FaStaNMF: A Fast and Stable Non-Negative Matrix Factorization for Gene Expression

Michael D. Sweeney, Luke A. Torre-Healy, Virginia L. Ma, Margaret A. Hall, Lucie Chrastecka, Alisa Yurovsky, and Richard A. Moffitt

***Abstract*—Gene expression analysis of samples with mixed cell types only provides limited insight to the characteristics of specific tissues. *In silico* deconvolution can be applied to extract cell type specific expression, thus avoiding prohibitively expensive techniques such as cell sorting or single-cell sequencing. Non-negative matrix factorization (NMF) is a deconvolution method shown to be useful for gene expression data, in part due to its constraint of non-negativity. Unlike other methods, NMF provides the capability to deconvolve without prior knowledge of the components of the model. However, NMF is not guaranteed to provide a globally unique solution. In this work, we present FaStaNMF, a method that balances achieving global stability of the NMF results, which is essential for inter-experiment and inter-lab reproducibility, with accuracy and speed. Results: FaStaNMF was applied to four datasets with known ground truth, created based on publicly available data or by using our simulation infrastructure, RNAGinesis. We assessed FaStaNMF on three criteria – speed, accuracy, and stability, and it favorably compared to the standard approach of achieving reproduceable results with NMF. We expect that FaStaNMF can be applied successfully to a wide array of biological data, such as different tumor/immune and other disease microenvironments.**

***Index Terms*—Deconvolution, gene expression, NMF, transcriptomics.**

I. Introduction

MANY Problems in Biology Require the Precise Knowledge of Expression of Specific Cell types. Very importantly, Cancer Research Relies on the Ability to Faithfully Assess Tumor-Specific Expression Profiles in Parallel With Information From Surrounding Tissue or Immune Cells That are Found in Bulk Patient samples, i.e., Biopsy Specimens [1]. Isolating Cells Using Flow cytometry, or Single Cell Sequencing Is powerful, but They Do Not Scale well. These Methods are expensive, Require Intensive Equipment resources, and Cannot Be Used on the Majority of the World's Existing samples, Which are Formalin-Preserved [2], [3]. Deconvolution Is a Cheaper *in Silico* Solution That Is Applicable to Archival data, and Comprises a Class of Algorithms Aimed At Providing Robust Insight Into the Complex Processes Occurring in Individual Cell types, Which Is Our focus, As Well More Generally Unmixing Individual Expression Profiles From Heterogenous samples.

In recent times, there has been tremendous growth in the development of in silico gene expression deconvolution and analysis tools for cancer analysis. There are methods that are only interested in inferring tumor purity (proportion of tumor cells in the bulk sample) that use DNA methylation profiles, such as InfiniumPurify [4], or DNA copy number variation, such as ABSOLUTE [5]. These methods are not appropriate for extraction of gene-level expression profiles of tumor microenvironments.

The transcriptomics-based methods seek to deconvolve both the proportion of each cell type in each sample and the corresponding cell type expression profiles, but present various limitations such as assumptions about the number of cell types present in the sample or requiring a-priori pure expression profiles. For example, DeMixT [6] (while limited to only three compartments) requires the input of pure expressions from 2 compartments to estimate the expression of the 3rd compartment and their proportions. ESTIMATE [7] is limited to providing a score of tumor purity. DSA [8] requires a-priori lists of marker genes and CIBERSORT [9] requires a "signature matrix" comprised of barcode genes that are enriched in each cell type of interest. The assumptions and requirements make these tools non ideal for complex microenvironments.

Other deconvolution analysis tools make use of various standard decomposition methods as their primary workflow. Principal component analysis (PCA) is employed in GO-PCA [10], and pcaExplorer [11], while differential gene expression analysis (DEA) using the DESeq2 package is used in GENAVi [12] and BEAVR [13]. These deconvolution tools, while useful for RNAseq analysis exploration, are not appropriate for unmixing individual expression profiles from heterogenous samples, in part due to producing negative values.

Manuscript received 26 October 2022; revised 12 July 2023; accepted 15 July 2023. Date of publication 19 July 2023; date of current version 10 December 2024. *(Corresponding author: Alisa Yurovsky.)*

Michael D. Sweeney, Virginia L. Ma, Margaret A. Hall, and Alisa Yurovsky are with the Department of Biomedical Informatics, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: mikesweeney013@gmail.com; virginia_ma@college.harvard.edu; margaret.hall@stonybrookmedicine.edu; alisa.yurovsky@gmail.com).

Luke A. Torre-Healy is with the Department of Biomedical Informatics, MSTP, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: luke.torrehealy@stonybrookmedicine.edu).

Lucie Chrastecka is with the Department of Pharmacological Sciences, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: lucie.chrastecka@stonybrook.edu).

Richard A. Moffitt is with the Department of Biomedical Informatics, Department of Pathology, Department of Pharmacological Sciences, Stony Brook Cancer Center, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: richard.austin.moffitt@emory.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TCBB.2023.3296979, provided by the authors.

Digital Object Identifier 10.1109/TCBB.2023.3296979

The application of non-negative matrix factorization (NMF) to transcriptomic data deconvolution was introduced as an alternative to PCA; NMF jettisons PCA's constraint of orthogonality — which typically requires linear combinations of components with arbitrary signs [14]. The non-negativity constraint of NMF make sense in aiding the interpretation for transcriptomic data. Furthermore, comparison studies of matrix factorization methods have indicated that NMF and extensions of NMF offer the most promising results for identifying clusters and subtypes when compared to methods such as extensions of PCA [15], partitioning around medoids (PAM) [16], hierarchical clustering (HC) [16], [17], and DESeq2 [17].

As the best of the sample-composition-agnostic transcriptomic data deconvolution methods, NMF has been implemented by a number of packages and wrappers [18], [19]. Prior work, DECODER [20], uses NMF and leverages NMF's consensus matrix, to select the best number of metasamples for mixed sample deconvolution by iteratively applying NMF to subsets of the expression matrix.

We present FaStaNMF (Fast and Stable NMF), a new method that balances accuracy and speed with achieving global stability of the NMF results, which is essential for reproducibility (https://github.com/rmoffitt/aged). We extend the idea of taking advantage of the NMF ***consensus matrix*** to identify the most informative genes for each metasample from DECODER, and use it for iterative seeding of the NMF runs, to achieve a fast, stable, and accurate result. We do not compare FaStaNMF against DECODER because the latter is focused on selecting the best number of metasamples, as opposed to producing a fast and reproduceable result, and most importantly, because it is not based on an open-source platform.

We compare FaStaNMF against the standard approach of achieving reproduceable results with NMF, the best sample-composition-agnostic genomic data deconvolution method. Unlike the standard reproduceable results of NMF, FaStaNMF results are version and implementation independent, which is extremely important for being able to reproduce historical results for downstream analysis.

II. Methods

A. NMF Approach and the Consensus Matrix

In the context of transcriptomic data, NMF performs deconvolution on matrices (see Fig. 1(a)), where the rows represent genes and the columns represent samples. After running NMF, the multitude of genes represented in the gene expression data will be represented by the k most prominent metasamples in the gene expression matrix where k is the factorization rank specified by the user. NMF will deconvolve the gene expression data matrix, V, into two matrices W and H such that $V \approx W \times H$ (Fig. 1(d)). W will have i rows and k columns, with each column representing the gene expressions individual to this metasample, and H will have k rows and j columns, with each row representing the metasample's proportion for each sample. For example, matrix cell W_{ab} will represent coefficient of gene a in metasample b; matrix cell H_{cd} will represent the proportion of metasample c in sample d.

