Kathryn Graz
BE C175 Winter 2021

Final Report--Sanford *et al*.

Introduction/Motivation:

This project focuses on the re-implementation of Sanford *et al.*'s "Gene regulation gravitates toward either addition or multiplication when combining the effects of two signals", focusing on dimensionality reduction to determine the combined effects of signals on gene activity. The transcription of a single gene can be affected by multiple cell signals. It is pertinent within biological research to determine the relationship between overlapping cell signals for technologic or medicinal use. Sanford *et al.*'s model determines that most combined cell signals show either an additive or multiplicative effect. That is, two cell signals that affect the same gene will often result in a response equal to the sum or product of the signals individually. We hope to use PLSR and PCA techniques to further analyse the effect combined gene signals on transcription activity.

In this project, I focused on the PCA comparison between the normal and log-normal distributions, and motivating why that was necessary to capture the additive and multiplicative effects. This is my first python-heavy class, and I had not worked with pandas (or dataframes) before, so I definitely struggled with the large tpm dataframe and the pandas functions. I also motivated the R2X analysis of PCA worked with Dennis, and how the R2X could explain gene type. I also analyzed the PLSR results for the paper. Overall, this was a team effort. Leon gave me a pandas crash course, Dennis taught me how to condense the large dataframe and deal with bad values for the PCA, and Qing helped consolidate and rationalize the results.

Problem Definition:

We used a python jupyter notebook and Pandas. Our question from the paper was on similarities in gene regulations to predict gene transcription amount, and what are the most effective combinations for treatments.
Our project is interesting because it tracks how drugs affect cell signaling and transcriptional response and can be widely applied to many different aspects and fields of bioengineering. The project shows us how medicines/ drugs can affect cells, whether they are therapeutic or they can uncover previously unseen relationships that can be utilized in other areas of research. Because PCA can assist in determining if the drug is additive or not, this can be used as a primer for more complicated experiments.

PCA: Normal and Log-Normal

Methods:

Because we had 37 different treatments and over 20,000 genes, dimensionality reduction was necessary to find any meaningful relationships within the data. We first used PCA to try and find a relationship.

PCA by nature captures additive responses, because principal components are additive combinations of the data. We surmised that running PCA on the basic dataset would show us which gene/treatment combinations displayed additive responses. We had to clean up our data by selecting only data points we were interested in; for the PCA analysis, this was just the tpm values, along with the gene names and sample IDs.

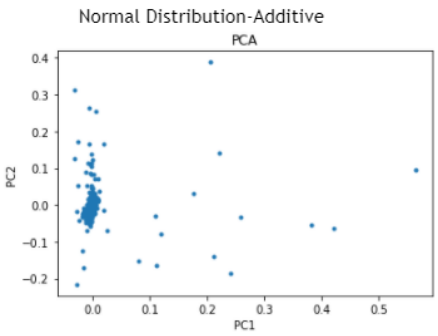
However, we also wanted a way to determine the multiplicative effects of genes within our dataset. We used log-transformation, because it turns a multiplicative process additive, as shown in the following log relationship:

$$log(x * y) = log(x) + log(y)$$

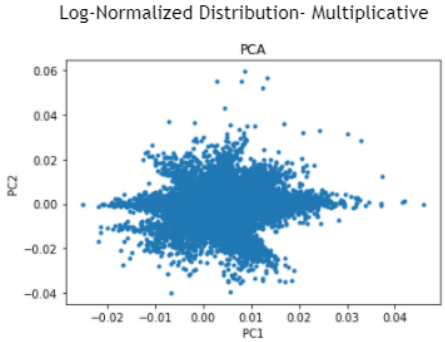
We initially struggled in forming our dataset because there were many 0 values within our tpm dataframe, values which cannot be log transformed. We figured, however, that these values were not very interesting to our question because zero-value results would not factor into our additive or multiplicative PCA interpretations. We simply held these values at 0 in our transformation to yield our complete dataset for multiplicative genes.

