Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Omanović Amra

Projekcija visokodimenzionalnih podatkov ob upoštevanju domenskih omejitev

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Polona Oblak
Somentor: prof. dr. Blaž Zupan

Ljubljana, 2018

University of Ljubljana
Faculty of Computer and Information Science

Omanović Amra

Knowledge-constrained projection of high-dimensional data

master's thesis

the 2nd cycle master's study programme
computer and information science

Supervisor: Assoc. Prof. Dr. Polona Oblak
Co-supervisor: Prof. Dr. Blaž Zupan

Ljubljana, 2018

Acknowledgments

First of all, I would like to thank all of the staff at the Faculty of Computer and Information Science, who helped me as a foreign student to engage in a new way of studying, all of which was within a new environment.

I would also like to thank my supervisor Assoc. Prof. Polona Oblak for her patience, guidance, selfless support and all the knowledge that I gained I should thank her.

I am also grateful to my co-supervisor Prof. Blaž Zupan for his time in introducing me to a new world of science, and his committed support in helping to solve all manner of problems.

A special thanks must also go to my parents and a brother, who have always supported me during my studies.

I would also like to extend my gratitude to my friends and relatives, especially those in Slovenia who accepted me as their child, who looked forward to my success and always motivated me on my education journey.

Omanović Amra, 2018

"Education breeds confidence. Confidence breeds hope. Hope breeds peace."

— Confucius

Contents

List of used acronmys

SVD	Singular Value Decomposition
KKT	Karush–Kuhn–Tucker
STRING	Search Tool for the Retrieval of Interacting Genes/Proteins
CD	Cluster of Differentiation
PCA	Principal Component Analysis
HLDA	Human Leukocyte Differentiation Antigens
AML	Acute Myeloid Leukemia
KS	Kolmogorov-Smirnov

Povzetek

Naslov: Projekcija visokodimenzionalnih podatkov ob upoštevanju domenskih omejitev

Projekcija visokodimenzionalnih podatkov se običajno pripravi z zmanjšanjem dimenzionalnosti, ki se predstavi v latentnem prostoru, kar omogoča smiselno vizualizacijo. Pripravili smo sintetične podatke, ki odražajo gensko izražanje v pravih podatkovnih zbirkah. Metode smo kasneje testirali na pripravljenih sintetičnih in pravih podatkih. V tem delu smo obravnavali naloge z izvajanjem regularizirane SVD metode, z uporabo L_0-norme in L_1-norme. Modelu je bila dodana informacija z regularizacijo dveh dodatnih matrik sosednosti. Pokazali smo, da so te metode dale boljše rezultate kot standardni SVD.

Ključne besede

projekcija podatkov, latentni prostori, regularizacija, podatkovna veda, genomika posameznih celic

Abstract

Title: Knowledge-constrained projection of high-dimensional data

Projection of high-dimensional data is usually done by reducing dimensionality of the data and transforming the data to the latent space. We created synthetic data to simulate real gene-expression datasets and we tested methods on both synthetic and real data. With this work we address the visualization of our data through implementation of regularized singular value decomposition (SVD) for biclustering using L_0-norm and L_1-norm. Additional knowledge is introduced to the model through regularization with the two prior adjacency matrices. We show that L_0-norm SVD and L_1-norm SVD give better results than standard SVD.

Keywords

data projection, latent spaces, regularization, data science, single-cell genomics

Razširjeni povzetek

Nedavni napredek na področju biotehnologije je povzročil ustvarjanje orodij molekularne biologije, kateri nam lahko pomagajo opazovati modelne organizme in ljudi, ki omogočajo zbiranje velikih količin podatkov. Primer takšne nedavne tehnologije je enocelično RNA zaporedje [3, 20], kar je pripomoglo ustvariti podatke o celicah in ekspresijah genov. Takšni podatki postajajo veliki in lahko vključujejo tisoče celic in več deset tisoč genov. Računalniški pristopi so potrebni za zmanjšanje dimenzionalnosti podatkov in njihovo predstavitev v latentnem prostoru, ki bi lahko vodili do vizualizacije podatkov.
Projekcija podatkov in izbira lastnosti na takšnih področjih morata obravnavati več nalog, ki vključujejo obvladovanje podatkovnih podatkov, vključitev dodatnih informacij (npr. genske ontologije [1]) in izkoriščanje redkosti vhodnih podatkov. V tem delu smo obravnavali naloge z izvajanjem regularizirane SVD metode-razcepa s singularnimi vrednostmi. Modelu bomo dodali informacije z regularizacijo.

I Kratek pregled sorodnih del

V [11] so avtorji predlagali L_0-normo, graf-regulariziran redki SVD za gručenje visokodimenzionalnih podatkov. Delo se opira na stare podatke, ki jih razlaga graf. V regularizirani SVD so trije glavni vidiki: redki SVD, graf-regulariziran SVD in povezava med SVD metodo in PCA metodo. Gručenje skozi SVD [9] je orodje za analizo za prepoznavanje interpretiranih pove-

zav vrstic in stolpcev v matrikah visokodimenzionalnih podatkov. V [16] so predlagali vključitev izbire stabilnosti za izboljšanje redkega SVD pristopa. Njihov S4VD algoritem najde stabilne gruče in ocenjuje verjetnost selekcije genov in vzorcev, k pripadajočim gručam. Kaznovani razcep matrike (PMD) [18] ima za posledico regularizirano različico SVD. Pri tej metodi avtorji uporabijo tudi kazni v višini L_1 in metoda je bila prikazana na javno dostopnem podatkovnem nizu podatkov o ekspresiji genov. V [15] so avtorji predlagali novo metodo PCA, in sicer redko PCA prek regulariziranega SVD (sPCA-rSVD). Ta metoda zagotavlja enotno obravnavo obeh klasičnih večvrstnih podatkov in visokodimenzionalnih podatkov z nizkim vzorcem.

II Predlagana metoda

V tem delu smo se odločili predstaviti metodi L_0-norm SVD in L_1-norm SVD. Obe metodi izkoriščata poznavanje matrik sosednosti za vrstice in stolpce in sicer ena od opisanih metod uporablja kot parameter regularizacije normo L_1, druga pa normo L_0. Oba algoritma sta posplošitvi metode SVD, katera je dejansko matrična faktorizacija s katero matriko faktoriramo nadalje v tri nove matrike. Analizirali smo prvo matriko, ki je matrika levih singularnih vektorjev. Pri vizualizaciji obeh metod smo uporabili podatke iz dveh stolpcev, kar pomeni da smo izračunali dva singularna vektorja.

III Eksperimentalna evaluacija

Naša eksperimentalna evaluacija je sestavljena iz dveh delov: vrednotenje rezultatov sintetičnih podatkov in vrednotenje rezultatov resničnih genskih izrazov. Sintetične podatke smo sintetizirali na naslednji način:

- konstruiranje matrike X: domnevali smo, da imamo pet različnih vrst celic: T celico, B celico, dendritično celico, NK celico in granulocitom. Izbrali smo 200 celic vsake vrste, tako da na koncu naša matrika X vsebuje 1000 vrstic. Za vsak tip celice označujemo, kateri geni so ustrezni

markerski geni. Nato smo za naše 78 predhodno ujemajočih se genov (stolpcev) postavili vrednost večjo od nič, če je gen markerski za to vrsto celice. Svojo matrico X smo sešteli s šumno matriko. Na koncu smo dodali 1000 naključnih genov (stolpcev), da bi bolje simulirali resnične podatkovne zbirke genskih izrazov, kjer večina genov ni markerskih.

- konstruiranje matrike A_2: za matriko sosednosti za stolpce smo vzeli vrednosti iz baze podatkov STRING za ujemajočih 78 genov, za drugih 1000 naključnih genov smo določili verjetnost 0.3, da so povezani (sosednji).

Da bi prikazali, kako delujejo standardni SVD, L_1-norm SVD in L_0-norm SVD in kako se spremeni vizualizacija glede na različne vrednosti parametrov, smo se odločili določiti nekaj parametrov in spremeniti le en parameter. Zanima nas, kako dobri so SVD, L_1-norm SVD in L_0-norm SVD pri odkrivanju različnih vrst celic, ko ne vemo, kakšen je tip celice. Zato zdaj izvajamo le zdrave celice. V tem delu naše analize ne moremo uporabiti ocene **silhuete** kot merila učinkovitosti, saj ne poznamo tipov celic. Pregledali bomo grafe porazdelitvenih funkcij srednjih vrednosti markerskih genov in videli, kakšna je razlika med njimi. Za oceno razlike med porazdelitvami smo uporabili test Kolmogorov-Smirnov (KS-test) za dva vzorca. Ta preizkus je neparametrični preizkus, ki primerja zbirne porazdelitve dveh podatkovnih nizov. Ugotoviti poizkuša, ali se dve podatkovni skupini bistveno razlikujeta. Prednost KS-testa je, da ne daje nobene predpostavke o porazdelitvi podatkov. Ničelna hipoteza je, da sta bili obe skupini vzorčeni iz populacij z enakimi porazdelitvami. Preizkuša vsakršno kršitev te ničelne hipoteze - različnih medijev, različnih odstopanj ali različnih porazdelitev. Če je vrednost p majhna, lahko sklepamo, da sta bili obe skupini vzorčeni iz populacij z različnimi porazdelitvami. Populacije se lahko razlikujejo glede na mediano, variabilnost ali obliko porazdelitve. To lahko vidimo tako za algoritme kot za vse vrste celic, p-vrednost je manjša od 0.05. Ker test Kolmogorov-Smirnov ne primerja nobenega določenega parametra, ne poroča o nobenem intervalu zaupanja. Interval zaupanja je vrsta intervalne ocene, ki bi lahko vsebovala pravo vre-

dnost nepoznanega populacijskega parametra. Najpogosteje se uporablja interval zaupanja 95%, tudi drugi pa se lahko uporabijo. Izračunali smo interval zaupanja 95%, s katerim smo ugotovili, da ima L_1-norm SVD največjo povprečno vrednost markerskih genov.
Obenem smo na resničnih podatkih analizirali, kako metode med seboj razlikujejo zdrave in AML celice. Pri tej analizi, merjenja učinkovitosti naših metod smo se odločili za uporabo silhuete. Silhueta se nanaša na metodo interpretacije in potrjevanja skladnosti v gručah podatkov. Izkazalo se je, da ima metoda L_1-norm SVD najboljše rezultate.

IV Sklep

Pregledali smo dva algoritma reguliranega SVD: L_0-norm SVD in L_1-norm SVD [11]. Implementacija je bila izvedena v programskem jeziku Python, koda pa je javno dostopna na Githubovem repozitoriju. Testirali smo metode sintetičnih podatkov in resničnih podatkov o izraženosti genov iz aplikacije scOrange: "mononuklearne celice kostnega mozga z AML" na dva načina:

1. metode preskušanja elementov na celotnem naboru podatkov in ocenjevanje uspešnosti: kako različne metode razlikujejo med zdravimi in AML celicami?

2. testiranje samo na zdravih celicah in ocenjevanje učinkovitosti z uporabo markerskih genov: kako različne metode delijo povprečne ocene markerjev?

Glede na vrednost parametrov smo pri obeh metodah dobili različne rezultate, torej različne vizualizacije. Da pa bi dosegli najboljši rezultat, smo spremenili parametre in ugotovili, da pri preveliki regularizaciji parametrov pride do slabše vizualizacije. V obeh testih so bili najboljši rezultati pri L_1-norm SVD.

Chapter 1

Introduction

Recent advances in biotechnology have resulted in molecular biology tools that can help us observe model organisms and humans, through analysis of the collection of large volumes of gathered experimental data. An example of a such recent technology is the single-cell RNA sequencing [3, 20], which can gather the data on gene expressions in a collection of cells, one cell at a time. Such data can include thousands of cells and can record expression of full compendium of genes. Mammalian genomes where single-cell RNA has been recently applied includes typically about 20 000 genes. Computational approaches are required to reduce the dimensionality of such data and present it in a latent space that could lead to data visualisation.

In such domains, data projection and feature selection, need to address several problems. These include coping with data volume, the incorporation of additional knowledge (e.g. gene ontologies [1]), and capitalizing on the sparseness of the input data. Within this work, we will address the problems through the implementation of regularized singular value decomposition (SVD) for biclustering. Additional knowledge of adjacency matrices will be introduced to the model through regularization.