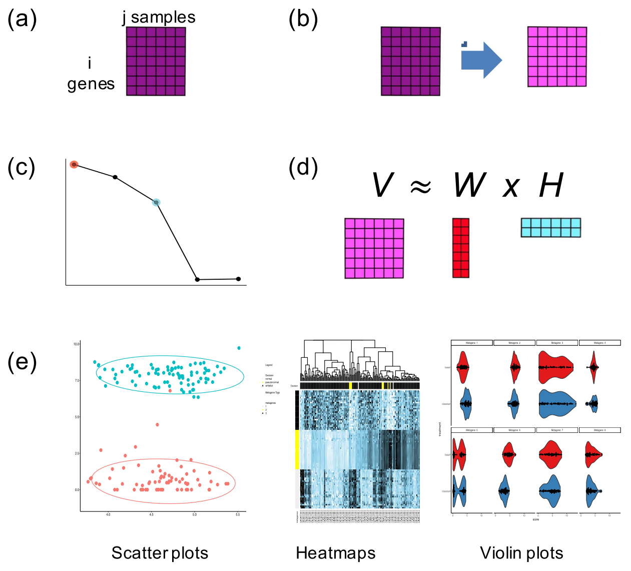

Fig. 1. NMF workflow with FaStaNMF visualizations. (a) Input matrix with gene expressions. (b) A gene expression matrix in its original form is rarely optimal for deconvolution. Clearing rows with variance less than one, since genes that are equally expressed across all samples will contribute nothing significant to the deconvolution, will help speed up the pipeline and reduce noise. Furthermore, performing deconvolution on read counts or expression data that is not transformed with a transformation such as a log transformation or a variance-stabilizing transformation can lead to substantially skewed or slanted results. (c) FaStaNMF provides a convenient visualization and the cophenetic correlation coefficient generator to aid in the selection of k – the optimal factorization rank for NMF. (d) Non-negative matrix factorization (NMF) is performed on the data. (e) FaStaNMF package provides a visualization tools help interpret the results of the deconvolution.

Before running NMF, it is good practice to clean up and normalize the data (Fig. 1(b)). Rows with a variance of less than one can be removed, as genes expressing such little variance across samples will not have a significant effect on the results of NMF, but will contribute to the downstream runtime. Clearing rows with low variance is typically only recommended for gene expression matrices with counts that have not already been transformed. Next, the data is optionally transformed with log(1+x) or a variance stabilizing transformation (VST). A log(1+x) transformation is appropriate prior to NMF as the transformation will output non-negative values for values of zero in the untransformed input matrix. VST will transform the values to become approximately homoscedastic, having constant variance unrelated to their mean values; VST is better at stabilizing variance than log, but should be applied carefully to sparse data [21].

After the optional data transformation, the desirable factorization rank, k, should be selected for NMF. The most common approach to finding an optimal factorization rank is to use the cophenetic correlation coefficient as a measure [18]. Brunet et al. [14] also proposed the cophenetic correlation coefficient as a most appropriate measure for clustering stability. To obtain a cophenetic correlation coefficient, NMF is run multiple times to acquire a ***consensus matrix***, which is the average connectivity matrix over these runs, and the cophenetic correlation coefficient is a measure of the robustness and stability of the clusters using the ***consensus matrix*** [14], [18]. The cophenetic correlation coefficient is between zero and one, and the closer the cophenetic correlation coefficient is to one, the more robust and stable the clusters are [18].

Due to its importance, a cophenetic correlation coefficient plot generator (Fig. 1(c)) is included as an optional preprocessing step for FaStaNMF, displaying the cophenetic correlation coefficient for each factorization rank k in an optionally consecutive range of ranks. The FaStaNMF cophenetic correlation coefficient generator will highlight two ranks for the user: the rank with the maximum cophenetic correlation coefficient and the rank directly before the biggest decrease in slope before the first positive slope in the graph is witnessed. Brunet et al. [14] proposes using a factorization rank where the magnitude of the cophenetic correlation coefficient begins to decrease. The ranks suggested by the FaStaNMF cophenetic correlation coefficient generator and the cophenetic correlation coefficients themselves are mere suggestions, as any rank has the potential to provide insight into the data (see Supplement Fig. 1). However, since its first applications to gene expression data, researchers seeking subtypes in gene expression data have repeatedly used the cophenetic correlation coefficient as a guide for an optimal factorization rank k for NMF [14], [20], [22].

NMF (Fig. 1(d)) works by iteratively modifying W and H until their product approximates the gene expression matrix V. The initial values of W and H (called seed) can be user-specified, initialized randomly, or with the results of algorithms such as Independent Component Analysis or Nonnegative Double Singular Value Decomposition. The objective of the NMF optimization is to minimize the cost function, subject to the non-negativity constraints. The cost function, or the distance between V and W $\times$ H, can be based on the Frobenius Distance or the Kullback-Leibler divergence, and an optional regularization term can be added to enforce smoothness or sparsity properties on W and H [23]. NMF iteratively solves the problem by building matrices W and H that minimize the cost function at each iteration, using a variety of algorithms, including those by [14] and [24].

Finally, a variety of visualizations can be used to interpret the results of the convolution and infer biological meaning. FaStaNMF package provides scatterplot, heatmap, and violin plot generators (Fig. 1(e)) to interpret the results of the deconvolution.

NMF is a stochastic approach, and care must be taken to achieve inter-experiment and inter-lab reproducibility of results. A trivial solution for result reproducibility is to use the same seed, which initializes the random number generators, with subsequent NMF runs, which will result in the identical factorization. A more reasonable approach is to combine setting the seed with running the NMF multiple times (Fig. 2(a)). NMF returns the best fit over all the runs, i.e., the result of the run where the factorization achieves the lowest approximation error. A popular NMF implementation recommends setting the number of runs to nrun = 200 to achieve a stable and reasonable result [18].

While effective, we find several problems with the standard NMF approaches. First, running the entire dataset nrun times does not scale well for increasingly larger datasets. Secondly,

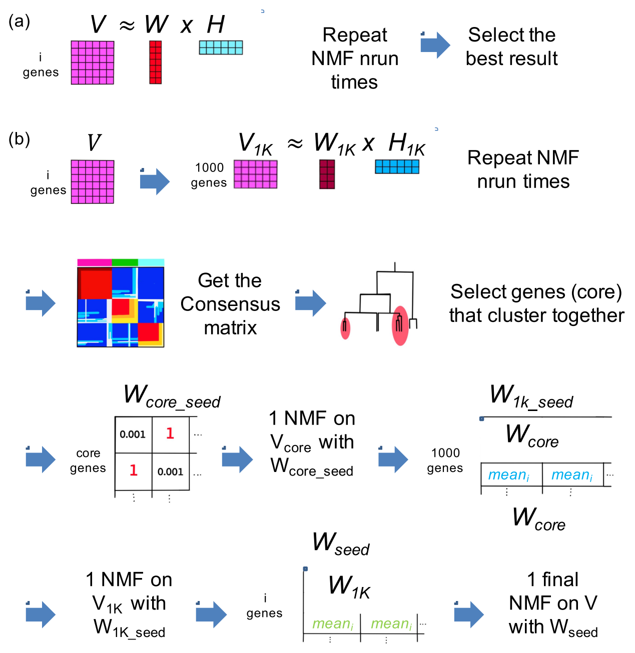

Fig. 2. Regular NMF vs FaStaNMF flow. (a) To achieve a stable and reasonable result, regular NMF is run n times, performing deconvolution on the entire dataset each time, and the best result is selected. (b) FaStaNMF flow performs n deconvolutions on the reduced dataset of 1000 most variable genes and leverages the most impactful genes from the consensus matrix to seed the final NMF runs.

the seed is tied to the implementation and version of NMF, which implies the possibility of different starting points and different results of the optimization when a different version or implementation is used.

