Comp683: Computational Biology

Lecture 21

April 15, 2024

Today + Announcements

- Finishing Up Graph Alignment
- Multiomics Factor Analysis (MOFA)
- Homework 2 is online and due April 26, https://github.com/natalies-teaching/Comp683_CompBio_2024/tree/main/Homework2
- Final project presentations will be on the last two days of class (April 25 and April 30). Note that you can just give us an update. The writeup will be due May 3. Sign up here for a 10 min slot https://docs.google.com/spreadsheets/d/1LvzE54589TLroS8McWfQ6uAscSf9uOLs_FXnLyScMw4/edit?usp=sharing

Review Questions

1. What was the meaning of the **w** and **x** inferred through the Mashup appproach?
2. How were the **x** and **w** used to estimate the random walk with restart probabilities?

Overview of Regal Method

Regal → Representation Based Graph Alignment.

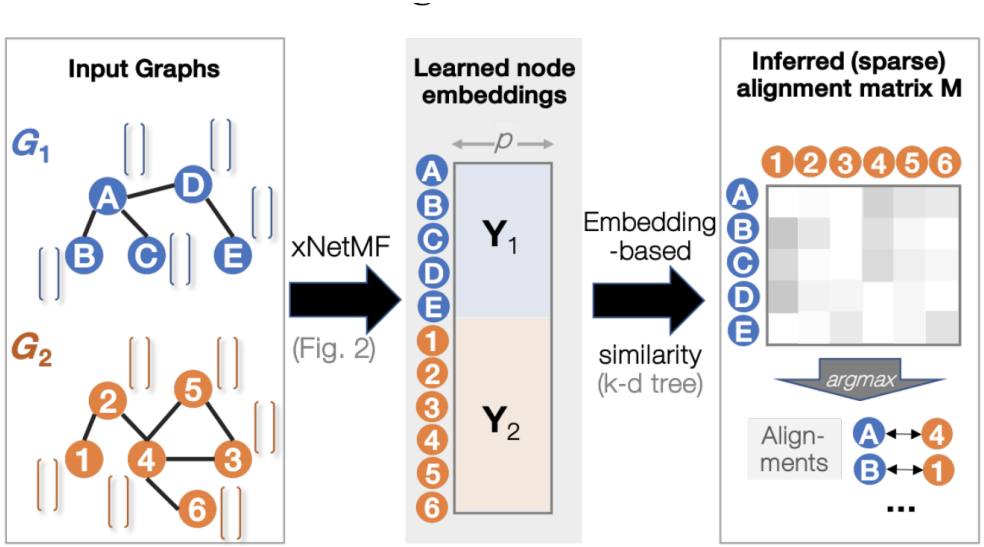


Figure: from from Heimann *et al.* CIKM 2018. Similar to Node2vec, the authors want to find a representation for each node

From our earlier example, we would get a mapping of cell-populations from one dataset to the other.

Calculating Node Similarity Within and Between Graphs

Define distance between nodes u and v in terms of structure ($\mathbf{d}$), or attributes ($\mathbf{f}$) as,

$$\text{sim}(u, v) = \exp\left[-\gamma_s \cdot \|\mathbf{d}_u - \mathbf{d}_v\|_2^2 - \gamma_a \cdot \text{dist}\left(\mathbf{f}_u, \mathbf{f}_v\right)\right]$$

Each $\mathbf{d}_u$ is defined as,

$$\mathbf{d}_u = \sum_{k=1}^{K} \delta^{k-1} \mathbf{d}_u^k$$

Here, δ is a discount factor for greater hop distances.

Expensive Formulation

Given a factorization approach, write the $n \times n$ similarity matrix, $\mathbf{S}$ as,

$$\mathbf{S} \approx \mathbf{Y}\mathbf{Z}^T$$

Here, $\mathbf{Y}$ gives the node-to-embedding matrix (which is what we want).
Intuitively, we want the following to be as close as possible to 0,

$$||\mathbf{S} - \mathbf{Y}\mathbf{Z}^T||$$

($\mathbf{Z}$ and $\mathbf{Y}$ are both $n \times p$ matrices)

An Approximation with Landmarks

- The punchline is that $\mathbf{S}$ will be approximated with a low-rank matrix, $\tilde{\mathbf{S}}$, which is never explicitly computed!
- The solution is to choose $p << n$ 'landmark' nodes, chosen across both graphs (G_1, G_2).
- Compute similarity between each node and each landmark. This produces an $n \times p$ matrix, $\mathbf{C}$.
- Further, the $p \times p$ landmark-to-landmark submatrix can also be extracted ($\mathbf{W}$).

The Low-Rank Matrix, $\tilde{\mathbf{S}}$

Finally, the low-rank matrix $\tilde{\mathbf{S}}$ is given as,

$$\tilde{\mathbf{S}} = \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}$$

Here $\mathbf{W}^{\dagger}$ is computed as the pseudoinverse of $\mathbf{W}$

- The problem is that $\tilde{\mathbf{S}}$ is still an $n \times n$ matrix, and we still need to find the embedding matrix, $\mathbf{Y}$

Approximation for the Embedding Matrix, $\mathbf{Y}$

Theorem: Given graphs, $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2)$ with an $n \times n$ joint combined structural and attribute-based similarity matrix $\mathbf{S} \approx \mathbf{Y}\mathbf{Z}^T$, its node embedding matrix $\mathbf{Y}$ can be approximated as,

$$\tilde{Y} = CU\Sigma^{1/2}. \tag{1}$$

Here, $\mathbf{C}$ is the $n \times p$ matrix of similarities between the n nodes and the p chosen landmarks and $\mathbf{W}^\dagger = \mathbf{U}\Sigma\mathbf{V}^T$ is the full rank SVD of the pseudoinverse of the small $p \times p$ landmark similarity matrix matrix, $\mathbf{W}^\dagger$.

