Predict how DNA, RNA & protein measurements co-vary in single cells

Open Problems - Multimodal Single-Cell Integration

ALEX ROBERTS, 11/11/2022

Table of Contents

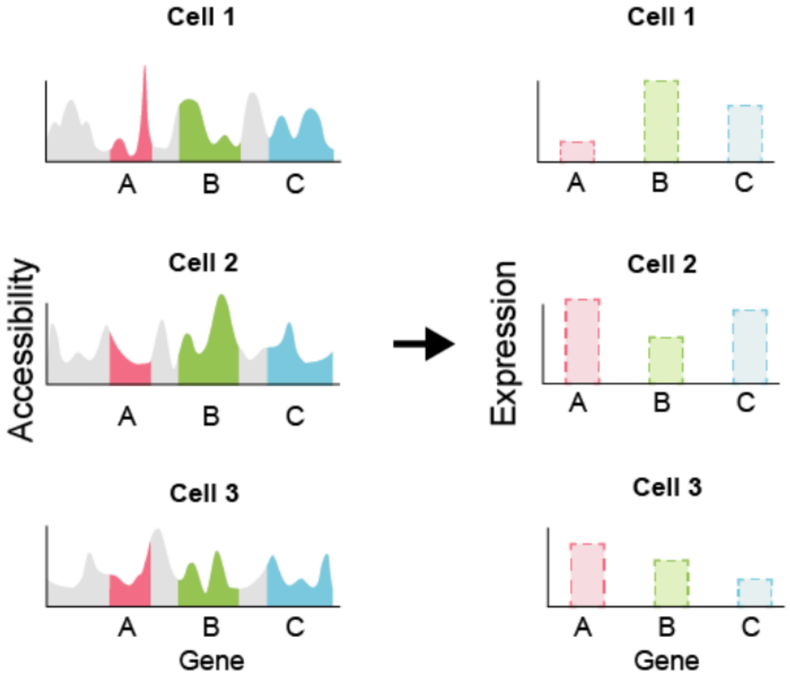
Introduction

Goal of the Competition

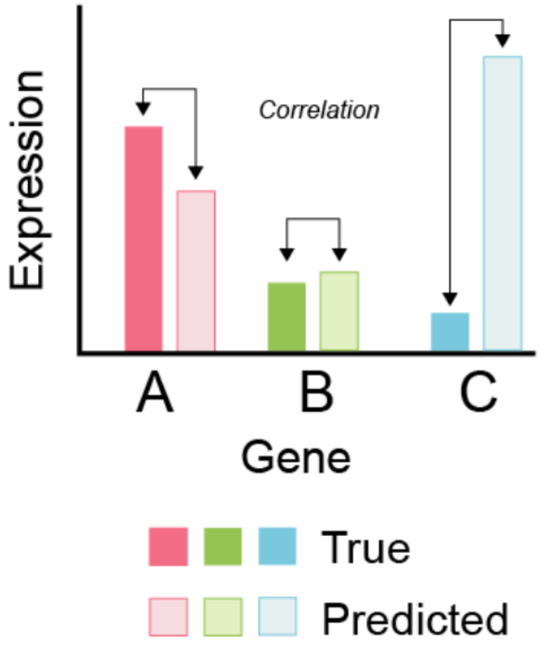
Predict how DNA, RNA, and protein measurements co-vary in single cells as bone marrow stem cells develop into more mature blood cells.

Develop a model trained on a subset of a 300,000-cell time course dataset of CD34+ hematopoietic stem and progenitor cells (HSPC) from four human donors at five time points generated for this competition by Cellarity, a cell-centric drug creation company.

Task



Metric



by ‘Open Problems in Single-Cell Analysis’ on Kaggle

History

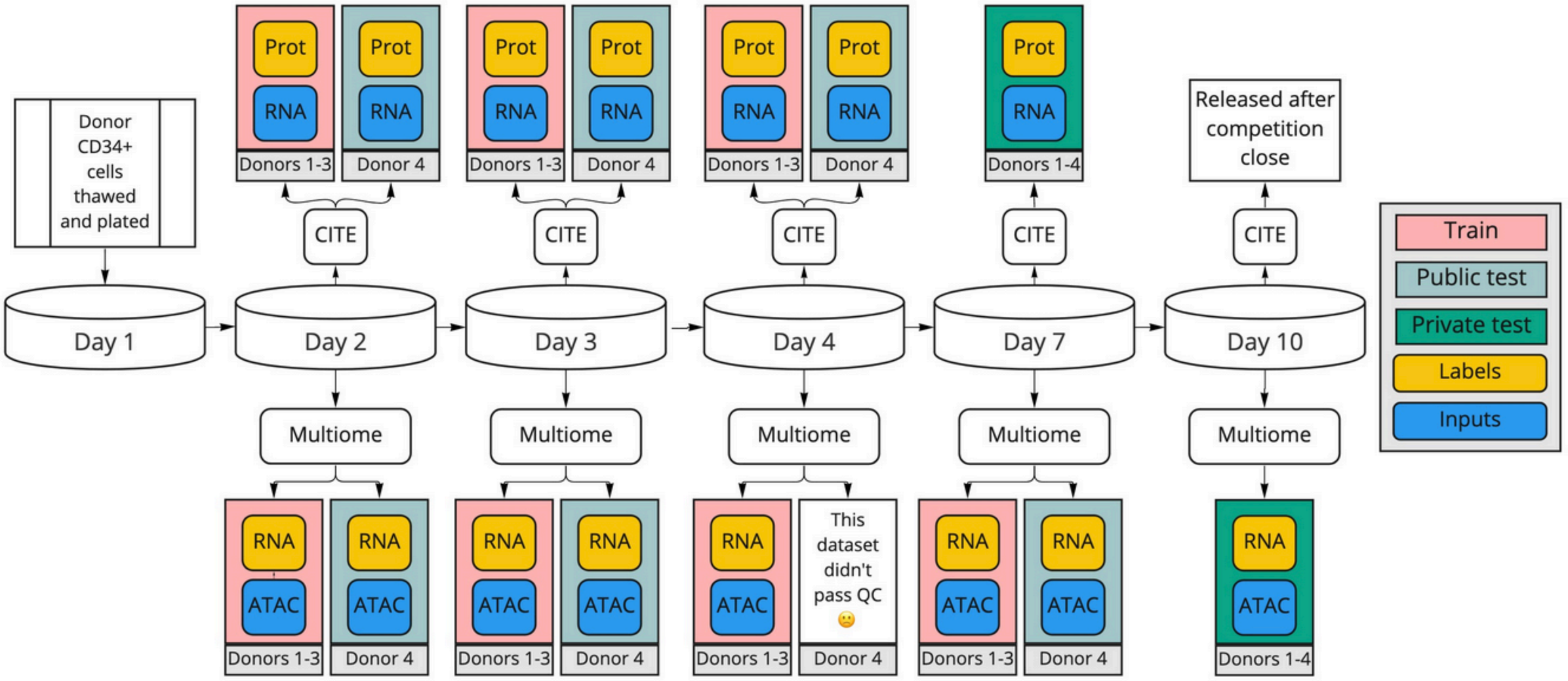
- In the past decade, the advent of single-cell genomics has enabled the measurement of DNA, RNA, and proteins in single cells.
- These technologies allow the study of biology at an unprecedented scale and resolution.
- Among the outcomes have been detailed maps of early human embryonic development, the discovery of new disease-associated cell types, and cell-targeted therapeutic interventions.
- Moreover, with recent advances in experimental techniques it is now possible to measure multiple genomic modalities in the same cell.

Task

Each assay technology measures two modalities. The Multiome kit measures **chromatin accessibility** (DNA) and **gene expression** (RNA), while the CITEseq kit measures **gene expression** (RNA) and **surface protein levels**.

Following the central dogma of molecular biology: **DNA → RNA→Protein**, your task is as follows:

- For the **Multiome** samples: given **chromatin accessibility**, predict **gene expression**.
- For the **CITEseq** samples: given **gene expression**, predict **protein levels**.



by ‘Open Problems in Single-Cell Analysis’ on Kaggle

Train/Test Splits

Multiome	Day 2	Day 3	Day 4	Day 7	Day 10
Donor 1	Train				Private Test
Donor 2					
Donor 3					
Donor 4	Public Test				

CITEseq	Day 2	Day 3	Day 4	Day 7
Donor 1	Train			Private Test
Donor 2				
Donor 3				
Donor 4	Public Test			

Scoring

- Pearson correlation coefficient is used to rank submissions.
- For each observation in the Multiome data set, the correlation between the ground-truth gene expressions and the predicted gene expressions is computed.
- For each observation in the CITEseq data set, the correlation between ground-truth surface protein levels and predicted surface protein levels is computed.
- The overall score is the average of each sample's correlation score.

In statistics, the **Pearson correlation coefficient** (**PCC**, pronounced /ˈpɪərsən/) — also known as **Pearson's *r***, the **Pearson product-moment correlation coefficient** (**PPMCC**), the **bivariate correlation**,[1] or colloquially simply as **the correlation coefficient**[2] — is a measure of linear correlation between two sets of data. It is the ratio between the covariance of two variables and the product of their standard deviations; thus, it is essentially a normalized measurement of the covariance, such that the result always has a value between −1 and 1. As with covariance itself, the measure can only reflect a linear correlation of variables, and ignores many other types of relationships or correlations. As a simple example, one would expect the age and height of a sample of teenagers from a high school to have a Pearson correlation coefficient significantly greater than 0, but less than 1 (as 1 would represent an unrealistically perfect correlation).

by Wikipedia

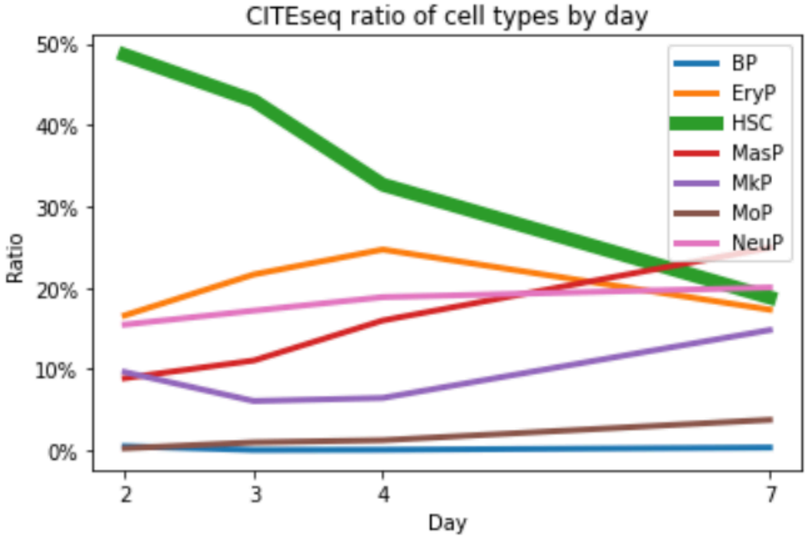
1197 teams are competing. Prizes are $15000 (1st), $7000 (2nd), $3000 (3rd).

The top 500 teams score between 81.2% and 81.5% on the public leaderboard.

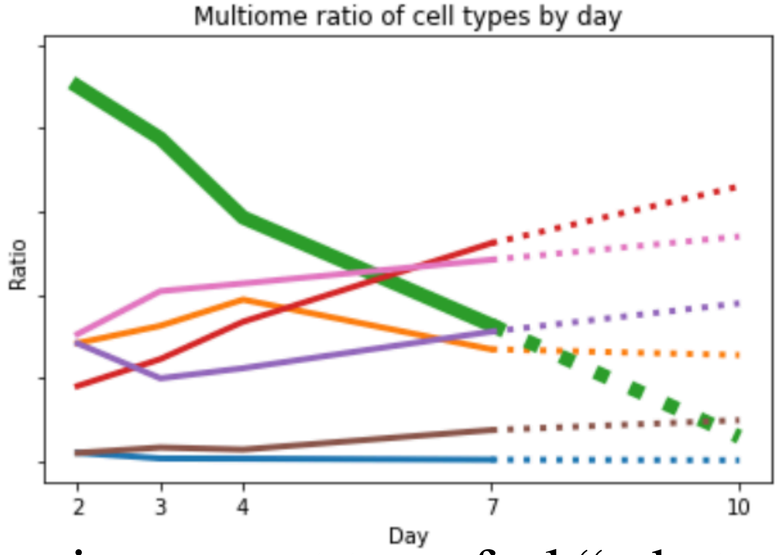
Competition hosted by Open Problems in Single-Cell Analysis partnering with Cellarity, Chan Zuckerberg Biohub, the Chan Zuckerberg Initiative, Helmholtz Munich, and Yale.

Challenges

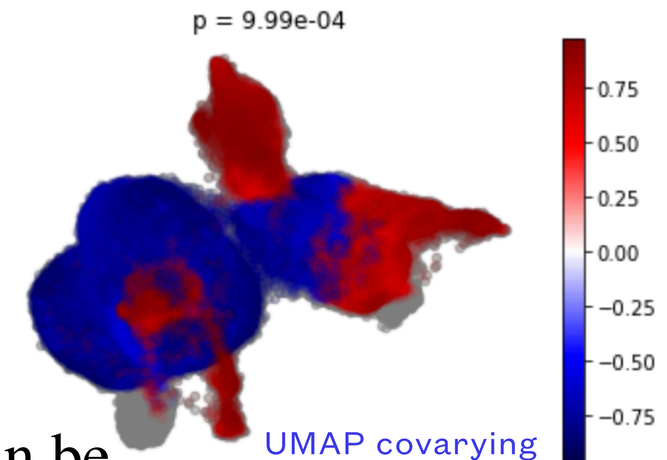
Challenges - Times Series



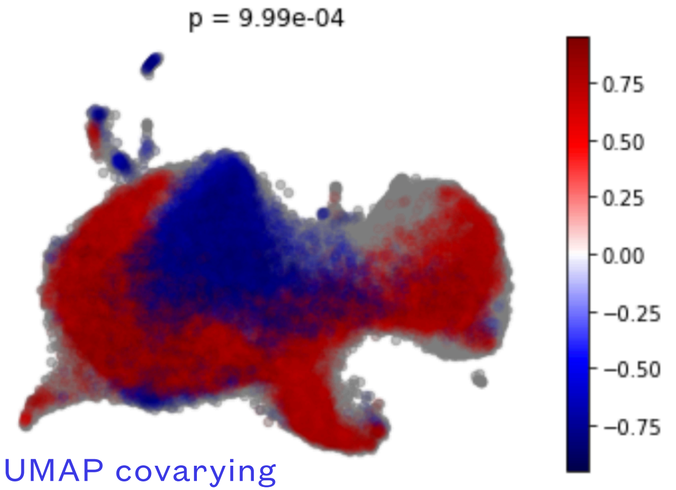
by @AmbrosM



- Current pipelines for single-cell data analysis treat cells as static snapshots, even when there is an underlying dynamical biological process.
- **Accounting for temporal dynamics alongside state changes over time is an open challenge in single-cell data science.**



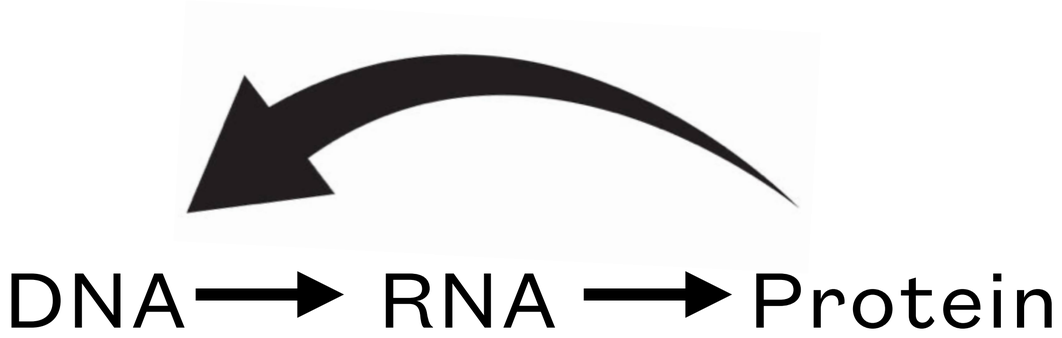
UMAP covarying neighborhood analysis (CNA) of gene expression levels with respect to date



UMAP covarying neighborhood analysis (CNA) of protein levels with respect to date

