21-241 Final Project: Analyzing Drug Response Using PCA

Sarah Cross, Aashna Kulshrestha

December 8th, 2023

1 Overview

Analyzing and predicting how small molecules change gene expression across different cell types is a remarkably complex problem that is extremely important in the field of drug discovery, as new medicines are typically costly and extremely expensive to develop. If able to accurately predict how a small molecule upregulates or downregulates a set of genes, therapeutics could be developed to target genes known to be associated with specific illnesses entirely computationally, significantly accelerating and expending the development of new medicines.

This project, using classic Linear Algebra techniques, attempts to divulge certain insights into the mechanisms and patterns found in drug perturbation, looking in particular at how to reduce down the dimensionality of differential gene expression using PCA in order to observe patterns amongst similar drugs. Additionally, by obtaining the principal components of a set of 18,211 genes, this may serve as a better input for a model predicting drug perturbation, as models being trained on relatively few samples (634 samples, in this case) with many inputs (18,211 genes) are likely to be subjected to extreme overfitting (eg. the models have enough parameters that they can "remember" the exact points given during training, and retain high accuracy during training while receiving extremely low accuracy when exposed to new data).

2 Background

2.1 Biological Background

This project uses a dataset developed by *Open Problems in Single-Cell Analysis*, containing single-cell gene expression profiles (with 18,211 genes) taken from human peripheral blood mononuclear cells (PBMCs). The cells were treated with 144 compounds, then gene expression was measured and post-processed [6].

The researchers stipulated that PCA was likely to perform well on this particular problem, as many genes are co-regulated. This can happen for a variety of reasons; for example, genes may be grouped into the same operon and may be transcribed into proteins together, or may be part of the same metabolic pathway and coregulate one another [2] [9]. As a result, it is likely that there are not 18,211 genes that are changing completely independently of one another, and that many are linearly dependent on one another, making principal component analysis a viable pre-processing technique for this dataset in particular [4] [3].

2.2 Linear Algebra Background

The first step to performing PCA (Principal Component Analysis) is to pre-process and center the relevant data from a chosen dataset. This ensures that, rather than capturing the overall average as

a significant variable, the resulting components are only looking at variance from *within* the dataset. In terms of this project, the dataset chosen was already centered.

Because different cell types inherently respond differently to small molecules, the researchers filtered the data based on cell type, focusing specifically on the NK and T-regulatory cell types individually, as the most data was available for these specific types of cells (not all drugs were tested on all cells because of the cytotoxic effects of certain drugs on certain cells). Both the NK and T-regulatory cells contained 146 available drug samples, with 18211 genes tested. As such, we filtered the dataset by these cell types, leaving us with A, a m *x* n matrix where $m = 18211$ and $n = 146$ for each cell type.

Next, we must find the Singular Value Decomposition (SVD) of A.

$$A = \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}$$

$$U\Sigma V^T = \begin{pmatrix} u_1 & \dots & u_m \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} \begin{pmatrix} v_1^T \\ \dots \\ v_n^T \end{pmatrix}$$

To continue performing PCA, we must choose a rank k approximation. Our thought process for choosing an appropriate k was based on the reconstruction accuracy for the dataset. Through trial-and-error, we found that when $k = 100$, the original dataset can be reconstructed with a 99.4% accuracy. Due to the high accuracy rate, we determined our k to be 100.

SVD according to the rank k approximation:

$$U\Sigma V^T = \begin{pmatrix} u_1 & \dots & u_k \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{pmatrix} \begin{pmatrix} v_1^T \\ \dots \\ v_k^T \end{pmatrix}$$

Finally, we can plot the first three principal components to obtain a summarized version of the dataset. The results of our findings are shown in *Section 4: Graphs and Results.*

3 Code

Note: the @printf command does not allow for new lines; however, for readability, we placed each argument on a new line.

```
# Import relevant packages
import Pkg;
Pkg.add("DataFrames")
Pkg.add("Statistics")
Pkg.add("LinearAlgebra")
Pkg.add("CSV")
Pkg.add("JSON")
using DataFrames, Statistics, Plots
using LinearAlgebra, Printf, CSV, DelimitedFiles, JSON

# Read file + push into dataframe
path = "nk_cells.csv"
```