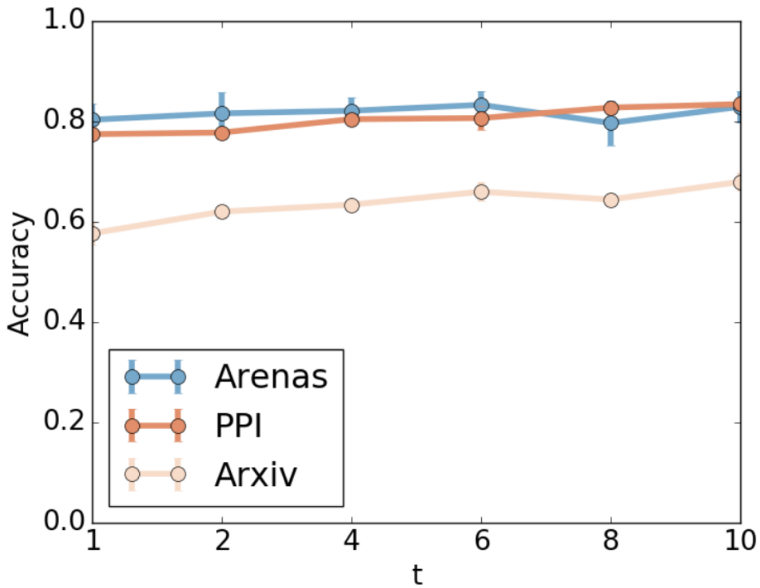
So to Summarize the Entire xNetMF

Algorithm 2 xNetMF $(G_1, G_2, p, K, \gamma_s, \gamma_a)$

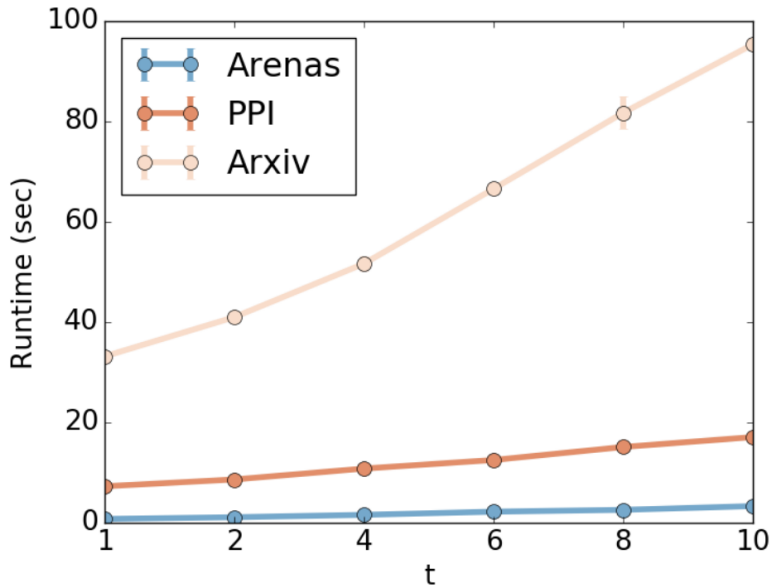
========== STEP 1. Node Identity Extraction ==========
for node u in $\mathcal{V}_1 \cup \mathcal{V}_2$ **do**
  for hop k up to K **do** ▷ counts of node degrees of k-hop neighbors of u
    $\mathbf{d}_u^k$ = CountDegreeDistributions($\mathcal{R}_u^k$) ▷ $1 \leq K \leq$ graph diameter
  end for
  $\mathbf{d}_u = \sum_{k=1}^{K} \delta^{k-1} \mathbf{d}_u^k$ ▷ discount factor $\delta \in (0, 1]$
end for
========== STEP 2. Efficient Similarity-based Representation ==========
========== STEP 2a. Reduced $n \times p$ Similarity Computation ==========
$\mathcal{L}$ = ChooseLandmarks(G_1, G_2,p) ▷ choose p nodes from G_1, G_2
for node u in $\mathcal{V}$ **do**
  for node v in $\mathcal{L}$ **do**
    $c_{uv} = e^{-\gamma_s \cdot ||\mathbf{d}_u - \mathbf{d}_v||_2^2 - \gamma_a \cdot \text{dist}(\mathbf{f}_u, \mathbf{f}_v)}$
  end for
end for ▷ Used in low-rank approx. of similarity graph (not constructed)
========== STEP 2b. From Similarity to Representation ==========
$\mathbf{W} = \mathbf{C}[\mathcal{L}, \mathcal{L}]$ ▷ Rows of $\mathbf{C}$ corresponding to landmark nodes
$[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{SVD}(\mathbf{W}^\dagger)$
$\tilde{\mathbf{Y}} = \mathbf{C}\mathbf{U}\mathbf{\Sigma}^{\frac{1}{2}}$ ▷ **Embedding**: *implicit* factorization of similarity graph
$\tilde{\mathbf{Y}} = Normalize(\tilde{\mathbf{Y}})$ ▷ **Postprocessing**: make embeddings have magnitude 1
$\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2 = \text{Split}(\tilde{\mathbf{Y}})$ ▷ **Separate** representations for nodes in G_1, G_2
return $\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2$

We Still Have Another Step (matching nodes between graphs)

But a question of interest is, how many landmarks do we need? These landmarks will be our effective embedding dimension.



(a) Accuracy w.r.t. # of landmarks



(b) Runtime w.r.t. # of landmarks

Figure: Here $p = t \log_2 n$

The Last Step: Distances Between Nodes in Different Graphs via Embeddings

Letting $\tilde{Y}_1$ and $\tilde{Y}_2$ be the embeddings for graphs 1 and 2 respectively, then pairwise node similarities between the graphs can be computed as,

$$\text{sim}_{emb}\left(\tilde{\mathbf{Y}}_1[u], \tilde{\mathbf{Y}}_2[v]\right) = e^{-\left\|\tilde{\mathbf{Y}}_1[u]-\tilde{\mathbf{Y}}_2[v]\right\|_2^2}$$

- Match nodes according to these similarity scores.
- **Soft Scoring Approach:** From kd-tree, find $\alpha << N$ top matches of nodes from the other graph.

Experimental Evaluation

Given an adjacency matrix, $\mathbf{A}$, generate a random permutation matrix, $\mathbf{P}$ and generate a new network as,

$$\mathbf{A}' = \mathbf{PAP}^T$$

Further, remove edges from $\mathbf{A}'$ with probability p_s, and permute attributes with probability, p_a.