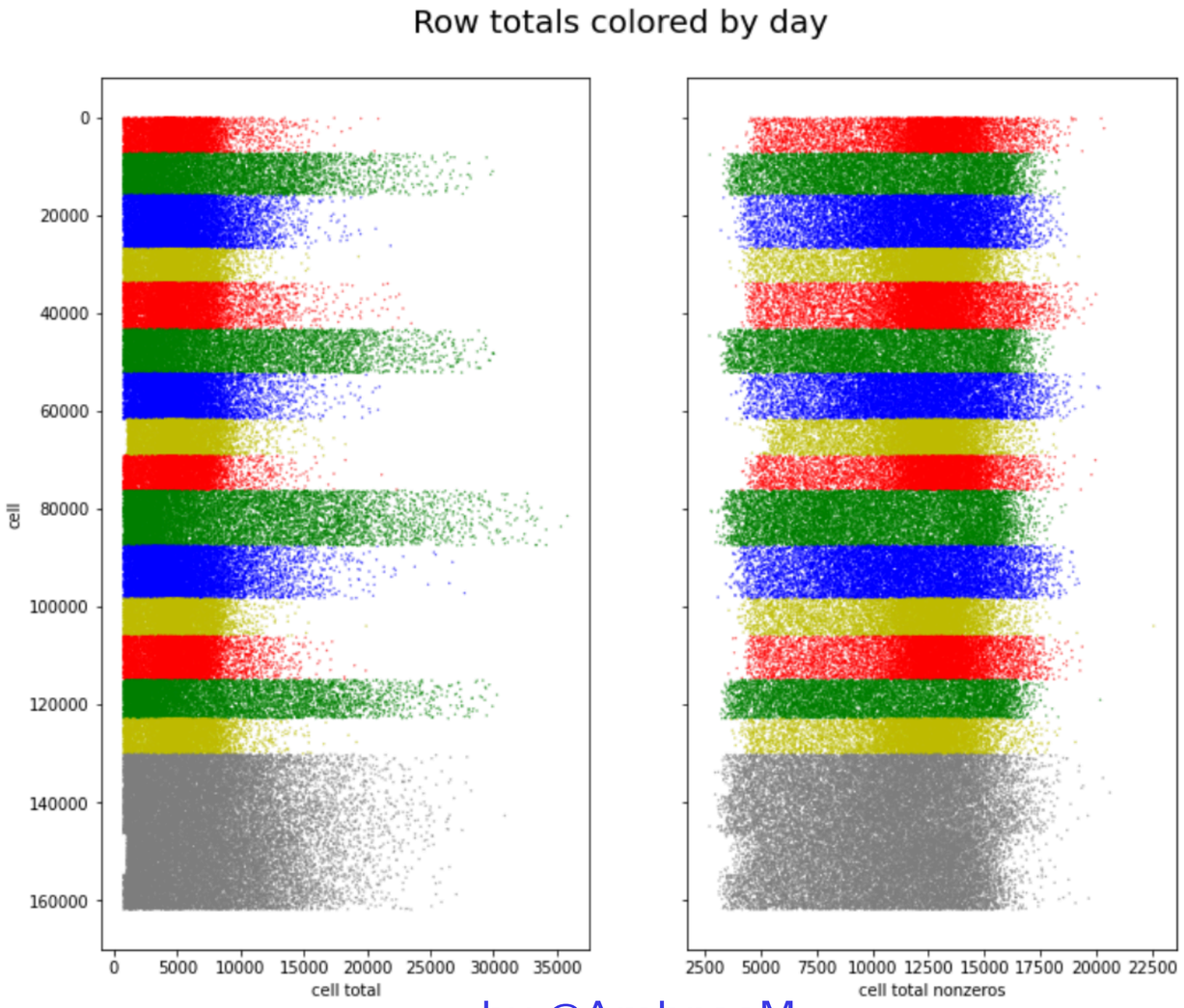
Organizers want to unfurl “what progress can be made in **predicting changes in genetic dynamics over time** through interdisciplinary collaboration.”

Challenges - Gene regulation



- Understanding how a single genome gives rise to a diversity of cellular states is the key to gaining mechanistic insight into how tissues function or malfunction in health and disease.
- **These processes are regulated by feedback: for example, a protein may bind DNA to prevent the production of more RNA.**
- This genetic regulation is the foundation for dynamic cellular processes that allow organisms to develop and adapt to changing environments.
- Being able to solve the prediction problems over time may yield new insights into how **gene regulation influences differentiation as blood and immune cells mature**.

Challenges - Batch effects



by @AmbrosM

- While multimodal single-cell data is increasingly available, data analysis methods are still scarce.
- Due to the small volume of a single cell, measurements are sparse and noisy.
- Differences in molecular sampling depths between cells (sequencing depth) and technical effects from handling cells in batches (batch effects) can often overwhelm biological differences.

Left: Different colors are cells collected on different days (bands are individual donors). x-value is the sum of all gene availabilities for all genes. Day 3 has much higher availabilities. Right: # of non-zeros is about the same.

Challenges - Big Data

Multiome input

The Multiome dataset is much larger than the CITEseq part and way too large to fit into 16 GByte RAM:

- train inputs: 105942 * 228942 float32 values (97 GByte)
- train targets: 105942 * 23418 float32 values (10 GByte)
- test inputs: 55935 * 228942 float32 values (13 GByte)

- 98% for Multiome and 78% for CITEseq data are zeroes -> have to use sparse arrays.
- Implications:

made available by @fabiencrom

  - have to do SVD instead of PCA. Luckily most sklearn models support sparse arrays.
  - need over 100GB of RAM. Luckily, organizers provide free cluster access with 130GB RAM, 300GB storage so no issue (except GPU RAM limitation).

Dataset description

Data collection

- The dataset for this competition comprises single-cell multiomics data collected from mobilized peripheral CD34+ hematopoietic stem and progenitor cells (HSPCs) isolated from four healthy human donors.
- Measurements were taken at five time points over a ten-day period. During this time, cells were cultured with StemSpan SFEM media supplemented with CC100 and thrombopoietin (TPO) and incubated at 37ºC. Media was changed every 2-3 days. No additional media supplements were added to the cell culture conditions.
- From each culture plate at each sampling time point, cells were collected for measurement with two single-cell assays. The first is the **10x Chromium Single Cell Multiome ATAC + Gene Expression** technology (Multiome) and the second is the **10x Genomics Single Cell Gene Expression with Feature Barcoding technology** technology using the TotalSeq™ -B Human Universal Cocktail, V1.0 (CITEseq).

File and Field Descriptions

- **metadata.csv**
  - `cell_id` - A unique identifier for each observed cell.
  - `donor` - An identifier for the four cell donors.
  - `day` - The day of the experiment the observation was made.
  - `technology` - Either `citeseq` or `multiome`.
  - `cell_type` - One of the above cell types or else `hidden`.

cell_id	day	donor	cell_type	technology
c2150f55becb	2	27678	HSC	citeseq
65b7edf8a4da	2	27678	HSC	citeseq
c1b26cb1057b	2	27678	EryP	citeseq
917168fa6f83	2	27678	NeuP	citeseq
2b29feeca86d	2	27678	EryP	citeseq
...	...	...	...	...
96a60b026659	10	31800	hidden	multiome
d493e546991e	10	31800	hidden	multiome
05666c99aa48	10	31800	hidden	multiome
121f946642b5	10	31800	hidden	multiome
b847ba21f59f	10	31800	hidden	multiome

281528 rows × 4 columns

cell_id	day	donor	cell_type	technology
c2150f55becb	2	27678	HSC	citeseq
65b7edf8a4da	2	27678	HSC	citeseq
c1b26cb1057b	2	27678	EryP	citeseq
917168fa6f83	2	27678	NeuP	citeseq
2b29feeca86d	2	27678	EryP	citeseq
...	...	...	...	...
96a60b026659	10	31800	hidden	multiome
d493e546991e	10	31800	hidden	multiome
05666c99aa48	10	31800	hidden	multiome
121f946642b5	10	31800	hidden	multiome
b847ba21f59f	10	31800	hidden	multiome

281528 rows × 4 columns

The metadata table

The metadata table (which describes training and test data) shows us:

- There is data about 281528 unique cells.
- The cells belong to five days, four donors, eight cell types (including one type named 'hidden'), and two technologies.
- The metadata table has no missing values.

Insight:

- Every cell is used only on a single day and then discarded. There are no time series over single cells.
- The two technologies do not share cells. It looks like we may create two completely independent models, one per technology, even if they share the same four donors. It's two Kaggle competitions in one!
- As the models are independent, it is a good idea to work with two separate notebooks, one for CITEseq, the other one for Multiome.
- Donor and cell_type are categorical features, which can be one-hot encoded.

by @AmbrosM

File and Field Descriptions

Multiome

- **train/test_multi_inputs.h5** - ATAC-seq peak counts transformed with TF-IDF using the default `log(TF) * log(IDF)` output (chromatin accessibility), with rows corresponding to cells and columns corresponding to the location of the genome whose level of accessibility is measured, here identified by the genomic coordinates on reference genome GRCh38 provided in the 10x References - 2020-A (July 7, 2020).

Column Examples and Widths

```
array(['GL000194.1:114519-115365', 'GL000194.1:55758-56597',
       'GL000194.1:58217-58957', ..., 'chrY:7836768-7837671',
       'chrY:7869454-7870371', 'chrY:7873814-7874709'], dtype=object)
```

width: 228942

- **train_multi_targets.h5** - RNA gene expression levels as library-size normalized and log1p transformed counts for the same cells.

```
array(['ENSG00000121410', 'ENSG00000268895', 'ENSG00000175899', ...,
       'ENSG00000162378', 'ENSG00000159840', 'ENSG00000074755'],
      dtype=object)
```

width: 23418

train length: 105942

CITEseq

- **train/test_cite_inputs.h5** - RNA library-size normalized and log1p transformed counts (gene expression levels), with rows corresponding to cells and columns corresponding to genes given by `{gene_name}_{gene_ensemble-ids}`.

```
array(['ENSG00000121410_A1BG', 'ENSG00000268895_A1BG-AS1',
       'ENSG00000175899_A2M', ..., 'ENSG00000162378_ZYG11B',
       'ENSG00000159840_ZYX', 'ENSG00000074755_ZZEF1'], dtype=object)
```

width: 22050

- **train_cite_targets.h5** - Surface protein levels for the same cells that have been dsb normalized.

```
array(['CD86', 'CD274', 'CD270', 'CD155', 'CD112', 'CD47', 'CD48', 'CD40',
       'CD154', 'CD52', 'CD3', 'CD8', 'CD56', 'CD19', 'CD33'],
      dtype=object)
```

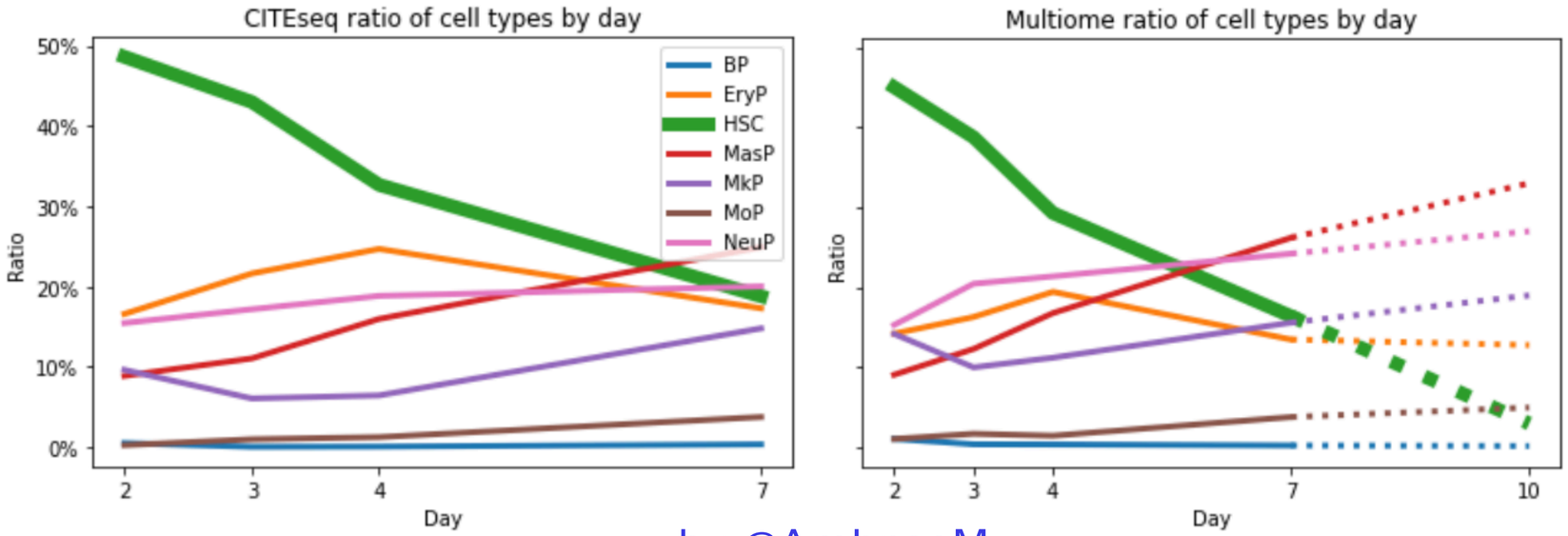
width: 140

train length: 70898

Cell Types

To help guide your analysis, we performed a preliminary cell type annotation based on the RNA gene expression using information from the following paper: https://www.nature.com/articles/ncb3493. Note, cell type annotation is an imprecise art, and the concept of assigning discrete labels to continuous data has inherent limitations. You do not need to use these labels in your predictions; they are primarily provided to guide exploratory analysis. In the data, there are the following cell types:

- *MasP* = Mast Cell Progenitor
- *MkP* = Megakaryocyte Progenitor
- *NeuP* = Neutrophil Progenitor
- *MoP* = Monocyte Progenitor
- *EryP* = Erythrocyte Progenitor
- *HSC* = Hematoploetic Stem Cell
- *BP* = B-Cell Progenitor



by @AmbrosM

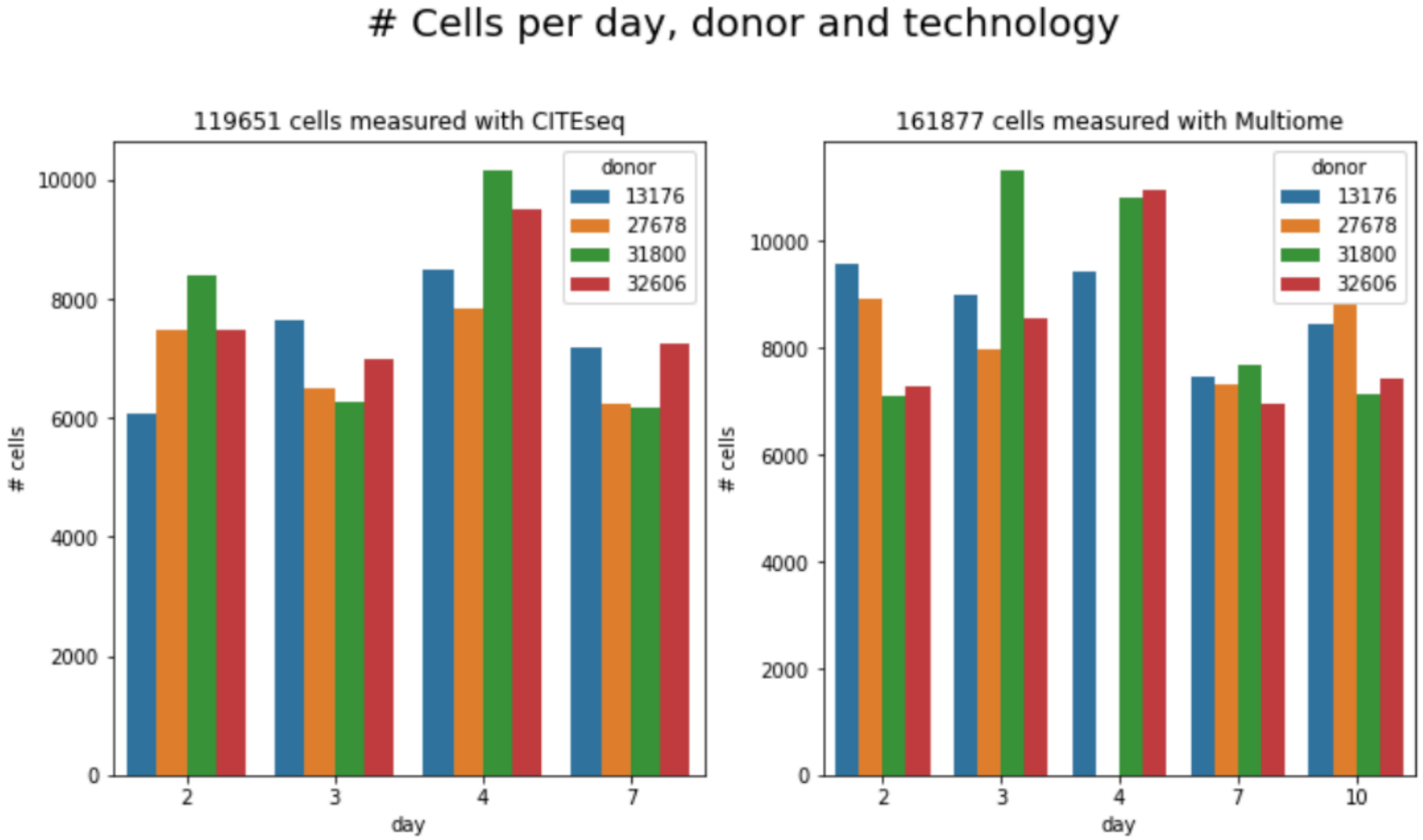
Exploratory Data Analysis (EDA)

by @AmbrosM with presenter’s comments in blue

Metadata

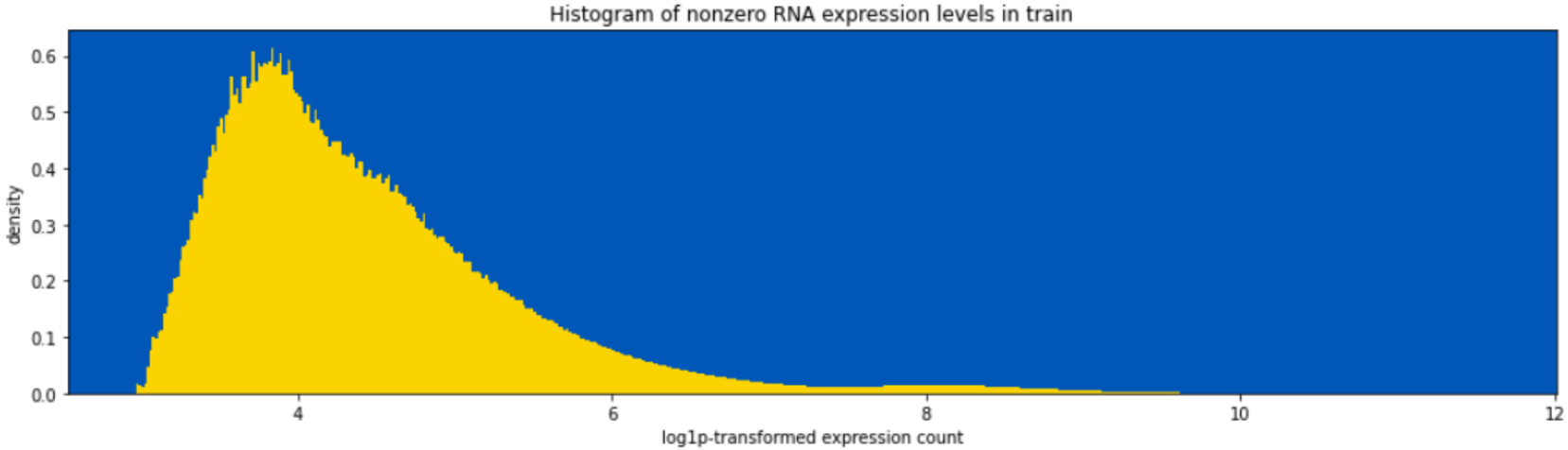

Metadata distribution
day
2
3
4
7
10
0
10000
20000
30000
40000
50000
60000
70000
cells
donor
13176
27678
31800
32606
0
10000
20000
30000
40000
50000
60000
70000
cells
cell_type
BP
EryP
HSC
MasP
MkP
MoP
NeuP
hidden
0
10000
20000
30000
40000
50000
60000
70000
80000
cells
technology
citeseq
multiome
0
20000
40000
60000
80000
100000
120000
140000
160000
cells