1.1 Related work

By computing singular value decomposition (SVD) we want do discover "concepts". By a "concept" we mean a new knowledge which shows relationship between rows and columns and as a result we have content-aggregated data that we are looking for. We observe certain data which is in the space that we can observe, and we want to map it to a latent space where similar data points are closer together. We want that latent space to capture the structure of our data.

Min *et al.* [11] proposed a L_0-norm sparse graph-regularized SVD for biclustering high-dimensional data. The paper relies on old data explained by the graph. In regularized SVD there are three main aspects: sparse SVD, graph-regularized SVD and the relationship between SVD and PCA.

Biclustering via sparse singular value decomposition is an analysis tool for identifying interpretable row-column associations within high-dimensional data matrices. Lee *et al.* [9] proposed sparse SVD which forces the left and the right singular vectors to be sparse. They tested algorithms on a lung cancer microarray dataset, on a food nutrition dataset and on a simulated datasets.

Sill *et al.* [16] proposed to incorporate stability selection to improve sparse SVD approach. Their S4VD algorithm finds stable biclusters and estimates the selection probabilities of genes, and the samples which belong to the biclusters. In a simulation study, their S4VD algorithm outperformed the sparse SVD algorithm and two other SVD-related biclustering methods in recovering artificial biclusters and in being robust to noisy data.

Penalized matrix decomposition (PMD) results in a regularized version of the SVD [18]. The data matrix is approximated and singular vectors minimize the squared Frobenius norm, subject to penalties on those vectors. In this method Witten *et al.* used L_1 penalties and the method was demonstrated on a publicly available gene expression data set. They showed that when this method is applied to a cross-products matrix, it results in a method for penalized canonical correlation analysis and this is tested on a simulated

and genomic data.

Principal component analysis (PCA) is a widely used tool for data analysis and dimension reduction, but since the principal components can be difficult to interpret, Shena *et al.* [15] proposed sparse PCA via regularized SVD (sPCA-rSVD). They used the relation of PCA with singular value decomposition (SVD) of the data matrix. This method provides a uniform treatment of both classical multi-variant data and high-dimension-low-sample-size data.

Osher *et al.* [13] introduced l_1 optimization for sparse vectors, L_1 optimization for finding functions with compact support, and computing sparse solutions from measurements that are corrupted by unknown noise, while Lu *et al.* [21] presented how l_0-norm minimization problems can be reformulated to an equivalent rank minimization problem and then by applying the penalty decomposition, we solve the latter problem. Further use of singular value decomposition in transforming genome-wide expression data is described by Alter *et al.* [12]. They showed that SVD is a useful mathematical framework for processing and modelling genome-wide expression data, in which for the mathematical variables and operations we may assign biological meaning.

The penalized singular value decomposition, for a (noisy) data matrix, when the left singular vector has a sparse structure and the right singular vector is a discretized function is presented by Hong *et al.* [6]. It is shown, that the value of only one parameter has to be chosen. They tested proposed approach on the artificial and real dataset. More detailed, a sparse SVD for high-dimensional data is explained by Yang *et al.* [19]. They proposed a new approach for approximating a large, noisy data matrix and they compared the method with two other existing methods, and showed that their algorithm is computationally faster.

From these results we have learned how the general form of penalized matrix decomposition looks like and how different penalties can be used. We also learned applications of SVD methods on gene-expression data, so now we proceed with two most common penalties: L_0-norm and L_1-norm and how they transform our synthetic and real data.

1.2 Methodology and contributions

We reviewed SVD by paying special attention to properties of the input data. In particular, we consider the sparseness of the matrix and relatedness of cells and relatedness of genes. We analysed the methods to optimize functions with additional constraints. Some of the additional constraints were Lagrangian multipliers and KKT conditions. Our approach benefited from data sparseness by modifying numerical approaches for eigenvector computation. Python programming language was used and the resulting method was tested on the visualisation of recently published data sets from single-cell genomics.

The first step was modification of the SVD method [11] by implementing and adapting the algorithm to our input data. We constructed data visualization which relied on the first two components of SVD. The project resulted in a Python code that is published on GitHub [1].

The rest of the thesis is structured as follows: in Subsection 2.4 and Subsection 2.5 we explain theoretical background of L_1-norm SVD and L_0-norm SVD through foundations of norms, mathematical optimization and singular value decomposition. In Section 3 we explain our gene-expression dataset and how we use knowledge of STRING database in creating adjacency matrices, which represent our additional knowledge necessary for regularization process. In Section 4 we present how methods work on synthetic and real gene-expression data, we discuss results, visualizations and compare them. Our results showed that we got better visualizations with L_0-norm SVD and L_1-norm SVD than with the standard SVD.

[1] `https://github.com/Ejmric/L0-and-L1-Norm-SVD`

Chapter 2

Theoretical background

2.1 L_1-norm and L_0-norm

A norm of a vector assigns strictly positive length or size to each vector in a vector space. The higher the norm the bigger the vector.

Definition 2.1. *A **vector norm** is a function from $\mathbb{R}^n$ to $\mathbb{R}$, with certain properties. If $x \in \mathbb{R}^n$, we represent its norm by $||x||$. The defining properties of the vector norm are:*

(i) $||x|| \geq 0$ *for all* $x \in \mathbb{R}^n$ *and also* $||x|| = 0$ *if and only if* $x = 0$,

(ii) $||\alpha x|| = |\alpha| \cdot ||x||$ *for all* $\alpha \in \mathbb{R}, x \in \mathbb{R}^n$ *(positive homogeneity),*

(iii) $||x + y|| \leq ||x|| + ||y||$ *for all* $x, y \in \mathbb{R}^n$ *(triangle inequality).*

For every real number $p \geq 1$, we define $||x||_p = \sqrt[p]{\sum_i |x_i|^p}, p \in \mathbb{R}^n$, which is a vector norm. In particular, we are interested in a special case, when $p = 1$, which we call L_1-norm.

Definition 2.2. *L_1-**norm** of $x \in \mathbb{R}^n$ represents the sum of absolute values of the components of the vector x:*

$$||x||_1 = \sum_{i=1}^{n} |x_i|$$

The L_1-norm is often called *Manhattan norm*. It is used in finding the sparsest solution, the solution that has fewest non-zero elements and this problem is regarded as L_1-optimisation [13]. We will also use the so-called L_0-norm:

Definition 2.3. *L_0-**norm** represents the total number of non-zero elements of a vector:*

$$||x||_0 = |\{i, x_i \neq 0\}|$$

L_0-norm is actually not a norm. If we look at the condition (ii) in Def. 2.1 we can see that the L_0-norm does not satisfy it. We can multiply x by any non-zero scalar and it does not change the L_0- norm. L_0-norm is actually a cardinality function, a measure of the "number of elements of the set".

There are many applications that use L_0-norm, also in finding the sparsest solution. Finding the lowest L_0-norm is called the optimisation problem of L_0-norm [21]. Compared to L_1-norm, L_0-norm can enforce a desirable level of sparsity.

2.2 Mathematical optimization

Mathematical optimization is a branch of applied mathematics which is useful in many different fields. The basic optimization problem consists of:

(i) the objective function $f(x)$. This is the output that we are trying to maximize or minimize,

(ii) variables $x_1, x_2...$ are the inputs,

(iii) constraints are equations or inequations that restrict the variables. They can be equality constraints $h_n(x)$ or inequality constraints $g_n(x)$. There are no strict inequalities and g_n defines domain of f, $f : D_f \to \mathbb{R}$ (then g_n restricts domain of f).

An optimization problem can be represented in the following way:

$$\begin{aligned} \text{Find max/min} \quad & f(x) \\ \text{under constraints} \quad & h_n(x) = 0 \\ & g_n(x) \geq 0 \quad \text{or} \quad g_n(x) \leq 0 \end{aligned} \tag{2.1}$$

In order to get a proper form of optimization problem, the general conversions that can be used are:

- interchange of $\leq$ with $\geq$ or interchange of $\geq$ with $\leq$. It is done by multiplying with -1.
- conversion to inequality: $x = y \Leftrightarrow x \leq y$ and $y \geq x$.
- interchange of $\leq$ with $=$: $x \leq y \Leftrightarrow y = x + t$ and $t \geq 0$, where t is "slack variable".

The subfield of mathematical optimization in which we are interested is convex optimization, where we want to find the minimum of the convex function f over convex sets. So, our basic problem is:

$$\begin{aligned} \text{Find min} \quad & f(x) \\ \text{with constraints} \quad & h_n(x) = 0 \\ & g_n(x) \leq 0 \end{aligned}$$

Definition 2.4. *A real-valued function $f : \mathbb{R}^n \to \mathbb{R}$ defined on an n-dimensional interval is called* ***convex*** *if the line segment between any two points on the graph of the function lies above or on the graph in an Euclidean space. Then for all $x_1, x_2 \in \mathbb{R}^n$ and $t \in [0, 1]$:*

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$

Function f is called ***concave*** *if and only if $-f$ is convex.*

We can also define convexity using derivatives:

Definition 2.5. *The **derivative of** $f(x)$ **with respect to** x is the function $f'(x)$ and is defined as:*

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Definition 2.6. *The **second derivative of** f is the derivative of the derivative of f:*

$$f'' = (f')'$$

Definition 2.7. *Function f is **convex** if and only if $f''(x) \geq 0$ for all x.*

These three definitions are for functions of one variable. For multivariable functions, we need to introduce partial and total derivatives.

Definition 2.8. *The **partial derivative** of a function $f : \mathbb{R}^n \to \mathbb{R}$ at the point $a = (a_1, a_2, ..., a_n) \in \mathbb{R}^n$ with respect to the i-th variable x-i is defined as:*

$$\frac{\partial}{\partial x_i} f(a) = \lim_{h \to 0} \frac{f(a_1, ..., a_{i-1}, a_i + h, a_i + 1, ..., a_n) - f(a_1, ..., a_i, ..., a_n)}{h}$$

Example The partial derivative of $f(x, y) = 3x^2y + 2y^2$ with respect to x is $6xy$. Its partial derivative with respect to y is $3x^2 + 4y$.

Definition 2.9. *A **total derivative** of a multivariable function is equal to the sum of the partial derivatives with respect to each variable times the derivative of that variable with respect to the independent variable.*

We will also define derivative over vector [5]:

Definition 2.10. *Let f represent a function, defined on a set S, of a vector $x = (x_1, ..., x_m)^T$ of m variables. Suppose that S contains at least some interior points, and let $c = (c_1, ..., c_m)$ represent an arbitrary one of those points. Further, let u_j represent the jth column of I_m. Consider the limit*

$$\lim_{t \to 0} \frac{f(c + tu_j) - f(c)}{t}.$$

When this limit exists, it is called the jth (first-order) partial derivative of f at c.

Definition 2.11. *The **gradient** is a vector of derivatives for each variable of a function and its symbol is usually* ∇.

One of the operations that preserve convexity is composition: if f and g are convex functions and g is non-decreasing over a univariate domain, then $h(x) = g(f(x))$ is convex.

Some of the examples of convex functions:

- The function $f(x) = x^2$ has $f''(x) = 2 > 0$, so f is a convex function.
- The exponential function $f(x) = e^x$ is convex.
- The function -log $det(X)$ on the domain of positive-definite matrices is convex.
- Every norm is a convex function, by the triangle inequality and positive homogeneity. For $A, B \in \mathbb{R}^n$ and $\alpha, \beta \in [0, 1], \alpha + \beta = 1$:

$$||\alpha A + \beta B|| \leq ||\alpha A|| + ||\beta B|| = |\alpha|||A|| + |\beta|||B|| \tag{2.2}$$

Now, let us define biconvex set and biconvex function [4].

Definition 2.12. *The set set $B \subseteq X \times Y$ is called a **biconvex set** on $X \times Y$ or biconvex for short, if B_x is convex for every $x \in X$ and B_y is convex for every $y \in Y$.*

Definition 2.13. *A function $f : B \to \mathbb{R}$ on a biconvex set $B \subseteq X \times Y$ is called a **biconvex function** on B if:*

1. *$f_x(\bullet) := f(x, \bullet) : B_x \to \mathbb{R}$ is a convex function on B_x for every fixed $x \in X$*
2. *$f_y(\bullet) := f(\bullet, y) : B_y \to \mathbb{R}$ is a convex function on B_y for every fixed $y \in Y$.*

Definition 2.14. ***Convex optimization** is to minimize a convex $f(x)$ on a convex set D.*

Definition 2.15. *A real-valued function f defined on domain $X \in \mathbb{R}^n$ has a* ***local minimum*** *point at $x^* \in X$ if there exists some $\epsilon > 0$ such that $f(x^*) \leq f(x)$ for all $x \in X$ within distance ϵ of x^*. The function has a* ***local maximum*** *point at $x^* \in X$ if there exists some $\epsilon > 0$ such that $f(x^*) \geq f(x)$ for all $x \in X$ within distance ϵ of x^*.*

Convex optimization has some nice properties: every local minimum is global, theoretically it is well explained, there are numerical efficient algorithms for it, and it comes from applications. Some of the applications are in norm approximation and regularization, semidefinite programming, linear matrix inequalites, convex relaxation, and parameter estimations [10].