When NMF is run multiple times, in order to select the best run which would minimize the cost function, the ***consensus matrix*** - the average connectivity matrix over all these runs – is generated by default. Our idea is to take advantage of the information in the ***consensus matrix*** to identify the most consistently co-clustered genes with which to build a set of metasamples to seed a final deterministic NMF run, to achieve a fast, stable, and accurate result, which is implementation and version agnostic.

B. FaStaNMF Approach

Our approach takes advantage of the NMF ***consensus matrix*** introduced in the previous section. Due to the nature of gene expression profiles, we don't expect most of the features to be informative for the purposes of deconvolution, however, we are interested in acquiring the full expression profiles for each metasample. Briefly, FaStaNMF is run multiple times on a reduced dataset to generate the ***consensus matrix***, and the most impactful genes are selected to seed the final NMF runs (Fig. 2(b)).

First, we select the 1000 most variable genes from the gene expression matrix V, creating V_{1K}. While we empirically found this value to give stable results, we leave it as a parameter with default set to 1000. We run regular NMF deconvolution with nrun = 200 on V_{1K}. The ***consensus matrix*** - the average connectivity matrix over all these runs – is generated by default.

We then use the ***consensus matrix*** to select mutually exclusive sets of genes that respectively cluster together most of the time. Briefly, a dendrogram is generated using hierarchical clustering of the ***consensus matrix***. The dendrogram is iteratively probed at different levels to select genes that cluster together. We start looking for the largest sets of genes that cluster together more than 95% of the time. The subset of gene expressions is then extracted for the combined set of genes, where each gene is among the most frequently clustered genes for one of the factors, is called V_{core}. It is intuitive that the core genes will be among the most variable genes in the set.

Next, we create the NMF seed, $W_{core^{\cdot}seed}$, by setting the values of the core genes for each respective metasample to 1, while setting the rest of the values to 0.001. A single NMF deconvolution is performed on V_{core} with $W_{core^{\cdot}seed}$, providing a good starting point to coax the result toward a solution that reflects the consensus result. We then take the result of this deconvolution, W_{core}, and then extend it to comprise the original 1000 most variable genes, to create the next seed, $W_{1K^{\cdot}seed.}$

At this point, we wanted to ensure that the extra genes we added to the seed would have meaningful values that would not interfere with the scale of the NMF deconvolution. We filled $W_{1K^{\cdot}seed}$, for each added gene *i*, for each factor, with the row-wise gene means for that gene from V, multiplied by the scale factor. The scale factor was calculated by averaging the ratio of the gene-wise row means from W_{core} by the gene-wise row means from V_{core}.

Lastly, we take the result of this deconvolution, W_{1K} and extend it in the manner described in the previous paragraph to create W_{seed}, the seed for the final NMF run. It should be noted that in the implementation, we swapped the orders H and W for the seed, in order to aid in the dendrogram processing step. This design decision necessitated the two-step extension process, in order to produce the final *W* matrix, with each column representing the gene expressions individual to this metasample, and *H* matrix with each row representing the metasample's proportion for each sample.

C. RNAGinesis: Creating Datasets for Benchmarking Genomic Data Deconvolution

Transcriptomic data deconvolution has been performed on many public datasets, leading to new discoveries, such as new pancreatic adenocarcinoma subtypes [22] and work on inflammatory bowel disease [25]. However, in order to benchmark the genomic deconvolution algorithms, it is necessary to have datasets annotated with ground truth – i.e., the gene expression profile for each metasample, as well as the proportion of each metasample in each sample. There is little such publicly available data that does not require extensive pre-processing, with a notable exception of experimental data mixing known quantities of liver lung or brain tissues [26], referred to here as LiverLungBrain.

We have created an infrastructure, called RNAGinesis (https://github.com/rmoffitt/rnaGinesis) for the purpose of generating datasets with known ground truth for the benchmarking of deconvolution datasets. RNAGinesis also provides conversion for existing publicly available datasets (such as LiverLungBrain) into a ***common format***.

While there are multiple use cases for transcriptomic data deconvolution, a lot of recent work has focused on tumor samples [16], [22], [27], [28]. Given recent developments in sub-typing of PDAC [22], it is clear that looking specifically at stromal, tumor, and immune, and normal gene expression individually is crucial to understanding the complex tumor microenvironment.

Our RNAGinesis simulation is modeled on the Pancreatic Ductal Adenocarcinoma (PDAC) transcriptome samples from The Cancer Genome Atlas (TCGA). Among solid cancers, PDAC remains one of the deadliest, with an extremely low 5-year-survival rate of 4%. In addition, only about a third of a tumor sample consists of actual tumor tissue, indicating that the gene expression of a patient's tumor sample is very different from the expression of pure pancreatic tumor cells, a significant issue when it comes to studying the disease [29]. The abundance and composition of these PDAC transcriptome samples make them a good candidate for the simulation modeling.

Several properties of our model can be manipulated to fit a large number of scenarios: the number of cell types (or metasamples), the similarity of the gene expression between cell types, the distribution of the mixture of cell types, how much noise has muddled the data, and the sample size. This allows for the simulation of datasets that represent easiest and most difficult data to decompose and for the evaluation of how these different parameters affect the performance of various methods.

1) Simulation of Sample-Specific Proportions: We simulate the H matrix, with each row representing the cell type's (metasample's) proportion for each sample. The H matrix for k cell types is modeled with a Dirichlet distribution, which is a multivariate beta distribution with probability density function:

$$\frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}, \tag{1}$$

$$where\ B(\alpha) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)} \text{ and } \alpha = \alpha_1, \ldots, \alpha_K.$$

The α parameter can be estimated with known average proportions of the different cell types, which are referred to as p1, p2, $\cdots$, pk. The Dirichlet distribution is dynamic, so the proportion of a specific cell type in a single sample is, on average, equal to the average proportion, but also can deviate from it, creating a more realistically heterogeneous model. For example, if p1 = 0.1 and thus $\alpha 1 = 0.1$, then for 1000 samples, most will contain close to 10% to tissue type 1, but there will also be a few samples composed nearly entirely of that cell type. This is biologically interpretable: if a typical sample contains 10% infiltrate from normal tissue, it is still likely for a few samples in the cohort to be nearly all normal tissue, due to the macroscopic heterogeneity of patient specimens.

The k α parameters can be manipulated to resemble the proportions of tissue types found in different cancers and other mixed tissue samples, making it a flexible model that can be applied to the study of many different diseases. The result of using the Dirichlet distribution to simulate the proportions of k different tissue types in s samples are s draws from the distribution, assembled into H.

2) Simulation of Sample-Specific Expression: Gene expression for each cell type is modeled as

$$W_{ij} \sim N\left(\mu_j, \sum j\right) \tag{2}$$

using a multivariate normal distribution, where for the ith sample and jth cell type, μ_j is the average gene expression of each of the jth cell type genes and Σ_j is the covariance matrix of the genes for the jth cell type. W_{ij} is g × 1, and gene expression of all cell types is W, a g × k matrix.

We downloaded gene expression data from TCGA for pancreatic tumor samples, labeled as PAAD, the TCGA code for PDAC, consisting of the expression levels of over 52000 genes in 150 samples. The data was first filtered to remove any genes with 0 or near-0 expression levels. Selecting the highest expressing 5000 genes in the 150 samples, μ_0, the first metasample expression profile, was done by finding the average gene expression of each gene.

Using μ_0 and Σ_0, which are the average gene expression and covariance matrix for our first cell type, we simulate the averages and covariances for subsequent cell types through rearrangements of μ_0 and Σ_0. Two different rearrangement methods can be used, which will call Methods 1 and 2. Using Method 1, μ_0 is completely permuted to get μ_j and the same rearrangement is used to scramble the same rows and columns of the covariance matrix. This leads to the gene expression of each tissue type being completely orthogonal to the rest, which means there is low correlation between the expressions of the different tissues. On the other hand, with Method 2, μ_0 and Σ_0 are sorted into rank order according to average gene expression, followed by local rearrangement based on a swapping factor. This method conserves rank order and preserves some correlation structure between the gene expression of the different cell types. This is important, as even the most dissimilar mammalian cell types retain highly correlated gene expression profiles.