Results from Adding Noise

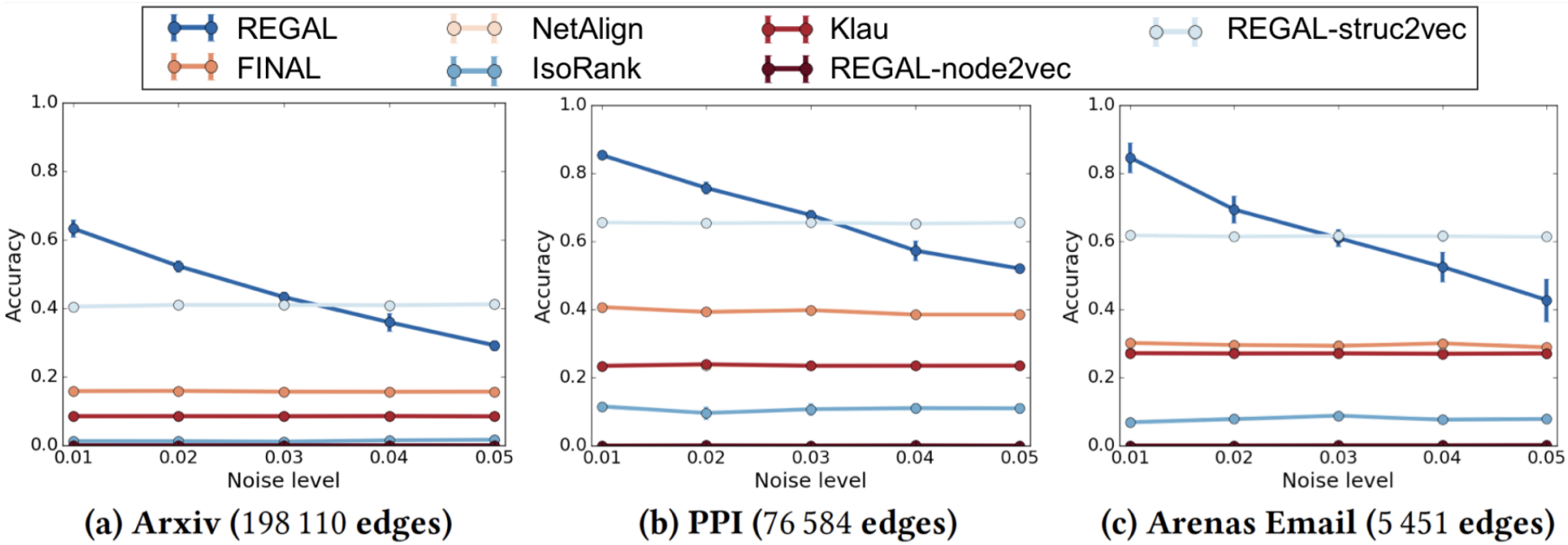


Figure: from Fig. 4, Replacing the embedding step with struct2vec produces better results when there is more noise.

REGAL Is More Ideal wrt Run-time

Dataset	Arxiv	PPI	Arenas
FINAL	4182 (180)	62.88 (32.20)	3.82 (1.41)
NetAlign	149.62 (282.03)	22.44 (0.61)	**1.89 (0.07)**
IsoRank	**17.04 (6.22)**	**6.14 (1.33)**	**0.73 (0.05)**
Klau	1291.00 (373)	476.54 (8.98)	43.04 (0.80)
REGAL-node2vec	709.04 (20.98)	139.56 (1.54)	15.05 (0.23)
REGAL-struc2vec	1975.37 (223.22)	441.35 (13.21)	74.07 (0.95)
REGAL	**86.80 (11.23)**	**18.27 (2.12)**	2.32 (0.31)

Figure: from Table 4. The methods that perform better in ‘noise‘ experiments also have higher run-time.

Conclusion

- REGAL computes embeddings via landmark points.
- The magic is in approximating the pairwise node similarity matrix through landmark points
- Performance is generally better than baselines for most ‘noise‘ levels.

Questions for You

- How do you think landmarks should be chosen? At random, or something more principled?
- What about extending this to more than two graphs?

Graph Alignment is a Hard Problem

- With REGAL, there was some fancy linear algebra required
- **Another reasonable assumption:** Nodes within the a common neighborhood in one graph should be mapped to nodes that are close (e.g. within or in a close neighborhood) in the other graph.
- This feels a bit like Leiden- where the partition will be ‘corrected‘ to make sure that communities contain graphs with a connected path between most pairs within the community.

Switching gears to another multiomics data integration strategy: MOFA

Notation and Goals

- Start with M data matrices, $\mathbf{Y}^1, \mathbf{Y}^2, \ldots, \mathbf{Y}^M$ on a set of N samples
- A Particular modality's matrix, $\mathbf{Y}_m$ has dimensions, $N \times D_m$

The goal is to write the input data matrix, $\mathbf{Y}^m$ as a product of a common factor matrix, $\mathbf{Z}$ (samples $\times$ factors) and a 'weight' matrix (e.g. one that relates features to factor space) as,

$$\mathbf{Y}^m = \mathbf{Z}\mathbf{W}^{mT} + \boldsymbol{\epsilon}^{\mathbf{m}}$$

Unpacking...

Assuming there are k factors and given

$$\mathbf{Y}^m = \mathbf{Z}\mathbf{W}^{mT} + \boldsymbol{\epsilon}^{\mathbf{m}}$$

,

- $\mathbf{Z} \in \mathbb{R}^{N \times K}$ relate the original samples to the factors
- $\mathbf{W}^m \in \mathbb{R}^{D_m \times K}$ relates the original features to the factors

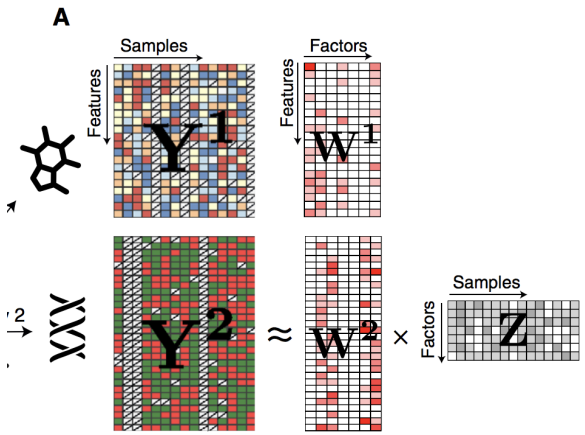


Figure: from Argelaguet *et al.* Molecular Systems Biology. 2018.

Overview of the Model

This is latent variable model and feature dependencies are attempted to be explained in terms of k latent classes (or 'factors').

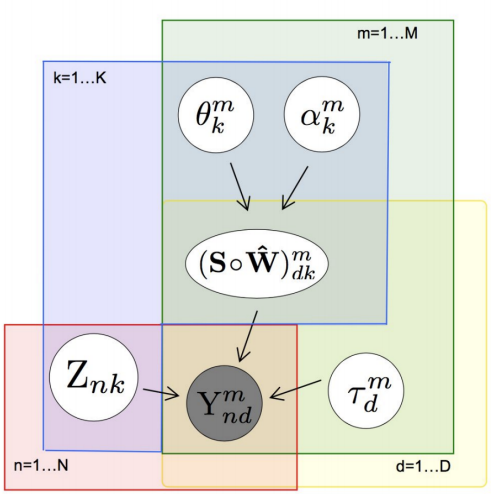


Figure: from Argelaguet *et al.* Molecular Systems Biology. 2018. As always, gray nodes are observed variables and white notes are the unobserved variables inferred by the model.

