Matching Pursuit (OMP) is a variant of Matching Pursuit that imposes orthogonality constraints on the selected atoms. OMP iteratively selects the atom with the highest inner product with the residual and orthogonalizes it with respect to the previously selected atoms [18]. This orthogonality constraint helps mitigate the issue of redundant atoms and improves the efficiency of the sparse coding process.

B. Dictionary Learning

Now that we have a method for computing the representation vector **x** we can turn our attention to finding the optimal

combination of **x** and the dictionary **D.** Given matrix of examples $\mathbf{Y} = [\mathbf{y_1}, \ldots, \mathbf{y_m}]$ this can be formulated as an optimization problem:

$$\min_{\boldsymbol{D},\boldsymbol{X}}\{\|\boldsymbol{Y}-\boldsymbol{DX}\|_F^2\}\ s.t.\ \|\boldsymbol{x}_i\|_0 \le \boldsymbol{T}\ \forall i \qquad (2)$$

where $\mathbf{X} = [\mathbf{x_1}, \ldots, \mathbf{x_n}]$ and **T** is the maximum number of non-zero coefficients.

The Method of Optimal Direction (MOD) is a classic approach for solving this problem [19]. MOD is an iterative algorithm that alternates between getting the sparse coding using a method such as OMP and updating the dictionary by finding the analytical solution to the problem given by $\mathbf{D} = \mathbf{XR}^+$ where $\mathbf{R}^+$ is the Moore-Penrose pseudoinverse. This process is repeated until convergence. MOD works well and is efficient for low-dimensional inputs **X**, but because it requires a matrix inversion it does not scale well with the dimensionality of **X**.

To address the limitations of MOD several other sparse dictionary learning algorithms have been developed, including algorithms based on stochastic gradient descent [20] and the Lagrange dual formulation [21]. However, the most popular algorithm is k-SVD which I employ in this paper and describe in detail below.

C. k-SVD Algorithm

The k-SVD algorithm is a widely used and effective method for dictionary learning [22]. Like MOD, the algorithm iteratively updates the dictionary and the sparse coefficient vectors. First, OMP or another approximation algorithm is employed the calculate the coefficients **X.** Second, the algorithm searches for a better dictionary **D** given the coefficients **X.**

Task: Find the best dictionary to represent the data samples $\{\mathbf{y}_i\}_{i=1}^N$ as sparse compositions, by solving

$$\min_{\mathbf{D},\mathbf{X}}\{\|\mathbf{Y}-\mathbf{DX}\|_F^2\} \quad \text{subject to} \quad \forall i,\ \|\mathbf{x}_i\|_0 \le T_0.$$

Initialization : Set the dictionary matrix $\mathbf{D}^{(0)} \in \mathbf{R}^{n\times K}$ with ℓ^2 normalized columns. Set $J = 1$.

Repeat until convergence (stopping rule):

- *Sparse Coding Stage*: Use any pursuit algorithm to compute the representation vectors $\mathbf{x}_i$ for each example $\mathbf{y}_i$, by approximating the solution of

$$i = 1, 2, \ldots,\ N, \quad \min_{\mathbf{x}_i}\{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2\} \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \le T_0.$$

- *Codebook Update Stage*: For each column $k = 1, 2, \ldots, K$ in $\mathbf{D}^{(J-1)}$, update it by
  - Define the group of examples that use this atom, $\omega_k = \{i |\ 1 \le i \le N,\ \mathbf{x}_T^k(i) \ne 0\}$.
  - Compute the overall representation error matrix, $\mathbf{E}_k$, by

$$\mathbf{E}_k = \mathbf{Y} - \sum_{j\ne k} \mathbf{d}_j \mathbf{x}_T^j.$$

  - Restrict $\mathbf{E}_k$ by choosing only the columns corresponding to ω_k, and obtain $\mathbf{E}_k^R$.
  - Apply SVD decomposition $\mathbf{E}_k^R = \mathbf{U}\boldsymbol{\Delta}\mathbf{V}^T$. Choose the updated dictionary column $\tilde{\mathbf{d}}_k$ to be the first column of $\mathbf{U}$. Update the coefficient vector $\mathbf{x}_R^k$ to be the first column of $\mathbf{V}$ multiplied by $\boldsymbol{\Delta}(1,1)$.
- Set $J = J + 1$.

k-SVD algorithm from [22]

The k-SVD algorithm is efficient because it updates one column of the dictionary at a time. The update of the k-th column is accomplished by solving a minimization problem. This process decomposes the residual matrix into a sum of rank-1 matrices, assuming all other terms are fixed except for the k-th column. The k-th column of the dictionary is then updated using the SVD approximation of the residual matrix.

To address the sparsity constraint, the algorithm defines a set ω_k that includes the indices of examples that use the atom $\mathbf{d}_k$ in the dictionary. This set is used to shrink the coefficient vector $\mathbf{x}_k$ by discarding the zero entries. By performing multiplications with the corresponding matrices, the minimization problem is transformed into a form that can be solved using SVD. The solution for $\mathbf{d}_k$ is obtained from the first column of **U**, and the coefficient vector $\mathbf{x}_k$ is obtained from the first column of the **V** multiplied by $\boldsymbol{\Delta}(1,1)$.

D. Dual Graph Regularized k-SVD

While k-SVD is a simple and effective algorithm for sparse dictionary learning it ignores any structure that may be present in the data matrix **Y**. This simplification can produce suboptimal results when **Y** is noisy, as is the case for many biological problems. To take advantage of structure in the underlying data Yankelevsky and Elad proposed an updated version of k-SVD called dual-graph-regularized dictionary learning (DGRDL) [23].