Results:

The following are the PCA graphs for both the normal and the log-transformed datasets:



Left: PCA on data. Shows additive genetic responses

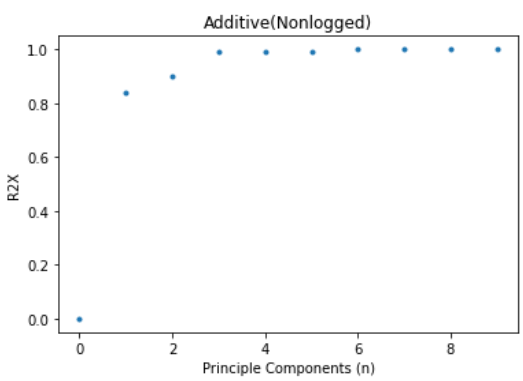


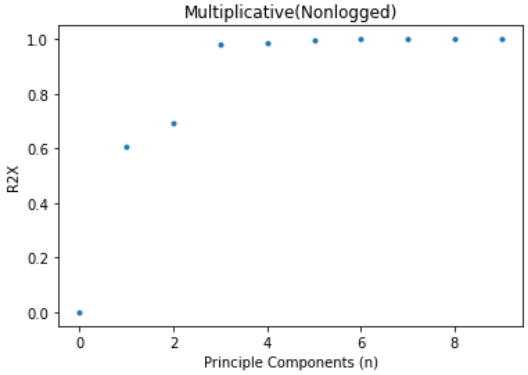
Right: Logged data points, then applied PCA Shows multiplicative genetic responses

The left and right graphs show the additive and multiplicative responses of the genes, respectively. It was difficult to draw concrete conclusions from these graphs. When we labelled the genes, we were able to tell which had similar additive and multiplicative responses (i.e., genes that were close together on either graph), and could determine vaguely how the genes behaved in general. However, there were not very many overarching patterns to go off of or predict with. To try and get more precise results, we turned to analysing the R2X values for differing principal components.

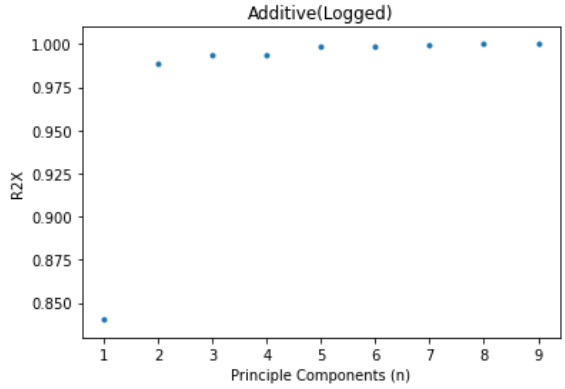
R^2X:

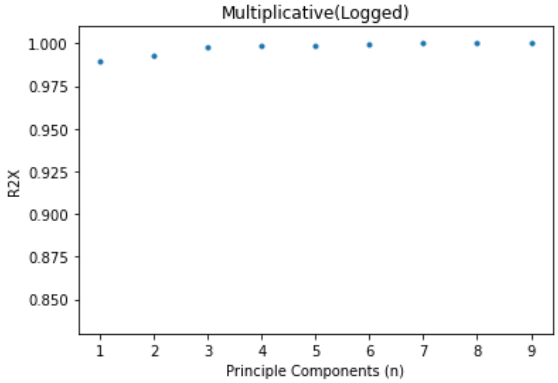
The original 37x27311 dataframe was manipulated into two 9x27311 dataframes. The first new dataframe was put through PCA analysis and the variance (R^2X) of a known additive gene (GPRC5A) and a known multiplicative gene (EPHB2) was plotted against the principal component numbers.





As expected, PCA explains more variance for the additive gene since principal components are additive combinations of the data. Then the second (logged) data frame was put through PCA analysis and GPRC5A and EPHB2 was again plotted for R^2X versus principal component number.





This also expectedly shows what we want because the manipulated logged data frame makes the multiplicative combinations look additive, which PCA captures better, as shown by the higher variance for the multiplicative gene.