However, for gene expression data, with increasing numbers of assayed gene variants, the covariance matrix of g^2 elements becomes very large. To implement the generation of multivariate normal data, we employed the Cholesky decomposition of covariance matrix Σ_0. According to the Cholesky decomposition, Σ_0 can be decomposed as follows:

$$\Sigma_0 = AA^T \tag{3}$$

then, the following is true:

$$W_{ij} = A_j Z + \mu_j \tag{4}$$

where Z ∼ N(0, I), or Gaussian data with covariance I (the identity matrix). In practice, we decompose Σ_0 into A once, and the decomposed covariance matrix A is re-ordered either with Method 1 or 2, resulting in A_j for each of the k tissue types. W_{ij} are simulated with j = 1 …k and combined to form $W_i = [W_{i1}, \cdots, W_{ik}]$, which is a g by k. W_i is simulated for sample i = 1 …s. Each W_{ij} is an independent random sample from the multivariate normal distribution of gene expression for the jth tissue type. This means that the jth column of each W_i are sampled from the same distribution with identical μ_j and Σ_j.

3) Simulation of Mixtures and Random Error Term: With $W_i \in R^{g\times k}$ and $H_i \in R^{k\times 1}$ both simulated to serve as the known proportions of the k cell types and the gene expression over g genes, respectively, for the ith sample, the two matrices can be multiplied to obtain simulated mixed gene expression data, which we will call $Y_i \in R^{g\times 1}$. This process is repeated for each of the s simulated samples to result in g × s mixed gene expression data matrix.

In order to introduce observation noise to the gene expression data, we devised a multiplicative error model. The error is denoted by $E_i = (e_{i1}, e_{i2}, \ldots, e_{ig})^T$, where the logarithm of each element in the vector follows the normal distribution N(0,σ). Thus, the general formula for simulating mixture gene expression with random error can be expressed as

$$Y_i = (W_i \times H_i) * E_i \tag{5}$$

where × is the usual matrix multiplication and ∗ is component-wise multiplication. This error model is consistent with RNAseq data which are counts in which variance is proportionate to the expectation.

4) Accuracy Evaluation Methodology and Post-Processing: An important aspect of benchmarking is a consistent evaluation methodology. We use the same methodology to evaluate the accuracy of NMF deconvolution on our Simulation Study (Section II-C5) and on our experiments comparing the accuracy of standard NMF vs FaStaNMF against ground truth.

FaStaNMF, NMF, and other similar deconvolutions output two matrices: one k × s matrix H of the proportion of each sample represented by each of the k cell types and (metasamples) and another g × k matrix W with the expression levels of g genes in each of the k metasamples. H is evaluated against $\widehat{H}$, the true proportions of each tissue type as described in Section II-C1. W is compared against $\widehat{W}$, i.e., $\mu_{1,\cdots,k}$, the average gene expression levels of each of the k cell types for g genes as described in Section II-C2.

The measure of the difference between H and $\widehat{H}$ for each of the k metasamples is defined as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{s}\left(\widehat{H}_i - H_i\right)}{s}} \tag{6}$$

where s is the number of samples. Evaluation of how well the deconvolution result estimates the expression levels of specific cell types in a sample is done with the cosine distance:

$$cosdist\left(W, \widehat{W}\right) = \frac{1 - corr\left(W, \widehat{W}\right)}{2} \tag{7}$$

where the correlation between the simulated gene expression levels for gene g in k tissue type and the level estimated by the convolution serves as the basis of comparison. Because different implementation of deconvolution can produce results that are comparable, but on a different scale (see Section II-B for our FaStaNMF approach that attempts to keep the scale constant), cosine distance, which uses correlation, is more appropriate than the absolute distance measurement when comparing gene expression profiles.

Before accuracy evaluation, there are post-processing steps that must be applied to the results of FaStaNMF, as well as the standard NMF results. The H matrix will have each column representing all the metasamples proportions for a sample. These proportions will be relative, and need to be normalized to sum to 1, for each sample, to be able to properly compare the results to ground truth.

Next, since NMF decomposes the gene expression data into k metasamples arbitrarily, this may not be in the same order as the simulated data, or another dataset with known metasamples and their proportions. H and W are re-ordered based on the best rearrangement that results in lowest error across all samples.

First, we generate all possible permutations of the metasamples order. For each permutation we calculate the sum of all the correlations between W and $\widehat{W}$ across all samples, and the sum of RMSE between H and $\widehat{H}$ across all samples, and multiple the two error sums to obtain the single error score for this permutation. We then selected the permutation with the minimum error score, and globally re-ordered the H and W matrices. We found that dividing the correlations by RMSE and then selecting for the maximum score yielded identical results.

5) Effects of Varying Simulation Parameters: We varied the simulation parameters to assess the effects of similarity of tissue specific gene expression as well as noise on the results of decomposition. We generated five different scenarios with varied amount noise and degree of similarity between the tissues, each of which included 20 simulations of 50 samples each.

For all simulations, we set the α parameters for the H matrix to $\alpha1 = 0.3$, $\alpha2 = 0.4$, $\alpha3 = 0.2$, $\alpha4 = 0.1$. These parameters were chosen based on proportions of tumor, stroma, immune and normal cells generally present in pathology slides for the PDAC tumors. Individual sample proportions in each simulation were modeled with the Dirichlet distribution, the multivariate beta distribution described in Section II-C1.

To generate the gene expression profiles for each sample (the W matrix), we sampled from the multivariate normal distribution as described in Section II-C2. To generate each individual tissue expression vector, we used Method 2 to rearrange μ_0 to get μ_j. The Rearrange Factors used were Low (swapping factor = 2), Medium (swapping factor = 80), and High (swapping factor = 4000). Different levels of noise were simulated by varying the error term σ: Low ($\sigma = 1.01$), Medium ($\sigma = 5$), and High ($\sigma = 10$). Scenarios designed to test the effect of the rearrange on the deconvolution have the Low noise setting, and scenarios designed to test the effect of the noise on deconvolution have the Medium rearrange level.

Fig. 3 demonstrates the effect of different rearrange level settings on the pairwise correlations between the tissues in the simulations. The histogram displays the combined pairwise correlations (Pearson) between means of the tissue expressions for the 20 runs in each of the scenarios (Low, Medium, High). High rearrange setting results in low (0, 0.25) pairwise correlations, setting up an easy to deconvolve scenario. Low rearrange settings yields extremely high (>0.9) correlations, setting up a problem that is more difficult to deconvolve due to extreme similarity between the tissues. Medium Rearrange setting results in mid-range (0.3, 0.7) pairwise wise correlations, creating a problem of medium difficulty.