Starts Simple Enough....

The the observed value of feature d in sample n from modality m is modeled as,

$$p(y_{nd}^m) = \mathcal{N}(y_{nd}^m \mid \mathbf{z}_{n,:}\mathbf{w}_{d,:}^{mT}, 1/\gamma_d^m)$$

- $\mathbf{w}_{d,:}^m$ gives the d-th row of $\mathbf{W}^m$
- $\mathbf{z}_{n,:}$ is the n-th row of the latent factor matrix, $\mathbf{Z}$.
- **Other priors**:
  - $p(z_{n,k} = \mathcal{N}(z_{n,k} \mid 0, 1)$
  - $p(\tau_d^m) = \mathcal{G}(\tau_d^m \mid a_0^\tau, b_0^\tau)$

Regularization in $\mathbf{W}^m$

In computing each $\mathbf{W}^m$ there two desired kinds of sparsity :

- View and factor-wise sparsity
- Feature-wise sparsity

The intuition is that $\mathbf{w}^m_{:,k}$ is shrunk to 0 if factor k does not drive any variation in view m.

In practice...

Sparsity is enforced for each **W** through appropriate priors. Specifically, they model **W** as a product of a Gaussian random variable, $\hat{w}$, and a Bernoulli random variable, s.

$$p(\hat{w}^m_{d,k}, s^m_{d,k}) = \mathcal{N}(\hat{w}^m_{d,k} \mid 0, 1/\alpha^m_k)\text{Ber}(s^m_{d,k} \mid \theta^m_k)$$

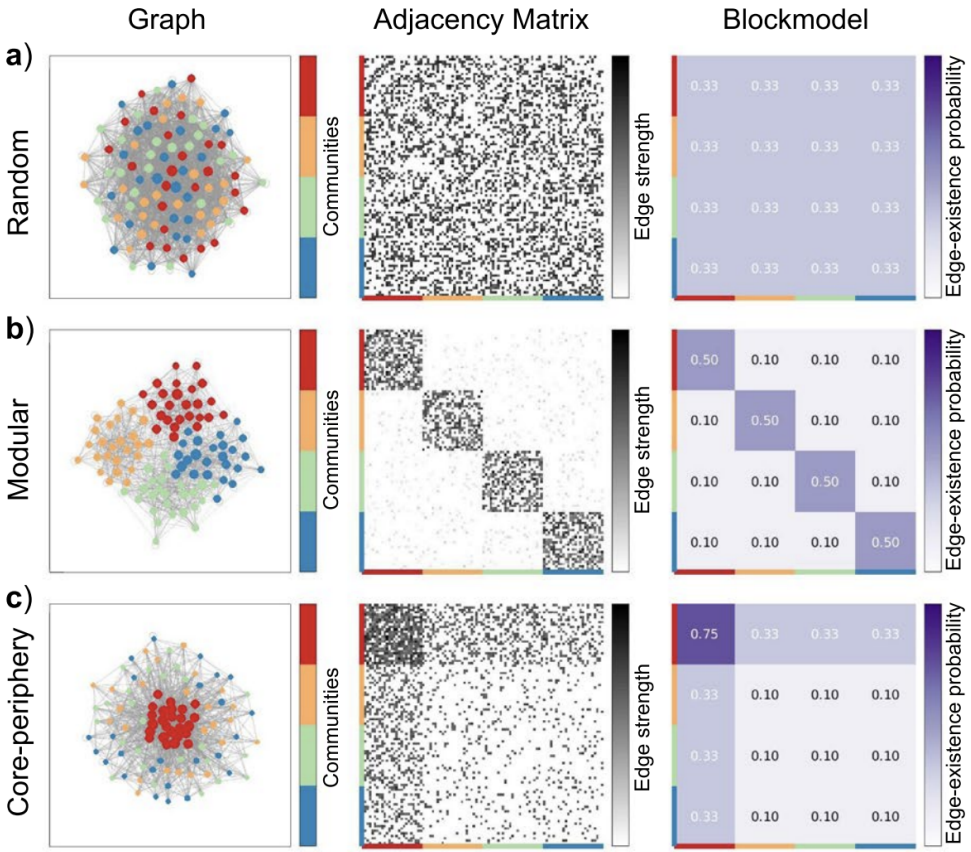
- $p(\theta^m_m) = \text{Beta}(\theta^m_k \mid a^\theta_0, b^\theta_0)$
- $p(\alpha^m_k) = \mathcal{G}(\alpha^m_k \mid a^\alpha_0, b^\alpha_o)$

Thinking More About Sparsity

Let's think about what happens with bernoulli probabilities:

- A value of θ_k^m close to 0 implies that most of the weights of factor k in view m tend towards 0.
- A θ_k^m close to 1 alternatively implies that most of the weights are non-zero (e.g. a non-sparse factor)

Living in the world of latent variables... remember stochastic block model?



Latent variable: node-to-community assignments

$$\log \mathcal{L}(\mathcal{X}, \mathcal{Z}) = \sum_i \sum_q Z_{iq} \log \alpha_q + \frac{1}{2} \sum_{i \neq j} \sum_{q,\ell} Z_{iq} Z_{j\ell} \log b\left(X_{ij}; \pi_{q\ell}\right)$$

- At some point in updating parameters via EM, we need to worry about the posterior or $P(\mathcal{Z} \mid \mathcal{X})$.

Dealing with Latent Variables

More help with factorized approximations...

- Our situation is the case of having an intractable posterior distribution of unobserved variables $p(\mathbf{X} \mid \mathbf{Y})$.
- Here, $\mathbf{X}$ is denoting our un-observed variables, and $\mathbf{Y}$ is denoting the observed variables

We can deal with $p(\mathbf{X} \mid \mathbf{Y})$ by approximating it with a factorized form,

$$q(\mathbf{X}) = \prod_i q(\mathbf{X}_i)$$

.

This can make inference more efficient.

Reminder About Latent Variables Here

Notably, $\mathbf{Z}$ and $(\mathbf{S} \circ \hat{\mathbf{W}})$

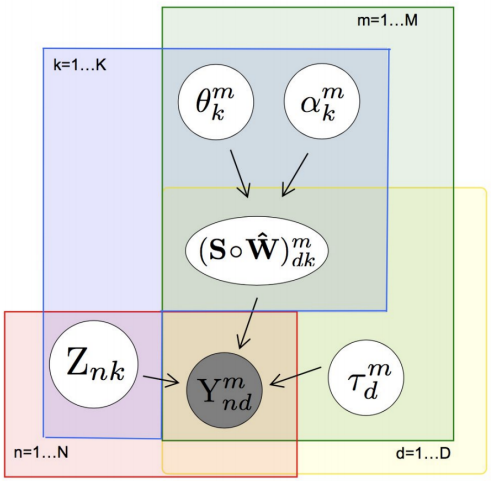


Figure: from Argelaguet *et al.* Molecular Systems Biology. 2018. As always, gray nodes are observed variables and white notes are the unobserved variables inferred by the model.