Metadata

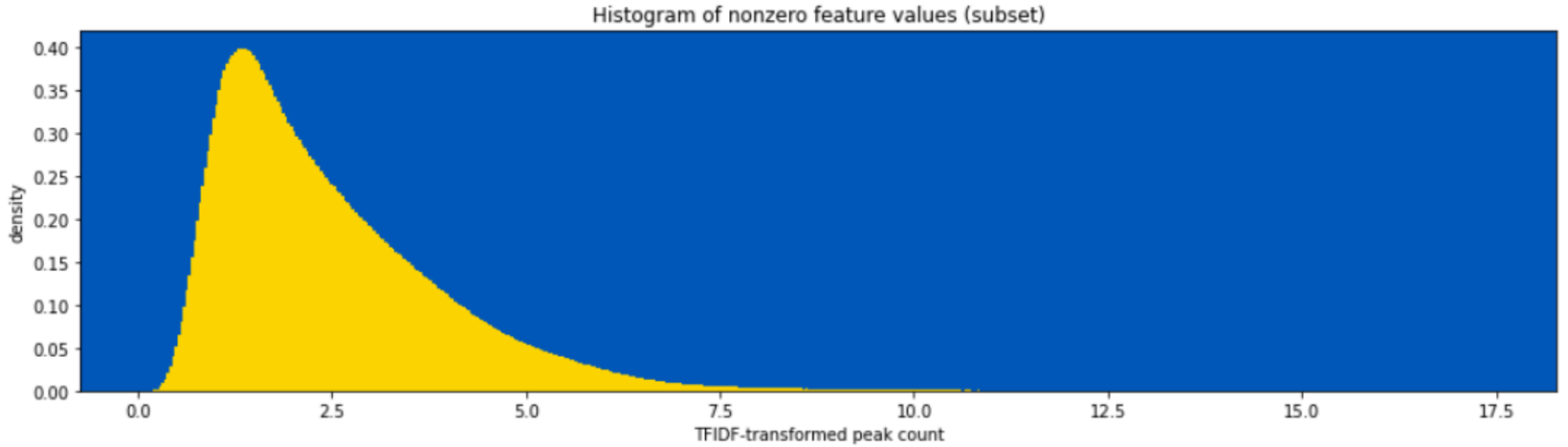

Cells per day, donor and technology
119651 cells measured with CITEseq
161877 cells measured with Multiome
cells
day
donor
13176
27678
31800
32606
0
2000
4000
6000
8000
10000
2
3
4
7
10

Non-zero entries

Multiome

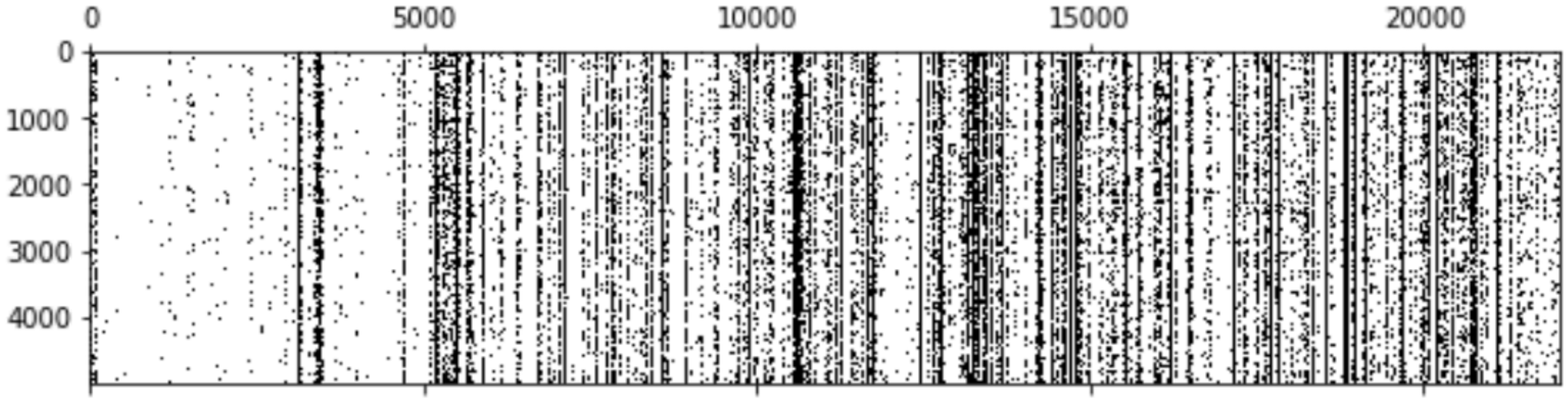


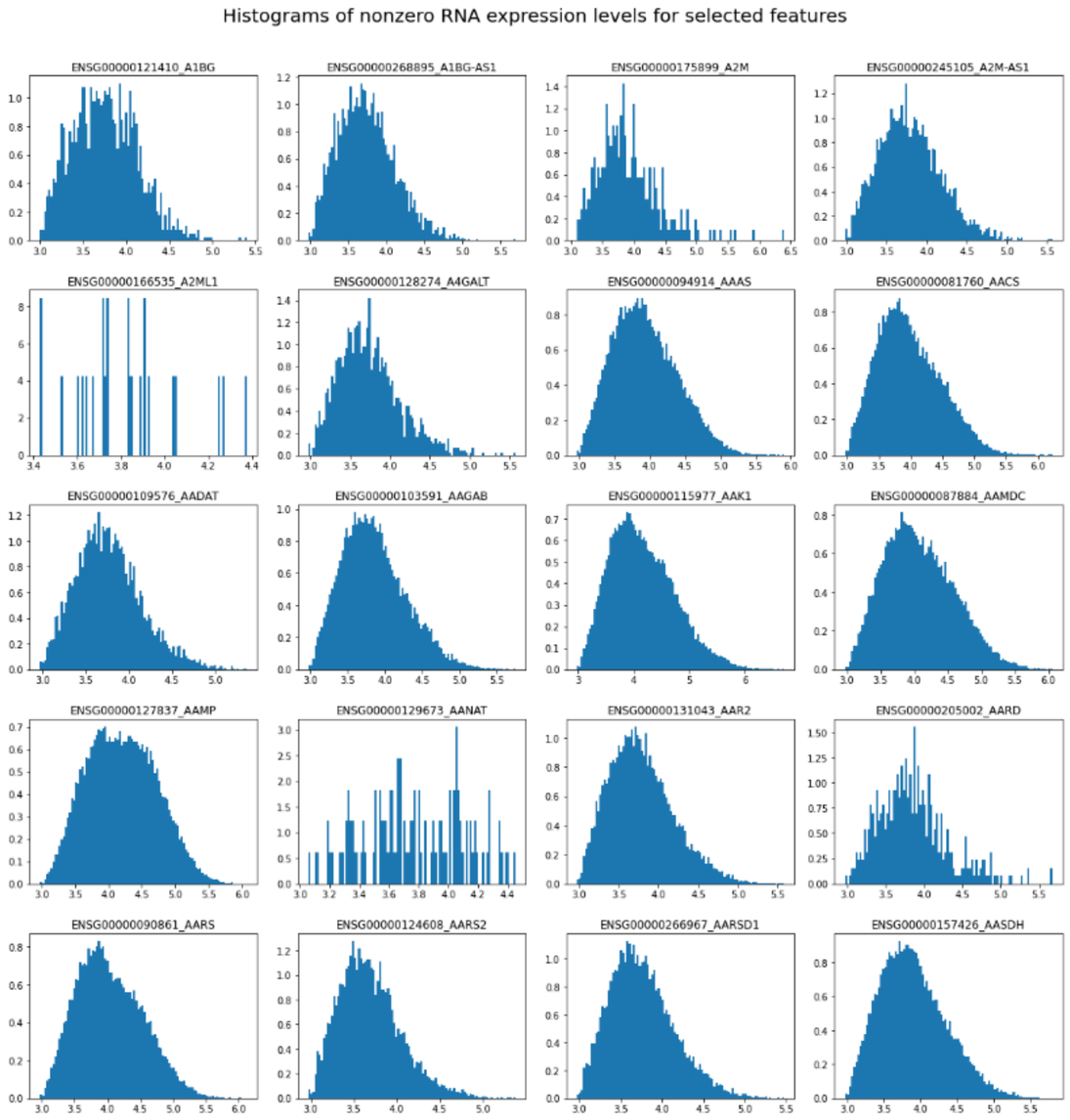
CITEseq



CITEseq input - Non-zero entries

```
plt.figure(figsize=(10, 4))
plt.spy(df_cite_train_x[:5000])
plt.show()
```

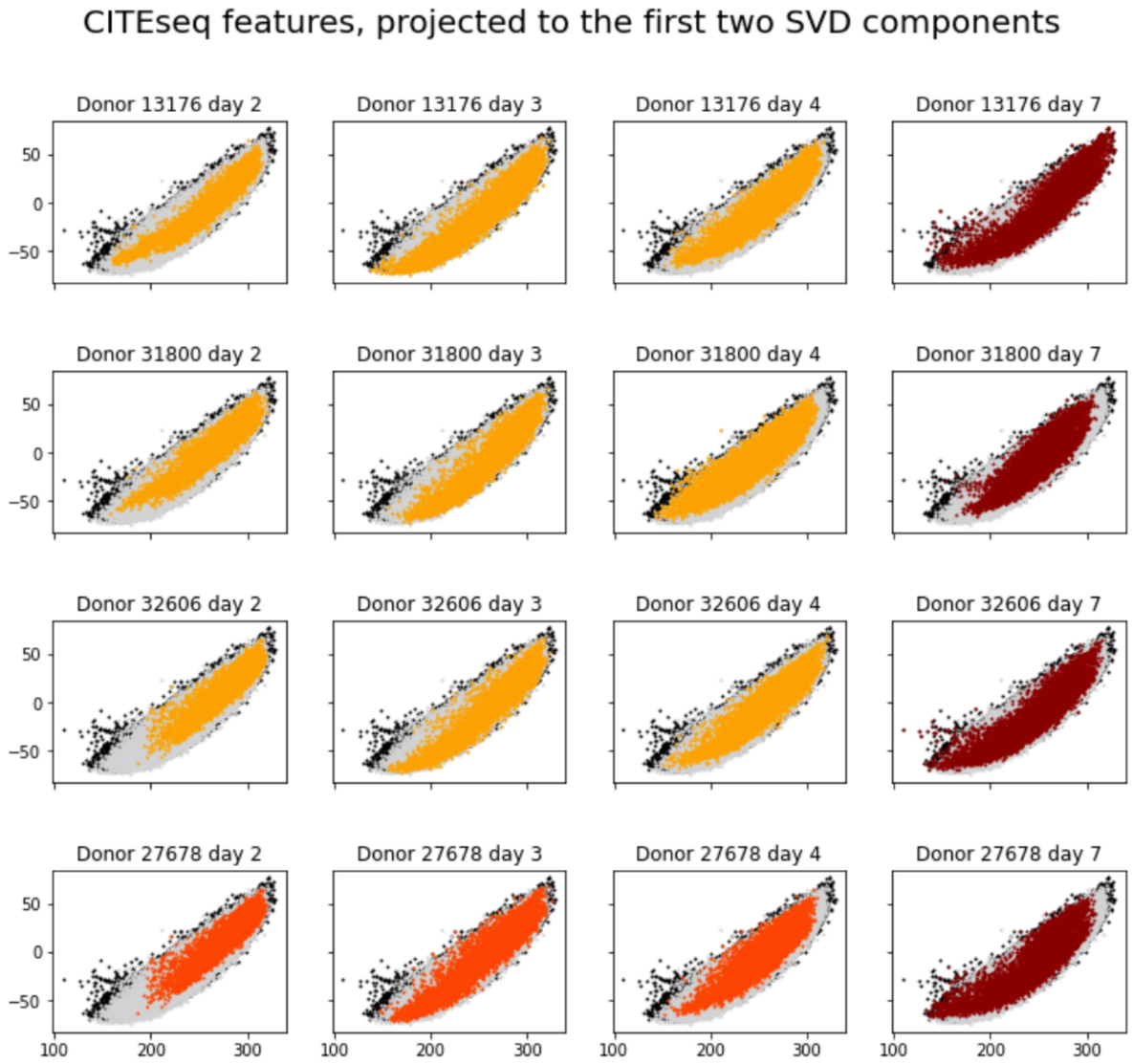




CITEseq input

Example RNA expression profiles

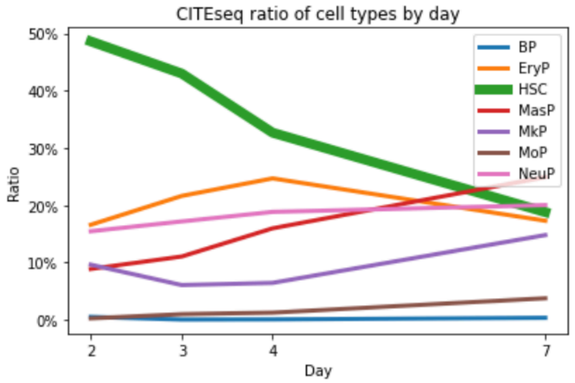
- All have non-zero minimum value apart from mostly zeroes.
- All are positive.
- Some are very sparse.
- Does zero mean ‘zero’ or ‘unknown’?
  -> turns out that leaving it as zero rather than replacing it with the mean improves performance.

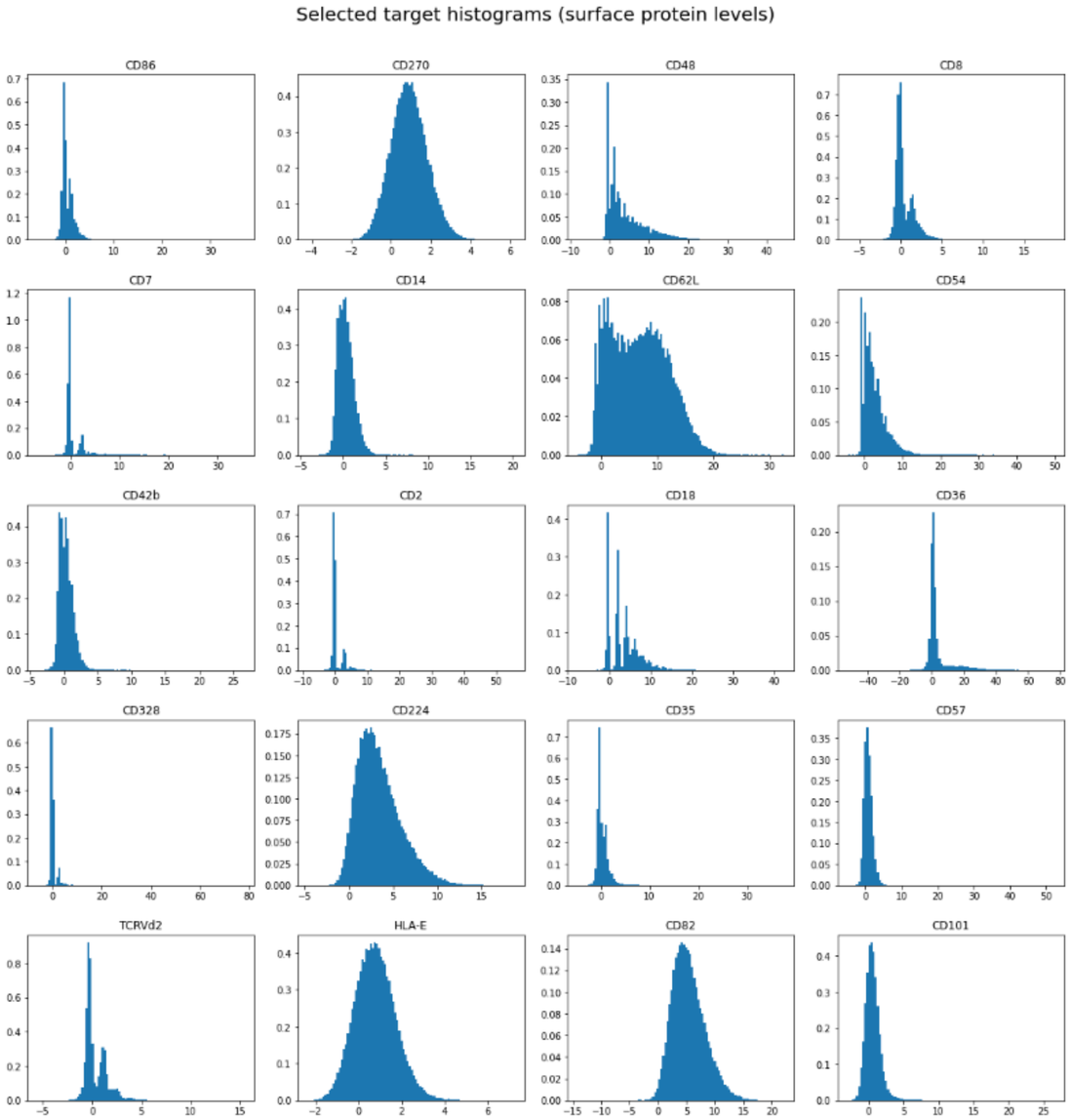


CITEseq input: SVD profiles

Day and Donor Dimensions

- 3rd and 4th donor, day 2 are the same (mistake by Cellarity)
- Plenty of variation among donors as well as days.
- Day 7 looks more ‘filled-out’ than other days, suggesting more diversity in cell type.