The method of Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to equality constraints. When we want to maximize (minimize) a multivariable function $f(x_1, ..., x_n)$ subject to the constraint $g(x_1, ..., x_n) = c$, then the method of Lagrange multipliers works like this:

- introduce a new variable λ and define a new function L:

$$L(x_1, ..., x_n, \lambda) = f(x_1, ..., x_n) - \lambda(g(x_1, ..., x_n) - c)$$

  The function L is called the "Lagrangian" and the new variable λ a "Lagrange multiplier".

- set the gradient of L equal to the zero vector:

$$\nabla L(x_1, ..., x_n, \lambda) = 0, \tag{2.3}$$

  where 0 means the zero vector and in this step we find the critical points of L.

- consider each solution of (2.3) and plug each one into f. Whichever gives the greatest(or the smallest) value is the maximum (or the minimum) point.

The method of Lagrange multipliers is generalized by the Karush–Kuhn–Tucker (KKT) conditions [2]. We defined our functions f, g, h in (2.1). KKT conditions can also take into account inequality constraints of the form $h(x) \leq c$. These conditions are first-order necessary conditions for a solution in nonlinear programming to be optimal, provided that some regularity conditions are satisfied. The KKT conditions for the solution $x \in \mathbb{R}^n$ are:

- **stationarity**
  for minimization:
  $$\nabla_x f(x) + \sum_{i=1}^{m} \nabla_x \lambda_i h_i(x) + \sum_{i=1}^{n} \eta_i \nabla_x g_i(x) = 0$$
  for maximization:
  $$\nabla_x f(x) + \sum_{i=1}^{m} \nabla_x \lambda_i h_i(x) - \sum_{i=1}^{n} \eta_i \nabla_x g_i(x) = 0$$
- **equality constraints**
  $$\nabla_\lambda f(x) + \sum_{i=1}^{m} \nabla_\lambda \lambda_i h_i(x) + \sum_{i=1}^{n} \eta_i \nabla_\lambda g_i(x) = 0$$
- **inequality constraints a.k.a. complementary slackness condition**
  $$\eta_i g_i(x) = 0, \forall i = 1, 2, ..., n$$
  $$\eta_i \geq 0, \forall i = 1, 2, ..., n$$

The KKT conditions are necessary to find an optimum, but not necessarily sufficient.

2.3 Singular value decomposition (SVD)

We shall assume that the reader is familiar with orthogonality, matrix factorization, eigenvalues and eigenvectors.

Definition 2.16. *The **singular value decomposition** (SVD) of a $n \times p$ real matrix X is a factorization:*

$$X = UDV^T$$

where U is an $n \times r$ orthogonal matrix, D is an $r \times r$ diagonal matrix and V is an $p \times r$ orthogonal matrix.

The diagonal entries of D are known as the singular values of X sorted in decreasing order. The columns of U and the columns of V are called the left-singular vectors and right-singular vectors of X, respectively.

The non-sparse singular vectors can be difficult to interpret. Many studies [19] impose sparsity on singular vectors which lead to better capturing inherent structures and patterns of input data.

By computing SVD we want do discover "concepts". By "concept" we mean a new knowledge which shows relationship between rows and columns and as a result we have content-aggregated data that we are looking for. We observe data which is in the observable space, and we want to map it to a latent space where similar data points are closer together. So we want that latent space captures the structure of our data.

Matrix U is the "row-to-concept" similarity matrix, V is the "column-to-concept" similarity matrix and D is the 'strength' of each concept.

Now, we review sparse graph-regularized penalty. Given a simple graph G, the adjacency matrix A of graph G and diagonal matrix D whose diagonal elements are the degrees of vertices in G, Laplacian matrix L is defined as:

$$L = D - A.$$

We can impose sparsity on singular vectors in SVD with the following penalty:

$$P(v) = \lambda_1 l + \lambda_2 v^T L v$$

where $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are two regularization parameters, and l can be L_0-norm penalty or L_1-norm penalty. The procedure of different penalties and how they work in SVD can be found in [6, 18].

In the following subsections, we present two algorithms: L_1-norm sparse graph-regularized SVD and L_0-norm sparse graph-regularized SVD which are generalizations of SVD algorithm.

2.4 L_1-norm sparse graph-regularized SVD

Frobenius norm for the vectors is equal to the Euclidean norm, but for the matrix $X \in \mathbb{R}^{m \times n}$ it is defined as:

$$||A||_F = \sqrt{(A, A)} = \sqrt{\text{tr}(A^T A)} \tag{2.4}$$

where tr is the trace (sum of the elements on the main diagonal) of the matrix A.

We have the following optimization problem [18]:

$$\begin{aligned} \underset{u,v}{\text{minimize}} \quad & ||X - duv^T||_F^2 \\ \text{subject to} \quad & ||u||_2 \leq 1, ||u||_1 \leq s_1, |u|^T L_1 |u| \leq s_2, \\ & ||v||_2 \leq 1, ||v||_1 \leq c_1, |v|^T L_2 |v| \leq c_2 \end{aligned} \tag{2.5}$$

where X is a matrix of size $n \times p$, d is a positive singular value of X, u and v are column vectors of dimension $n \times 1$ and $p \times 1$ respectively, L_1 and L_2 are Laplacian matrices of adjacency matrices for rows and columns (A_1 and A_2), and s_1, s_2, c_1 and c_2 are given parameters.

We want to show that our objective function $||X - duv^T||_F^2$ in (2.5) is biconvex according to u and v. By the definition of biconvex function we need to show convexity when u is fixed over v and when v is fixed over u. Without loss of generality because of symmetry between u and v we will show convexity by fixing u over v.

Let $A = X - duv^T$ and let the objective function be $||A||_F^2$. The function $A \to ||A||_F$ is convex, which is a part of norm properties (2.2). Moreover, the square function x^2 is increasing and convex on $[0, \infty]$, so $A \to ||A||_F^2$ is the composition of a convex function with a convex increasing function,

which makes it convex as well. By this we showed that our objective function $||X - duv^T||_F^2$ is biconvex to u and v.

For the matrix properties we used [7].

Proposition 1. *Some of the properties of a trace function* tr *are:*

(a) $\text{tr}(A^T) = \text{tr}(A)$

(b) $\text{tr}(AB) = \text{tr}(BA)$

(c) $\text{tr}(ABC) = \text{tr}(BCA) = \text{tr}(CAB)$

(d) $\text{tr}(vu^T X) = \text{tr}(u^T Xv) = u^T Xv$

(e) $\text{tr}(vu^T uv^T) = \text{tr}(u^T u)(v^T v) = (u^T u)(v^T v) = ||u||^2||v||^2$

(f) $\text{tr}(A + B) = \text{tr}(A) + \text{tr}(B)$

where A, B, C are matrices and v, u are column vectors.

Norm $||\cdot||_F$ arises from $(A, B) = \text{tr}(A^T B)$, where $A, B \in \mathbb{R}^{m\times n}$. From Proposition 1. we also have that $(A, B) = \text{tr}(A^T B) = \text{tr}(B^T A) = \text{tr}(AB^T) = \text{tr}(BA^T)$.

Using property of biconvexity we will first fix u and optimize over v:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & ||X - duv^T||_F^2 \\ \text{subject to} \quad & ||v||_2 \leq 1, ||v||_1 \leq c_1, |v|^T L|v| \leq c_2 \end{aligned} \tag{2.6}$$

Using definition 2.4 of Frobenius norm we can write our objective function as:

$$\begin{aligned} ||X - duv^T||_F^2 &= \text{tr}((X - duv^T)^T(X - duv^T)) = \\ &= \text{tr}(X^T X - dvu^T X - dX^T uv^T + d^2 vu^T uv^T) \end{aligned} \tag{2.7}$$

where we used $(AB)^T = B^T A^T$.

Further use of properties from Proposition 1 and the fact that $\text{tr}(X^T uv^T) = \text{tr}((X^T uv^T)^T) = \text{tr}(vu^T X) = \text{tr}(u^T Xv))$ transforms the right-hand side of (2.7) in:

$$||X - duv^T||_F^2 = ||X||_F^2 - 2du^T Xv + d^2||u||^2||v||^2$$

Since u, v can be assumed to be nonzero, we now prove that without loss of generality we can assume $||u||_2 = 1$ and $||v||_2 = 1$.

$$duv^T = d \cdot ||u|| \cdot ||v|| \cdot \frac{u}{||u||} \cdot \frac{v^T}{||v||} = d'u'v'^T$$

for $d' = d \cdot ||u|| \cdot ||v||$, $u' = \frac{u}{||u||}$ and $v' = \frac{v^T}{||v||}$. By dividing the vector by its norm we got new vectors u' and v'. We can check that $||X' - d'u'v'^T||_F^2 = ||X - duv^T||_F^2$:

$$d' \cdot u'^T \cdot X \cdot v' = d \cdot ||u|| \cdot ||v|| \cdot \frac{u^T}{||u||} \cdot X \frac{v^T}{||v||} = du^T Xv$$

We will still denote vectors as u and v. Like this we gained:

$$||X - duv^T||_F^2 = ||X||_F^2 - 2du^T Xv + d^2$$

Minimizing this function is equivalent to minimizing $-u^T Xv$, because d is a positive value and $||X||_F^2$ is positive, so we do not have to take it into account while optimizing.

If we put that $z = X^T u$ which implies $z^T = u^T X$ then $-u^T Xv = -z^T v$ and since z, v are vectors then we can change the places to $-v^T z$. Now we have the following optimization problem:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & -v^T z \\ \text{subject to} \quad & ||v||_2 \leq 1, ||v||_1 \leq c_1, |v|^T L|v| \leq c_2 \end{aligned} \tag{2.8}$$

We want to remove the absolute operator in the condition $|v|^T L|v| \leq c_2$ since it is generally not a convex condition.

Theorem 2.1. *Suppose v^* is an optimal solution of (2.8), then $v_i^* z_i \geq 0$ for all i, where $1 \leq i \leq n$ and v_i, z_i are coordinates of vectors v, z respectively.*

Proof. Suppose that the Theorem 2.1 is false: it exists $i : v_i^* z_i < 0$.

We first construct a vector v' which satisfies: $v'_j = v_j^*$ for all $j \neq i$ and $v'_i = -v_i^*$. Obviously the number of non-zero elements of v' and v^* is the same, $||v'||_0 = ||v^*||_0$, and their Euclidean norms are equal $||v'||_2 = ||v^*||_2$.

Let $f_1 = -v^{T*}z$ and $f_2 = -v^{T'}z$. Then $f_1 - f_2 = (-v^* + v')^T z = -2v_i^* z_i > 0$. So $f_1 - f_2 > 0$ which means that $f_1 > f_2$, i.e. $-v^{T*}z > -v^{T'}z$.

We found a vector v' which corresponds to a smaller objective value than v^* and that leads to a contradiction, so the theorem is true. □

Based on the Theorem 2.1 we can change our minimization problem to:

$$\begin{aligned} &\underset{v}{\text{minimize}} && -v^T|z| \\ &\text{subject to} && ||v||_2 \leq 1, ||v||_1 \leq c_1, v^T L v \leq c_2, v_k \geq 0, \text{for all} \quad k \end{aligned}$$

where $|z| = (|z_1|, ..., |z_p|)^T$.