For MOFA : Mean Field Approximation

The assumption under the mean-field approach is that $q(\mathbf{X})$ factorized over M disjoint groups of variables as,

$$q(\mathbf{X}) = \prod_{i=1}^{M} q(\mathbf{x_i})$$

.

In the case of MOFA,

$$\begin{aligned} q(\mathbf{Z}, \mathbf{S}, \hat{\mathbf{W}}, \boldsymbol{\alpha}, \boldsymbol{\tau}, \boldsymbol{\theta}) &= q(\mathbf{Z}) q(\boldsymbol{\alpha}) q(\boldsymbol{\theta}) q(\boldsymbol{\tau}) q(\mathbf{S}, \hat{\mathbf{W}}) \\ &= \prod_{n=1}^{N} \prod_{k=1}^{K} q\left(z_{n,k}\right) \prod_{m=1}^{M} \prod_{k=1}^{K} q\left(\alpha_k^m\right) q\left(\theta_k^m\right) \prod_{m=1}^{M} \prod_{d=1}^{D_m} q\left(\tau_d^m\right) \prod_{m=1}^{M} \prod_{d=1}^{D_m} \prod_{k=1}^{K} q\left(\hat{w}_{d,k}^m, s_{d,k}^m\right) \end{aligned}$$

Update via EM - $q(\mathbf{Z})$

$$q(\mathbf{Z}) = \prod_{k=1}^{K} \prod_{n=1}^{N} q\left(z_{nk}\right) = \prod_{k=1}^{K} \prod_{n=1}^{N} \mathcal{N}\left(z_{nk} \mid \mu_{z_{nk}}, \sigma_{z_{nk}}\right)$$

where

$$\sigma_{z_{nk}}^2 = \left(\sum_{m=1}^{M} \sum_{d=1}^{D_m} \langle \tau_d^m \rangle \left\langle (s_{dk}^m \hat{w}_{dk}^m)^2 \right\rangle + 1 \right)^{-1}$$

$$\mu_{z_{nk}} = \sigma_{z_{nk}}^2 \sum_{m=1}^{M} \sum_{d=1}^{D_m} \langle \tau_d^m \rangle \langle s_{dk}^m \hat{w}_{dk}^m \rangle \left(y_{nd}^m - \sum_{j \neq k} \left\langle s_{dj}^m \hat{w}_{dj}^m \right\rangle \langle z_{nj} \rangle \right)$$

$q(\hat{\mathbf{W}}, \mathbf{S})$

$$q(\hat{\mathbf{W}}, \mathbf{S}) = \prod_{m=1}^{M} \prod_{d=1}^{D_m} \prod_{k=1}^{K} q\left(\hat{w}_{dk}^m, s_{dk}^m\right) = \prod_{m=1}^{M} \prod_{d=1}^{D_m} \prod_{k=1}^{K} q\left(\hat{w}_{dk}^m \mid s_{dk}^m\right) q\left(s_{dk}^m\right)$$

See appendix of paper for particular updates for $q(\cdot)$s. The point is how this is factorized.

Moving into Results: Summary

After having inferred $\mathbf{W}^m s$ and $\mathbf{Z}$, several things can be done, such as, imputation, annotation of factors, etc.

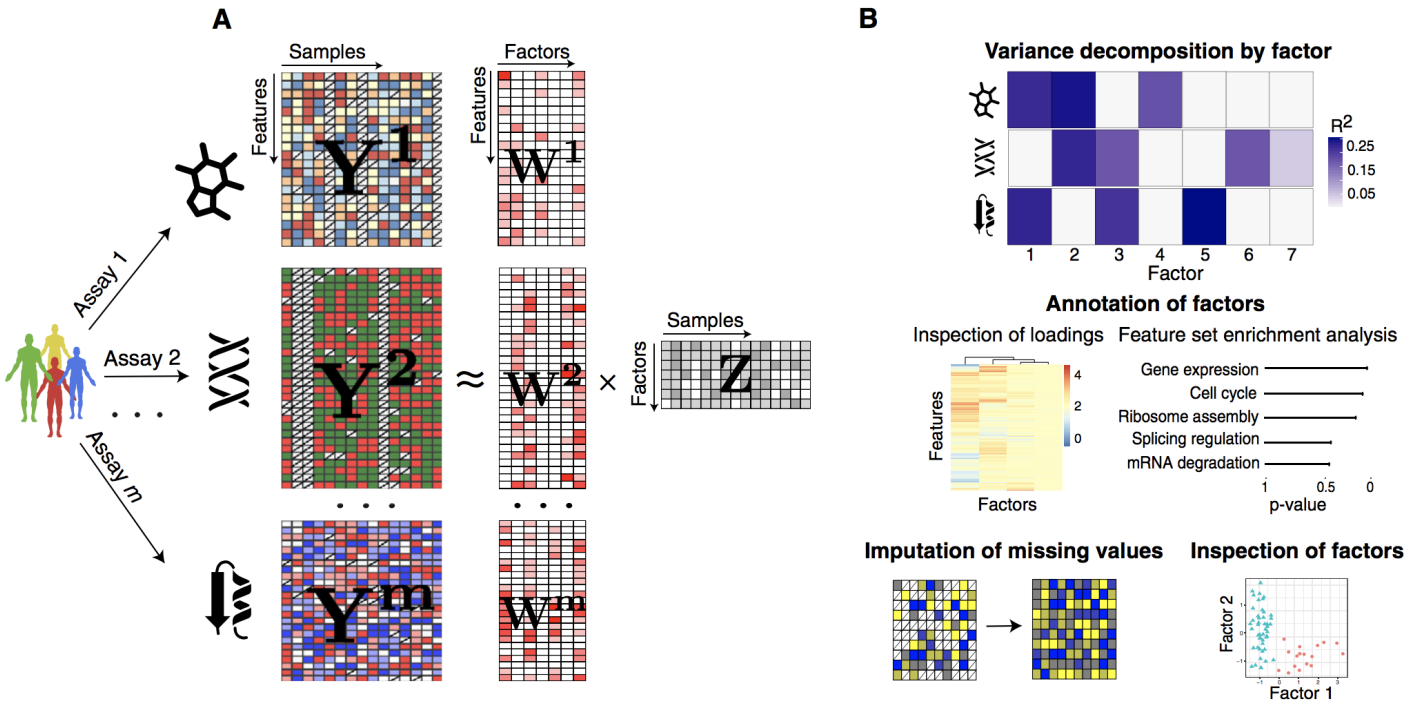


Figure: from Argelaguet *et al.* Molecular Systems Biology. 2018.