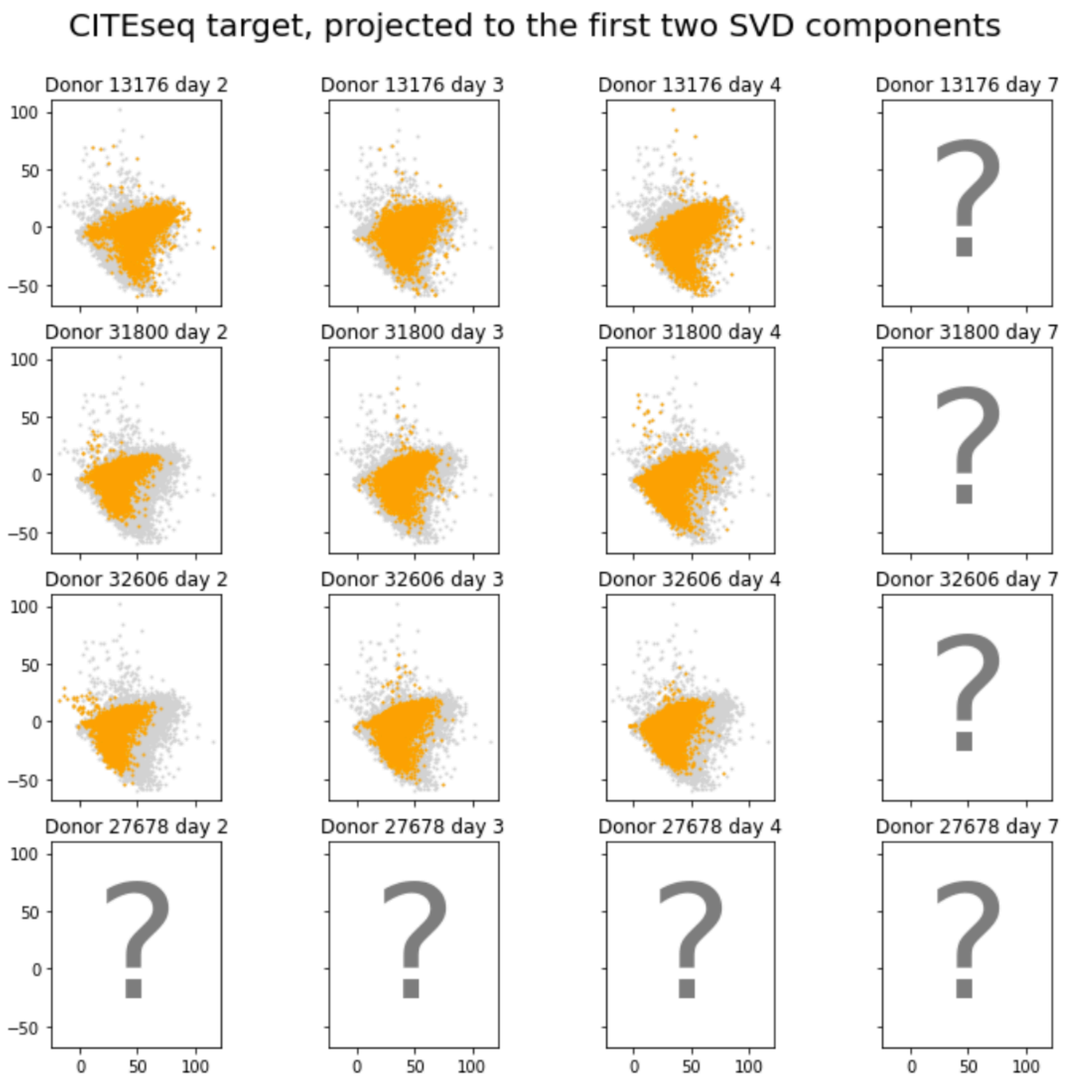




CITEseq target

Example protein concentration profiles

- No zero values, some negative values.
- Variety of distributions - normal, bimodal, discrete. Outliers are present.
- Remember the final metric is the Pearson correlation of predictions to true values for each cell (row standardized).



CITEseq target: SVD profiles

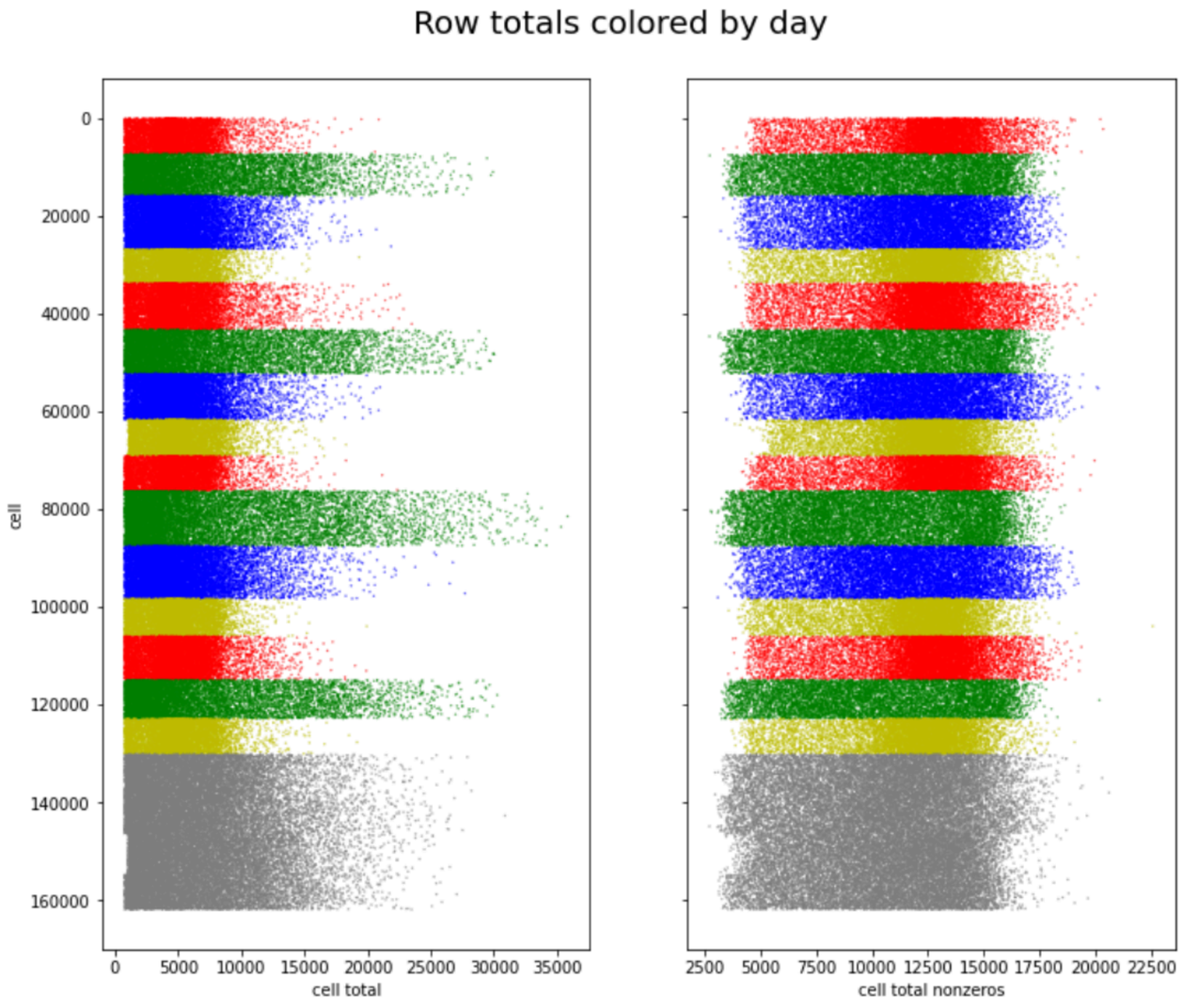
Day and Donor Dimensions

- Again plenty of variation among donors as well as days.

	Gene	Protein
0	ENSG00000114013_CD86	CD86
1	ENSG00000120217_CD274	CD274
2	ENSG00000196776_CD47	CD47
3	ENSG00000117091_CD48	CD48
4	ENSG00000101017_CD40	CD40
...	...	...
146	ENSG00000102181_CD99L2	CD9
147	ENSG00000223773_CD99P1	CD9
148	ENSG00000204592_HLA-E	HLA-E
149	ENSG00000085117_CD82	CD82
150	ENSG00000134256_CD101	CD101

CITEseq: RNA-protein matches
Selecting Specific RNAs

- 151 genes carry the name of one of the proteins as suffix
- Can assume that they are especially important.
- In practice, only 84 of these need to be included. @pourchot



Multiome input

Batch effects

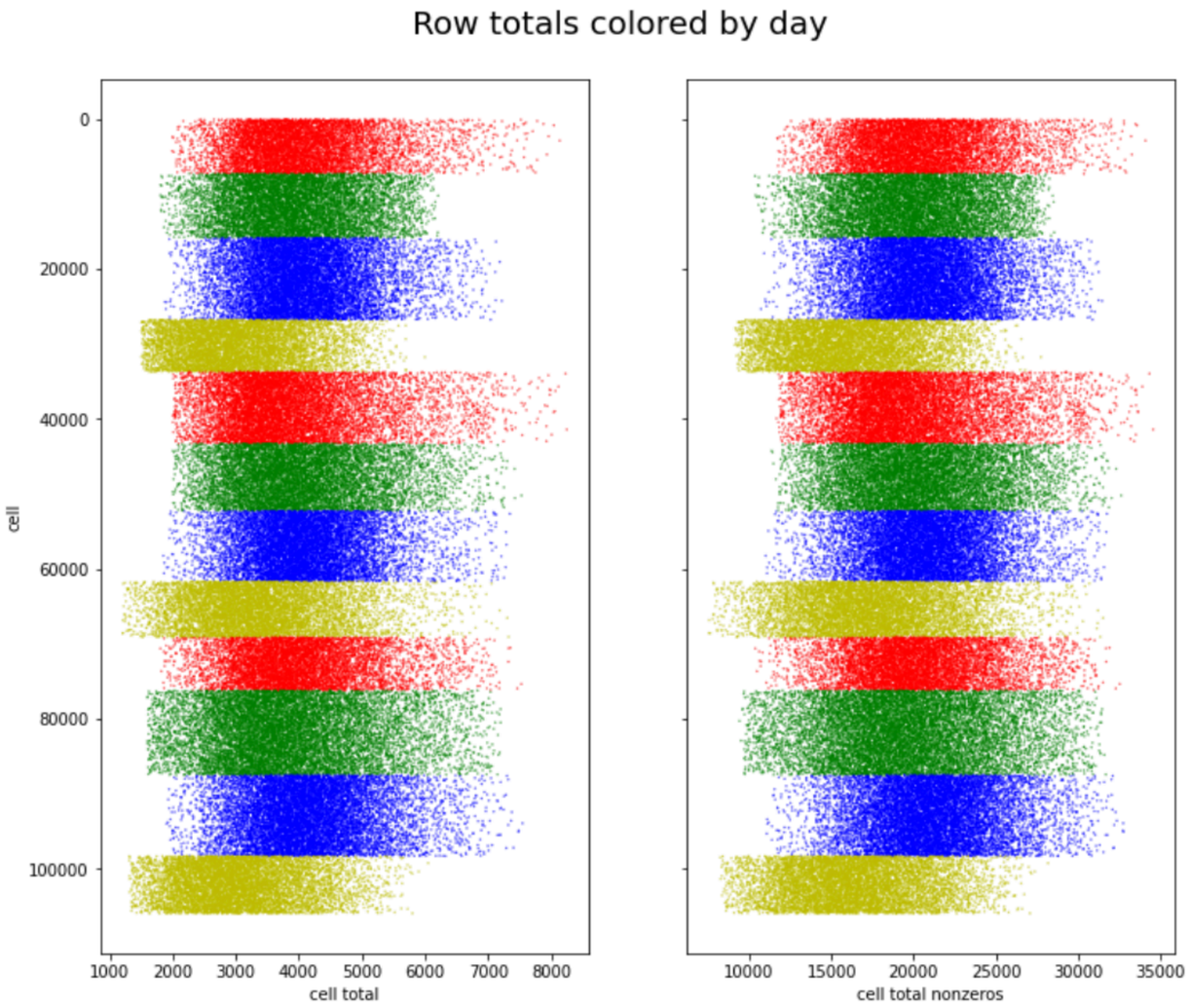
Biological experiments are notorious for batch effects: If you repeat the same experiment several times, you get systematic differences in the measurements. We can show these batch effects for the ATACseq measurements. Because of the huge data size, we look only at the row sums of the data (i.e. the sum of all measured values of a cell). In the diagram, we plot these sums for every row of the train and test datasets, and we color the dots by the day of the experiment (day 2 is red, day 3 is green and so on).

The diagram clearly shows that day 3 (green) gave the highest measurements, day 7 (yellow) the lowest. There are differences among donors as well, albeit smaller ones.

Insight:

- We should try to normalize the data in a preprocessing step so that the batch effects don't affect the predictions.
- Because the batch effects depend more on the day than on the donor, we should use a `GroupKFold` on days for cross-validation to validate that the differences between the days don't break the model.

In practice, I ‘gave up’ and built one new model for each day.