We evaluate the results of deconvolution with the single run of NMF (nrun = 1), using the metrics described in Section II-C4. The effect of Rearrange Factor on the difficulty of deconvolution are shown in Fig. 4, and confirm the intuition from Fig. 3. Low

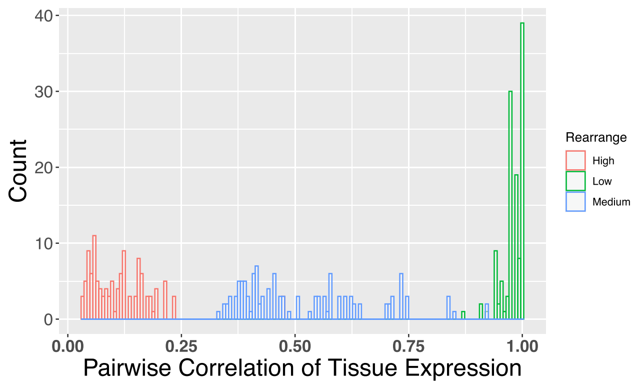

Fig. 3. Histogram of the pairwise correlations of tissue expressions for the simulations with different levels of Rearrange between the tissue expression vectors. High Rearrange setting results in low (0, 0.25) pairwise correlations between simulated tissues, making them very different from each other, and setting up an easy to deconvolve scenario. Low Rearrange settings yields extremely high (>0.9) correlation between simulated tissues, setting up a problem that is more difficult to deconvolve due to extreme similarity between the tissues. Medium Rearrange setting results in mid-range (0.3, 0.75) pairwise wise correlations, creating a problem of medium difficulty.

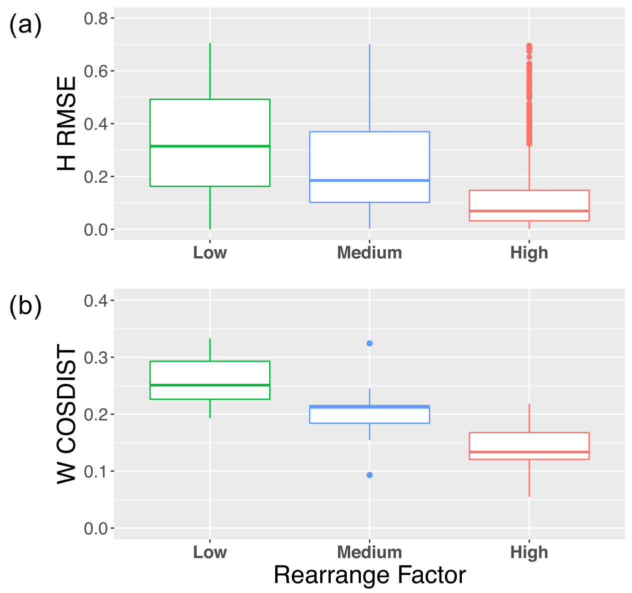

Fig. 4. The effect of rearrange factor on the hardness of deconvolution. Low rearrange results in similar tissue expression profiles, making the problem harder, and resulting in high error both for the H proportions matrix (measured with RMSE), and for the W expressions matrix (measured with the cosine distance). Medium and high rearrange scenarios have respectively lower errors for both the H and the W, which is consistent with their respective medium and low pairwise tissue expression correlations. The higher the Rearrange Factor, the more difference there is between the simulated tissues, resulting in an easier deconvolution problem.

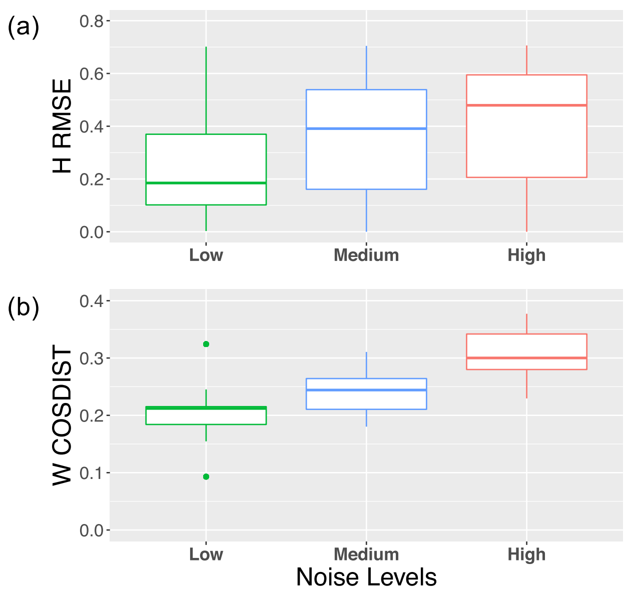

Fig. 5. The effect of Noise Level on the hardness of deconvolution. Increasing the noise levels results in higher error both for the H proportions matrix (measured with RMSE), and for the W expressions matrix (measured with the cosine distance).

Rearrange results in higher error both for the H proportions matrix and for the W expressions. Medium and High Rearrange scenarios have respectively lower errors for both the H and the W. The higher the Rearrange Factor, the more difference there is between the simulated tissues, resulting in an easier deconvolution problem.

The effect of Noise Level on the hardness of deconvolution is demonstrated in Fig. 5. Increasing the noise levels results in higher error both for the H proportions matrix (measured with RMSE), and for the W expressions matrix (measured with the Cosine Distance).

III. Results

Both for the internal FaStaNMF deconvolution and for the standard NMF evaluation, we used the standard NMF R implementation [18]. Unless otherwise specified, default parameters were used for random seeding, as well as for the optimization algorithm (Brunet) [14].

A. Datasets Used

Using the RNAGinesis infrastructure detailed Section II-C, we created and evaluated four datasets to compare FaStaNMF against the standard NMF approach. The Simulation dataset is synthetic data with the first tissue based on the PDAC expression profiles. ResampledSingleCell comprises mixed tissue samples based on sampling of three types of single cells from pancreas scRNAseq. PDACFiveTissue dataset is five compartment mixture sampled from a PDAC single cell sequencing experiment. LiverLungBrain dataset contains microarray expression of liver, brain, and lung tissues in known proportions.

1) Simulation Dataset: The simulated dataset contains 50 samples, with expression levels for 5000 synthetic genes for three synthetic metasamples. We created a toy scenario which would correspond to a very easy deconvolution problem.

Individual sample proportions, were modeled with the Dirichlet distribution, the multivariate beta distribution described in Section II-C1, with the α parameters for the H matrix set to $\alpha1 = 0.3$, $\alpha2 = 0.5$, $\alpha3 = 0.2$.

For this experiment, individual W matrices were created to be identical. The expressions of the first tissue were set to μ_0 from the RNAGinesis simulation based on the PDAC expression profiles. Expressions for the other two tissue were created using Method1 – complete permutation of the values, resulting in orthogonal gene expressions for the metasamples (see Supplement Fig. 4(a)).

2) ResampledSingleCell Dataset: We obtained the data from the single cell transcriptome atlas of the human pancreas [30] using scRNAseq R package. The data contained expression values for 12385 genes across 2126 single cells, corresponding to 10 tissues.

Using the RNAGinesis infrastructure and the raw data from [30], we created a new benchmarking dataset – SingleCellMuraro; SingleCellMuraro has 50 mixed tissue samples with the known ground truth: each sample has the gene expression data matrix, V, and two matrices W and H such that V = W × H. Briefly, we obtained the pure cell type gene expression profiles as centroids from the three most abundant cell types (alpha, beta, and ductal) from this single-cell RNAseq data set, creating three metasamples. We then generated H, W and V matrices for each of the 50 mixed tissue samples.

To create the H matrix, we set $\alpha1 = 0.5$, $\alpha2 = 0.2$, $\alpha3 = 0.3$, representing alpha, beta, and ductal cells, and modeled the individual H proportions for each sample using the Dirichlet distribution, the multivariate beta distribution with probability density function as in Section II-C1.

To create the W matrix for each sample, we randomly sampled 500 cells of each of the three tissue types, and averaged the values for each tissue type, generating individualized metasample expression profiles (see Supplement Fig. 4(b)). Then for each sample j, the mixed expression matrix V_j is calculated, where $V_j = W_j \times H_j$.

3) PDACFiveTissue Dataset: We obtained the data from the PDAC single cell atlas [31], and selected all single cells which were classified as tumor (29K), normal (5K), stroma (21K), immune (2K), and endocrine (1K).