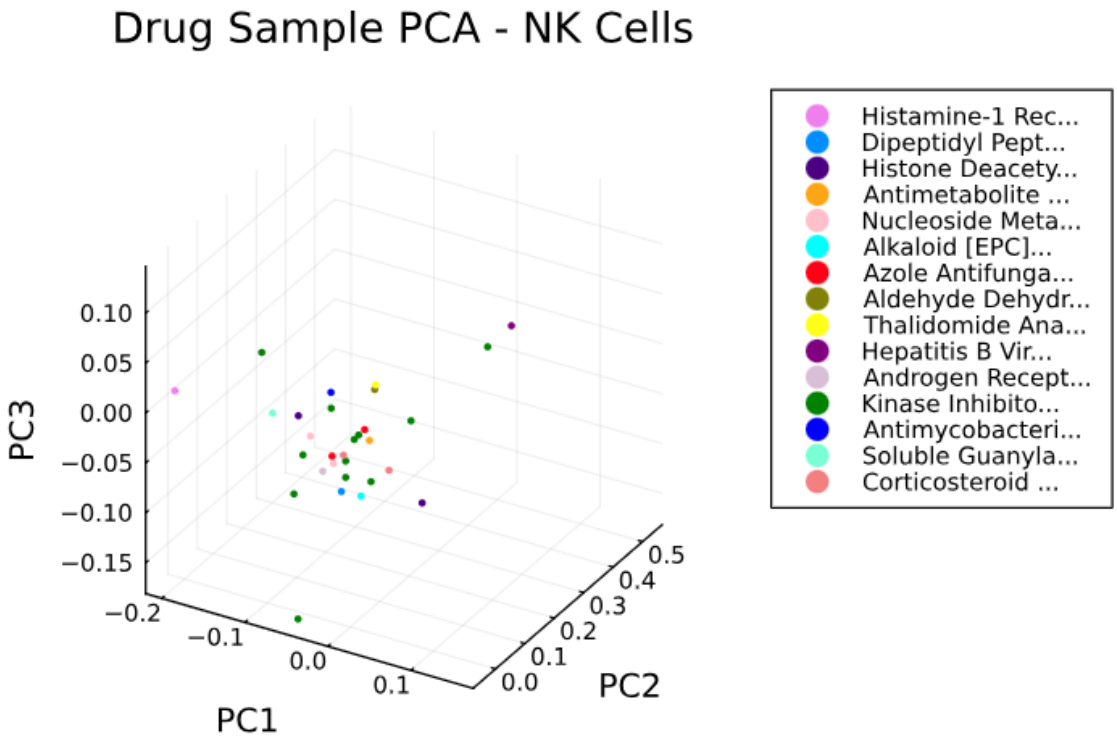
```
dframe = DataFrame(CSV.File(path, normalizenames=true))

# Display information about the file we are looking at
@printf "File %s:\n\tRows: %d\n\tCols: %d\n\tFirst Col Names: %s"
    path
    nrow(dframe)
    ncol(dframe)
    first(names(dframe), 20)

# There are several types of cells in the files:
"""
- 275 B cells - produce antibody molecules - lymph
- 97  T cells CD4+ - type of T cell, releases cytokines to activate immune cells
("helper cells") - myeloid
- 446 T regulatory cells - type of T cell, modulate immune system +
prevent autoimmune disease by maintaining tolerance to self-antigens
(essentially prevent body from attacking itself) - myeloid
- 405 NK cells - part of innate immune system, "natural killer cells",
allows body to fight back wo any previous "memory" of a pathogen - lymph
- 172 T cells CD8+ - type of T cell, cytoxic T cell ("killer cells")  - myeloid
"""
# We've selected NK cells in particular.

# Now we're going to make the matrix whose rows are composed of the col values for
# each gene
# Ex: drug A [gene a, gene b, ..., gene n]
data = Matrix(dframe[:, Not(["Column1", "sm_name", "cell_type"])])

# (1) Perform SVD
U, sigma, V = svd(transpose(data))

# Create a rank n SVD decomposition
k = 100
rank_k_U = U[:,1:k]
rank_k_sigma = Diagonal(sigma[1:k])
rank_k_V = V[1:k,:]

# Apply model to set
Y = rank_k_V
# Rank k estimation of data
X = rank_k_U * rank_k_sigma * rank_k_V

# Display the difference between the actual data + rank-n data projection
new_comp = data - transpose(X)
@printf "Difference between data:\n\tMean: %f\n\tSTD: %f\n\tAccuracy: %f%%"
    mean(new_comp)
    std(new_comp)
    (100 - 100 * mean((data - transpose(X)) / data))

# Standardize down data
m = mean(Y)
s = std(Y)

Z = (Y .- m)
```

```
# Calculate covariance matrix
c = cov(Z)
heatmap(1:size(c,1),
    1:size(c,2), c,
    xlabel="Drugs", ylabel="Drugs",
    title="Drug-Gene Reaction Similarities")

c = c - Matrix(1I, size(c,1), size(c,1))
display(max)
display(argmax(c))
display(dframe[!, "sm_name"][argmax(c)[1]])
display(dframe[!, "sm_name"][argmax(c)[2]])

# Let's label points by their corresponding EPC (pharmacologic class)
# if available, else do not show
# (1) Load drug information
info = JSON.parsefile("drug_information_indexed.json")

# (3) Map these to colors
map = Dict(
    "Azole Antifungal [EPC]"=>"red",
    nothing=>colorant"transparent",
    "Kinase Inhibitor [EPC]"=>"green",
    "Hepatitis B Virus Nucleoside Analog Reverse Transcriptase Inhibitor [EPC]"=>"purple",
    "Androgen Receptor Inhibitor [EPC]"=>"thistle",
    "Histone Deacetylase Inhibitor [EPC]"=>"indigo",
    "Dipeptidyl Peptidase 4 Inhibitor [EPC]"=>"dodgerblue",
    "Aldehyde Dehydrogenase Inhibitor [EPC]"=>"olive",
    "Alkaloid [EPC]"=>"cyan",
    "Antimycobacterial [EPC]"=>"blue",
    "Nucleoside Metabolic Inhibitor [EPC]"=>"pink",