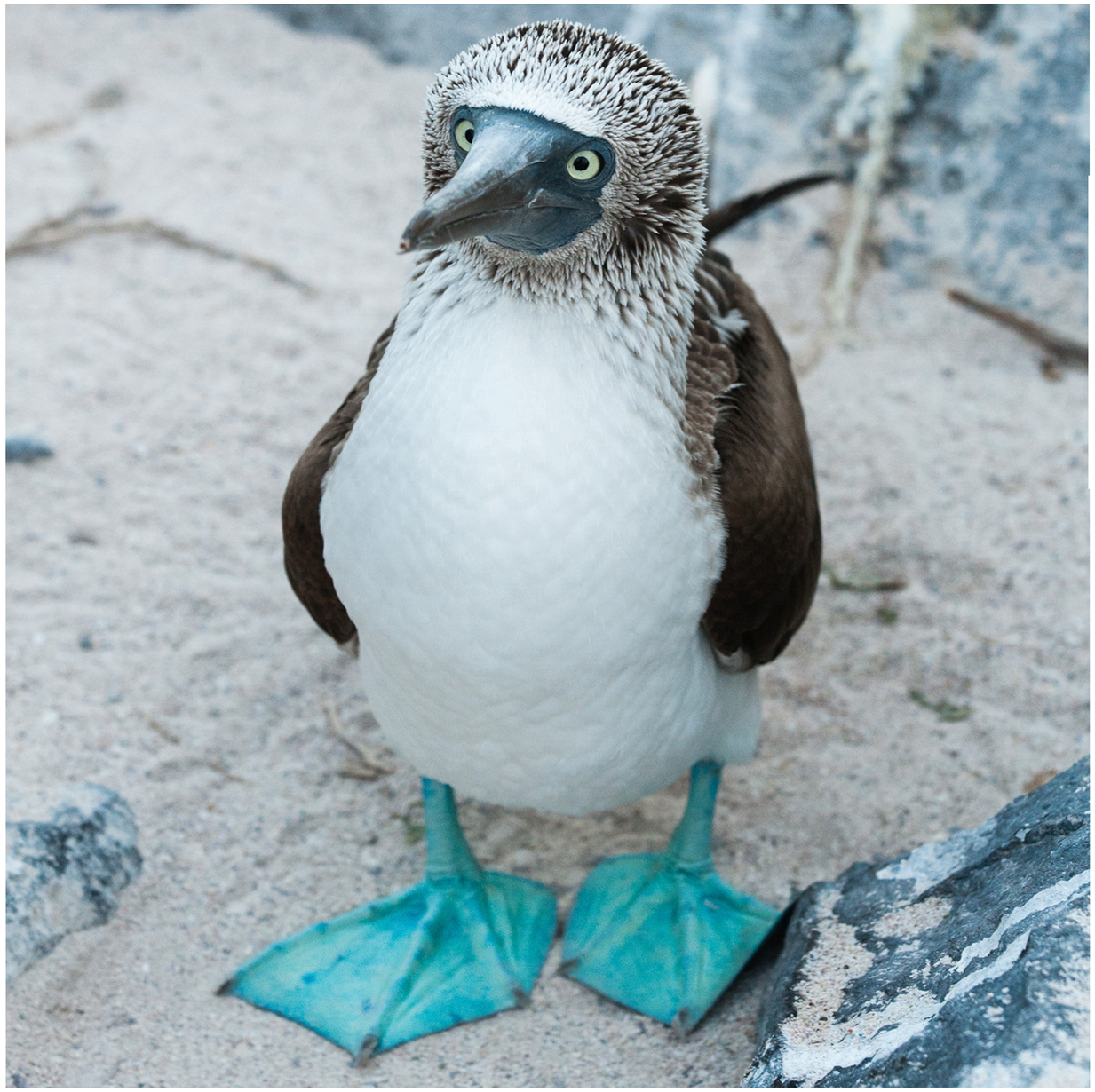


Multiome target

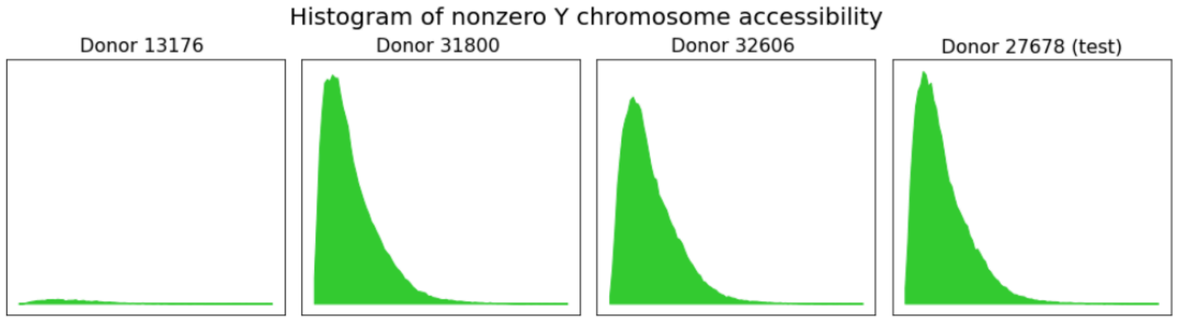
Above, we saw the batch effects in the ATACseq data; below we see that the gene expression dataset has its own, independent, batch effects: For the gene expression data, day 7 (yellow) clearly has the lowest totals. Day 3 (green), which deviated the most from the other days for the ATACseq data, is not far from the mean for the gene expressions. But again, differences among days seem to exceed differences among donors.

The good news here is that the competition metric does not depend on shifts in the target values: Even if for day 7 (and maybe for day 10) all measurements are shifted by a constant, the predictions can still achieve a good correlation with the true values. With a mean-squared-error metric, this would be different.

But we clearly see that
there is something strange
about day 7



Y Chromosome



The histograms of the Y chromosomes illustrate the diversity of the donors: Donor 13176 seems to have (almost) no Y chromosome (maybe the few nonzero values are measuring errors). It appears that the donors are one woman and three men.

Insight:

- If we assume that the true values of donor 13176 are all zero, these data show us the magnitude of the measurement error.
- Maybe we can create a new (binary) feature for the presence of the Y chromosome.
- The diagram reminds us that the donors are different and that our model should be robust to these differences.

Model Building

by @AmbrosM with presenter’s comments in blue

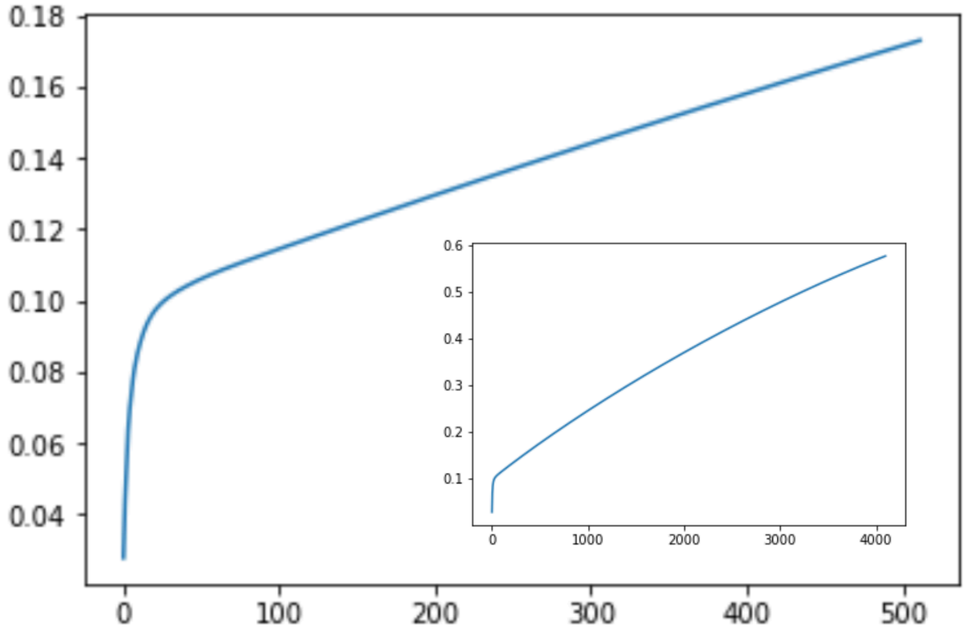
Strategy

Our solution strategy has five elements:

1. **Dimensionality reduction:** To reduce the size of the 10.6 GByte input data, we project the 22050 features to a space with only 64 dimensions by applying a truncated SVD. To these 64 dimensions, we add 144 features whose names shows their importance.
2. **The model:** The model is a sequential dense network with four hidden layers.
3. **The loss function:** The competition is scored by the average Pearson correlation coefficient between the predictions and the ground truth. As this scoring function is differentiable, we can directly use it as loss function for a neural network. This gives neural networks an advantage in comparison to algorithms which use mean squared error as a surrogate loss. coded by @lucasmorin
4. **Hyperparameter tuning with KerasTuner:** We tune the hyperparameters with KerasTuner.
5. **Cross-validation:** Submitting unvalidated models and relying only on the public leaderboard is bad practice. The model in this notebook is fully cross-validated with a 3-fold GroupKFold.

PCA motivation for CITEseq data

- There is next to no information contained beyond ~64 PCA components
- Cumulative Explained Variance vs. # of PCA components



Need almost as many PCA components

as # of different RNAs considered (23000)

Strategy	Correlation
Mean	71.8
4 PCA	85.8
8 PCA	87.7
16 PCA	88.1
32 PCA	88.6
64 PCA	88.8
512 PCA	88.8
256 PCA Magic	88.2
256 PCA +	89.0

by presenter

Model building

We read train and test datasets, keep the important columns and convert the rest to sparse matrices. by @fabiencrom

We now define two sets of features:

- `constant_cols` is the set of all features which are constant in the train or test datset. These columns will be discarded immediately after loading.
- `important_cols` is the set of all features whose name matches the name of a target protein. If a gene is named 'ENSG00000114013_CD86', it should be related to a protein named 'CD86'. These features will be used for the model unchanged, that is, they don't undergo dimensionality reduction.

We apply the truncated SVD to train and test together. The truncated SVD is memory-efficient. We concatenate the SVD output (64 components) with the 144 important features and get the arrays `X` and `Xt`, which will be the input to the Keras model.

SVD is PCA that can use sparse matrices

Finally, we read the target array `Y`:

multiome Y data needs to be standardized
because we use 'mse'

The model

Our model is a sequential network consisting of a few dense layers. The hyperparameters will be tuned with KerasTuner.

We use the `negative_correlation_loss` defined above as loss function.

The hidden layers have sizes (256,512,512).
We use 0.1 drop-out in-between each layer.
We apply L2 regularization.

Tuning with KerasTuner

Now we let KerasTuner optimize the hyperparameters. The tunable hyperparameters are:

- the sizes of the hidden layers
- the regularization factors
  also dropout

model implementation by @pourchot (base model by @ambrosM)

Model evaluation - CITEseq

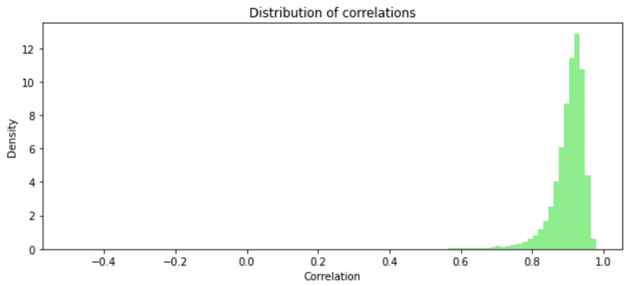
Cross-validation

For cross-validation of the tuned model, we create three folds. In every fold, we train on the data of two days and predict the third one. This scheme mimics the situation of the private leaderboard, where we train on three days and predict the fourth one (see EDA).

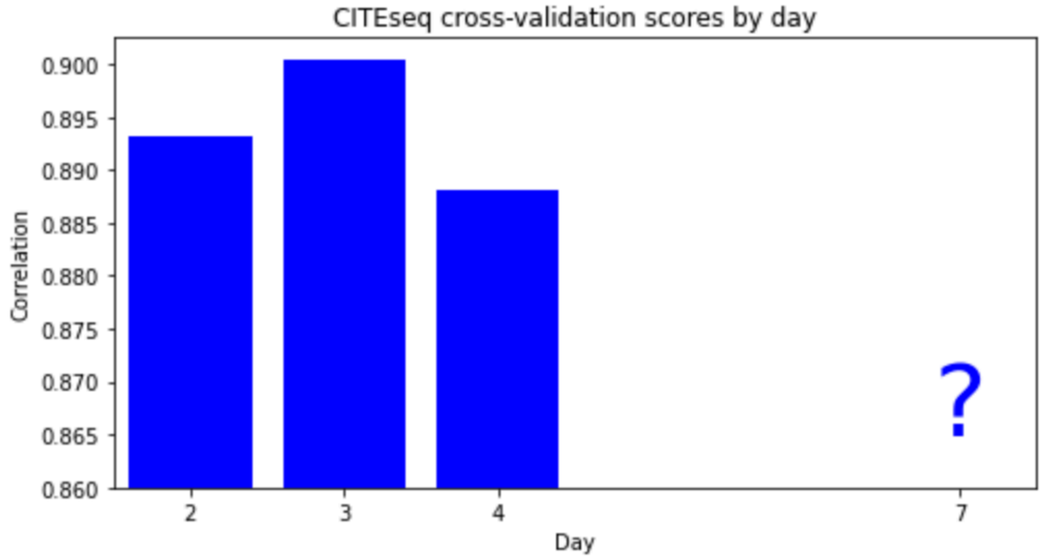
The models are saved so that we can use them to compute the test predictions later.

```
kf = GroupKFold(n_splits=N_SPLITS)
score_list = []
for fold, (idx_tr, idx_va) in enumerate(kf.split(X, groups=met
a.day)):
```

```
Fold 0 day 4:  47 epochs, corr =  0.88816
Fold 1 day 2:  54 epochs, corr =  0.89320
Fold 2 day 3:  53 epochs, corr =  0.90046
Average  corr = 0.89394
CPU times: user 15min 20s, sys: 1min 2s, total: 16min 23s
Wall time: 7min 50s
```



If we compare the cv scores for the days, we see that validation on day 3 (with training on days 2 and 4) gives the highest correlation, maybe because this is the only train-test split which doesn't lead to an extrapolation. Day 4 has the lowest correlation, perhaps because of the diversity of the cell types and because we're extrapolating into the future. We can guess that the score for day 7 (private leaderboard) will be even lower.

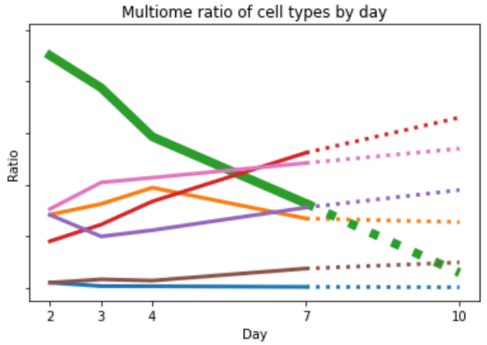


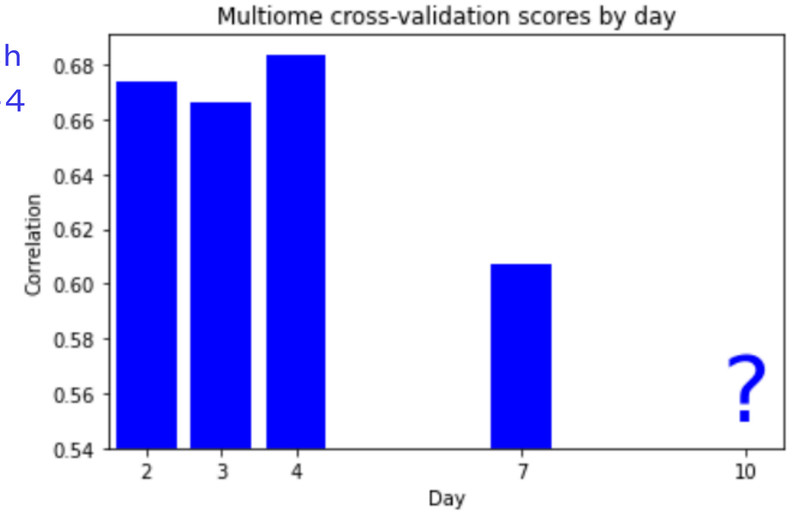
Model evaluation - Multiome

- Can predict 128 PCA components of RNA data (gene expression) with DNA data (gene availability) with 98% correlation:

```
973/973 [==============================] - 2s 2ms/step

 --------- FOLD 0 -----------
Day 4, correlation =  0.98223
INFO:tensorflow:Assets written to: ./submissions/multi_model_0/assets
model saved
901/901 [==============================] - 2s 2ms/step

 --------- FOLD 1 -----------
Day 3, correlation =  0.98009
INFO:tensorflow:Assets written to: ./submissions/multi_model_1/assets
model saved
1438/1438 [==============================] - 2s 2ms/step

 --------- FOLD 2 -----------
Day 2, correlation =  0.96975
INFO:tensorflow:Assets written to: ./submissions/multi_model_2/assets
model saved
```

Room for improvement: Day 7 is much more difficult to predict than days 2-4





- However, after inverse PCA, only 66% correlation remains

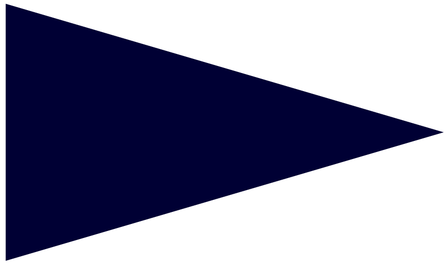
Conclusion: It is just as easy to predict protein levels from gene expression as from gene availability.