DGRDL enforces smoothness of the dictionary **D** and the coefficients **X** with respect to structure in **Y** using graph Laplacians. Specifically, the dictionary atoms are smooth with respect to the signal similarity graph and the coefficients are smooth with respect the feature similarity graph. This is accomplished by adding additional terms to the dictionary learning objective:

$$\min_{\boldsymbol{D},\boldsymbol{X}}\{\|\boldsymbol{Y}-\boldsymbol{DX}\|_F^2\} + \alpha\mathrm{Tr}(\mathbf{D}^T\mathbf{L}\mathbf{D}) + \beta\mathrm{Tr}(\mathbf{X}\mathbf{L}_C\mathbf{X}^T) \qquad (3)$$
$$s.t.\ \|\boldsymbol{x}_i\|_0 \le \boldsymbol{T}\ \forall i$$

Where $\mathbf{L} \in \mathbb{R}^{M\times M}$ and $\mathbf{L_C} \in \mathbb{R}^{N\times N}$ are the Laplacian matrices derived from similarity graphs in **Y** and α and β are hyperparameters. The optimization of this objective is more complicate than the standard one, but it follows the same steps as regular k-SVD with **X** and **D** being updated iteratively.

I chose to use this version of k-SVD to model the Perturb-seq data since the Webster method demonstrated that it performed well for the analogous problem of modeling CRISPR-Cas9 fitness screens across cell lines [6]. For Perturb-seq data, **L** is the Laplacian matrix derived from the gene response similarity graph and $\mathbf{L_C}$ is the Laplacian matrix derived from the gene perturbation similarity graph. This means that the dictionary of transcriptional programs should vary smoothly with respect to transcriptional response similarity and the coefficient matrix should vary smoothly with respect to gene knockout similarity.

III. Methods

A. Sparse dictionary learning

Like the Webster method, several modifications were made to the DGRDL method [6]. First, nearest-neighbor graphs using cosine similarity were used for regularization, with the number of neighbors in both the gene response and gene perturbation spaces being hyperparameters. Second, the dictionary was pre-initialized using k-medoids since this leads to faster convergence. Finally, the matrix X was scaled by a factor 1/sqrt(n) to convert the coefficients into units of standard deviation which makes them analogous to the loading coefficients in PCA.

To implement DGRDL, I used the convenient MATLAB wrapper published by Pan et al. (https://github.com/joshbiology/graph_dictionary_learning) [6]. To validate my implementation, I generated a simple synthetic Perturb-seq dataset. First, I made a dictionary in $\mathbb{R}^{25\times2}$ with two randomly generated transcriptional program with a correlation coefficient of 0.2. Second, I made a loading matrix in $\mathbb{R}^{2\times60}$ with 20 perturbations mapping to program 1, 20 perturbations mapping to program 2, and 20 perturbation mapping to both programs. Finally, I generated the synthetic data in $\mathbb{R}^{25\times60}$ by multiply the dictionary and the loading matrices and adding gaussian white noise with a standard deviation of 1. I then applied DGRDL to this matrix with K=2, T=2, and neighbor graph degree = 1.

For real the Perturb-seq data, DGRDL was run with T=4 and the 15 nearest neighbors in both the gene response and gene perturbation space. Since these were the hyperparameter settings used by the Webster method. To select K, the number of gene programs, all K values between 10 and 300 were tested incrementing K by 5. The optimal K was then identified based on an elbow in the K vs reconstruction error plot.

B. Perturb-seq data

Single cells representing each gene knockout were averaged to generate one differential expression profile for each perturbation. The expression of each gene in this profile was then z-scored relative to negative control perturbations. This generated a 11,258 gene by 8,248 perturbation matrix, where each entry is the differential expression of a given gene in response to a gene knockout relative to negative controls.

To match the clustering analysis conducted by Replogle et al. [10], this matrix was subset to only include variable genes and strong perturbations. Strong perturbations were defined by three criteria: (i) at least 50 differentially expressed gene at a significance of $p < 0.05$ using the Anderson-Darling test; (ii) at least 25 cell passing quality control filters; and (iii) significant knockdown of the target gene. The union of two sets of genes were used as features: (i) the top 10 differentially expressed genes from the selected perturbations and (ii) highly expressed genes in the top 30% of variance. This procedure generated a 3,384 gene by 1,946 perturbation matrix. The perturbation responses where centered and scaled, then the differential expression of each gene across the perturbations was centered.

C. Transcription factor activity

The ChEA database (2016 release) of transcription factor targets was used to estimate transcription factor activity across the gene knockouts [24]. Specifically, for each transcription factor the average response of its target genes was calculated for each perturbation. This generated an activity profile of length 1,946 for each transcription factor. To look for alignment between transcription factors and learned transcriptional programs each of these profiles was correlated with the length 1,946 loading vector for each dictionary element. Pearson correlation was used. As a baseline PCA was run on the Perturb-seq matrix with rank = 80 to generate analogous dictionary and loading matrices.

D. Annotating perturbations and programs

The loading heatmap in Fig. 3 was annotated with protein complexes from CORUM (2022 release) [25]. For genes that are members of multiple complexes in CORUM the largest complex was used. To simplify the heatmap only genes that are part of a complex with more than 10 proteins are shown.