To create the H matrix, we set $\alpha1 = 0.2$, $\alpha2 = 0.2$, $\alpha3 = 0.2$, $\alpha4 = 0.2$, $\alpha5 = 0.2$, representing tumor, normal, stroma, immune, and endocrine cells, and modeled the individual H proportions for each sample using the Dirichlet distribution, the multivariate beta distribution with probability density function as in Section II-C1.

To create the W matrix for each sample, we randomly sampled 500 cells of each of the five tissue types, and averaged the values for each tissue type, generating individualized metasample expression profiles for each of the 50 samples (see Supplement Fig. 4(c)). Then for each sample j, the mixed expression matrix V_j is calculated, where $V_j = W_j \times H_j$.

4) LiverLungBrain Dataset: We downloaded GSE19830 [26], containing 42 mixed samples of liver, brain, and lung tissues in different, known, mixing proportions with three replicates. There are micro-array expression values of 31099 genes for the each of the mixed tissue samples. The data was log-transformed with microarray normalization (RMA).

We converted the data from [26] with RNAGinesis to create the LiverLungBrain dataset in common format. Ground-truth gene expression profiles for the metagenes were obtained by averaging the replicates with 100% proportion of liver, lung, and brain tissue, respectively (see Supplement Fig. 4(d)).

B. Speed

To compare the runtime of FaStaNMF vs. NMF, we measured user time under comparable server utilization settings. For each algorithm, and for each dataset, we evaluate several duplicate nrun settings, ranging from nrun = 100 to nrun = 1000 to generate error bars (Fig. 6).

We see that while NMF outperforms FaStaNMF for datasets with a small number of genes (Fig. 6(a) Simulation and Fig. 6(b) ResampledSingleCell), FaStaNMF is significantly faster for LiverLungBrain, a large dataset which is more representative of typical RNAseq data, as well as of the future datasets (Fig. 6(d)). The transition happens between 12 and 16 thousand genes, as FaStaNMF is faster for PDACFiveTissue (Fig. 6(c)).

NMF spends its time on re-running the algorithm on the entire dataset nrun times. In contrast, FaStaNMF performs nruns on a small, constant size subset of the entire dataset. FaStaNMF spends a significant proportion of time on parsing the dendrogram, whose structure is dependent on the ***consensus matrix***.

To show unambiguously that the difference in speed is due to the number of genes and not the dataset composition, we generated two Gaussian noise datasets (sampling from a distribution with mean = 5000 and sd = 1000) with 5K and 30K genes respectively. Supplement Fig. 5 demonstrates that FaStaNMF is slower for the 5K genes dataset, and faster for 30K genes dataset.

Based on these results, we expect the FaStaNMF speedup to become noticeable when the number of genes is representative of whole-transcriptome experimentation. While the number of known protein coding genes has been fairly stable, the investigation of splice variants is rapidly increasing [32]. For example, the Chess dataset, a new comprehensive collection of human genes, used deep RNA sequencing to observe more than 100000 new gene isoforms and a smaller number of new genes to the previous catalogue of human genes and transcripts [33].

As the number of genes (or features) increases, the performance difference between NMF and FaStaNMF is expected to increase. While filtering datasets before NMF can address speed issues, the ability to generate NMF scores for all genes in a dataset facilitates robust and unbiased downstream applications and analyses.

C. Accuracy

To compare the accuracy of FaStaNMF vs. NMF, we measured the error of deconvolution of the H proportions matrix with RMSE, and the error of deconvolution of the W gene expressions matrix with the cosine distance, as detailed in Section II-C4. We compared the accuracy of a single NMF run (nrun = 1), vs. NMF

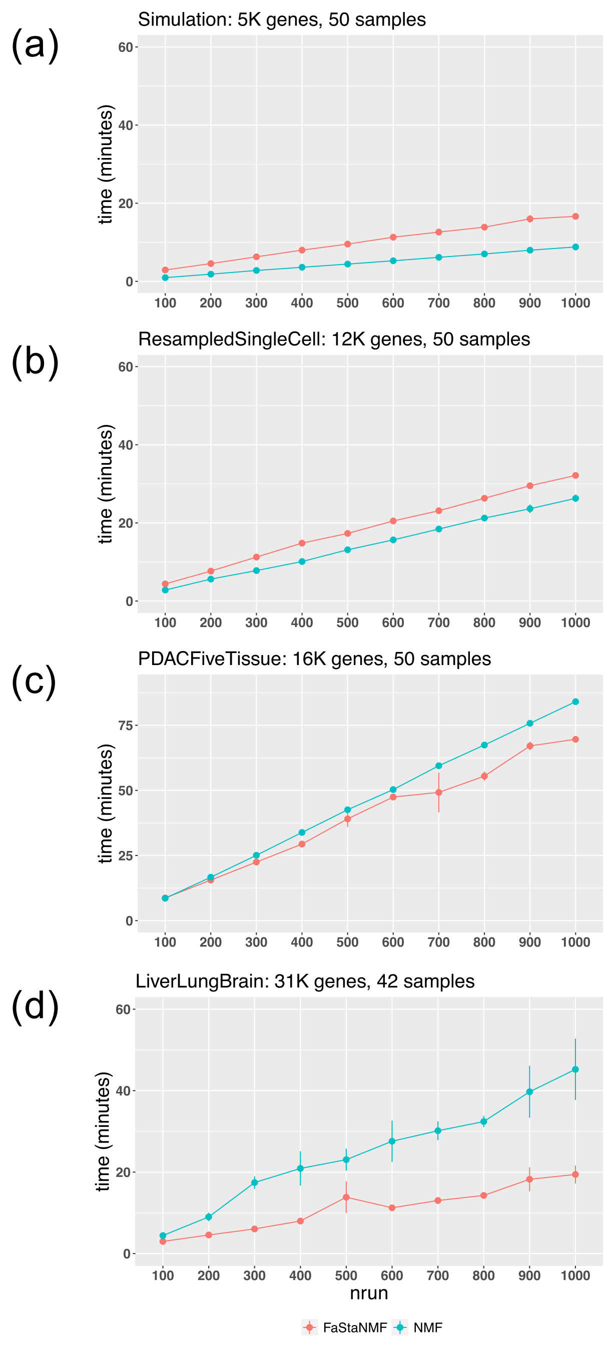

Fig. 6. Speed of FaStaNMF vs. NMF. We measured the runtime in minutes for different nrun settings for each algorithm, with replicate settings to create the error bars. We note that while NMF outperforms FaStaNMF for datasets with a small number of genes (a) simulation and (b) ResampledSingleCell). The transition happens between 12 and 16 thousand genes, as FaStaNMF is faster for (c) PDACFiveTissue. FaStaNMF is significantly faster for (d) LiverLungBrain, a large dataset which is more representative of real experimental scenarios.

with nrun = 200, vs FaStaNMF with nrun = 200 on our four datasets. Each setting was replicated ten times with a different random seed to generate a distribution of results.