We will solve this optimization problem using Lagrangian form. First let us write all constraints in appropriate form and add Lagrangian multipliers:

- $||v||_2 \leq 1$, which is equal to $v^T v \leq 1$ and $v^T v - 1 \leq 0$. Let the corresponding Lagrangian multiplier be $\frac{1}{2}\eta \geq 0$ (which results in $\eta(v^T v - 1)$).
- $||v||_1 \leq c_1$, which is equal to $\sum_i^p |v_i| \leq c_1$ and $\sum_i^p v_i \leq c_1$. Then we have that $\sum_i^p v_i - c_1 \leq 0$. Let the corresponding Lagrangian multiplier be $\lambda \geq 0$ (which results in $\lambda(\sum_i^p v_i - c_1)$).
- $v^T L v \leq c_2$, which is equal to $v^T L v - c_2 \leq 0$ and let the corresponding Lagrangian multiplier be $\frac{1}{2}\sigma \geq 0$ (this results in $\sigma(v^T L v - c_2)$).
- $v_k \geq 0$ for all k, which is equal to $-v_k \leq 0$ and let the corresponding Lagrangian multiplier be $\tau \geq 0$ (it results in $-\tau_k v_k, \forall k$).

Now, we will formulate the Lagrangian form as:

$$\begin{aligned} M(v, \eta, \lambda, \sigma, \tau_1, ...\tau_p) = &- v^T|z| + \frac{1}{2}\eta(v^T v - 1) + \lambda(\sum_i^p v_i - c_1) + \\ &+ \frac{1}{2}\sigma(v^T L v - c_2) - (\sum_i^p \tau_i v_i) \end{aligned} \tag{2.9}$$

We added $\frac{1}{2}$ in front of some elements so that there would be no canceling after the derivation process.

Rules for the first order derivatives [8], where v is a vector, a is a scalar and L is a matrix:

$$\frac{\partial a}{\partial x} = 0 \tag{2.10}$$

$$\frac{\partial x^T a}{\partial x} = \frac{\partial a x^T}{\partial x} = a \tag{2.11}$$

$$\frac{\partial x^T x}{\partial x} = 2x \tag{2.12}$$

$$\frac{\partial x^T L x}{\partial x} = (L + L^T)x$$

$$\text{For symmetric } L\text{: } \frac{\partial x^T L x}{\partial x} = 2Lx \tag{2.13}$$

We can write sums in (2.9) in a different form before derivation:

$$\sum_i^p v_i = \begin{bmatrix} v_1 & v_2 & \cdots & v_p \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = v^T \cdot e$$

$$\sum_i^p \tau_i v_i = \begin{bmatrix} v_1 & v_2 & \cdots & v_p \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_p \end{bmatrix} = v^T \cdot \tau$$

Using the rules (2.10), (2.11), (2.12) and (2.13), the derivative of M over v is:

$$\frac{\partial M}{\partial v} = -|z| + \eta v + \lambda e + \sigma L v - \tau = 0 \tag{2.14}$$

We know that $L = D - A$, so we can replace L in (2.14) with $D - A$. The easiest way to learn vector v is to use a coordinate descent method, so the subgradient of v_k just by coordinates in (2.9) is:

$$\frac{\partial M}{\partial v_k} = -|z_k| + \eta v_k + \lambda + \sigma d_k v_k - \sigma A_k v - \tau_k = 0, \tag{2.15}$$

where d_k is the degree of node k and A_k is the kth row of the adjacency matrix A.

The complementary slackness KKT condition (2.2) gives:

- if $v_k > 0$ then $\tau_k = 0$
- if $v_k = 0$ then $\tau_k > 0$

So, if $v_k > 0$ then $\tau_k = 0$, and from (2.15) we have:

$$-|z_k| + \eta v_k + \lambda + \sigma d_k v_k - \sigma A_k v = 0$$

$$\eta v_k + \sigma d_k v_k = |z_k| - \lambda + \sigma A_k v$$

$$(\eta + \sigma d_k) v_k = |z_k| - \lambda + \sigma A_k v$$

$$v_k = \frac{|z_k| - \lambda + \sigma A_k v}{\eta + \sigma d_k} \tag{2.16}$$

If $v_k = 0$ then $\tau_k > 0$, and we can merge this case with (2.16) in:

$$v_k = \frac{\max(|z_k| + \sigma A_k v - \lambda, 0)}{\eta + \sigma d_k}, k = 1, 2, ...$$

Let $v'_k = \max(|z_k| + \sigma A_k v - \lambda, 0)$ and $v' = (v'_1, v'_2, ..., v'_p)^T$. To meet the normalizing condition we have $v' = \frac{v'}{||v'||_2}$. In the end using Theorem 2.1 the optimal solution of (2.6) is:

$$v = v' \bullet \text{sign}(z),$$

where "$\bullet$" is element-wise product.

In the same way, with a fixed v while optimizing u, we can get vector u. When we have u and v, our objective function becomes a quadratic function in d, so the minimum is only related to d, and we can control iteration by monitoring the change of d.

Algorithm 1 shows the L_1-norm sparse-graph regularized SVD algorithm [11].

An input to the Algorithm 1 is a data matrix, two adjacency matrices (for rows and columns) and four parameters: λ_u (regularization parameter for

Algorithm 1 L_1-norm SVD

Input: data matrix $X \in \mathbb{R}^{n \times p}$; prior networks $A^1 \in \mathbb{R}^{n \times n}$ and $A^2 \in \mathbb{R}^{p \times p}$; parameters $\lambda_u, \lambda_v, \sigma_u, \sigma_v$

Initialize $\mathbf{v}$ with $||v||_2 = 1$
repeat
  Let $z = Xv$, $A = A^1$ and $u = |u|$
  for $i = 1$ to n **do**
    $u_i = \max(|z_i| + \sigma_u A_i u - \lambda_u, 0)$
  end for
  $u = \frac{u}{||u||_2}$
  $u = u \bullet \text{sign}(z)$
  Let $z = X^T u$, $A = A^2$ and $v = |v|$
  for $k = 1$ to p **do**
    $v_k = \max(|z_k| + \sigma_v A_k v - \lambda_v, 0)$
  end for
  $v = \frac{v}{||v||_2}$
  $v = v \bullet \text{sign}(z)$
  $d = z^T v$
until d convergence
return u, v, d

the left singular vector), λ_v (regularization parameter for the right singular vector), σ_u (importance of the prior graph A_1) and σ_v (importance of the prior graph A_2). Lines 3-8 compute u singular vector and lines 9-14 compute v singular vector.

2.5 L_0-norm sparse graph-regularized SVD

Similar to 2.4 we have the following optimization problem:

$$\begin{aligned} \underset{u,v}{\text{minimize}} \quad & ||X - duv^T||_F^2 \\ \text{subject to} \quad & ||u||_2 \leq 1, ||u||_0 \leq k_u, |u|^T L_1 |u| \leq c_1, \\ & ||v||_2 \leq 1, ||v||_0 \leq k_v, |v|^T L_2 |v| \leq c_2 \end{aligned}$$

The difference between those two problems are that here we have the L_0-norm constraints instead of L_1-norm constraints.

We will first fix u, optimize over v and let $z = X^T u$ and similar to 2.4 we obtain:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & -v^T z \\ \text{subject to} \quad & ||v||_2 \leq 1, ||v||_0 \leq k_v, |v|^T L |v| \leq c_2 \end{aligned} \tag{2.17}$$

Similarly to Theorem 2.1 now we have:

Theorem 2.2. *Suppose v^* is an optimal solution of (2.17), then $v_i^* z_i \geq 0$ for all $i, 1 \leq i \leq n$.*

Based on the Theorem 2.2 we can change our minimization problem to:

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & -v^T |z| \\ \text{subject to} \quad & ||v||_2 \leq 1, ||v||_0 \leq k_v, v^T L v \leq c_2, v_k \geq 0, \forall k \end{aligned}$$

where $|z| = (|z_1|, ..., |z_p|)^T$.

We will solve this optimization problem using Lagrangian form:

$$M(v, \eta, \sigma, \tau_1, ...\tau_p) = -v^T |z| + \frac{1}{2}\eta(v^T v - 1) + \frac{1}{2}\sigma(v^T L v - c_2) - (\sum_i^p \tau_i v_i) \tag{2.18}$$

We will deal with the following constraint $||v||_0 \leq k_v$ later.

The optimal solution of (2.18) satisfies:

$$\frac{\partial M}{\partial v} = -|z| + \eta v + \sigma L v - \tau = 0$$

The subgradient of v_k in (2.18) is:

$$\frac{\partial M}{\partial v_k} = -|z_k| + \eta v_k + \sigma d_k v_k - \sigma A_k v - \tau_k = 0 \tag{2.19}$$

where d_k is the degree of node k, $L = D - A$ and A_k is the kth row of the adjacency matrix A.

The complementary slackness KKT condition gives:

- if $v_k > 0$ then $\tau_k = 0$
- if $v_k = 0$ then $\tau_k > 0$

So, if $v_k > 0$ then $\tau_k = 0$, and from (2.19) we have:

$$-|z_k| + \eta v_k + \sigma d_k v_k - \sigma A_k v = 0$$

$$\eta v_k + \sigma d_k v_k = |z_k| + \sigma A_k v$$

$$(\eta + \sigma d_k) v_k = |z_k| + \sigma A_k v$$

$$v_k = \frac{|z_k| + \sigma A_k v}{\eta + \sigma d_k} \tag{2.20}$$

If $v_k = 0$ then $\tau_k > 0$, and we can merge this case with (2.20) in:

$$v_k = \frac{\max(|z_k| + \sigma A_k v, 0)}{\eta + \sigma d_k}, k = 1, 2, ...$$

Let $v'_k = \max(|z_k| + \sigma A_k v - \lambda, 0)$ and $v' = (v'_1, v'_2, ..., v'_p)^T$.

Definition 2.17. *The **order statistics** of a random sample $X_1, ..., X_n$ are the sample values placed in ascending order. The kth smallest X value is normally called the kth order statistic, denoted by $|X|_{k_v}$.*

The first order statistic is the smallest sample value (i.e. the minimum), once the values have been placed in order. For example, in the sample 9, 2, 11, 5, 7, 4 the first order statistic is two. The second order statistic is the next smallest value, which is in the same sample, equal to four.

Definition 2.18. *The **indicator function** is a function defined on a set X that indicates membership of an element in a subset A of X, having the value one for all elements of A and the value zero for all elements of X not in A.*

To satisfy the condition $||v||_0 \leq k_v$, that the value of L_0-norm of vector v has to be less or equal to k_v, we force the $p - k_v$ elements of v with the smallest absolute values to be zeros:

$$v' = v \bullet I(|v| \geq |v|_{k_v})$$

where $I(\cdot)$ is the indicator function, "$\bullet$" denotes element-wise product and $|v|_{k_v}$ is the kth order statistic of $|v|$.

To meet the normalizing condition we have $v' = \frac{v'}{||v'||_2}$. In the end using Theorem 2.2 the optimal solution of (2.17) is:

$$v = v' \bullet \text{sign}(z)$$

In the same way with a fixed v while optimizing u, we can get vector u. When we have u and v, our objective function becomes a quadratic function about d, so the minimum is only related to d, and we can control iteration by monitoring the change of d.

Algorithm 2 shows the L_0-norm sparse-graph regularized SVD algorithm [11].

An input to the Algorithm 2 is a data matrix, two adjacency matrices (for rows and columns) and four parameters: k_u (regularization parameter for the left singular vector), k_v (regularization parameter for the right singular vector), σ_u (importance of the prior graph A_1) and σ_v (importance of the prior graph A_2). Lines 3-9 compute u singular vector and lines 10-16 compute v singular vector.

We calculated second coordinates of singular vectors u, v for L_1-norm SVD and L_0-norm SVD naming the methods again on the new data matrix $X = X - du^T v$.

Algorithm 2 L_0-norm SVD

Input: data matrix $X \in \mathbb{R}^{n \times p}$; prior networks $A^1 \in \mathbb{R}^{n \times n}$ and $A^2 \in \mathbb{R}^{p \times p}$; parameters $k_u, k_v, \sigma_u, \sigma_v$

Initialize **v** with $||v||_2 = 1$
repeat
  Let $z = Xv$, $A = A^1$ and $u = |u|$
  for $i = 1$ to n **do**
    $u'_i = |z_i| + \sigma_u A_i u$
  end for
  $u = u' \bullet I(|u'| \geq |u'|_{k_u})$
  $u^* = \frac{u}{||u||_2}$
  $u = u^* \bullet \text{sign}(z)$
  Let $z = X^T u$, $A = A^2$
  for $k = 1$ to p **do**
    $v'_k = |z_k| + \sigma_v A_k v$
  end for
  $v = v' \bullet I(|v'| \geq |v'|_{k_v})$
  $v^* = \frac{v}{||v||_2}$
  $v = v^* \bullet \text{sign}(z)$
  $d = z^T v$
until d convergence
return u, v, d

Chapter 3

Gene expression data

Data visualization is one of the most important steps in the analysis of high-dimensional data. Plots that reveal relationships between columns or between rows are more complicated due to the high dimensionality of data. If we are able to reduce down to two dimensions, we can then easily present the data in a scatter plot like visualizations.