Gene set enrichment analysis was used to annotate transcriptional programs. Specifically, a hypergeometric enrichment test was used to quantify the overlap between the top 30 differentially expressed genes in the transcriptional program and gene sets from the Hallmark, GO Biological Process, and Canonical collections. P-values were corrected for multiple hypothesis testing using the Benjamini-Hochberg method.

IV. Results

The goal of this project is to factorize genome-scale Perturb-seq data into a dictionary of atoms and a sparse representation matrix. I refer to these atoms as ‘transcriptional programs’ as they represent a basis in the gene transcriptional response space. The representation matrix is referred to as the loading matrix and can be thought of as the extent to which each perturbation (gene knockout) modulates each transcription program. Fig. 1A is a schematic showing how the transcriptional response to the knockout of a fictitious gene X can be represented as the weighted sum of four transcriptional programs from the dictionary.

A. Validation

To validate my implementation or dual-graph-regularized dictionary learning (DGRDL) and provide intuition for how dictionary learning can be applied to Perturb-seq data, I generated a synthetic Perturb-seq dataset with two transcriptional programs and 60 perturbations, some of which exhibited pleiotropy, activating both transcriptional programs (Fig. 1B and 1C). Since real biological data is noisy, I added gaussian white noise to this matrix. I then applied DGRDL to the synthetic Perturb-seq dataset to reconstruct the dictionary and loading matrices. As expected, the reconstructed matrices

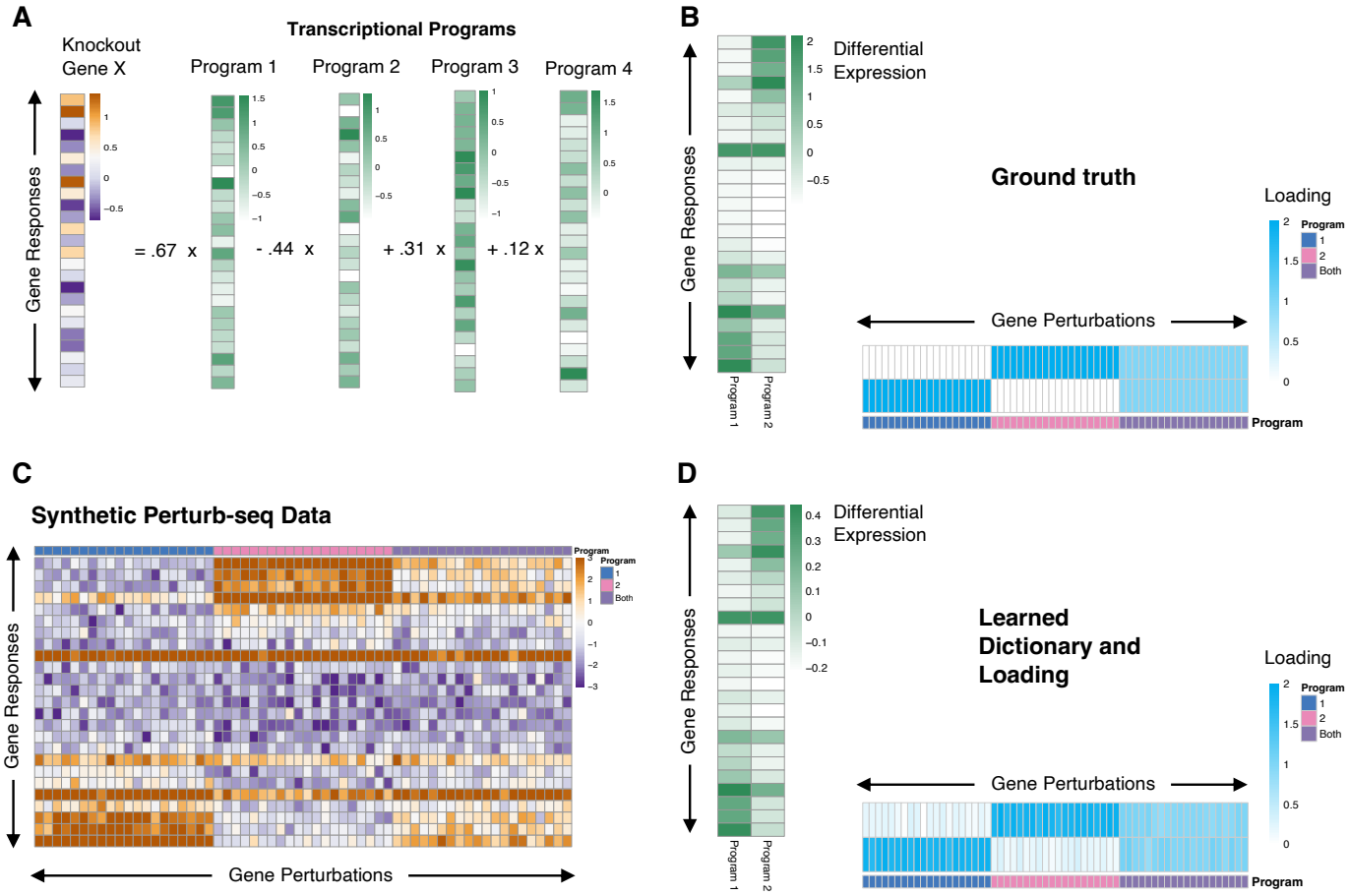

Fig. 1. A) Schematic showing how the transcriptional response to knocking out gene X can be modeled as the weighted sum of four transcriptional programs from the dictionary. B) Synthetic data with two transcriptional programs and 60 perturbations where a third of the perturbations exhibit pleiotropy, activating both transcriptional programs. C) Synthetic Perturb-seq data generated by multiplying the two matrices from B and adding white noise. D) DGRDL reconstructed dictionary and loading matrices from synthetic Perturb-seq data.

were similar to the ground truth matrices indicating that my DGRDL implementation was working (Fig. 1D).