Fig. 7 shows, that as expected, a single run of NMF gives the worst accuracy for the H matrix. We demonstrate that the accuracy of FaStaNMF is better — the error is lower — than NMF for three out of four datasets for the same nrun setting. While this is

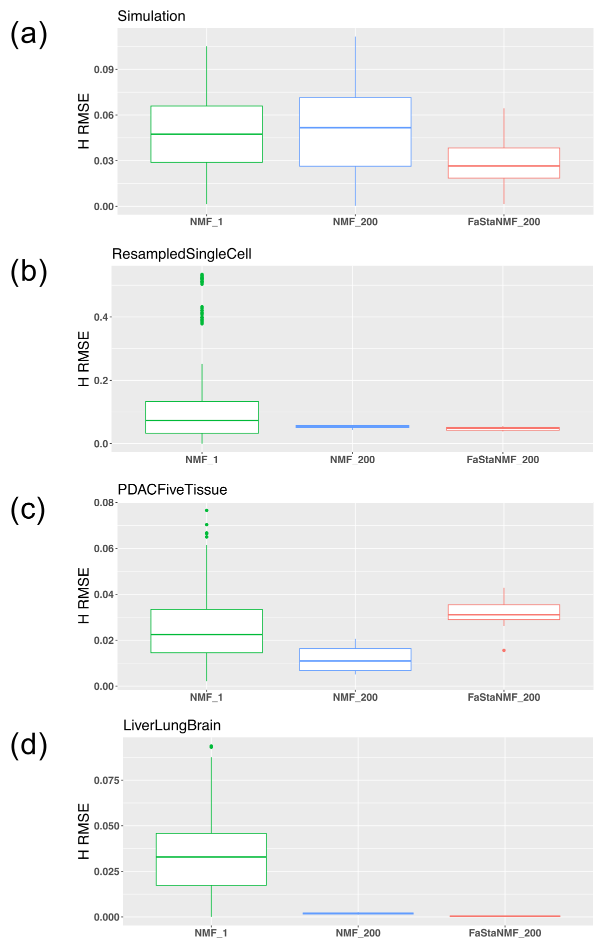

Fig. 7. Accuracy of deconvolution for the H proportions matrix (measured with RMSE). For each of the four datasets, presented in panels (a), (b), (c), and (d), we measured the accuracy of running NMF with nrun = 1, vs. NMF with nrun = 200, vs FaStaNMF with nrun = 200. Each setting was replicated ten times. As expected, a single run of NMF gives the worst accuracy. We demonstrate that the accuracy of FaStaNMF is better – the error is lower – than NMF for three out of four datasets for the same nrun setting. While this is most pronounced for the simulation dataset (a) with p-value $< 2.2e\text{-}16$, a subtle but still visible accuracy improvement exists for the other datasets: ResampledSingleCell (b) with p-value = 0.01, and LiverLungBrain (d) with p-value = 2.5e-06. In this figure we demonstrate the magnitude of error on a zero-based axis; Supplement Fig. 2 zooms in on the difference between FaStaNMF and NMF for nrun = 200.

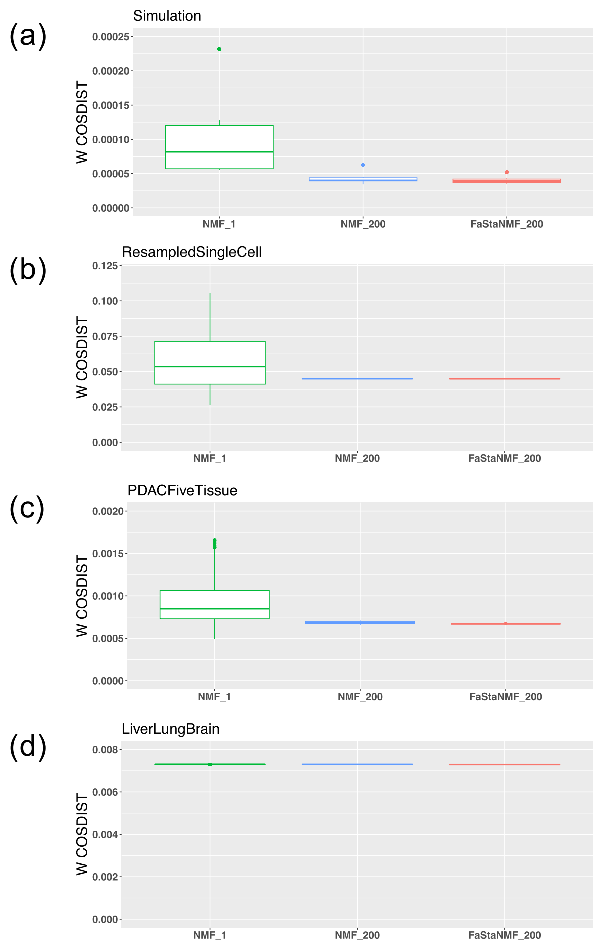

Fig. 8. Accuracy of deconvolution for the W gene expression matrix (measured with cosine distance). For each of the four datasets, presented in panels (a), (b), (c), and (d), we measured the accuracy of running NMF with nrun = 1, vs. NMF with nrun = 200, vs FaStaNMF with nrun = 200. Each setting was replicated ten times. We see that the accuracy of W deconvolution is much higher for W, than for H (see Fig. 7). As expected, a single run of NMF gives the worst accuracy, except for the LiverLungBrain (c), where it performs equally well. For the Simulation dataset (a), FaStaNMF shows small visible improvement over NMF for the same nrun setting (p-value = 1.2e-08), while for the other datasets, the accuracy improvement is minimal, but still statistically significant: p-value = 6.3e-09 for ResampledSingleCell (b), p-value = 0.009 for PDACFiveTissue (c), and p-value = 4.2e-07 for LiverLungBrain (d). In this figure we demonstrate the magnitude of error on a zero-based axis; Supplement Fig. 3 zooms in on the difference between FaStaNMF and NMF for nrun = 200.

most pronounced for the Simulation dataset (Fig. 7(a)), a subtle but still discernable improvement exists for the ResampledSingleCell and LiverLungBrain datasets (Fig. 7(b) and (d)). The p-values for the improvement of accuracy of FaStaNMF_200 over NMF_200 were measured with Welch Two Sample t-test, and showed statistical significance for three datasets, with $p < 2.2e\text{-}16$ for Simulation, p = 0.01 for ResampledSingleCell and p = 2.5e-06 for LiverLungBrain.

Fig. 8 shows that a single run of NMF gives the worst accuracy for the W matrix, except for the LiverLungBrain (Fig. 8(d)), where it performs similarly well. The p-values for the improvement of accuracy of FaStaNMF_200 over NMF_200 were measured with Welch Two Sample t-test, and showed statistical significance for all four datasets. For the Simulation dataset (Fig. 8(a)), FaStaNMF shows small visible improvement with p-value = 1.2e-08, while for the other datasets, the accuracy improvement is minimal, but still statistically significant: p-value = 6.3e-09 for ResampledSingleCell (Fig. 8(b)), p-value = 0.009 for PDACFiveTissue (c), and p-value = 4.2e-07 for LiverLungBrain (Fig. 8(d)).

Overall, we note that the accuracy of deconvolution is better for W than for H. We also demonstrate that overall, FaStaNMF performs better than NMF in terms of accuracy of deconvolution, given the same settings.

D. Stability

Another measure of deconvolution is stability of the top (most prominently expressed) genes for each metasample. The stability of this list is extremely important for the downstream

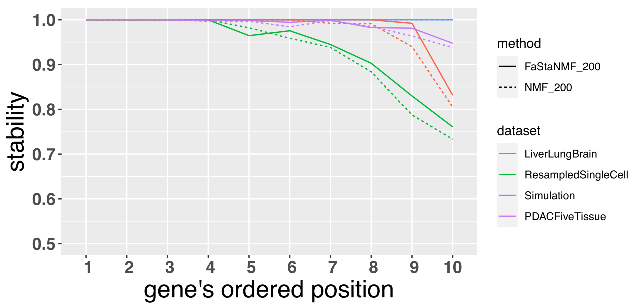

Fig. 9. Stability of genes staying at the top of the ordered metasample list across 50 runs of each method. We show perfect stability for Simulation dataset for both NMF and FaStaNMF in the top 10 positions (lines overlap), which is not surprising given the relative simplicity of the deconvolution problem that was set up, with low pairwise correlations of tissue expression. For the ResampledSingleCell dataset, we see that FaStaNMF has higher stability than NMF for positions 6-10; however, this difference is not statistically significant. For the PDACFiveTissue and LiverLungBrain datasets, FaStaNMF has higher stability than NMF, with the Wilcox paired rank sum test p-value of 0.05 for both.

application; chiefly, interpretation of exactly which cell types are found to exist in a sample.