Getting Intuition About Variance Explained by Factors

- **Variance Explained by Factor k in View m :** Compute the fraction of variance explained by factor k in view m as,

$$R^2_{m,k} = 1 - \left(\sum_{n,d} y^m_{nd} - z_{nk} w^m_{kd} - \mu^m_d \right)^2 / \left(\sum_{n,d} y^m_{nd} - \mu^m_d \right)^2$$

Here, μ_d is the feature-wise mean.

Variance Explained per Modality

- **Variance Explained In Each Modality Over All Features** : Compute the fraction of variance explained per view as,

$$R_m^2 = 1 - \left(\sum_{n,d} y_{nd}^m - \sum_k z_{nk} w_{kd}^m - \mu_d^m \right)^2 / \left(\sum_{n,d} y_{nd}^m - \mu_d^m \right)^2$$

Here, μ_d^m is the feature-wise mean.

Results in Dataset 1 : CLL dataset

Find data here
`https://rdrr.io/github/bioFAM/MOFAdata/man/CLL_data.html`

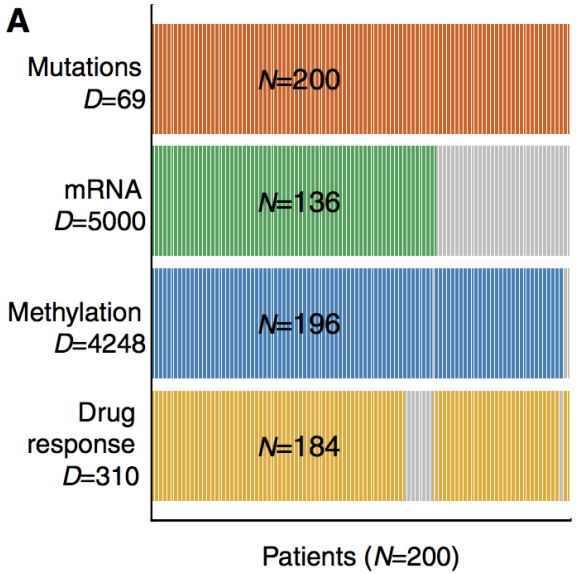


Figure: from Argelaguet *et al.* Molecular Systems Biology. 2018. Modalities and present/missing features for each patient.

Visualization of Samples

Samples can be projected into two dimensions based on the first two factors inferred in the matrix, **Z**

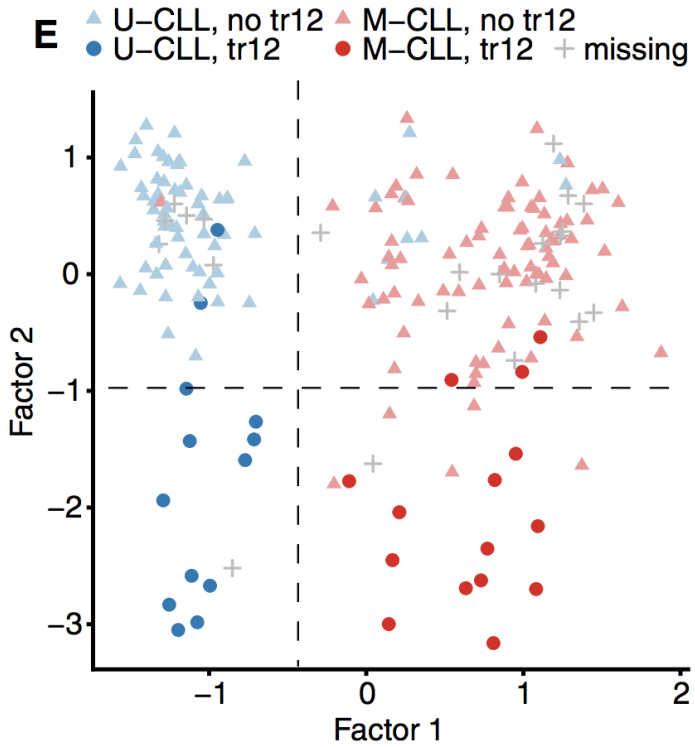


Figure: colors denote something related to the status of the tumor. Shape indicates chromosome 12 trisomy status.

Jointly Predicting Clinical Outcome in Contrast to a Single Type of Data

Experiment :

- Predict time to treatment for $N = 174$ patients using multivariate Cox regression trained using the 10 factors from MOFA
- Compare this performance on the prediction accuracy on individual modalities.
    - In this case, reduce each individual modalities to the top 10 PCs.

Results Predicting Time to Treatment

Note that the Y-axis is simply a statistic reflecting goodness of fit between true and predicted times to treatment (Use top PCs instead of regular feature space).

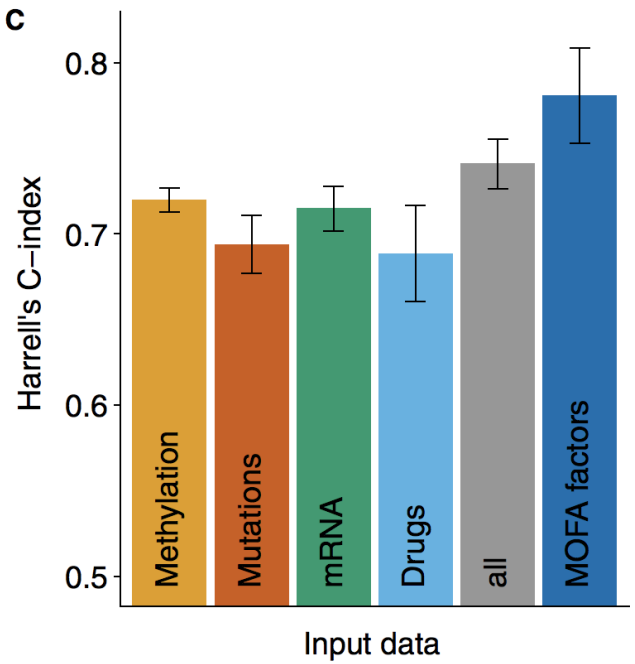


Figure: from Fig. 4 of Argelaguet *et al.* Molecular Systems Biology. 2018.

Instead of using PC Representations for Each Modality

Instead of using PC representations for each modality, the authors also compared to prediction with all features from each individual modality. Again, predicting time to treatment.