B. Determining the number of transcriptional programs

To determine the number of distinct transcription programs (dictionary atoms) in the Perturb-seq dataset I tested all K values between 10 and 300 with a step size of 5. The reconstruction error strictly decreased with increasing K as larger K values correspond to more model parameters (Fig. 2A). However, the marginal decrease in reconstruction error flattened out around K = 80 (Fig. 2B). This elbow suggested that there are approximately 80 distinct transcriptional programs that can be affected by knocking out genes in this cancer cell line.

80 transcriptional programs is notably higher than the 34 transcriptional clusters that Replogle et al. identified in their analysis of this dataset [10]. But this discrepancy is not unexpected given that sparse dictionary learning enables more flexible program discovery. K-means clustering is analogous to running sparse dictionary learning with T = 1 [22], meaning that each perturbation must be assigned to a single transcriptional program. By running sparse dictionary learning with T = 4, I can discover transcriptional programs that are not the dominant component for any given perturbation. For example, transcriptional programs can be a minor component of the transcriptional response that is shared across many perturbations.

C. Comparing transcriptional programs to transcription factor activity

A cells transcriptional response to a perturbation is mediated by DNA binding transcription factors. In short, transcription factors or their regulators sense the state of the cell and then turn up or down the expression of a set of target genes in response. This set of target genes can be identified by disrupting the transcription factor in question or looking at where it binds in the genome [24]. Since transcription factors are the key determinants of transcriptional variation in the cell, we desire a basis where each transcriptional program aligns with a distinct transcription factor or set of transcription factors that act together.

To assess how well the dictionary learned by DGRDL on the perturb-seq data aligns with known transcription factor

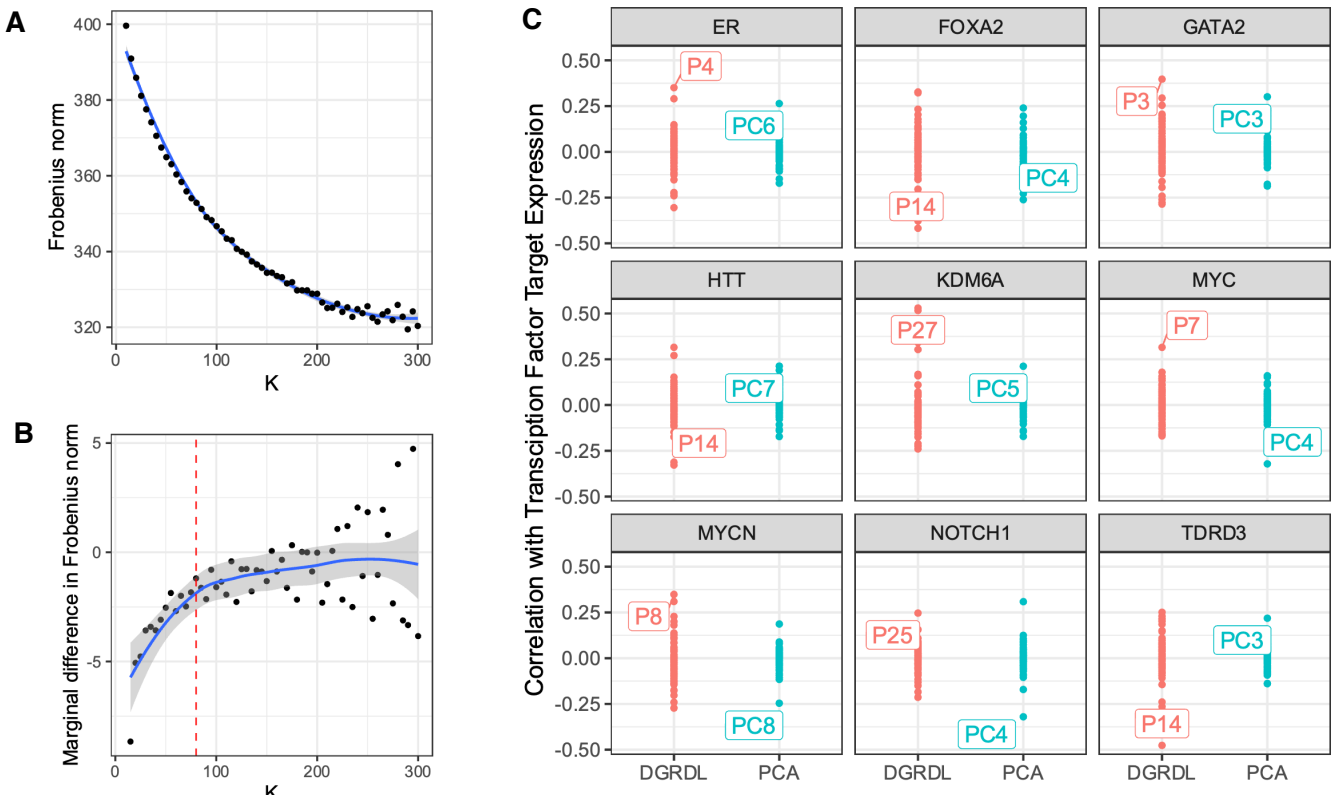

Fig. 2. A) Frobenius norm between Perturb-seq response matrix and DGRDL reconstruction. T is fixed at 4 while K values between 10 and 300 are tested with a step size of 5. B) Diminishing returns in this objective are observed at K=80 (dotted line). C) 9 randomly selected transcription factors. Distribution of correlations across perturbations between transcription factor target expression and transcriptional program loading for DGRDL and PCA.

targets, I scored the activity of each transcription factor across the perturbations by calculating the average response of its target genes. I then looked at the distribution of correlations between transcription factor activities and loadings. If a dictionary element perfectly aligns with a transcription factor its correlation should be 1 or -1 meaning the transcription factor is active for the same perturbations as the dictionary element is active.