3.1 Bone marrow mononuclear cells with AML dataset

"Bone marrow mononuclear cells" dataset represents gene expressions in bone marrow mononuclear cells from a patient with acute myeloid leukemia (AML) and two healthy donors that are used as controls. The data we have considered includes a sample of 1000 cells and 1000 genes with the highest dispersion. This is a sample dataset which includes cells from three separate experiments with datasets published on 10x Genomics single-cell data sets page: AML027 Pre-transplant BMMCs, Frozen BMMCs (Healthy Control 1), and Frozen BMMCs (Healthy Control 2) [22]. The Table 3.1 shows part of our dataset looks like.

Type	HBG1	HBG2	S100A9	S100A8	GNLY	LYZ
healthy	0	0	2.279	2.761	0	4.037
healthy	0	0	0	0	0	0
healthy	0	0	0	0	0	1.056
AML	2.397	1.276	0	0	0	1.276
AML	0	0	0	0	0	0
AML	0.943	2.985	0	0	0	0

Table 3.1: The sample of dataset "Bone marrow mononuclear cells"

3.2 STRING database

The STRING database [1] provides a critical assessment and integration of protein–protein interactions, including direct (physical) as well as indirect (functional) associations. We uploaded our 1000 genes as a ".txt" file to the STRING database [17]. The report from STRING is shown in Table 3.2.

	value
number of nodes	872
number of edges	5325
average node degree	12.2
avg.local clustering coefficient	0.412
expected number of edges	2553
PPI enrichment p-value	$1 \cdot 10^{-16}$

Table 3.2: Network analysis from STRING database

The Figure 3.1 shows us the network, but since it is a large network, it becomes hard to interpret it. The first step is to export it to the ".tsv" file.

[1] `http://string-db.org`

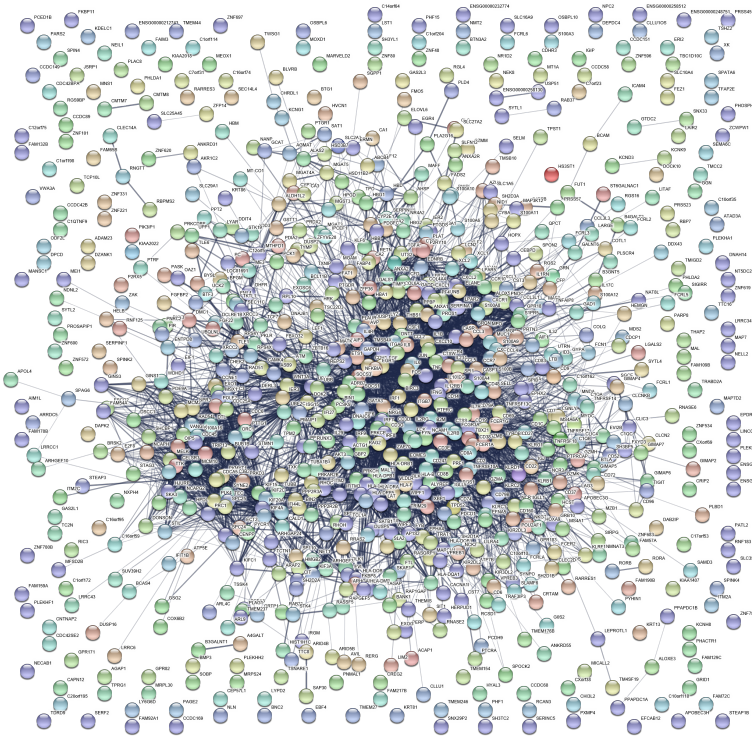


Figure 3.1: Gene network graph from STRING database. We uploaded all genes from "Bone marrow mononuclear cells with AML" dataset to the database and got the image above as the result.

The sample of this file can be viewed in Table 3.3.

node1	node2	co-expression
ORC1	MCM2	0.757
CDC45	MCM2	0.828
CHEK2	ATM	0.055
MCM2	CDC6	0.582
UBA52	RPS4X	0.956
EFNA5	EPHA4	0.381

Table 3.3: Sample of our gene network, where genes are represented as "node1" and "node2" and their co-expression score as obtained from STRING database

3.3 Human CD markers

Cluster of differentiation (CD) molecules are cell surface markers useful for the identification and characterization of leukocytes. The CD nomenclature was developed and is maintained through the Human Leukocyte Differentiation Antigens (HLDA) workshop started in 1982. New CD markers were established at the HLDA9 meeting held in Barcelona in 2010.

We downloaded the official ".pdf" file of Human CD Markers [2], Figure 3.2, but we could not automatically convert it to a ".csv" file, so we did it manually.

Human CD Markers handbook considers eleven types of cells, each type has a different number of "+" and "-" gene markers (Table 3.5). We will consider only markers that are overexpressed for a given cell type, that is, that are marked with "+". Genes can have more than one name or symbol, especially when the same gene is known by different scientific, informal, and historical names. That is why we decided to match gene names from "Bone

[2] `http://www.bdbiosciences.com/documents/cd_marker_handbook.pdf`

Notation	Meaning
+	Positive (co-expression is greater than 0)
-	Negative (co-expression is less than 0)
empty	Neutral (co-expression is 0)

Table 3.4: The meaning of different notations in Figure.3.2

marrow mononuclear cells with AML dataset" with genes from Human CD markers. From the initial 19 matched genes we came to matched 78 genes. We used scOrange software [3] and the procedure is shown in Figure 3.3.

[3] http://singlecell.biolab.si

Cell type	# of "+"	# of "-"
T Cell	223	81
B Cell	185	88
Dendritic Cell	125	67
NK Cell	131	131
Stem Cell/Precursor	111	36
Macrophage/Monocyte	219	73
Granulocyte	135	122
Platelet	50	124
Erythrocyte	29	149
Endothelial Cell	110	69
Epithelial Cell	94	50

Table 3.5: Cell types and number of associated overexpressed ("+") and underexpressed("-") marker genes in CD Marker handbook

CD	Alternative name	Ligands & Associated Molecules	T Cell	B Cell	Dendritic Cell	NK Cell	Stem Cell/Precursor	Macrophage/ Monocyte	Granulocyte	Platelet	Erythrocyte	Endothelial Cell	Epithelial Cell	Function
CD1a	R4, T6	B-2-Microglobulin, CD74	+	+	+	-		+	-	-	-			Antigen presenting protein
CD1b	R1, T6	B-2-Microglobulin, MHC II	+	+	+	-		+	-	-	-			Antigen presenting protein
CD1c	M241, R7, T6	B-2-Microglobulin	+	+	+	-		+	-	-	-			Antigen presenting protein
CD1d	R3G1	B-2-Microglobulin	+	+	+	-			-	-	-		+	Antigen presenting protein
CD1e	cR2	B-2-Microglobulin			+	-			-	-	-		+	Antigen presentation of glycolipids
CD2	E-rosette R, Erythrocyte R, T11, LFA-2	CD58, CD48, CD59, CD15, LFA-3	+	+		+			-	-	-			Cell adhesion between T cells and other cells
CD3	T3	TC3	+	-		-	-	-	-	-	-	-	-	A complex of subunits that meditates T cell signal transduction
CD3d	TiT3 complex	TC3	+											Part of the CD3/TCR complex that meditates T-cell receptor signal transduction
CD3e	T3e	TC3	+											Part of the CD3/TCR complex that meditates T-cell receptor signal transduction
CD3g	T3g	TC3	+											Part of the CD3/TCR complex that meditates T-cell receptor signal transduction

Figure 3.2: First few CD markers from human genome. The rows refer to genes and the columns represent cell types. The sign "+" means that the gene is a marker gene for that cell type.

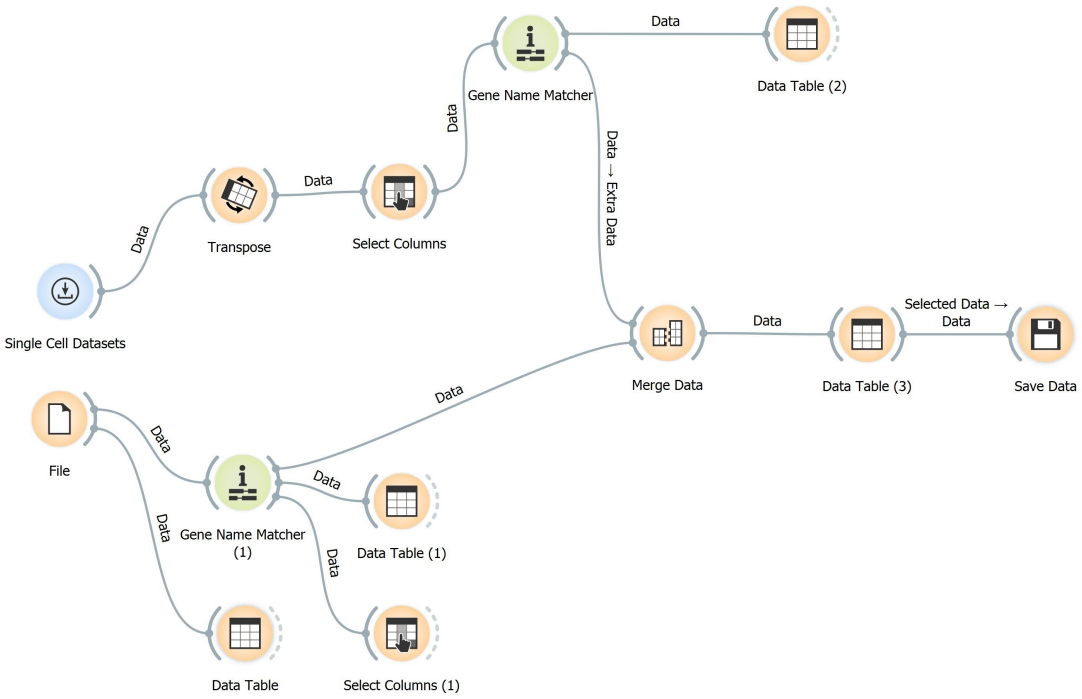


Figure 3.3: Matching gene names from "Bone marrow mononuclear cells with AML dataset" with Human CD markers in scOrange software. We uploaded our dataset and Human CD Markers. On CD Markers we performed "Gene Name Matcher" and after that we merged data to get matched gene names.

Chapter 4

Experimental evaluation

Our experimental evaluation consists of two parts: evaluation on synthetic data and evaluation on real gene-expression data. Evaluation will use the following matrices:

- with X we shall denote the gene-expression matrix,
- with A_1 we shall denote the adjacency matrix corresponding to the graph in which vertices are cells and there is an edge between two vertices if and only if the cells are of the same type,
- with A_2 we shall denote the adjacency matrix in which vertices are genes and there is an edge between two vertices if and only if the genes have value greater than 0 in a matrix. For A_2 we used the matrix from STRING database.

We shall show the visualization of the first two components of each method. We can calculate more components, but we need only two of them, so that we can make a 2D-plot.

4.1 Evaluation of methods on synthetic data

In this part, we evaluate performance of three different algorithms with simulated (synthetic) data. The process of constructing synthetic data was the

following:

- **constructing a matrix** X: we assumed that we have five different cell types: T cell, B cell, dendritic cell, NK cell and Granulocyte. We chose to have 200 cells of each type, so that in the end our matrix X contains 1000 rows. For each cell type we look in the Human CD Markers to see what are the corresponding marker genes. Then, for our 78 previously matched genes (columns) we put value greater than zero for a gene that is a marker gene for that cell type. We added a noise matrix to our matrix X. In the end, we added 1000 random genes (columns) to better simulate real gene-expression datasets where most of the genes are not the marker ones.

- **constructing a matrix** A_2: for the adjacency matrix for columns we took the data from the STRING database for the matched 78 genes. For the remaining 1000 random genes we have considered that they are adjacent with probability 0.3.

In measuring the efficiency of our methods used **silhouette score**. Silhouette scoring is a method of interpretation and validation of consistency within clusters of data. The silhouette score measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters [14].