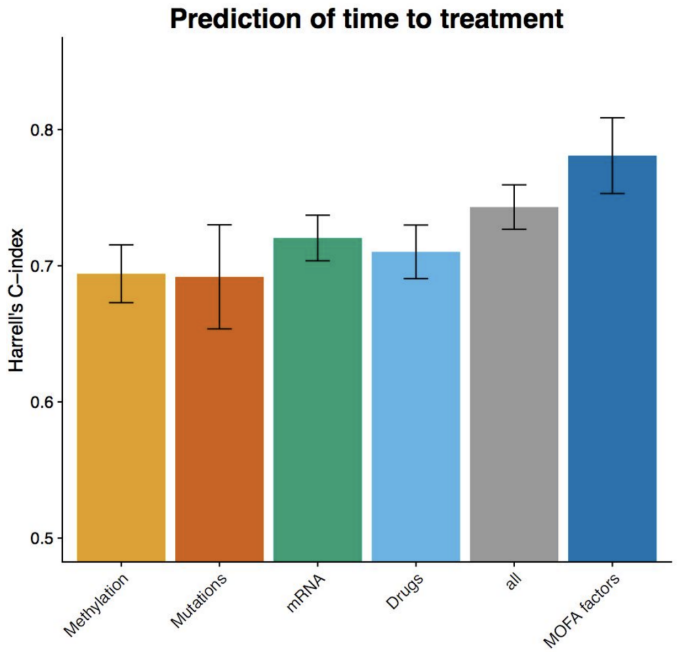


Figure: from Supp. Figure 20. Using all features still does not beat performance based on MOFA factors.

Certain Combinations of Modalities Can Turn out to be More Effective

Applying MOFA with individual modalities held out reveals sometimes certain combinations of modalities is more effective than just throwing everything in all together!

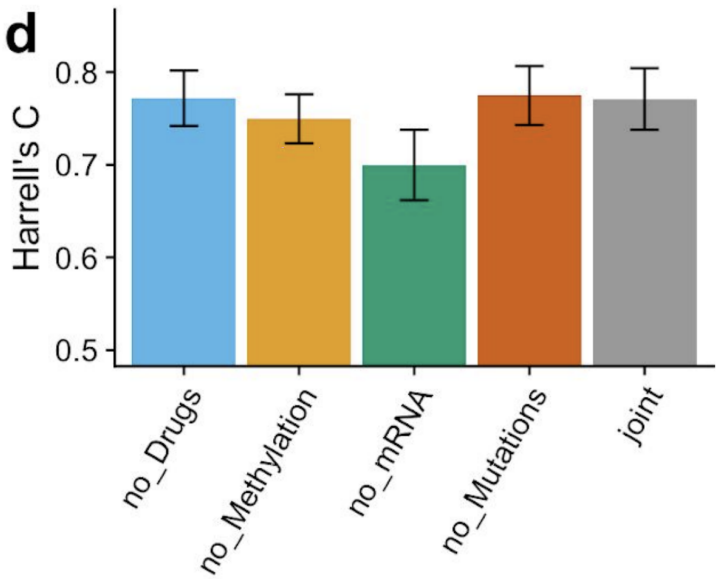


Figure: from Supplementary Figure 8.

Survival Analysis According to Particular Factors

Survival probability vs time-to-treatment for the individual MOFA factors

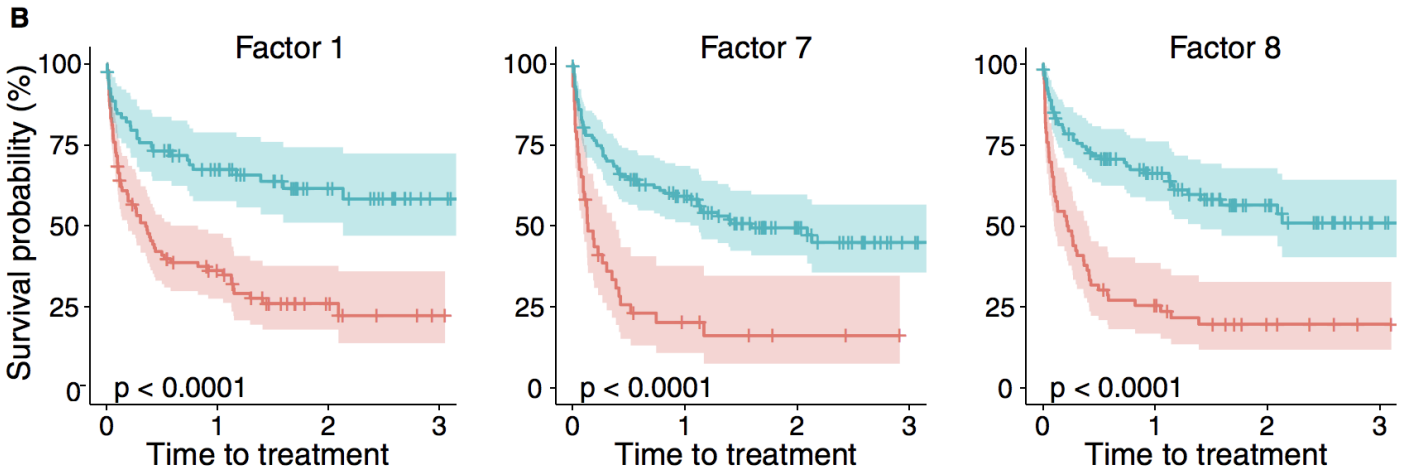


Figure: from Fig. 4. We can see distinct patterns based on survival probability against predicted time to next treatment.

Dataset 2 : Single-Cell Data for a Differentiation Trajectory

Data from a single-cell multiomics dataset. This is applied to 87 mouse embryonic stem calls, with 16 that were cultured in ‘2i1‘ media, giving them a naive pluripotent state. The remaining cells were serum grown and hence committed to a differentiation trajectory.

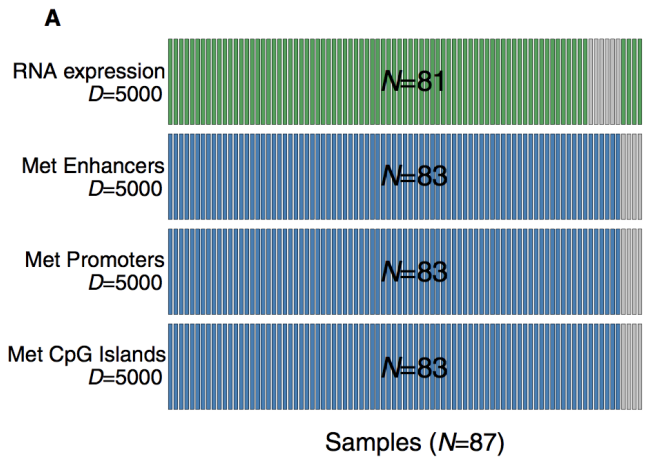


Figure: from Fig. 5. Distribution of samples profiled in each modality.