    "Antimetabolite [EPC]"=>"orange",
    "Thalidomide Analog [EPC]"=>"yellow",
    "Histamine-1 Receptor Antagonist [EPC]"=>"violet",
    "Soluble Guanylate Cyclase Stimulator [EPC]"=>"aquamarine",
    "Corticosteroid [EPC]"=>"lightcoral"
)

# (2) Get SM names + replace with 4th column
coloring = []
for drug in dframe[!, "sm_name"]
    if info[drug][3] != nothing
        push!(coloring, map[info[drug][3]])
    end
end

# Plot first three principal components

p = scatter(Y[1,:], Y[2,:], Y[3,:], marker=:circle, markersize=2, c=coloring, linewidth=0)
plot(p,xlabel="PC1", ylabel="PC2", zlabel="PC3")

# (2) Get SM names + replace with 4th column
coloring = []
```

```
correlation = []
Y_cutdown = []

# "Cluster" values that are of the same EPC together
# classification class are of a similar PCA
epc_clustering = Dict(
    "Azole Antifungal [EPC]"=>[],
    nothing=>[],
    "Kinase Inhibitor [EPC]"=>[],
    "Hepatitis B Virus Nucleoside Analog Reverse Transcriptase Inhibitor [EPC]"=>[],
    "Androgen Receptor Inhibitor [EPC]"=>[],
    "Histone Deacetylase Inhibitor [EPC]"=>[],
    "Dipeptidyl Peptidase 4 Inhibitor [EPC]"=>[],
    "Aldehyde Dehydrogenase Inhibitor [EPC]"=>[],
    "Alkaloid [EPC]"=>[],
    "Antimycobacterial [EPC]"=>[],
    "Nucleoside Metabolic Inhibitor [EPC]"=>[],
    "Antimetabolite [EPC]"=>[],
    "Thalidomide Analog [EPC]"=>[],
    "Histamine-1 Receptor Antagonist [EPC]"=>[],
    "Soluble Guanylate Cyclase Stimulator [EPC]"=>[],
    "Corticosteroid [EPC]"=>[],
)

let ii = 1
for drug in dframe[!, "sm_name"]
    push!(correlation, info[drug][3])
    push!(epc_clustering[info[drug][3]], X[ii,:])
    if info[drug][3] !== nothing
        push!(Y_cutdown, Y[:,ii])
        push!(coloring, map[info[drug][3]])
    end
    ii += 1
end
end

# Let's label points by their corresponding EPC (pharmacologic class)
# if available, else grey

plot(xlabel="PC1",
    ylabel="PC2",
    zlabel="PC3",
    title="Drug Sample PCA - NK Cells",
    legend = :outertopright)

for (name, Y) in map_2
    if name == nothing
        continue
    end
    # Get principal components
    a, b, c = [], [], []
    for i in Y
        push!(a, i[1])
        push!(b, i[2])
```

```
        push!(c, i[3])
    end
    scatter!(
    a, b, c,
    marker=:circle,
    markersize=2,
    markerstrokewidth=0,
    label=first(name, 15) * "...", color=map[name],
    linewidth=0)
end

display(plot!())

# Preview expression profiles, broken up by drug class.
display(epc_clustering)

# Now look at average points + distance from each other
for (epc_class, values) in epc_clustering
    @printf "DRUG CLASS %s:\n" epc_class
    display(mean(vec(values)))
    @printf "For drug class %s:\n\tMean: %f\n\tSTD: %f\n"
        epc_class
        mean(mean(vec(values)))
        mean(std(vec(values)))
end

@printf("ALL DRUG CLASSES:\n")
display(Y)
@printf("\tMean: %f\n\tSTD: %f\n", mean(mean(vec(X))), mean(std(vec(X))))
```