For this analysis, I compared DGRDL to PCA since PCA is commonly used to identify unique programs in transcriptional response datasets. I expect DGRDL to outperform PCA since it does not require the transcriptional programs to be orthogonal. As expected, for most transcription factors the best transcriptional program learned by DGRDL has a higher correlation than the best transcriptional program learned by PCA (Fig. 2C). For some transcription factors program pairs, such as KDM6A and Program 27, the correlation is particularly high suggesting that DGRDL has modeled the activity of this transcription factor. A few transcription factors, such as NOTCH1, did not have a good program match, but this is not unexpected since this transcription factor may not be modulated in this dataset or may act in concert with other factors. Moreover, there are more transcription factors than learned programs so we should not expect all of them to be modeled.

D. Analysis of protein complexes

To assess the quality the DGRDL factorization of the Perturb-seq transcriptional response matrix, I clustered the perturbations by their loadings and then annotate known proteins complexes (Fig. 3). Protein complexes nicely clustered in the loading space and were often uniquely distinguished by a specific transcriptional program. For example, usage of program 3 was mostly specific to genetic perturbations within the Mediator complex.

The loading heatmap also revealed numerous examples of pleiotropy, the genetic principle that that genes can have multiple functions. Pleiotropy is indicated by protein complexes that modulate multiple transcriptional programs and transcriptional programs that are modulated by multiple protein complexes. For example, reconstruction of the transcriptional response to perturbations in both ribosomal and tRNA-synthetase complexes uses transcription program 14. This makes sense because both ribosomes and tRNAs are required for translation. Disrupting translation should trigger a common transcriptional response in the cell.

Another nice example pleiotropy is program 79 which maps to genes in both the RNA polymerase II complex and the Mediator complex. I will explore this example in greater detail in the next section.