Visualizing Cells Based on **Z**

Using **Z** to plot cells based on the first two factors, the cells separate according to how they were cultured (which also corresponds to differentiation status!)

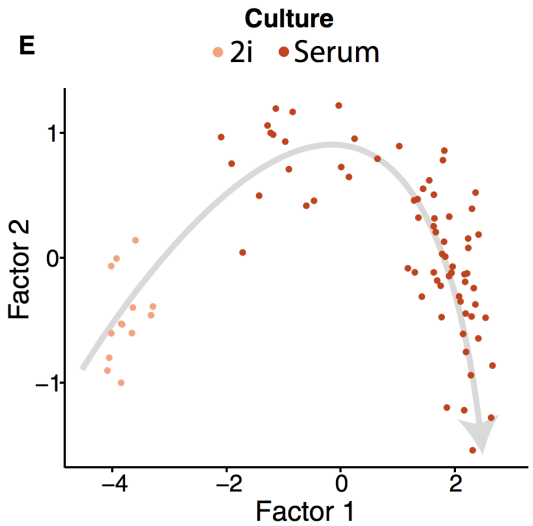


Figure: from Fig. 5.

Inferred Factors on Genes and How They Relate to Pluripotency and Differentiation

Using the $\mathbf{W}^m s$ for the mRNA data, the authors compared the ranks of genes involved in pluripotency and differentiation, respectively to the loadings.

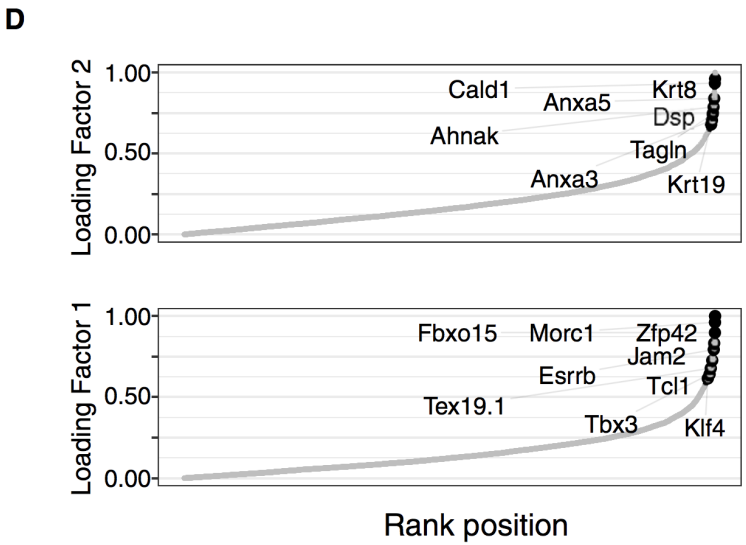


Figure: from Fig. 5. Pluripotency genes (top) vs differentiation related genes (bottom)

An Extension, MOFA+

A trivial extension....almost the same, except now we learn a **Z** per group.

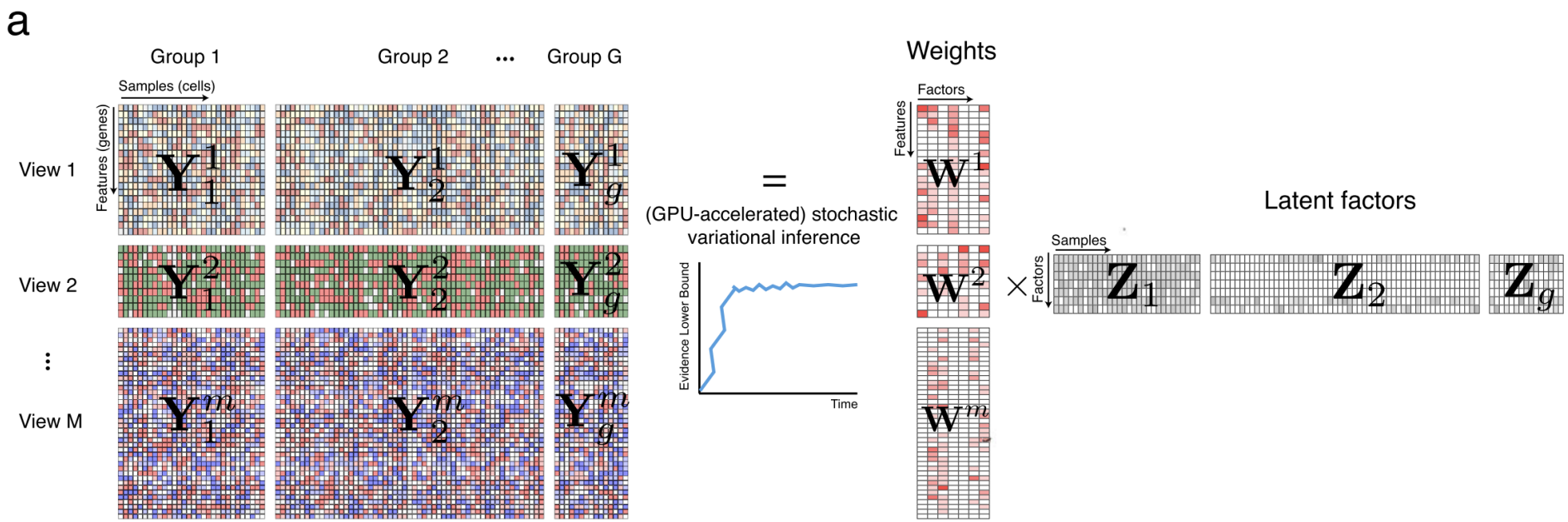


Figure: from Argalaguet *et al.* Genome Biology. 2020. Views are modalities. Groups are some partitioning of the samples.

So what do we think? Does anyone have any ideas for when this would be useful for your application?

Defining Y_{gm}

Similar to what we have seen with the regular MOFA,

$$Y_{gm} = Z_g W_m^T + \epsilon_{gm}$$

- Y_{gm} is the matrix of observations for the mth modality and gth group
- W_m is the weight matrix for the mth modality
- Z_g is the factor matrix for the gth group
- ϵ_{gm} is the residual noise for the mth modality in the gth group

Summary

- MOFA for decomposing a sample x feature matrix to (feature x factor)(factor x sample)
- MOFA+ for doing this calculation per group