4 Graphs and Results

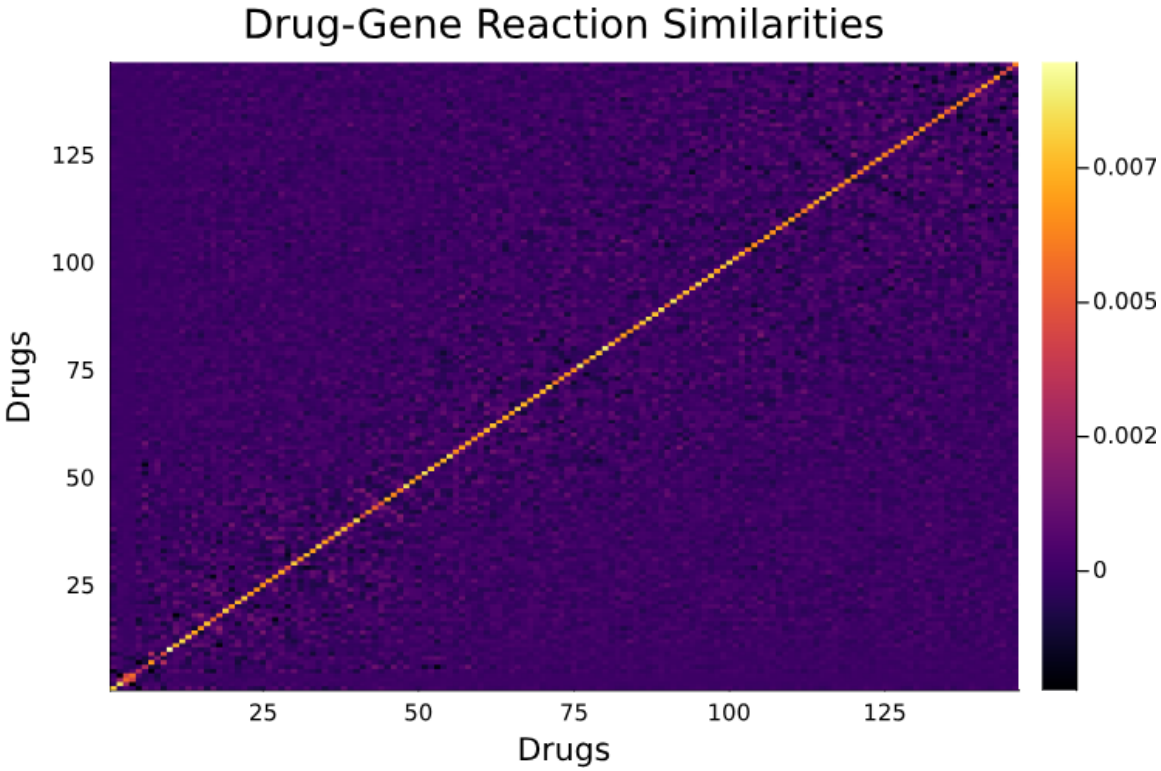


The graphs above displays the first three principal components of drug samples with pharmacological classifications (EPC) available in the OpenFDA Drug database [1] and cross-referenced with the National Drug Codes List [5], filtered by the type of cell (as different types of cells may respond very differently to the same type of drug, the researchers opted to analyze cell types separately, focusing on NK cells in particular). Ultimately it was found that there was a relatively weak correlation between drugs of the same pharmacological class, indicating that, although the drugs perform in the same way, they do not necessarily modify similar genes (however, only around 20 drugs had a corresponding EPC, which is likely not enough data to form a strong conclusion).