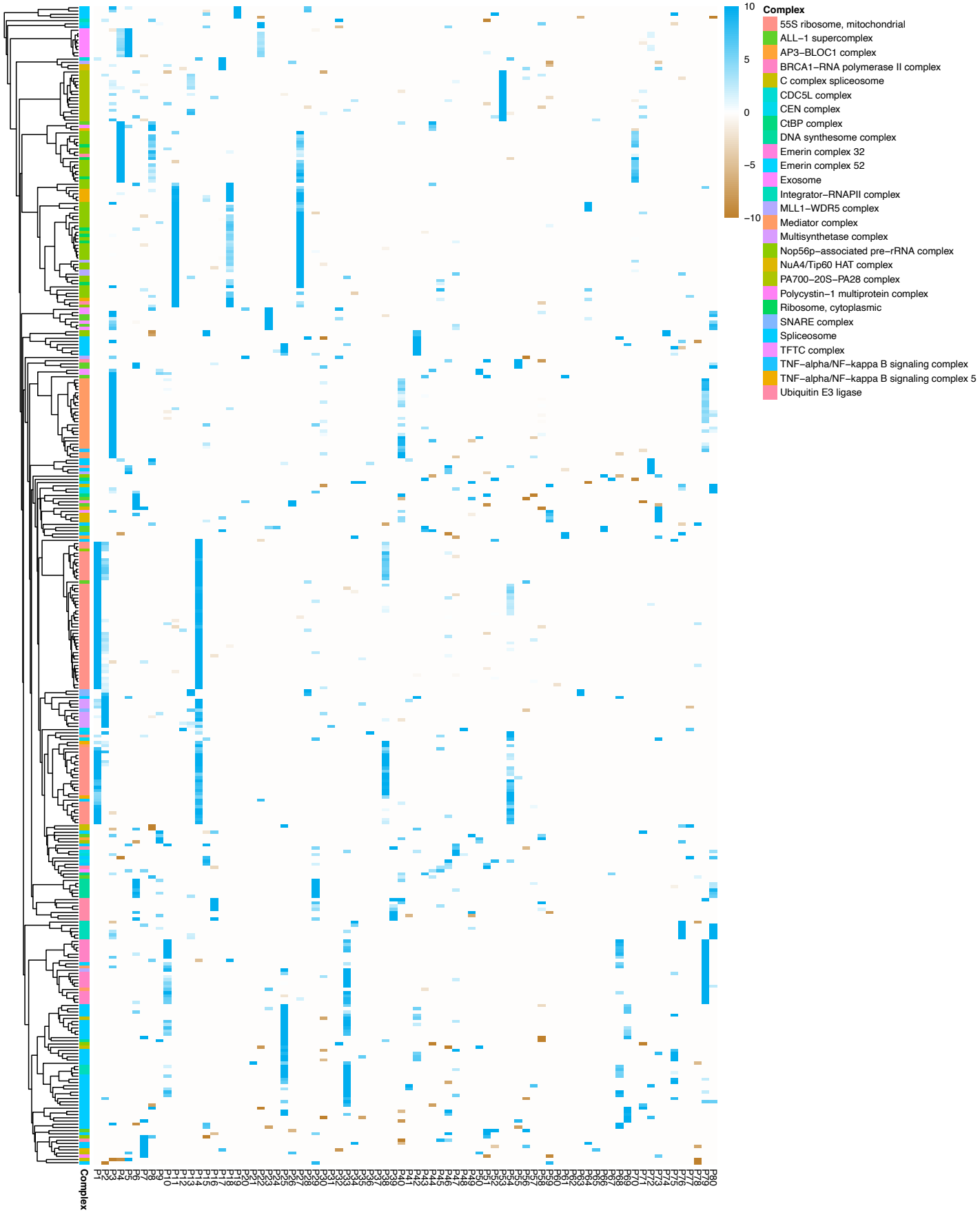

Fig. 3. Loading matrix from DGRDL run on Perturb-seq data with K=80 and T=4. Colors correspond to protein complexes from the CORUM database. Only genes that have protein complex annotations are shown.

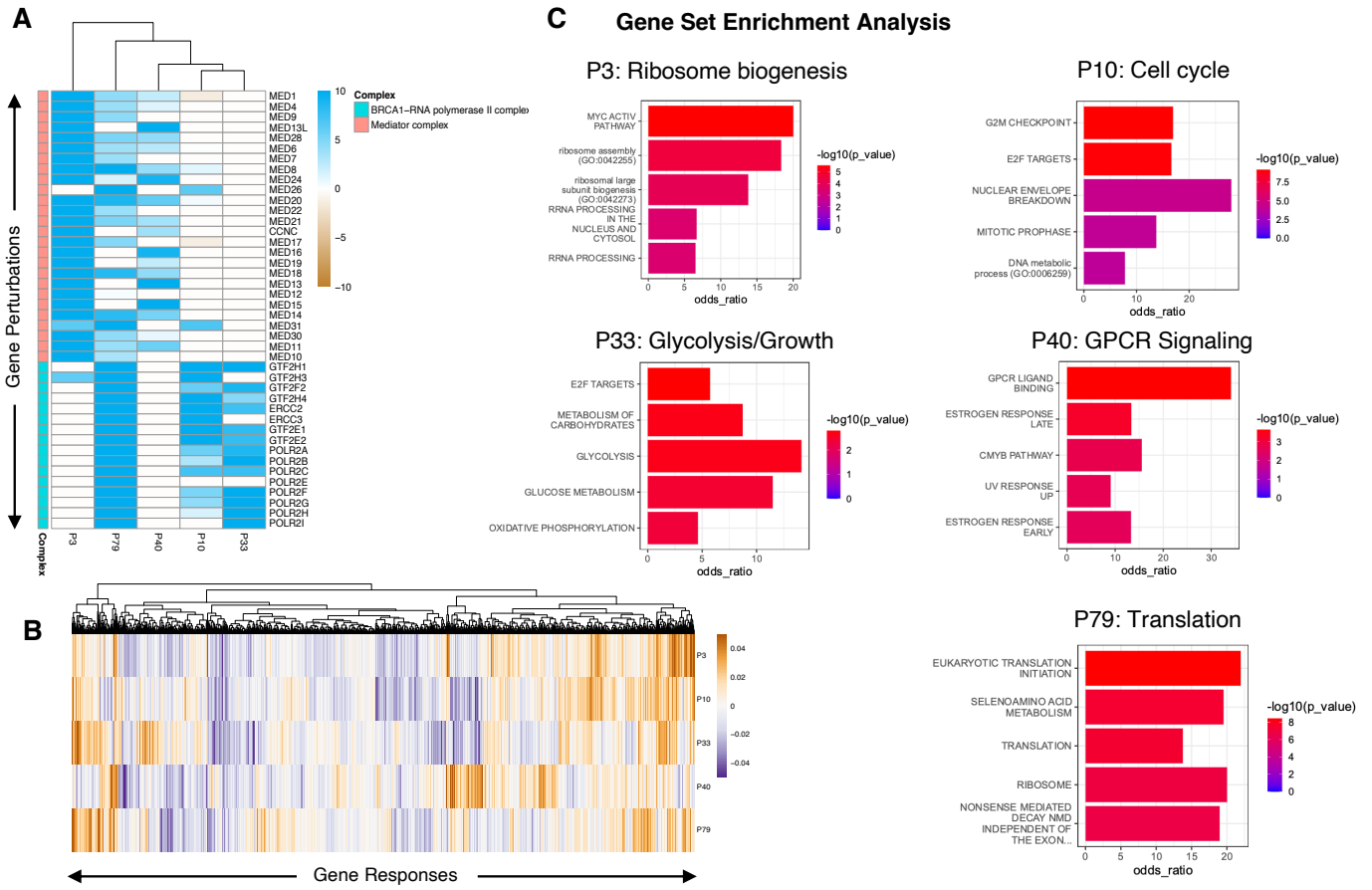

Fig. 4. A) Loadings for gene perturbation in the RNA polymerase II and mediator complexes. Only programs with significant loadings for these perturbations are shown. B) Transcriptional program dictionary corresponding to the loading matrix in A. C) Gene set enrichment analysis results for transcriptional programs in B.

E. Pleiotropic effects of Mediator complex disruptions

In the loading heatmap program 79 jumps out as it maps to two distinct groups of genetic perturbations, those in the RNA polymerase II complex, which is responsible for transcribing genes, and those in the Mediator complex, which regulates transcription by facilitating interactions between transcription factors and RNA polymerase II complex. To better understand this transcriptional program, I subset the DGRDL factorization to perturbations in these two complexes and programs with significant loading values for these perturbations.