We performed analysis on matrix X with 1000 cells and 1078 genes. In our experimental evaluation on synthetic data we are interested to see how good are standard SVD, L_1-norm SVD and L_0-norm SVD in discovering different types of cells. That is why in our analysis we do not have the adjacency matrix A_1 and therefore, the parameter for the importance of the prior graph A_1 is always equal to zero. The presence of matrix A_1 will make

the problem trivial and in the real datasets most of the time we do not own this knowledge.First, we performed standard SVD and the visualization is shown in Figure 4.1. The value of silhouette score for this visualization is 0.464.

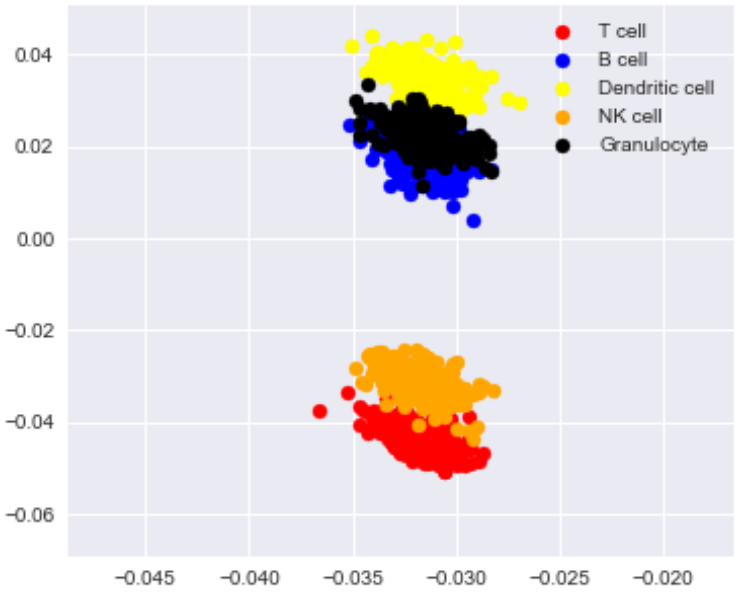


Figure 4.1: Standard SVD algorithm performed on synthetic data. The value of silhouette score for this visualization is 0.464.

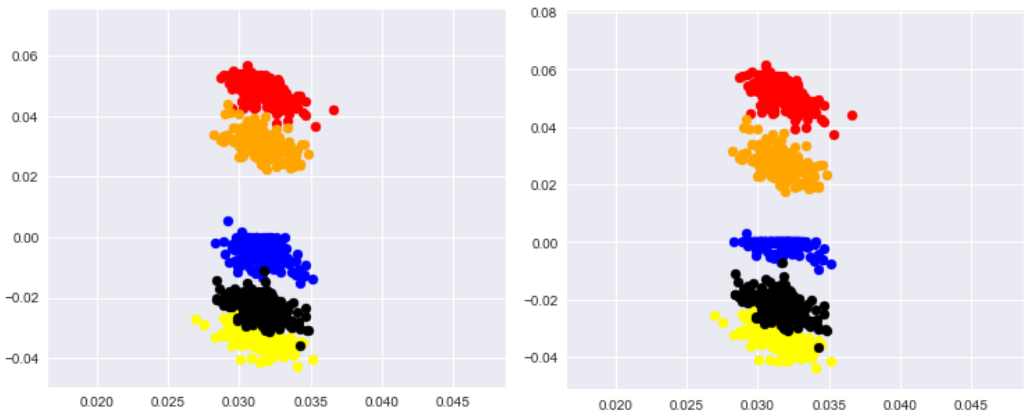
4.1.1 L_1-norm SVD

To demonstrate how the L_1-norm SVD and L_0-norm SVD work and how the visualization changes depending on different values of parameters, we chose to fix some parameters and to change only one parameter.

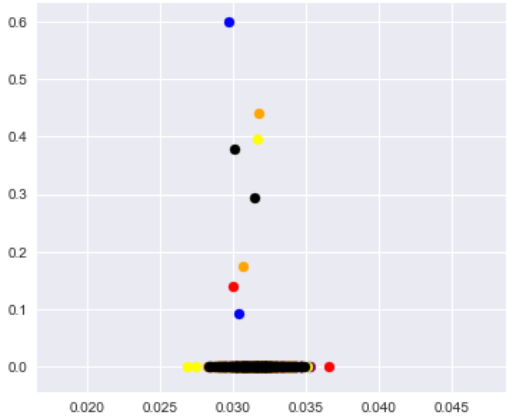
We can see different visualizations of L_1-norm SVD algorithm in Figure 4.2. We fixed σ_u, A_1 and σ_v with the following values: $\lambda_v = 1$, $\sigma_u = 0$ and $\sigma_v = 0.1$. We chose these values so that we can see how change of λ_u influence the visualization when value of λ_v is maximum and when importance of σ_v is small.

Table 4.1 shows how silhouette score changes with different values of λ_u. We

started with $\lambda_u = 0.1$ and we immediately got better result, 0.604, than with standard SVD, which has score 0.464. We are interested to see what is the maximum value of silhouette score which we can obtain by change of λ_u. We reached the maximum when $\lambda_u = 0.33$ (Figure 4.2 (b)) and score is equal to 0.645, which is a great improvement comparing to standard SVD and its score. For $\lambda_u = 0.4$ the score is -0.328, which is a sign that regularization parameter is too large. If we take a look in Algorithm 1, line 5, we can see that when we are updating coordinates for the left singular vector we are doing it by choosing the maximum between 0 and some value from which we substract λ_u. If we take too large value of λ_u, the algorithm will choose 0 as the coordinate update. This leads to a smaller silhouette score and worse visualization (Figure 4.2 (c)).



(a) $\lambda_u = 0.1$ **(b)** $\lambda_u = 0.33$



(c) $\lambda_u = 0.4$

Figure 4.2: L_1-norm SVD algorithm performed on *synthetic data* with parameters $\lambda_v = 1$, $\sigma_u = 0$, $\sigma_v = 0.1$ and different values of parameter λ_u. Synthetic data contains five types of cells: T cell (red), B cell (blue), dendritic cell (yellow), NK cell (orange) and Granulocyte (black).

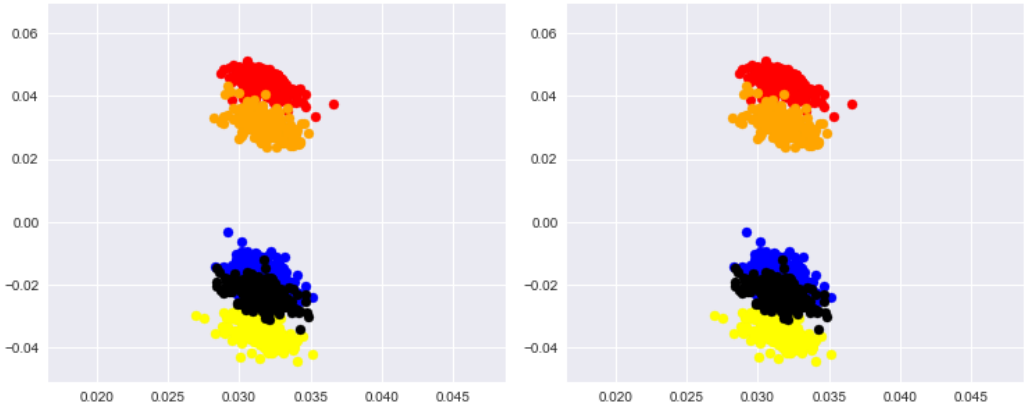
λ_u	Silhouette score
0.1	0.604
0.2	0.621
0.3	0.640
0.33	**0.645**
0.4	-0.328

Table 4.1: Silhouette scores for L_1-norm SVD performed on *synthetic data* with parameters $\lambda_v = 1$, $\sigma_u = 0$, $\sigma_v = 0.1$ and different values of parameter λ_u.

4.1.2 L_0-norm SVD

We fixed parameters k_u, k_v and σ_u with the following values: $k_u = 0$, $k_v = 0$ and $\sigma_u = 0$. The visualization of L_0-norm SVD is in Figure 4.3.

Table 4.2 shows how silhouette score changes by different values of σ_v. We started with $\sigma_v = 0.1$ and we got a result that is slightly better, 0.488, than with standard SVD. We are interested to see what is the maximum value of silhouette score which we can obtain with the change of σ_v. We reached the maximum when $\sigma_v = 0.9$ (Figure 4.3 (b)) and score is equal to 0.505. For $\sigma_v = 0.4$ the score is 0.502. With this we showed that putting too much importance of prior graphs can lead to worse silhouette score. In our case we got the adjacency matrix from STRING database and it did not find all genes and their interactions, thus the algorithm relies too much on data that is not necessarily correct. So, we have to be careful in regularization when depending on prior graphs.



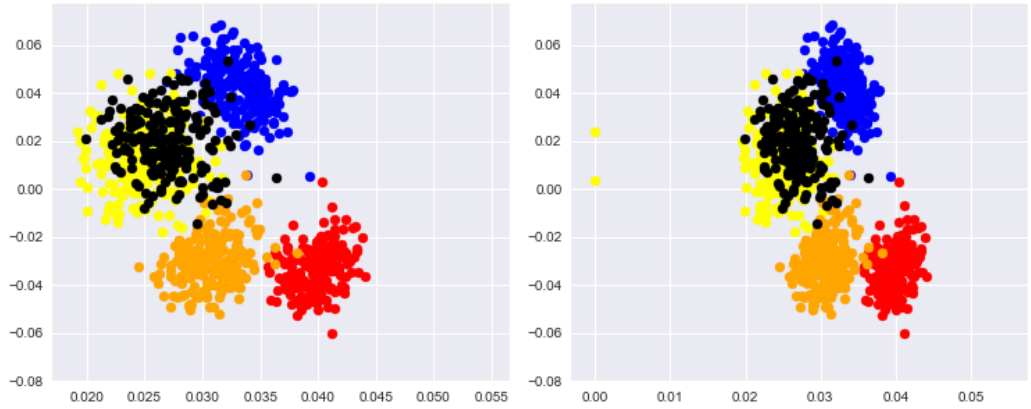
(a) $\sigma_v = 0.1$ **(b)** $\sigma_v = 0.9$

Figure 4.3: L_0-norm SVD algorithm performed on *synthetic data* with parameters $k_u = 0$, $k_v = 0$, $\sigma_u = 0$ and different values of parameter σ_v. Synthetic data contains five types of cells: T cell (red), B cell (blue), dendritic cell (yellow), NK cell (orange) and Granulocyte (black).

σ_v	Silhouette score
0.1	0.488
0.4	0.495
0.8	0.503
0.9	**0.505**
1	0.502

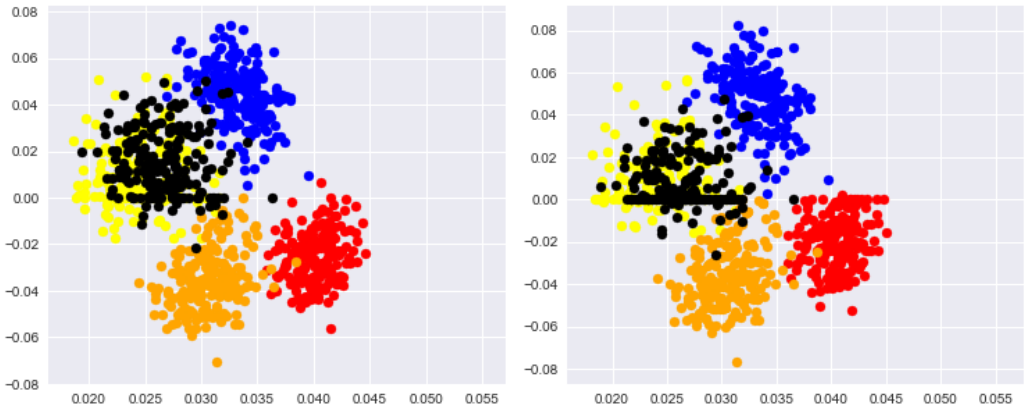
Table 4.2: Silhouette scores for L_0-norm SVD performed on *synthetic data* with parameters $k_u = 0$, $k_v = 0$, $\sigma_u = 0$ and different values of parameter σ_v.

We also present visualizations of synthetic data without 1000 random genes, when we only have marker genes as columns (Figure 4.5 and Figure 4.4). Here we can see that different cell types are separated better, since there are no random genes which create noise. This was an expected result.