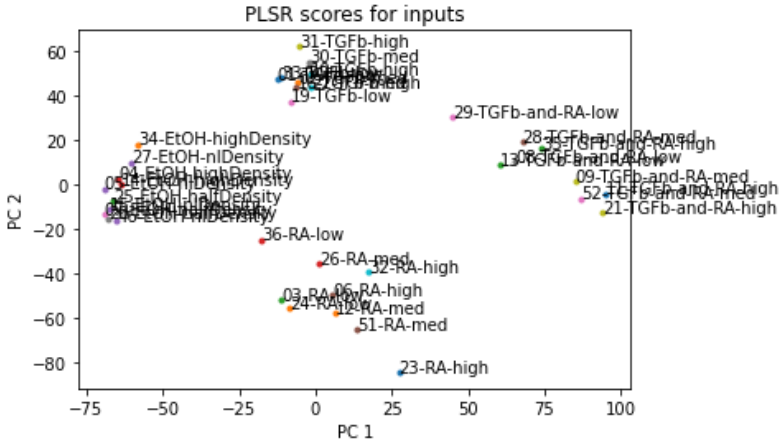
PLSR:

Tpm vs counts PLSR scores cluster:

We performed PLSR with tpm as input and gene count as output. Surprisingly, the scores plot

showing the different treatments had four distinct clusters for each type of treatment, regardless of dose concentration. This shows that specific types of signalling affects the behavior more than the amount of signalling.

The following are our PLSR results (notice the four distinct groups of treatments):



These four groups are indicative of some sort of underlying pattern between the cell treatments and genes. A further avenue of study would be to analyze these groups using K-means clustering to develop more precise parameters for the clusters and quantitatively look at relationships between the groups.

PLSR for Prediction: In an effort to predict transcription amount from retinoic acid and TGF-□, the data frame was reformatted by replacing the 'sampleID' string column with numerical 'TGFb' and 'Retinoic Acid' columns. The gene treatments were averaged and fitted for PLSR. We recorded the score and PLSR captured ~40% of the variance.

gene_name	TGFb	Retinoic Acid	A1BG	A1CF	A2M	A2ML1	A3GALT2	A4GALT	A4GNT
0	0.00	0	0.377997	0.035485	0.087716	0.263830	0.350411	12.057979	0.319075
1	0.00	50	0.345151	0.034716	0.127103	0.137183	0.000000	13.425112	0.000000
2	0.00	200	0.300917	0.000000	0.162882	0.082244	0.345012	13.311664	0.000000
3	0.00	400	0.203307	0.000000	0.211807	0.069556	0.335723	13.785032	0.000000
4	1.25	0	0.256296	0.000000	0.080188	0.125608	0.000000	11.649623	0.000000
5	1.25	50	0.351086	0.000000	0.054925	0.075671	0.385303	8.973253	0.000000
6	5.00	0	0.230624	0.035837	0.053812	0.110743	0.336408	9.336744	0.193978
7	5.00	200	0.362810	0.000000	0.054184	0.050074	0.676265	9.562037	0.000000
8	10.00	0	0.350926	0.032057	0.092482	0.130880	0.358890	10.128333	0.000000
9	10.00	400	0.419643	0.046901	0.000000	0.098149	0.376772	10.329859	0.000000

Figure 1

Prediction: After fitting the model we were then able to predict the outputs of sample ‘TGFb’ and ‘Retinoic Acid’. The first sample prediction mirrors the last treatment row in Figure 1. We can get a rough estimate of how far off our predictions are for our inputs instead of just looking at the score. The second sample input is an extrapolation which helps to determine the efficacy of additional amounts of protein.