There were five distinct transcriptional programs modulated by perturbations in the RNA polymerase II and Mediator complexes (Fig. 4A). Program 3 was activated by nearly every Mediator perturbation while the activation of program 40 was variable across the complex. In the RNA polymerase II complex, most perturbations triggered both program 10 and program 33. Program 79 was highly loaded for all RNA polymerase II and many Mediator perturbations. The transcriptional program dictionary revealed that these programs correspond to the up and down regulation of distinct sets of genes (Fig. 4B).

To understand what biological functions are regulated by these transcriptional programs I performed gene set enrichment analysis on the top differentially expressed genes in each program. This revealed that programs 10, 33, and 40 are related to the downregulation of cell growth and proliferation, program 3 is related to the downregulation of ribosome biogenesis, and program 79 is related to the downregulation of translation (Fig 4C). These are all core functions of the cell which makes sense given that transcription is required for the cell to make proteins and survive.

Combing annotation of the transcriptional programs with the loading matrix reveals how the cell responds to the inhibition of transcription and the multiple functions of the Mediator complex. Disruption of either RNA polymerase II or the Mediator complex inhibits translation (program 79) since mRNAs are no longer being synthesized. Perturbations to both complexes also inhibit cell division and growth, but it appears that this occurs in distinct ways for the two complexes. However, the Mediator complex is also required for transcription of ribosomal rRNA which is transcribed by a different polymerase (RNA polymerase I) and does not need translation, which is why program 3 (disruption of ribosome biogenesis) is specific to Mediator complex perturbations [26].

V. Discussion

The application of dual-graph-regularized dictionary learning (DGRDL) to Perturb-seq data is a novel use of sparse dictionary learning to characterize gene function. By accommodating the concept of genetic pleiotropy, this method is able to discover a wider array of transcriptional programs than can be discovered with clustering methods.

The DGRDL factorization of the Perturb-seq transcriptional response matrix, as illustrated in Fig. 3, reveals an intimate relationship between protein complexes and transcriptional programs. This makes sense because knocking out any essential gene in a protein complex should have a similar effect, disruption of the complex. How well these transcriptional programs align with functions in the cell and how interpretable they are remains to be seen. While I have shown that some transcriptional programs correspond to distinct transcription factors, many do not. Careful annotation of the transcriptional programs using gene set enrichment and other tools is required to understand what aspect of cell biology they capture and what aspect they do not. For example, a significant amount of regulation in the cell occurs at the post-translational level.

My results illustrate genetic pleiotropy, as I found that individual protein complexes could modulate multiple transcriptional programs and, conversely, single transcriptional programs could be influenced by different protein complexes. Of course, this outcome was likely given the way I performed dictionary learning. An interesting next step would be to learn the amount of pleiotropy as a per perturbation parameter with some type of hurdle model defining if there is enough evidence for additional functions to be assigned to a particular gene.

My analysis of the Mediator and RNA polymerase II complexes provides a nice an example of pleiotropy. Disruptions in these complexes activated a variety of transcriptional programs, including down regulation of growth, translation, and ribosome biogenesis. The specificity of program 3 (down-regulation of ribosome biogenesis) to Mediator complex perturbations demonstrated how this complex is required for both mRNA and rRNA transcription while the RNA polymerase II complex is not required for the transcription of some components of the ribosome [26]. It would be interesting to see what other perturbations activate program 3 as they are also likely to disrupt ribosome biogenesis in some way.

While my initial results are promising, there are several limitations I should note. First, my analysis is directly dependent on the quality of the Perturb-seq dataset. Any potential biases or errors in this initial data could affect my results. In particular, Perturb-seq differential expression estimates for lowly expressed genes tend to be noisy. I have tried to avoid this by subsetting the Perturb-seq transcriptional response matrix, but the attribution of low expression genes to expression programs is still limited by the noise. Second, while my model can identify transcriptional programs, it cannot directly infer causal relationships, and the learned transcriptional programs may not represent real axes of variation cell-state space. Third, while Perturb-seq does enable functional annotation of some genes that do not affect cell fitness, there are still many genes that do not elicit a strong transcriptional response when they are knocked out and have thus been filtered out [10]. Different methods will be required to discover the function of these genes.

This project took inspiration from the CRISPR-Cas9 fitness literature, particularly the Webster method [6]. I owe Pan et. al. much of the credit for developing this sparse dictionary learning framework. Here I have shown that this method can be applied to an analogous problem, learning transcriptional programs in Perturb-seq data. It will be interesting to see what other biological problems with similar structure sparse dictionary learning can be applied to. For example, it could be used to model morphological states in cell imaging studies of groups of proteins in proteomic studies. I am particularly excited about the potential of methods such as this to begin to learn joint axes of variation between different modalities, such as jointly learning gene function from Perturb-seq and fitness datasets.

Code

https://github.com/colganwi/perturb-seq-webster

References

[1] J. A. Doudna and E. Charpentier, “The new frontier of genome engineering with CRISPR-Cas9,” *Science (1979)*, vol. 346, no. 6213, Nov. 2014, doi: 10.1126/SCIENCE.1258096.

[2] C. Bock *et al.*, “High-content CRISPR screening,” *Nature Reviews Methods Primers 2022 2:1*, vol. 2, no. 1, pp. 1–23, Feb. 2022, doi: 10.1038/s43586-021-00093-4.

[3] A. Tsherniak *et al.*, “Defining a Cancer Dependency Map,” *Cell*, vol. 170, no. 3, p. 564, Jul. 2017, doi: 10.1016/J.CELL.2017.06.010.