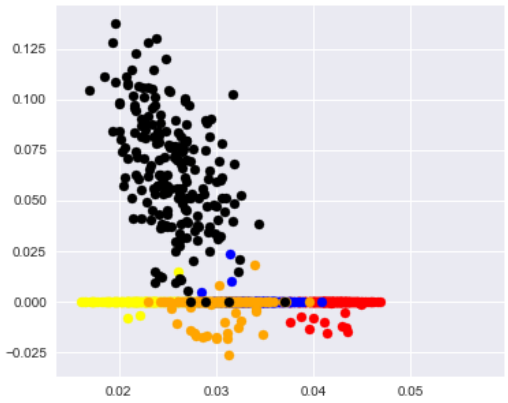


(a) $k_u = 0$ **(b)** $k_u = 2$

Figure 4.4: L_0-norm SVD algorithm performed on *synthetic data* with parameters $k_v = 0$, $\sigma_u = 0$, $\sigma_v = 1$ and different values of parameter k_u. Synthetic data contains five types of cells: T cell (red), B cell (blue), dendritic cell (yellow), NK cell (orange) and Granulocyte (black).



(a) $\lambda_u = 0.1$ **(b)** $\lambda_u = 0.2$



(c) $\lambda_u = 0.6$

Figure 4.5: L_1-norm SVD algorithm performed on *synthetic data* with parameters $\lambda_v = 1$, $\sigma_u = 0$, $\sigma_v = 0.1$ and different values of parameter λ_u. Synthetic data contains five types of cells: T cell (red), B cell (blue), dendritic cell (yellow), NK cell (orange) and Granulocyte (black).

4.2 Evaluation of methods on real dataset

4.2.1 Analysis on healthy and AML cells

In Subsection 4.1 we performed analysis without taking into consideration adjacency matrix A_1. The purpose of this subsection is to show that algorithms work well given the A_1. Visualization of standard SVD is in Figure 4.6

with score -0.002.

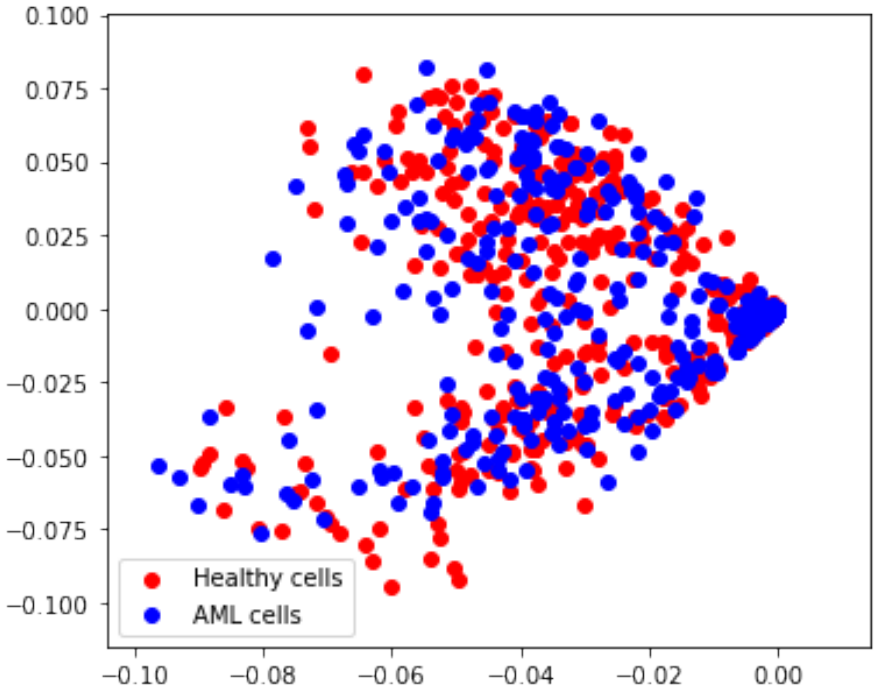
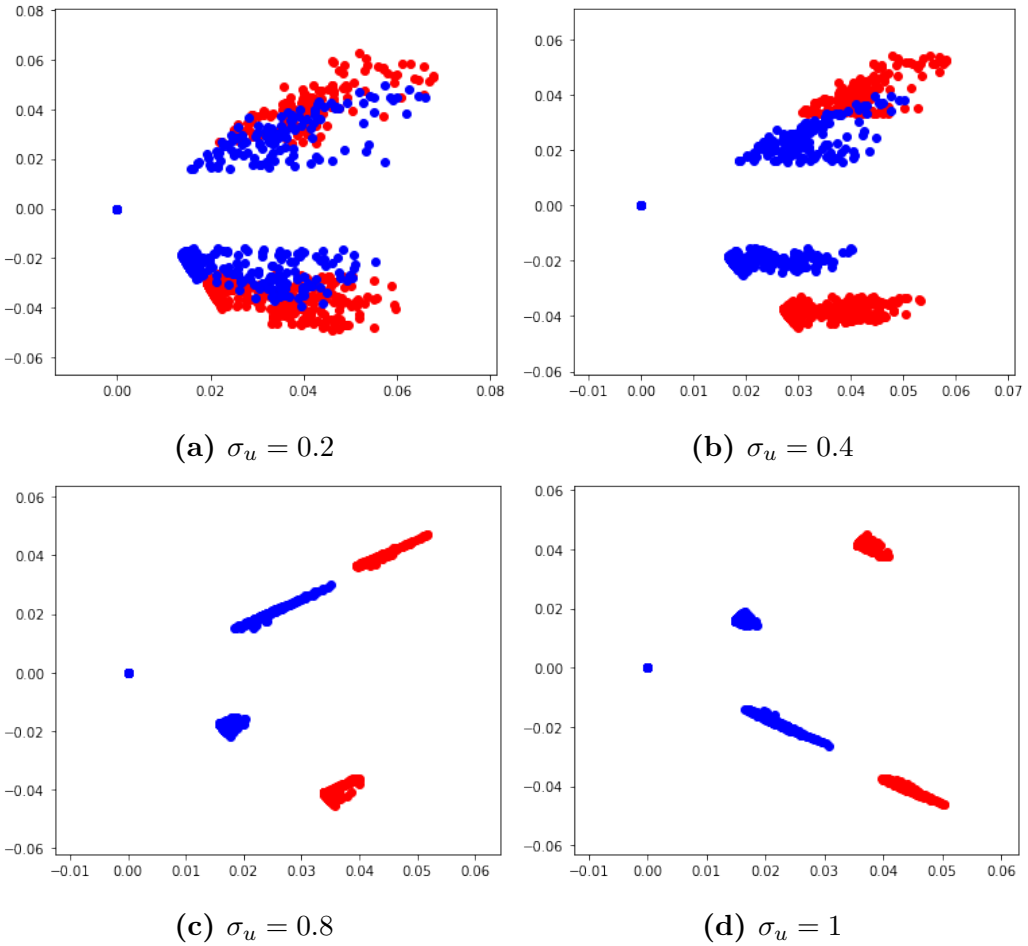


Figure 4.6: Standard SVD algorithm performed on dataset "Bone marrow mononuclear cells with AML".

L_1-norm SVD is shown in Figure 4.8 and L_0-norm SVD is in Figure 4.7.

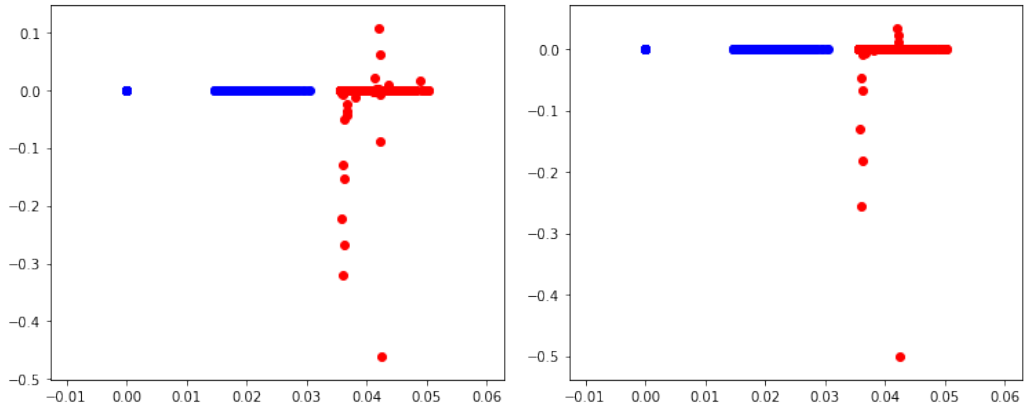


(a) $\sigma_u = 0.2$ **(b)** $\sigma_u = 0.4$

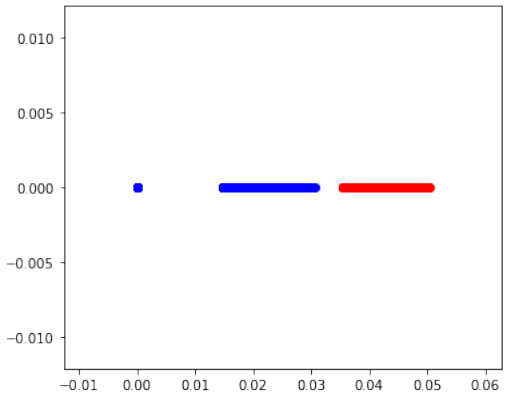
(c) $\sigma_u = 0.8$ **(d)** $\sigma_u = 1$

Figure 4.7: L_0-norm SVD algorithm performed on *Bone marrow mononuclear cells with AML* dataset with parameters $k_u = 0$, $k_v = 0$, $\sigma_v = 0.1$ and different values of parameter σ_u. Healthy cells are red and AML cells are blue.

Notice how the visualization for L_1-norm SVD and L_0-norm SVD is changing by increasing the importance of A_1. The best results for both methods are when the σ_u has the maximum value, 1. The best result has L_1-norm SVD which is 0.47. The values for both algorithms with different parameters are in Table 4.3 and Table 4.4.



(a) $\sigma_u = 0.2$ **(b)** $\sigma_u = 0.4$



(c) $\sigma_u = 1$

Figure 4.8: L_1-norm SVD algorithm performed on *Bone marrow mononuclear cells with AML* dataset with parameters $\lambda_u = 1$, $\lambda_v = 1$, $\sigma_v = 0.1$ and different values of parameter σ_u. Healthy cells are red and AML cells are blue.

σ_u	Silhouette score
0.2	0.06
0.4	0.16
0.8	0.22
1	**0.25**

Table 4.3: Silhouette scores for L_0-norm SVD performed on *"Bone marrow mononuclear cells with AML"* with parameters $k_u = 0$, $k_v = 0$, $\sigma_v = 0.1$ and different values of parameter σ_u.

σ_u	Silhouette score
0.2	0.41
0.4	0.43
0.8	0.47
1	**0.47**

Table 4.4: Silhouette scores for L_1-norm SVD performed on *"Bone marrow mononuclear cells with AML"* with parameters $\lambda_u = 1$, $\lambda_v = 1$, $\sigma_v = 0.1$ and different values of parameter σ_u.

4.2.2 Analysis on healthy cells

We are interested to see how good are standard SVD, L_1-norm SVD and L_0-norm SVD in discovering different types of cells, when we do not know what is the cell type. That is why we now perform analysis only on healthy cells. At the beginning, we selected one of the cell types, for example T Cell and we see which genes are markers (label "+" in 3.2) for the select cell type (CD1a, CD1b, CD1c...). Markers are collected in a set that is denoted by "X" and the number of elements in this set is denoted by "n".

On our dataset we performed standard SVD, L_0-norm SVD and L_1-norm

SVD.

The visualization of these methods is in Figure 4.9.