Feature Selection

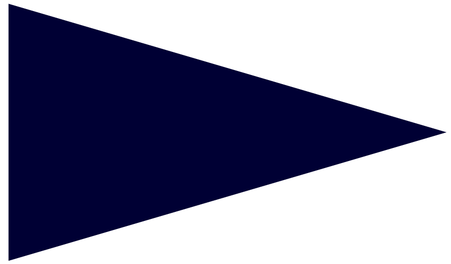
by presenter

Data structure

DNA: 200k+ genes



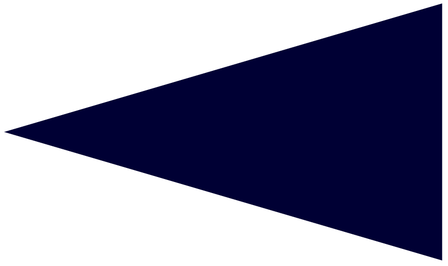
RNA: 20k+ expressions



140 Proteins

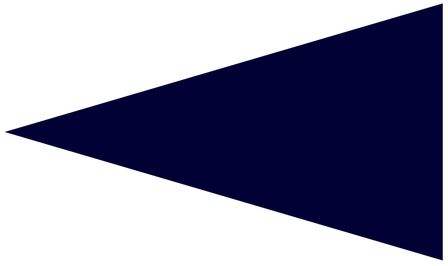
Best Predictive Performance

DNA PCA: 40 components



98% correlation

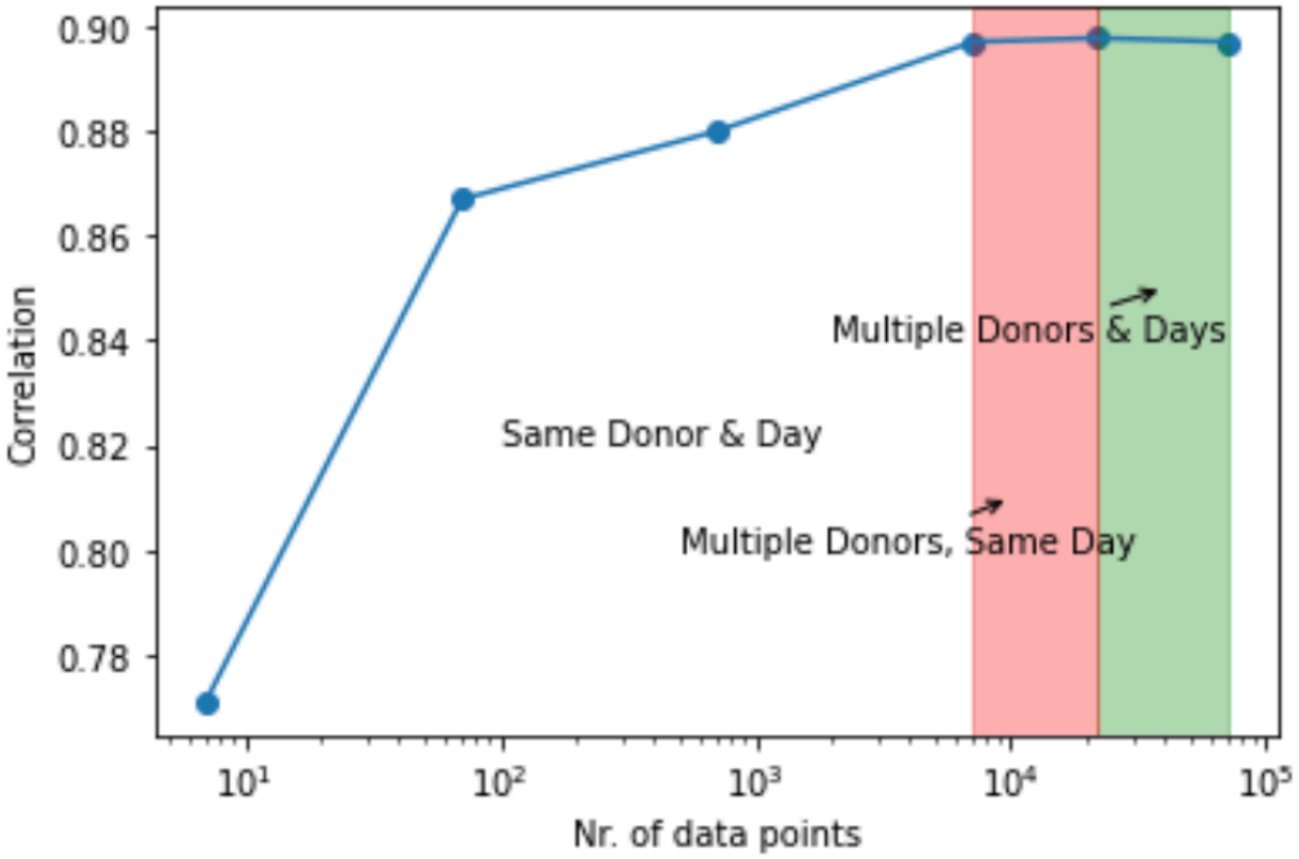
RNA PCA: 64 components



89% correlation

140 Proteins

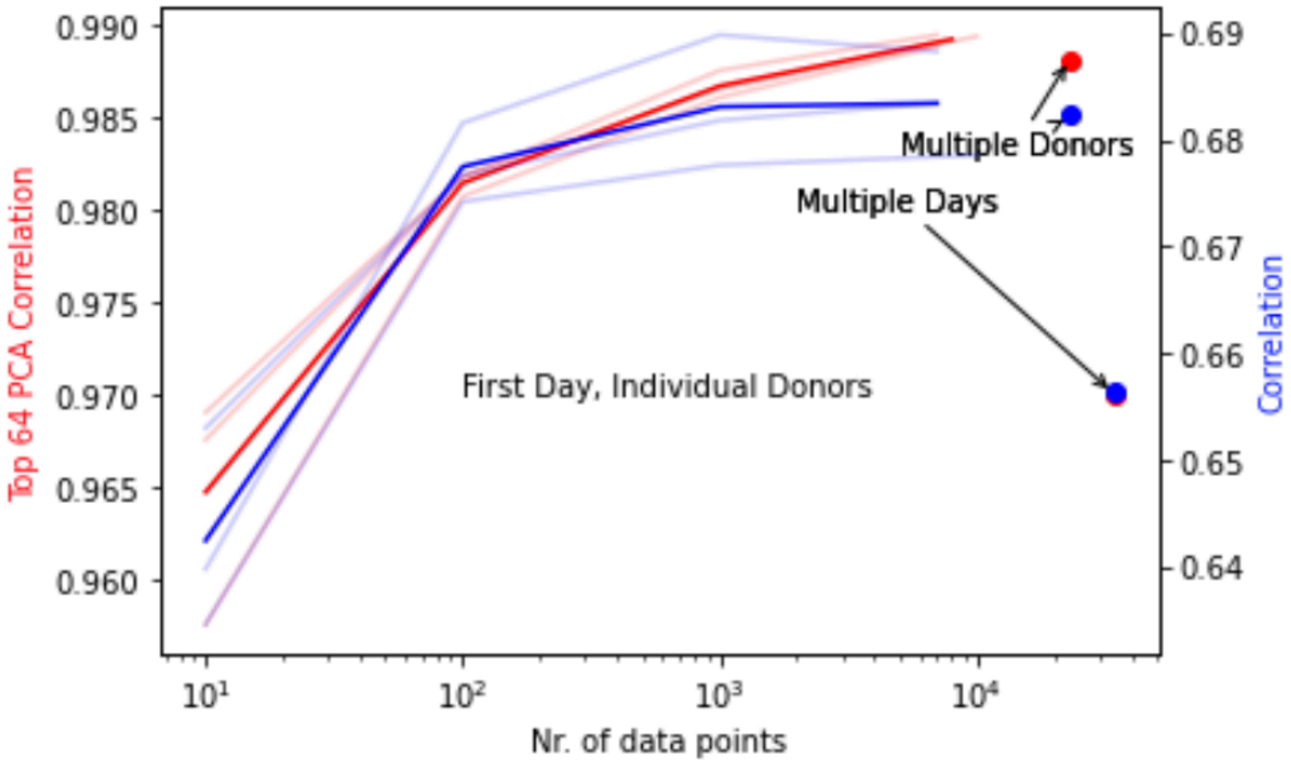
CITEseq data



How much data do we need?

- Performance improves with data all the way to 10k data points.
- If more data from the same day and donor were available, maybe we would have seen further improvement.
- Using data from different donors and days leads to flat-line performance (but perhaps more generalizable to other datasets).

Multiome data



How much data do we need?

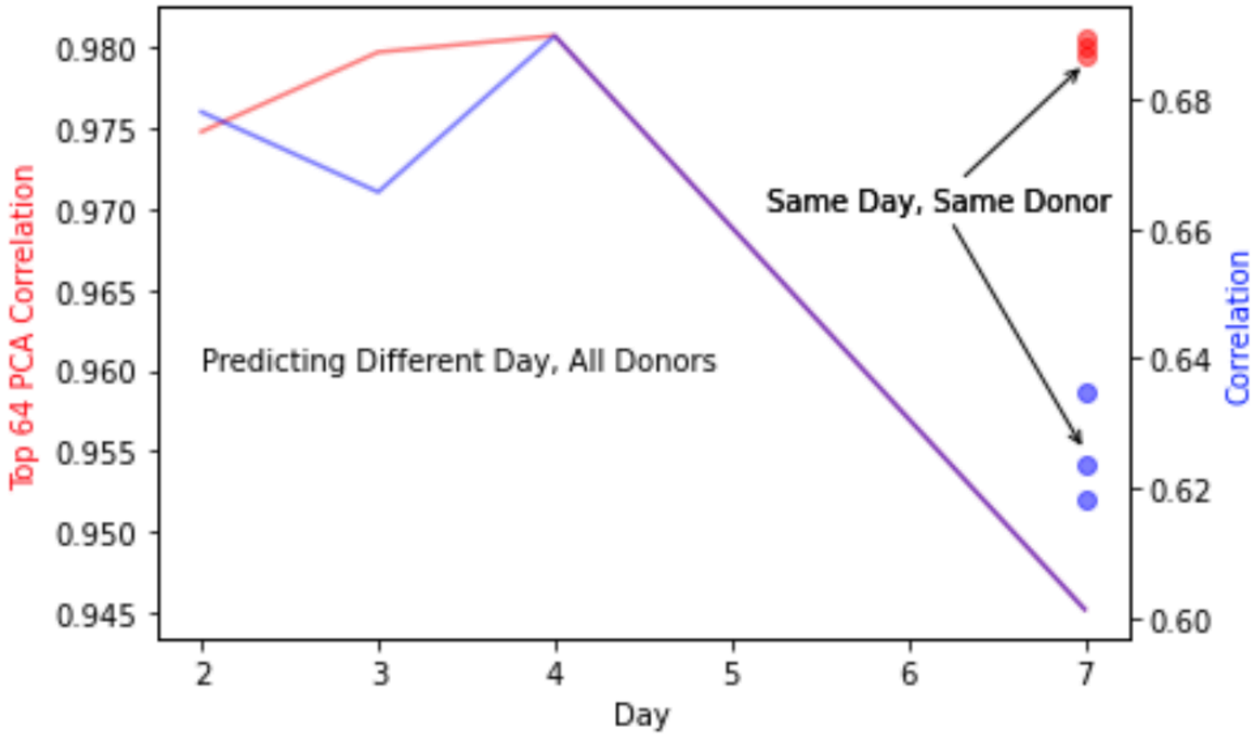
Predicting top 64 PCA vs all gene expressions

- Predicting the top 64 PCA components of RNA expression improves with more data, possibly even past 10k
- Predicting all of RNA expression (23k columns) saturates at 1000 data points, more uncertainty between different donors seen
- Multiome data is robust with respect to predicting different donors but not different days in media

Spotlight: Time Series Data

by presenter

Multiome data

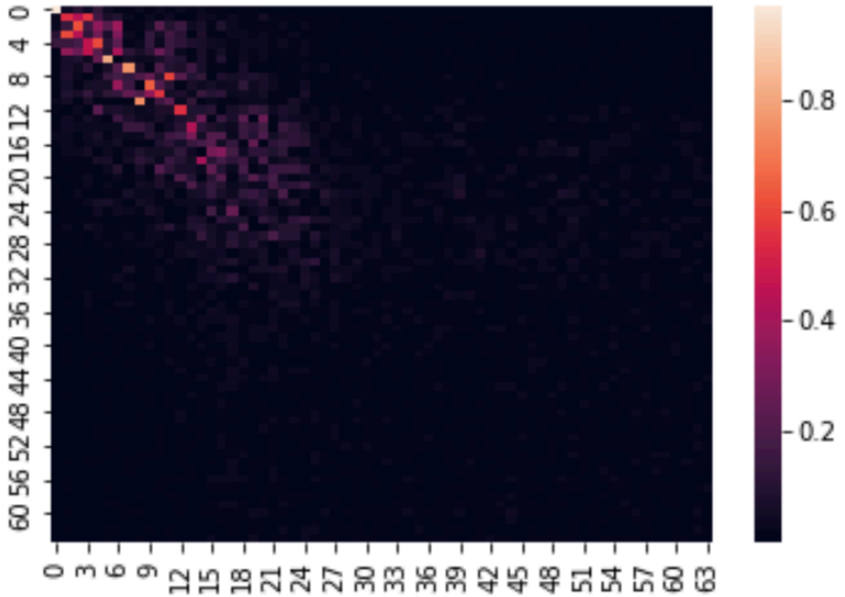


Spotlight: Different Days in Media

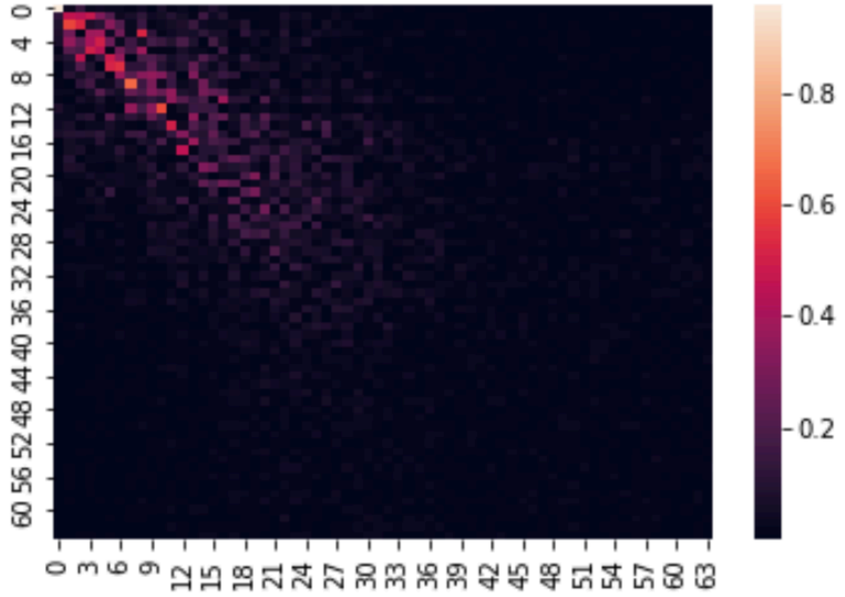
- Data from all other days is used to predict a certain day (GroupKFold).
- Day 7 is much more difficult to predict than 2,3 and 4.
- If we use data from the same day and donor, it cures the performance issue for the PCA prediction, but the overall prediction is only somewhat improved.
- The top 64 PCA components seem to be fundamentally different for day 7.
- More and more noise seems to sneak into the data over time that cannot even be predicted with data from the same day.
- Perhaps proteins are binding to DNA sites over time in media, obscuring the DNA->RNA connection?

Big question: Are the cells ‘spoiling’ in-vitro? Or differentiating?