	TGFb	Retinoic Acid	A1BG	A1CF	A2M	A2ML1	A3GALT2	A4GALT	A4GNT	AAAS	...	ZWILCH	ZWIN
0	10	400	0.362298	0.029668	0.054499	0.054887	0.518592	10.323523	-0.043468	40.068333	...	22.720921	52.02081
1	14	300	0.393041	0.042628	0.004124	0.067310	0.545893	8.713564	-0.029093	38.268409	...	20.779741	35.96603

2 rows × 22114 columns

Figure 2

Sample Treatment Scores

To view how treatments were affecting the tpm count we reformatted the dataframe where X=treatments(rows) x genes(columns) where the values=’count’(transcription count) and Y=treatments(rows) x genes(columns) where the values=’tpm’. We shrunk the matrix by averaging the treatments. After running and plotting PLSR we can see that retinoic acid seems to be a negative regulator of gene transcription on PC2 while TGF-□ is a positive regulator on

the same principal component. It's also obvious that the combination of the two treatments produces the greatest effect on the main principal component of PC1.

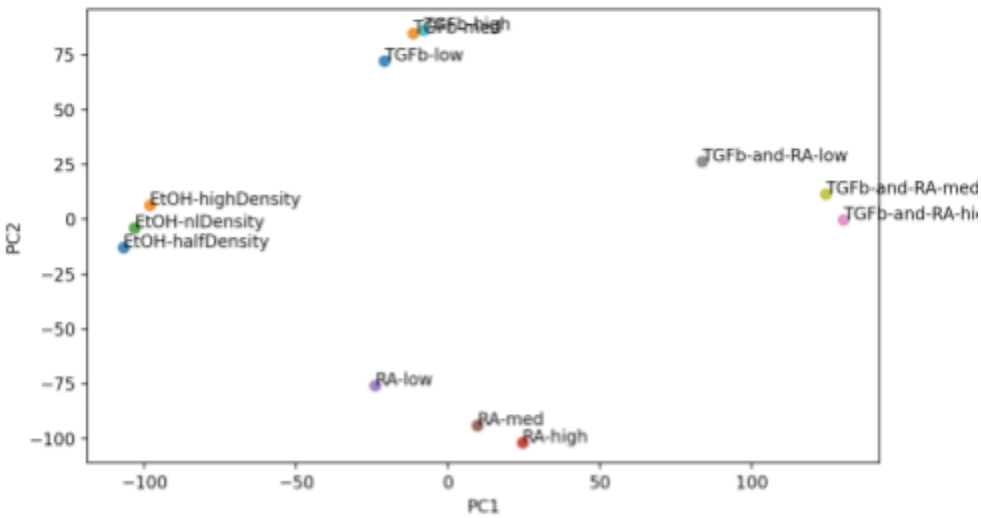


Figure 3

Neural Network

In an effort to compare PLSR prediction capabilities vs those of a neural network(NN) we used Tensorflow to create an NN with 3 input nodes, 2 hidden layers of 64 nodes, and 1 layer of output for regression. The reformatted datatable was fed into the NN and ran for 100 epochs. The results for regression were incredibly poor. Using the Keras Accuracy metric repeatedly showed accuracy at zero for every single epoch.

To try to improve the results we reduced the gene matrix output data to one single gene column 'AADAC'. Unfortunately the results were the same as before with poor regression estimation and zero accuracy. Upon further inspection we discovered that NNs need large samples of data to approximate a solution, and since we had only 10 samples which we averaged using Pandas Dataframe methods it likely could not converge on a solution. There was also the option of one-hot encoding the genes as columns for the input and genes with their 'tpm' count for the output, but the NN would take too long to converge and because many genes were only tested once or a handful of times we suspected our results would not dramatically improve.

Conclusion

We have found that PLSR was a really powerful analytical tool, which generated a high accuracy result in comparing the sheer number of outputs versus minimum inputs. However, the neural network will not be a useful analytical tool when we have a small amount of data. PCA was found very useful in describing data better with different treatments after log transformation, and that was because log transformed PCA can help us interpret the multiplicative effect of the data.

References

[1] Sanford, E. M., Emert, B. L., Coté, A., & Raj, A. (2020). Gene regulation gravitates toward either addition or multiplication when combining the effects of two signals. eLife, 9, e59388. https://doi.org/10.7554/eLife.59388

Git: https://github.com/leonaburime/be275_final_project