For NK Cells, the following information was found (in which standard deviation measures how different, on average, the gene expression data was across the drugs tested in that particular drug class): Excluding Azoles and Aldehyde drugs, the following appears to indicate that drugs within the same classification generally will result in similar gene expression profiles compared to how similar they are compared to drugs of all classes.

Drug Class	Standard Deviation
His-1 Receptor Antagonist	0.475053
Dipeptidyl Peptidase 4 Inhibitor	0.738885
Histone Deacetylase Inhibitor	0.868306
Antimetabolite	0.900148
Nucleoside Metabolic Inhibitor	1.317649
Alkaloid	0.660692
Azole Antifungal	0.612870
Aldehyde Dehydrogenase Inhibitor	2.026750
Thalidomide Analog	0.790018
Hepatitis B Virus Nucleoside Analog Reverse Transcriptase Inhibitor	1.174471
Kinase Inhibitor	1.041723
Androgen Receptor Inhibitor	0.727339
Antimycobacterial	0.919412
Soluble Guanylate Cyclase Stimulator	0.476024
All Drugs	1.904062

Table 1: Table comparing how different drug classes' resulting gene expression profiles are.



The graph above displays the covariance matrix for each drug and how similarly genes respond to perturbation. Lighter colors indicate that the drugs share similar differential gene expression data (the diagonal displays values of 1 because a drug will share 100% similarity with itself). From the data given, the covariance matrix displayed that the Vandetanib [8] and Idelalisib [7] drugs had the most similar gene expression profiles; this corroborates with existing information about both molecules, as both are anti-cancer drugs, kinase inhibitors, and both have a two-ring carbon, as well as 7 Hydrogen bond acceptor locations.

It was additionally found that a rank **100** matrix was able to form the original matrix with 99.5% accuracy, compared to the original 18,211 genes given, indicating that many genes can be represented as linear combinations of one another, and that the data can be significantly summarized down to

enable more efficient analysis and effective ML model training.

5 Bibliography

References

[1] URL: https://open.fda.gov/apis/ (visited on 12/08/2023).

[2] Piyush Agrawal, Vishaka Gopalan, and Sridhar Hannenhalli. "Predicting gene expression changes upon epigenomic drug treatment". In: *bioRxiv* (July 2023), p. 2023.07.20.549955. DOI: 10.1101/2023.07.20.549955. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10541107/ (visited on 12/08/2023).

[3] *Encyclopedia of Bioinformatics and Computational Biology*. en. URL: http://www.sciencedirect.com:5070/referencework/9780128114322/encyclopedia-of-bioinformatics-and-computational-biology (visited on 12/08/2023).

[4] *Gene regulation — Biological Principles*. URL: https://bioprinciples.biosci.gatech.edu/module-4-genes-and-genomes/4-7-gene-regulation/ (visited on 12/08/2023).

[5] *National Drug Codes List*. URL: https://ndclist.com/.

[6] *Open Problems – Single-Cell Perturbations*. en. URL: https://kaggle.com/competitions/open-problems-single-cell-perturbations (visited on 12/08/2023).

[7] PubChem. *Idelalisib*. en. URL: https://pubchem.ncbi.nlm.nih.gov/compound/11625818 (visited on 12/08/2023).

[8] PubChem. *Vandetanib*. en. URL: https://pubchem.ncbi.nlm.nih.gov/compound/3081361 (visited on 12/08/2023).

[9] Sergey V. Razin et al. "Co-Regulated Genes and Gene Clusters". In: *Genes* 12.6 (June 2021), p. 907. ISSN: 2073-4425. DOI: 10.3390/genes12060907. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8230824/ (visited on 12/08/2023).