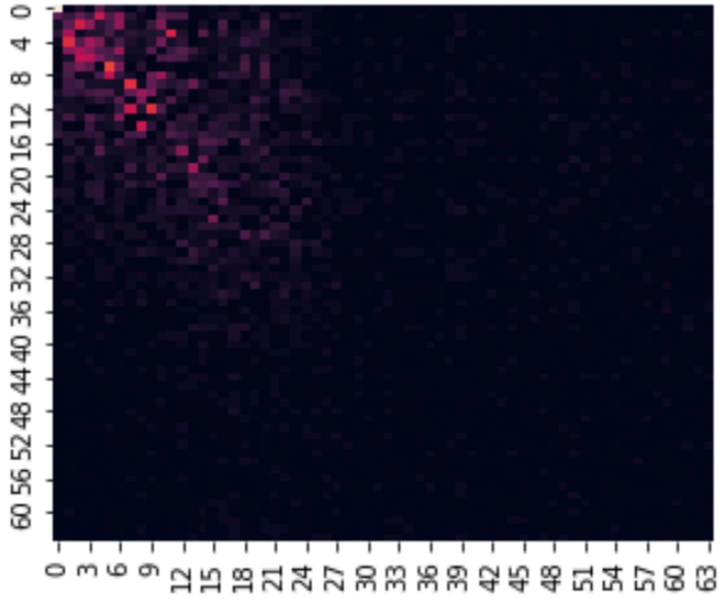
Comparing PCA components by day



i = 4; j = 2



i = 7; j = 4

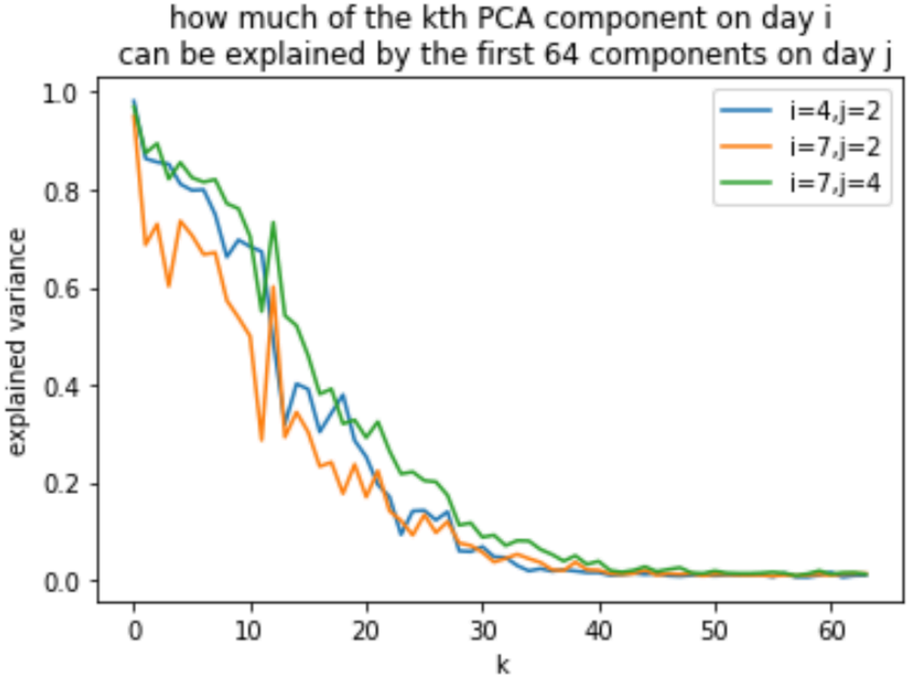


i = 7: i = 2

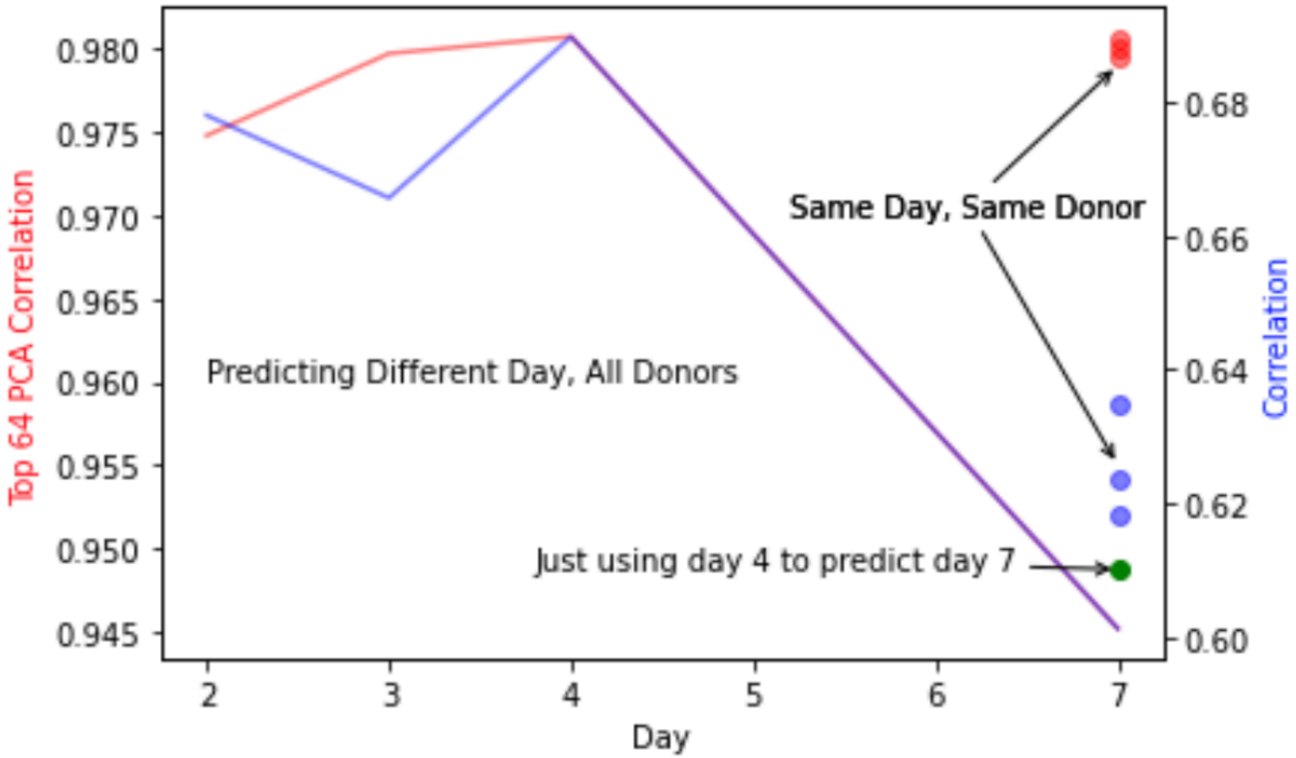
Rows: PCA components on day i

Columns: Amount of day i PCA explained by certain day j PCA

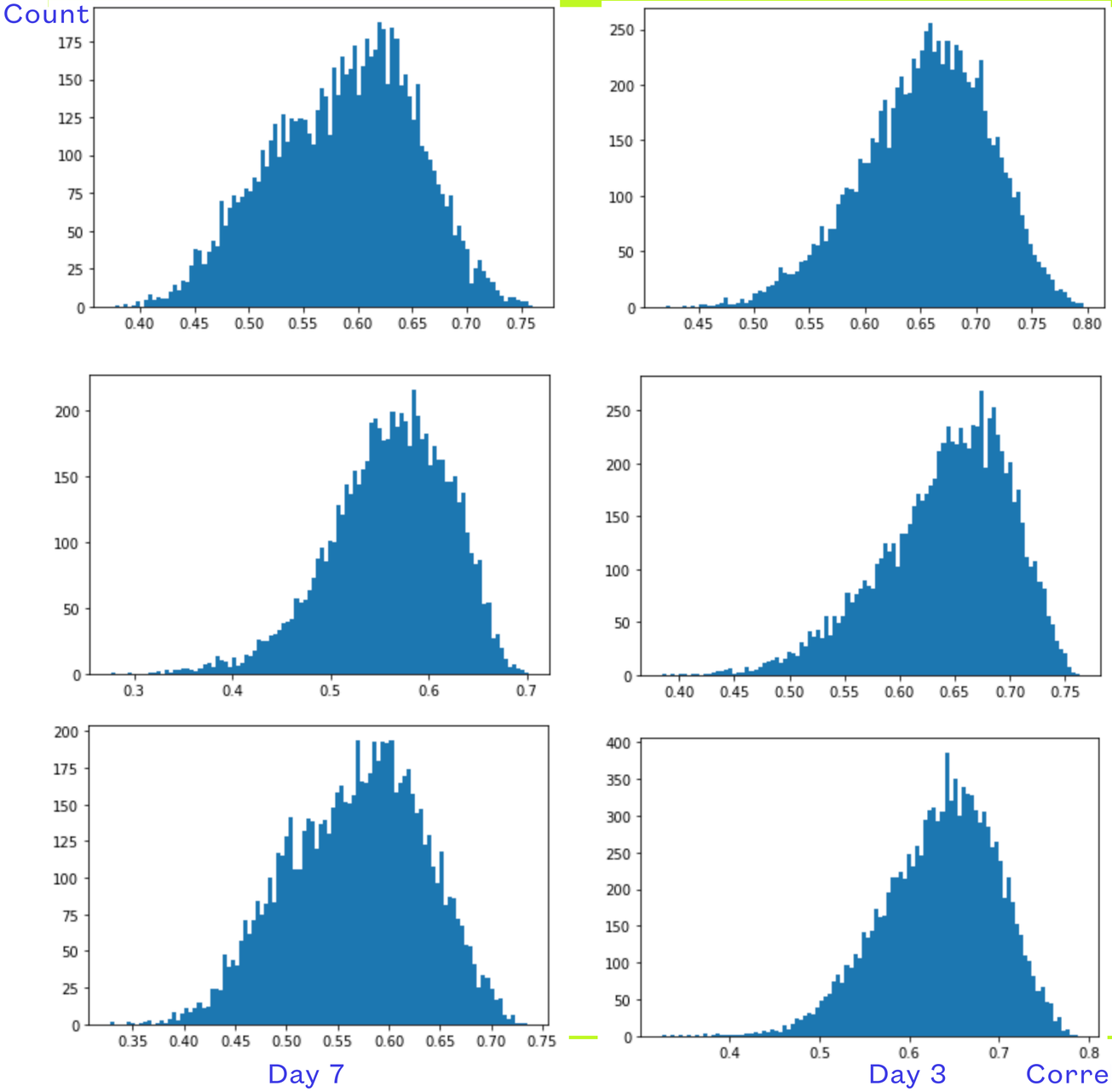
- While the first PCA component is independent of day, we see more diagonal correlations between days 2 and 4 than 7 and 4 or 7 and 2.
- However, overall explained variance is greatest in between days 4 and 7: a mystery given how poor the performance on day 7 is.
- It requires over 8k PCA components on any day to explain over 75% of all 64 PCA components on another day -> having data from the same day should be best.
- However, even though day 7 is not special in its PCA components, how they are expressed must be uniquely different so that it cannot be learned from data of another day.



Using most recent data



- Motivated by the similarity between days 4 and 7, we only use day 4 data to predict day 7 with some success over using all the data from days 2,3 and 4!
- However, using the day 7 PCAs to train on day 4 only further improved performance by 0.15%. Therefore, the difficulty is not learning the correct PCAs, it is that the rules for expressing them are different on day 7.



Trying to predict ‘differentiation’

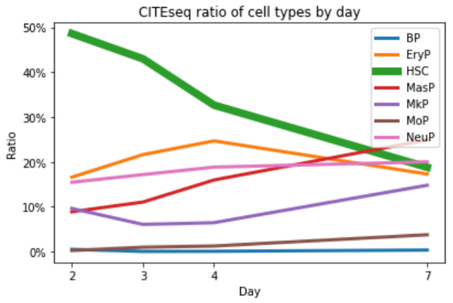
- The idea is to use a day 2 model on day 7 data. If there is a bimodal distribution, then maybe half the cells have started differentiating. Could fit a different NN for non-differentiated and differentiated cells.
- In reality, even though the distribution is fatter on day 7, fitting the worse-performing cells by themselves does not improve performance

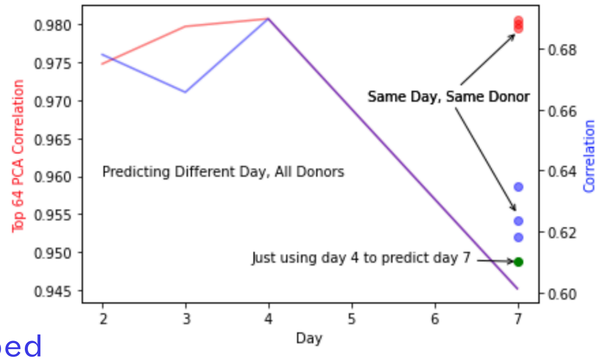
Performance of the 5 clusters predicted by KMeans

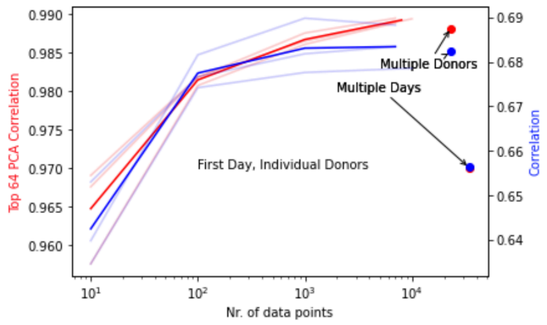
	size	correlation	64 PCA	512 PCA	1024 PCA	2048 PCA	4096 PCA
0	1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
1	2474	0.67578	0.96412	0.82946	0.75209	0.67592	0.66030
2	4989	0.62554	0.96897	0.84374	0.76559	0.68293	0.62151
3	1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
4	1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

	cell count	day 7 corr	day 4 corr
MasP	1751	0.62030	0.68760
NeuP	1733	0.61627	0.68479
MkP	1575	0.68074	0.68854
HSC	1282	0.63953	0.68380
EryP	900	0.66553	0.69479
MoP	224	0.66553	0.69479
BP	1	0.66553	0.69479

Grouped



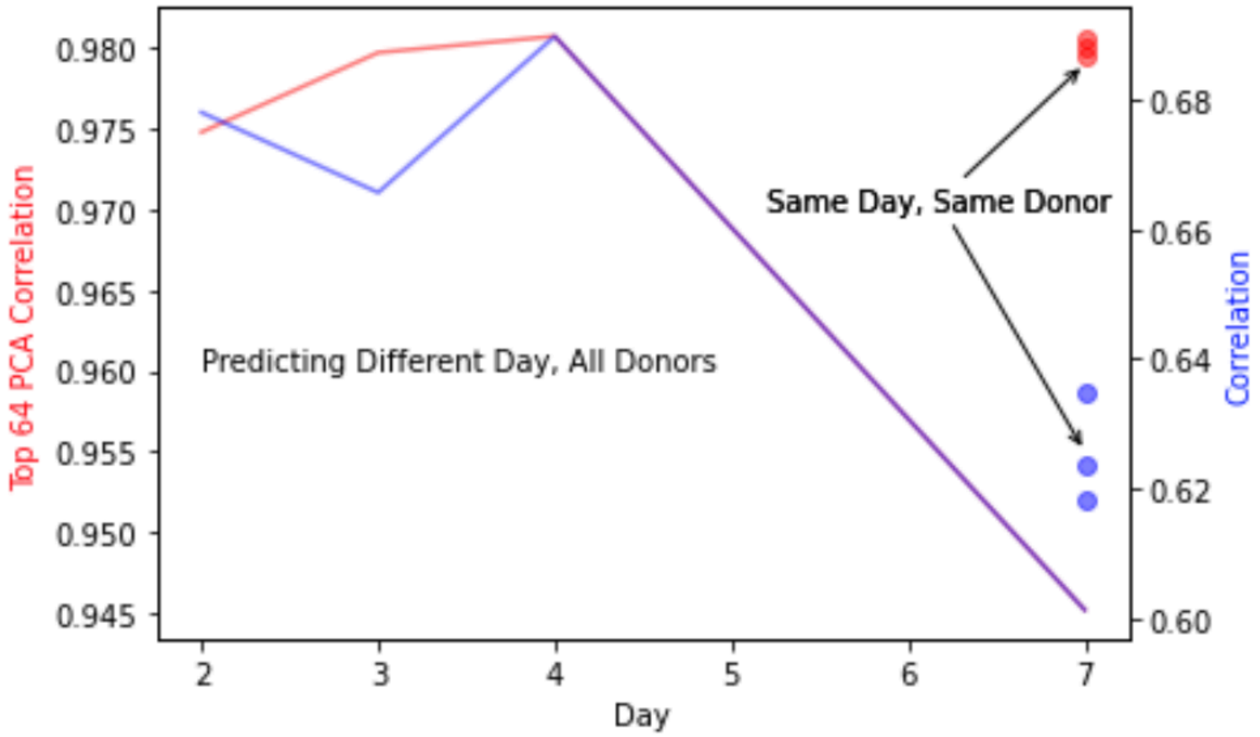




Trying to predict ‘differentiation’

- However, using KMeans on the full 200k column data, we can very clearly see two clusters emerging:
- There is no drop in top 64 PCA correlation, so our model can’t tell the difference between these clusters.
- Somewhere between PCA component 2048 and 4096, day 7 cells have a lot of noise.
- These PCA components don’t have much of an impact on protein production.
- By building different models for each type of cell or by using KMeans, we can improve performance by only 0.035%.
- We can see the issues are mainly with MasP, NeuP and HSC cells.

Multiome data



Strategy

- Approach to winning the competition:
  - 1. Predict rules of day 7 PCA component expression from previous days (reproduce red dots).
  - 2. Fit the PCA>2048 noise (blue dots).
- In practice, very difficult to do. Will be even more difficult on day 10.

Multiome	Day 2	Day 3	Day 4	Day 7	Day 10
Donor 1	Train				Private Test
Donor 2					
Donor 3					
Donor 4	Public Test				

Spotlight Conclusion:

Day 7 cells' protein expression (viz., first 64 components of RNA expression) has changed (maybe due to differentiating!). Here changed means it can be predicted from same-day cells but not other-day cells. Also, we hypothesize that random protein build-up has spoiled DNA-RNA connection for non-essential (PCA > 2048) components.

back to Feature Selection

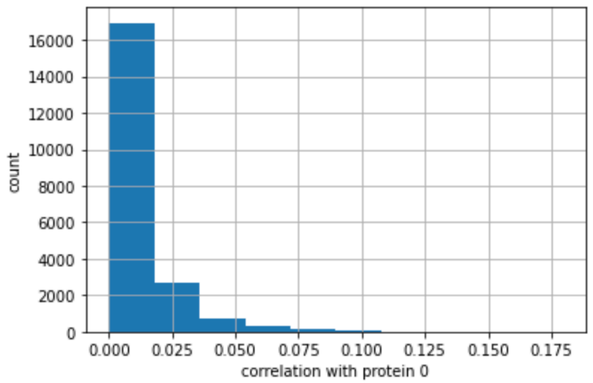
by presenter

Finding correlated features (CITEseq)

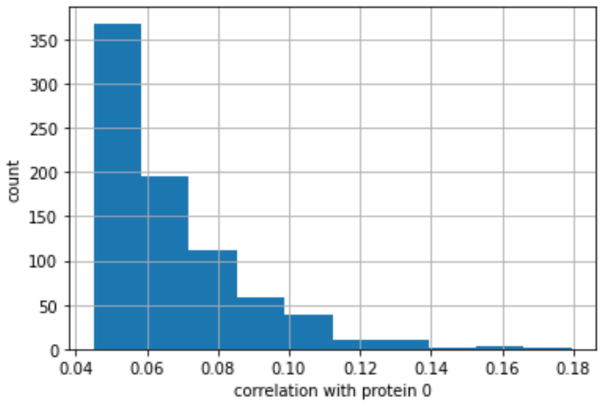
orders by @ahmedelfazouan

```
all_corrs = []
for j in range(140):
    my_corrs = []
    for i in range(X.shape[1]):
        my_corrs.append(np.corrcoef(X[:,i], Y_copy[:,j])[1, 0])
    my_corrs = list(pd.Series(my_corrs).apply(abs)\
                    .sort_values().iloc[-800:].index)
    all_corrs.append(my_corrs)
np.save('./corrs/cite_all_corrs.npy',all_corrs)
```

All genes



Top 800



- Simply appending individual columns to the X matrix and computing correlations
- Picking the top 800 correlated genes for each protein
- Outperforms our best PCA model by a lot for protein 0 using mse