[4] J. Pan *et al.*, “Interrogation of Mammalian Protein Complex Structure, Function, and Membership Using Genome-Scale Fitness Screens,” *Cell Syst*, vol. 6, no. 5, pp. 555-568.e7, May 2018, doi: 10.1016/j.cels.2018.04.011.

[5] C. J. Lord, N. Quinn, and C. J. Ryan, “Integrative analysis of large-scale loss-of-function screens identifies robust cancer-associated genetic interactions,” *Elife*, vol. 9, pp. 1–37, May 2020, doi: 10.7554/ELIFE.58925.

[6] J. Pan *et al.*, “Sparse dictionary learning recovers pleiotropy from human cell fitness screens,” *Cell Syst*, vol. 13, no. 4, pp. 286-303.e10, Apr. 2022, doi: 10.1016/J.CELS.2021.12.005.

[7] T. Hart *et al.*, “High-Resolution CRISPR Screens Reveal Fitness Genes and Genotype-Specific Cancer Liabilities,” *Cell*, vol. 163, no. 6, pp. 1515–1526, Dec. 2015, doi: 10.1016/j.cell.2015.11.015.

[8] D. Feldman *et al.*, “Pooled genetic perturbation screens with image-based phenotypes,” *Nature Protocols 2022 17:2*, vol. 17, no. 2, pp. 476–512, Jan. 2022, doi: 10.1038/s41596-021-00653-8.

[9] A. Dixit *et al.*, “Perturb-Seq: Dissecting Molecular Circuits with Scalable Single-Cell RNA Profiling of Pooled Genetic Screens,” *Cell*, vol. 167, no. 7, pp. 1853-1866.e17, Dec. 2016, doi: 10.1016/J.CELL.2016.11.038.

[10] J. M. Replogle *et al.*, “Mapping information-rich genotype-phenotype landscapes with genome-scale Perturb-seq,” *Cell*, vol. 185, no. 14, pp. 2559-2575.e28, Jul. 2022, doi: 10.1016/J.CELL.2022.05.013.

[11] F. W. Stearns, “One hundred years of pleiotropy: a retrospective,” *Genetics*, vol. 186, no. 3, pp. 767–773, Nov. 2010, doi: 10.1534/GENETICS.110.122549.

[12] M. Elad, “Sparse and redundant representations: From theory to applications in signal and image processing,” *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, pp. 1–376, 2010, doi: 10.1007/978-1-4419-7011-4/COVER.

[13] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, “An overview of JPEG-2000,” *Proceedings DCC 2000. Data Compression Conference*, pp. 523–541, Nov. 2000, doi: 10.1109/DCC.2000.838192.

[14] J. L. Starck, E. J. Candès, and D. L. Donoho, “The curvelet transform for image denoising,” *IEEE Transactions on Image Processing*, vol. 11, no. 6, pp. 670–684, Jun. 2002, doi: 10.1109/TIP.2002.1014998.

[15] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006, doi: 10.1109/TIP.2006.881969.

[16] B. K. Natarajan, “Sparse Approximate Solutions to Linear Systems,” *https://doi.org/10.1137/S0097539792240406*, vol. 24, no. 2, pp. 227–234, Jul. 2006, doi: 10.1137/S0097539792240406.

[17] S. G. Mallat and Z. Zhang, “Matching Pursuits With Time-Frequency Dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993, doi: 10.1109/78.258082.

[18] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, “Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition,” *Conference Record of the Asilomar Conference on Signals, Systems & Computers*, vol. 1, pp. 40–44, 1993, doi: 10.1109/ACSSC.1993.342465.

[19] K. Engan, S. O. Aase, and J. H. Husoy, “Method of Optimal Directions for frame design,” *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 5, pp. 2443–2446, 1999, doi: 10.1109/ICASSP.1999.760624.

[20] M. Aharon and M. Elad, “Sparse and Redundant Modeling of Image Content Using an Image-Signature-Dictionary,” *https://doi.org/10.1137/07070156X*, vol. 1, no. 3, pp. 228–247, Jul. 2008, doi: 10.1137/07070156X.

[21] H. Lee, A. B. Rajat, R. Andrew, and Y. Ng, “Efficient sparse coding algorithms,” *Adv Neural Inf Process Syst*, vol. 19, 2006.

[22] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006, doi: 10.1109/TSP.2006.881199.

[23] Y. Yankelevsky and M. Elad, “Dual Graph Regularized Dictionary Learning,” *IEEE Trans Signal Inf Process Netw*, vol. 2, no. 4, pp. 611–624, Dec. 2016, doi: 10.1109/TSIPN.2016.2605763.

[24] A. Lachmann, H. Xu, J. Krishnan, S. I. Berger, A. R. Mazloom, and A. Ma’ayan, “ChEA: transcription factor regulation inferred from integrating genome-wide ChIP-X experiments,” *Bioinformatics*, vol. 26, no. 19, p. 2438, Oct. 2010, doi: 10.1093/BIOINFORMATICS/BTQ466.

[25] G. Tsitsiridis *et al.*, “CORUM: the comprehensive resource of mammalian protein complexes–2022,” *Nucleic Acids Res*, vol. 51, no. D1, pp. D539–D545, Jan. 2023, doi: 10.1093/NAR/GKAC1015.

[26] G. M. Cooper, “Eukaryotic RNA Polymerases and General Transcription Factors,” 2000, Accessed: May 17, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK9935/