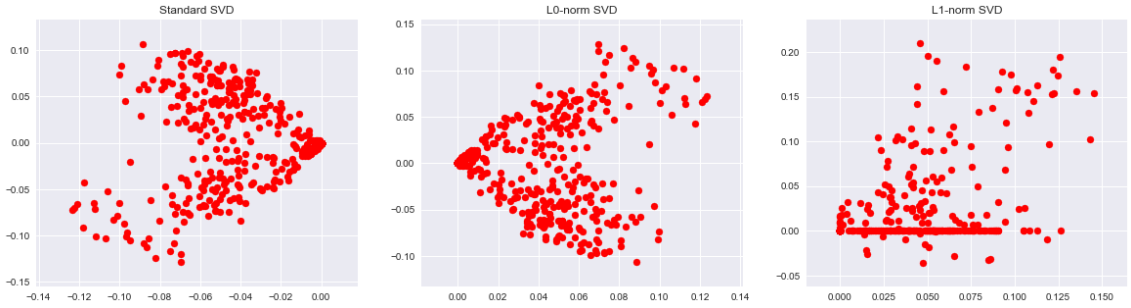
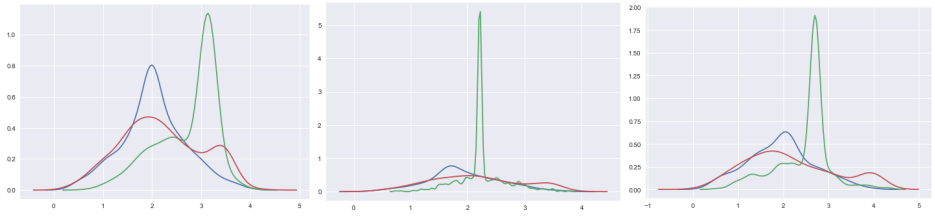


Figure 4.9: Standard SVD algorithm, L_0-norm SVD and L_1-norm SVD performed on healthy cells from "Bone marrow mononuclear cells with AML" dataset. We can see that standard SVD and L_0-norm differ slightly while in L_1-norm SVD there is a bigger regularization.

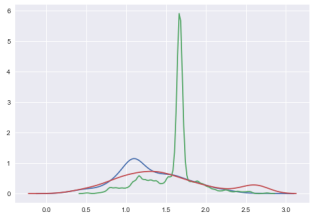
We chose each point from our visualization and "k" (in our case we put $k = 10$, since we have ten cell types) closest points around it. We calculated closest points using Euclidean distance. For each of the "k" points and our selected point, we look at the expression of the marker genes from our dataset (matrix X). For expression value 0 we have number 0 and for expression value > 0 we have number 1. We counted how many ones (1) we have and we denoted their number with "m". Score for this point is $\frac{m}{n} \cdot 100$. Then we made a distribution of all mean scores for standard svd, L_0-norm SVD and L_1-norm SVD and compared their graphs. We repeated this for all cell types. The goal was to show that methods L_1-norm SVD and L_0-norm SVD better combine cell types than standard SVD. In this part of our analysis we cannot use silhouette score as the measure for efficiency since we do not know the cell types. We shall take a look at distribution graphs of mean scores and see what is the difference among them.



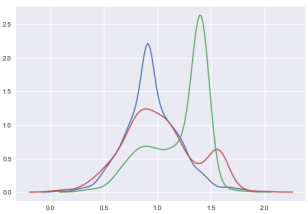
(a) T cell

(b) B cell

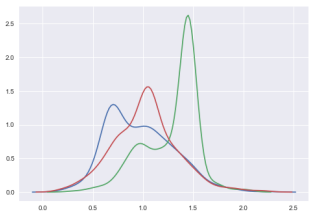
(c) NK cell



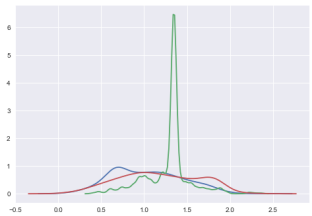
(d) Dendritic cell



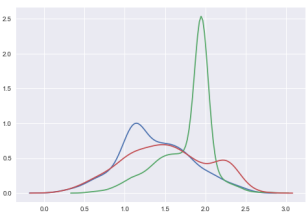
(e) Endothelial cell



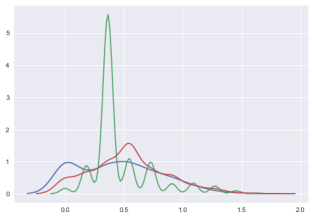
(f) Epithelial cell



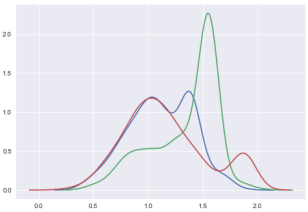
(g) Granulocyte



(h) Macrophage



(i) Platelet



(j) Stem-cell

Figure 4.10: Distributions of average means scores for different cell types. L_0-norm SVD is blue, L_1-norm SVD is green and standard SVD is red.

To evaluate the difference between the distributions we used the two sample Kolmogorov-Smirnov test (KS-test). This test is a non-parametric test that compares the cumulative distributions of two datasets. It tries to determine if two datasets differ significantly. The KS-test has the advantage of making no assumption about the distribution of data. The null hypothesis

is that both groups were sampled from populations with identical distributions. It tests for any violation of that null hypothesis – different medians, different variances, or different distributions. If the p-value is small, we can conclude that the two groups were sampled from populations with different distributions. The populations may differ in median, variability or the shape of the distribution.

We can see that for both algorithms and for all cell types p-value is smaller than 0.05. From this we can conclude that these distributions are different. Since the Kolmogorov-Smirnov test does not compare any particular parameter (i.e. mean or median), it does not report any confidence interval. A confidence interval is a type of interval estimate, that might contain the true value of an unknown population parameter. Most commonly, the 95% confidence interval is used, but also others can be used. We computed the 95% confidence interval and showed the results in Figure 4.11.

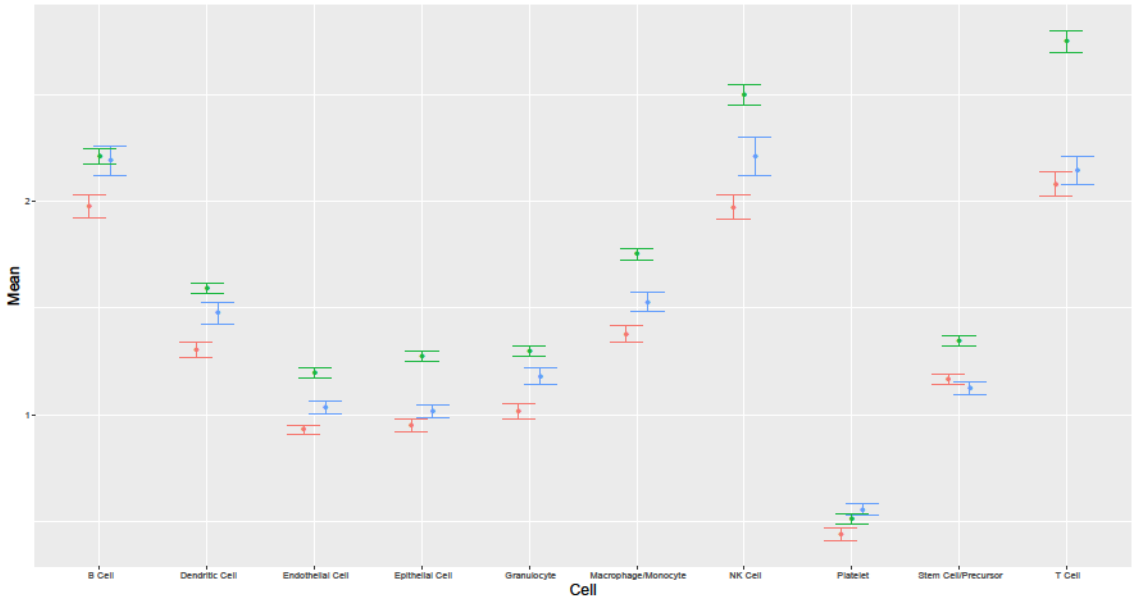


Figure 4.11: The 95% confidence interval for L_0-norm SVD (red), L_1-norm SVD (green) and standard SVD (blue). The L_1-norm SVD has the largest mean value in 9 out of 10 cell types.

From the confidence interval we see that for the most cell types L_1-norm SVD has the largest mean value (9 out of 10 cell types), while the mean scores for standard SVD and L_0-norm SVD are close. We can also notice that even if L_0-norm SVD and standard SVD have close values, we are more confident about the mean value of L_0-norm SVD algorithm than for standard SVD algorithm.

Chapter 5

Conclusion

In the thesis, we have reviewed two algorithms of regularized SVD: L_0-norm SVD and L_1-norm SVD [11]. Implementation was done in Python programming language and code is available on Github repository [1]. We tested the methods on synthetic data and on real gene-expression dataset to answer the following questions:

1. testing methods on the whole dataset and evaluating performance: how well different methods differentiate between healthy and AML cells?

2. testing only on healthy cells and evaluation of performance using marker genes: how different methods distribute mean marker scores?

We learned how to apply L_0-norm and L_1-norm on singular value decomposition method and what is the theoretical background behind it. Depending on the value of the parameters, both methods yielded different results, that is, different visualizations. In order to achieve the best result, we varied regularization parameters and found that in the case of over-regularization, the visualization is disturbed. In both tests, the best results were found with L_1-norm SVD. We showed that L_0-norm SVD and L_1-norm SVD better capture the structure of data than standard SVD.

[1] `https://github.com/Ejmric/L0-and-L1-Norm-SVD`

The continuation of this work can go in the way of relating this approach and data fusion approach [23] where we combine multiple sources of knowledge to get more accurate model. On each data source we can try to apply regularization and see is the result after data fusion better.

Bibliography

[1] The gene ontology project in 2008. *Nucleic Acids Research*, 36(SUPPL. 1), January 2008.

[2] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[3] James Eberwine, Jai-Yoon Sul, Tamas Bartfai, and Junhyong Kim. The promise of single-cell sequencing. 11:25–7, February 2014.

[4] Jochen Gorski, Frank Pfeuffer, and Kathrin Klamroth. Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical methods of operations research*, 66(3):373–407, 2007.

[5] David A. Harville. *Matrix Algebra From a Statistician's Perspective*. Springer, corrected edition, November 2000.

[6] Zhaoping Hong and Heng Lian. Sparse-smooth regularized singular value decomposition. *Journal of Multivariate Analysis*, 117:163–174, 2013.

[7] Roger A Horn, Roger A Horn, and Charles R Johnson. *Matrix analysis*. Cambridge university press, 1990.

[8] Michael Syskind Pedersen Kaare Brandt Petersen. The matrix cookbook. Available at `http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3274/pdf/imm3274.pdf`(2018/08/15).

[9] Mihee Lee, Haipeng Shen, Jianhua Z. Huang, and J. S. Marron. Biclustering via sparse singular value decomposition. *Biometrics*, 66 4:1087–95, 2010.

[10] Li Li. *Selected Applications of Convex Optimization*, volume 103. Springer, 2015.

[11] Wenwen Min, Juan Liu, and Shi-Hua Zhang. l_0-norm sparse graph-regularized SVD for biclustering. *CoRR*, abs/1603.06035, 2016.

[12] Patrick O. Brown Orly Alter and David Botstein. Singular value decomposition for genome-wide expression data processing and modeling. June 2000.

[13] Stanley Osher and Wotao Yin. Sparse recovery via l_1 and L_1 optimization. Technical report, CALIFORNIA UNIV LOS ANGELES DEPT OF MATHEMATICS, 2014.

[14] Sabine Schulte im Walde. *Experiments on the Automatic Induction of German Semantic Verb Classes*. PhD thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart, 2003. Published as AIMS Report 9(2).

[15] Haipeng Shen and Jianhua Z Huang. Sparse principal component analysis via regularized low rank matrix approximation. *Journal of multivariate analysis*, 99(6):1015–1034, 2008.

[16] Martin Sill, Sebastian Kaiser, Axel Benner, and Annette Kopp-Schneider. Robust biclustering by sparse singular value decomposition incorporating stability selection. *Bioinformatics*, 27 15:2089–97, 2011.

[17] Damian Szklarczyk, Andrea Franceschini, Stefan Wyder, Kristoffer Forslund, Davide Heller, Jaime Huerta-Cepas, Milan Simonovic, Alexander Roth, Alberto Santos, Kalliopi Tsafou, Michael Kuhn, Peer Bork, Lars Juhl Jensen, and Christian von Mering. STRING v10: protein-protein interaction networks, integrated over the tree of life. *Nucleic Acids Research*, 43(Database-Issue):447–452, 2015.

[18] Daniela M. Witten, Robert Tibshirani, and Trevor J. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10 3:515–34, 2009.

[19] Dan Yang, Zongming Ma, and Andreas Buja. A sparse singular value decomposition method for high-dimensional data. *Journal of Computational and Graphical Statistics*, 23(4):923–942, 2014.

[20] Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*, 347(6226):1138–1142, 2015.

[21] Yong Zhang Zhaosong Lu. Penalty decomposition methods for l_0-norm minimization. 2010.

[22] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8:14049, 2017.

[23] Marinka Žitnik and Blaž Zupan. Data fusion by matrix factorization. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):41–53, 2015.