Protein 0 performance

800 most correlated genes MSE PCAs

0.11999999731779099 0.13499999791383743

Optimizing MSE

All Proteins performance

800 most correlated genes per protein MSE 64 Overall PCAs

1.9386615403045362 1.9299999475479126

0.8289484171367852 Correlation 0.8856882177391285

Optimizing Correlation directly

PCAs

0.8943475781025629

No raw protein concentration information

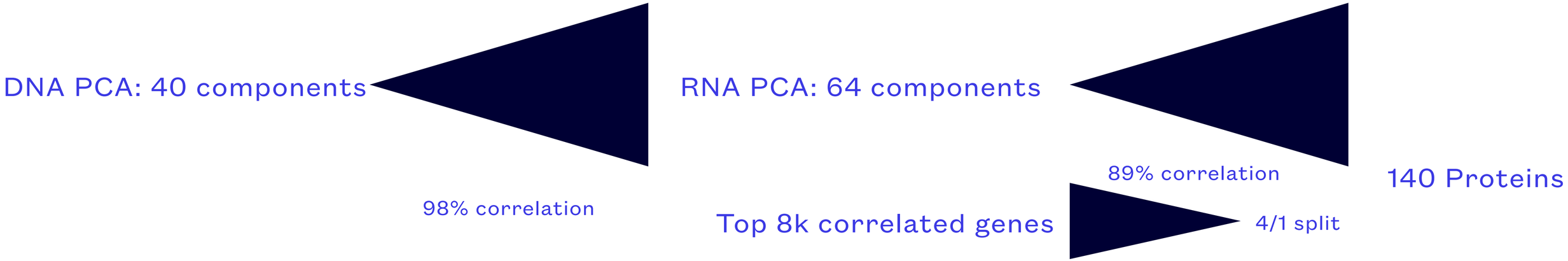
Because the 800 most correlated features are different for each protein, we can't use correlation as a loss with them

Quick-fix: Add up all most correlated features for all proteins: 8k features

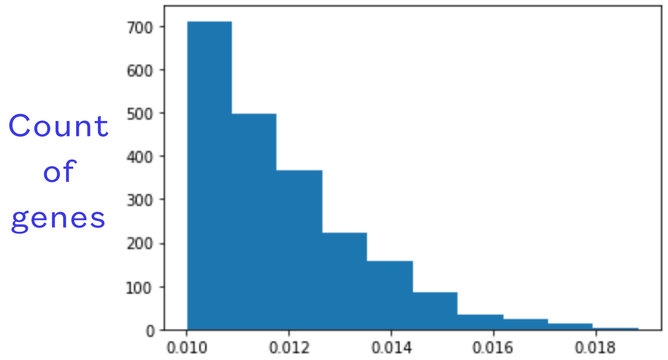
0.8919404922189595

Data structure

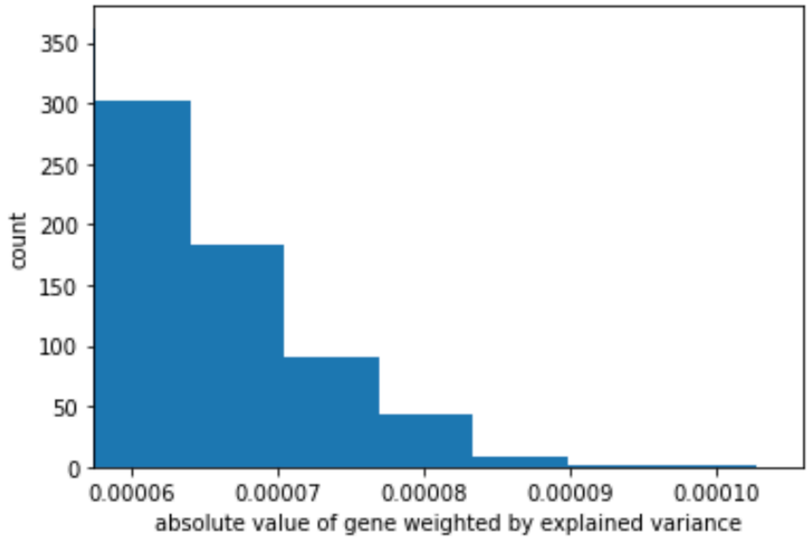
Best Predictive Performance



Finding correlated features (Multiome)



Absolute magnitude of gene in PCA component 0



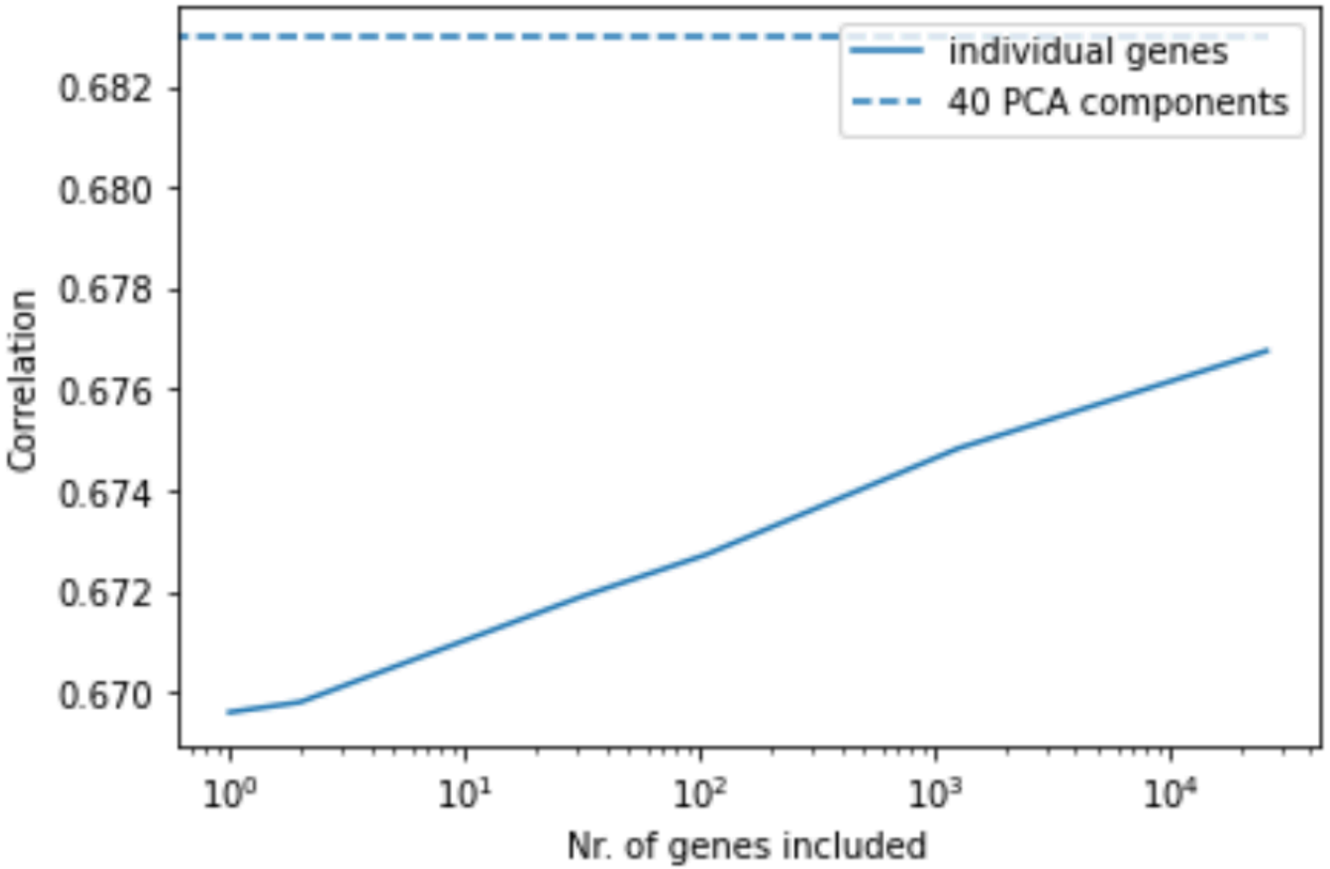
26k most prominent genes in PCA components

40 Overall PCAs

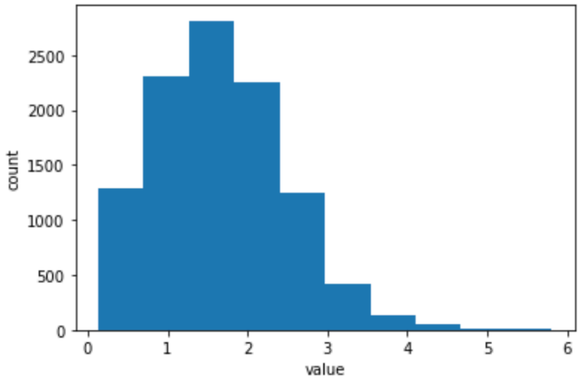
0.67676 Correlation 0.68295

- @ahmedelfazouan tells us the same approach doesn't work for Multiome
- However, we can look at the top genes in the top 40 PCAs components.

  idea by H.G. Martin
- Here, we choose genes with sum of absolute values in the top 40 PCA components weighted by their explained variance over a certain threshold
- Could again combine models, but this time with 9/1 split, only gaining 0.01%.
- On the plus side, now have model based on individual genes.



Other stats of one of these miracle genes (#3 in list): only 10% non-zero values.



Finding correlated features (Multiome)

- Changing nr. of genes included only changes predictions on a log scale.
- A single gene predicts all RNA expressions with 67% correlation and top 64 PCA components of RNA expression with 96.5% correlation!
- But correlation does not imply causation and in fact there are many such genes with equal predictability. Here are the most prominent:
  - 'chr14:105817718-105818626'
  - 'chr22:27611910-27612834'
  - 'chr1:47180897-47181792'
  - 'chr3:128415902-128416493'
  - 'chr20:44543967-44544901'
- So with 89%x96.5% = 85.9% correlation, all 140 protein concentrations can be deduced by any one of these single gene availabilities (assuming correlation is multiplicative).
- However, does not mean we can predict raw protein levels (see later slide).

Conclusion

by presenter

Data structure

DNA: 200k+ genes

RNA: 20k+ expressions

140 Proteins

Best Predictive Performance

DNA PCA: 40 components

98% correlation

RNA PCA: 64 components

89% correlation

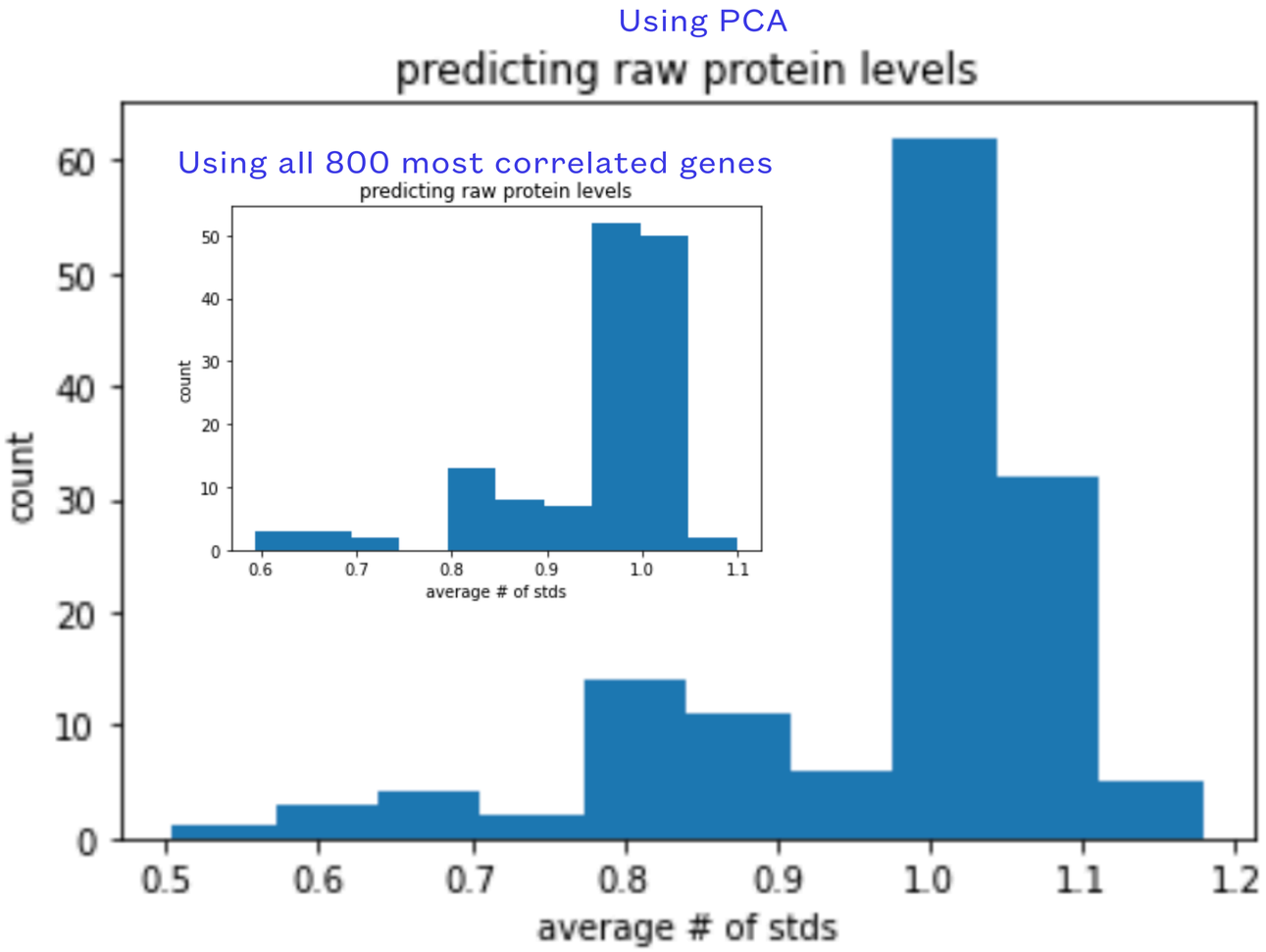
140 Proteins

or

Top x correlated genes
with small correlation penalty

Top 8k correlated genes

4/1 split



Predicting individual protein levels

Using ‘mse’ as loss

- Not normalizing the data, how well can we predict raw protein levels for individual proteins?
- The Pearson correlation doesn’t compare raw protein levels, only how well our prediction correlates to the cell-normalized protein levels.
- We are barely better than just predicting the average (worse for half - better for half). However, when we are better, we are a lot better -> cracked about 30 proteins
- The lesson is that it is MUCH easier to predict how proteins correlate than raw levels
- Furthermore, most of these protein concentrations can be predicted by a single gene!

What correlates together, expresses together. A success story of using PCA and correlation to predict protein levels from DNA data with 98%x89%=87% correlation (for cells < 7 days old)!

What we learned

- CITEseq:
  - The existing model can predict protein levels significantly better than the mean for about 30 proteins.
  - It excels at leveraging correlations to obtain Pearson correlation at the 89% level, largely due to the magic of PCA which is shown to be the language that is spoken by cells.
  - Also,@ahmedelfazouan showed that individual gene expressions can also be used for competitive predictions. For example, using the expressions of about 8000 genes strongly correlated with individual proteins.

What we learned

- Multiome:
  - PCA components of RNA expression can be expressed with above 98% correlation! Therefore, protein concentrations can be predicted from DNA availability alone with almost as much confidence as from RNA data.
  - It shows that while there are 200k+ ‘knobs’ for each cell in the individual-gene basis, the cell in practice (for healthy donors) only routinely utilizes about 40 ‘knobs’ which are the top 40 PCA components.
  - However, we show that by all but 2% of the correlations observed with the multiome data can be deduced from gene availability of any one single gene of a pool of genes with tiny differences in success. Adding each new gene only improves performance a tiny amount, further reinforcing the view that the PCA components are the true predictors because of how correlated the genes are.
  - We also show that one representative gene of this family has 10% non-zero values. Furthermore, to achieve the remaining 2% one needs to include contributions of over 20k genes.

What we learned

- Multiome:
  - On day 7, there is a strong drop in model performance due to two separate effects:
    - 1. The top PCA components of RNA expression can no longer be accurately predicted from previous days' data.
      - Since these directly contribute to protein production in our model, we hypothesize the cells have changed - perhaps differentiated.
    - 2. Between PCA components 2048 and 4096, performance drops strongly for one of two clusters found with KMeans - even when using same-day cells.
      - We hypothesize that random proteins inhibiting non-essential RNA production in this regime is responsible.

Ways to improve model

- More data - we showed that we can still improve with more cells per donor and day!
- Measuring all 3 modalities in the same cell would let us hopefully fit the noise on day 7 better (though not the change perhaps due to differentiation).
- Time-series data in the same cell would let us understand what is changing during differentiation, also perhaps with auxiliary measurements traditionally used to determine differentiation for validation of novel predictions.
- CITEseq data for day 7. Predicting a 3-day jump when the training data only contains a 2-day jump very early on is difficult.
- Analyzing the DNA data using KMeans clearly reveals 2 clusters for all but day 3 (when KMeans only fit 1 cluster). The clusters are very predictive of model performance on day 7. It would be interesting to know what these clusters are.

Failed attempts

Over 20 notebooks and 300 hours of work-time

- Things I tried that didn’t work:
  - Using cell annotations
  - RobustPCA
  - Feature Extraction by Intermediate Labels on Extremal Cells with Protein Spikes
  - Recurrent Neural Networks
  - Graph Neural Networks & KNN - edge information has 86% correlation vs node information 90% correlation -> nodes alone are superior. Also memory/batch issues.
  - MAGIC: Preprocessing and batch correction
  - CNA: Covarying neighborhood analysis (finally ran into singular matrix issue)
  - UMAP - worse than PCA
  - Feature Extraction by Punitive Regularization (this works at the 0.02% level)
  - ResNet - retraining model from previous day
  - Autoencoder - at best just as good as PCA
  - millipede - feature selection
  - episcanpy (extracted gene expression to true gene expression has 8% correlation)
  - match cell types to proportion seen on day 7 during training
  - forcing the day 4 data to learn how to work with day 7 PCAs
  - binarize inputs
  - combining predicted protein levels with DNA data to predict RNA data
  - TabNet

Acknowledgements

Thanks to all the Kagglers who taught me how to work with this kind of data!

Names on individual slides