To select the top genes, each metasample column of W is sorted in descending order of the difference between the value of a gene minus the mean of the values of this gene in the other metasamples. We selected top 10 genes for each metasample for each dataset. To compare FaStaNMF vs. NMF, for each of the datasets, we ran each algorithm 50 times, with different random seeds, and nrun $= 200$, the default values suggested by [18] to achieve stability. We used the results of the 50 runs to calculate the probabilities of stability for each dataset, for each metasample, and for each of the top 10 positions: the probability of a gene present in the 1st position for one of the runs staying in the top 10 positions for the subsequent runs, the probability of a gene present in the 2nd position for one of the runs staying in the top 10 positions for the rest of the runs, etc.

Fig. 9 shows our stability results. We see that we have perfect stability for Simulation dataset for both NMF and FaStaNMF in the top 10 positions, not surprising given the relative simplicity of the deconvolution problem that was set up, with low pairwise correlations of tissue expression. For the ResampledSingleCell dataset, we see that FaStaNMF has higher stability than NMF for positions 6-10; however, this difference is not statistically significant. For the PDACFiveTissue and LiverLungBrain datasets, FaStaNMF has higher stability than NMF, with the Wilcox paired rank sum test p-value of 0.05 for both.

We extended the stability calculations to examine the top 30 and the top 50 positions (see Supplement Fig. 6). In all cases, we found that FaStaNMF is either more stable than NMF with very low p-values, or stability varies but NMF is NOT more stable than FaStaNMF with statistical significance. We note that for the Simulation dataset, stability decreases as we look at the lower ranked positions. Aside from the experimental results presented here, we claim that FaStaNMF is more stable in principle, as it is not dependent on the implement or version of the NMF.

In order to the demonstrate that the stability of tissue-specific compartments is due to the underlying biological stability of gene expression, we ran GSEA on the three deconvolved compartments from the LiverLungBrain dataset. Running GSEA on the C8 collections from MSigDB, we found one compartment enriched for lung, one for brain, and one for gastro-intestinal pathways.

IV. Discussion

Our evaluation of the benefits of FaStaNMF vs. the standard NMF approach used evaluation methodology and datasets created with our RNAGinesis infrastructure and modeled on real biological data. RNAGinesis can simulate gene expression data representing mixtures of multiple cell types with different variances and gene expressions which are tuned for a variety of scenarios. This *in silico* design for the simulation of data with varying properties can be used for benchmarking existing methods and helping develop new ones for mixed-tissue deconvolution and studying tumor heterogeneity.

Our goal in developing FaStaNMF was to improve on NMF, where NMF is currently the best of the sample-composition-agnostic transcriptomic data deconvolution methods. In particular, while the standard NMF approach of setting the random seed and repeating NMF deconvolution a few hundred times on the entire expression matrix does produce a stable and reproduceable result, we found two problems which we overcome with FaStaNMF. One problem is that the reproducibility depends on the starting point, the saved seed, which ties the results to a particular version and implementation of NMF. The other problem is that as transcriptomic datasets are growing in size, the brute force approach of iteratively running NMF on the entire expression matrix will become increasingly time consuming.

We overcome both problems by taking advantage of the information in the ***consensus matrix*** to identify the most informative genes for each metasample and to use them for iterative seeding of the NMF runs. Having the bulk of the computation performed on a small constant-size subset will scale well for increasing transcriptomic datasets, while using a small set of most informative genes as a seed to the final NMF run achieves a stable, and accurate result.

V. Conclusion

FaStaNMF made improvements over NMF in the areas of accuracy, stability, and speed, with different performance gains depending on the size and composition of the data.

For speed, we see that while NMF outperforms FaStaNMF for datasets with a small number of genes, FaStaNMF is significantly faster for a large dataset which is more representative of real and future datasets. Future improvements will focus on making the FaStaNMF core, the parsing of the dendrogram, even faster.

For all four benchmark datasets, FaStaNMF performs better than NMF in terms of accuracy of deconvolution, given the same settings. While in some cases the improvement is tiny, it is statistically significant.

We found that FaStaNMF is either statistically significantly more stable than NMF, or stability varies but NMF is NOT more stable than FaStaNMF with statistical significance. To rephrase, FaStaNMF is better than or equivalent to NMF for stability.

Future areas of improvement include exploring the alternatives to the current error function, which is sensitive to outliers. The effect of outliers can be reduced with the log transformation, but this makes us lose sensitivity on the low end of the spectrum.

Another area of improvement will focus on compartment deconvolution that retains the benefits of NMF, while being able to deconvolve the true profiles of individual patients' tumors. Cancer research and the ability to offer individualized cancer therapies increasingly relies on being able to characterize the unique properties of patients' tumors by assaying the gene expression of individual samples. Implicit in this process is the ability to faithfully assess tumor-specific expression profiles in parallel with information from surrounding tissue or immune cells that are found in patient samples.

Acknowledgment

The results published are partly based on the data generated by the TCGA (https://www.cancer.gov/tcga). This work was supported in part by the Simons Foundation, in part by the URECA Program, and in part by the N S F CRA CI Fellowship.

References

[1] A. Marusyk, V. Almendro, and K. Polyak, "Intra-tumour heterogeneity: A looking glass for cancer?," *Nature Rev. Cancer*, vol. 12, no. 5, pp. 323–334, Apr. 2012.

[2] S. Chung and W. Shen, "Laser capture microdissection: From its principle to applications in research on neurodegeneration," *Neural Regeneration Res.*, vol. 10, pp. 897–898, 2015.

[3] A. Haque, J. Engel, S. A. Teichmann, and T. Lönnberg, "A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications," *Genome Med.*, vol. 9, no. 1, 2017, Art. no. 75.

[4] X. Zheng, N. Zhang, H.-J. Wu, and H. Wu, "Estimating and accounting for tumor purity in the analysis of DNA methylation data from cancer studies," *Genome Biol.*, vol. 18, no. 1, 2017, Art. no. 17.

[5] S. L. Carter et al., "Absolute quantification of somatic DNA alterations in human cancer," *Nature Biotechnol.*, vol. 30, no. 5, pp. 413–421, May 2012.

[6] Z. Wang et al., "Transcriptome deconvolution of heterogeneous tumor samples with immune infiltration," *iScience*, vol. 9, pp. 451–460, 2018.

[7] K. Yoshihara et al., "Inferring tumour purity and stromal and immune cell admixture from expression data," *Nature Commun.*, vol. 4, no. 1, 2013, Art. no. 2612.

[8] Y. Zhong, Y.-W. Wan, K. Pang, L. M. L. Chow, and Z. Liu, "Digital sorting of complex tissues for cell type-specific gene expression profiles," *BMC Bioinf.*, vol. 14, no. 1, 2013, Art. no. 89.

[9] A. M. Newman et al., "Robust enumeration of cell subsets from tissue expression profiles," *Nature Methods*, vol. 12, no. 5, pp. 453–457, 2015.

[10] F. Wagner, "GO-PCA: An unsupervised method to explore gene expression data using prior knowledge," *PLoS One*, vol. 10, no. 11, 2015, Art. no. e0143196.

[11] F. Marini and H. Binder, "pcaExplorer: An R/Bioconductor package for interacting with RNA-seq principal components," *BMC Bioinf.*, vol. 20, no. 1, 2019, Art. no. 331.

[12] A. L. P. Reyes et al., "GENAVi: A shiny web application for gene expression normalization, analysis and visualization," *BMC Genomic.*, vol. 20, no. 1, 2019, Art. no. 745.

[13] P. Perampalam and F. A. Dick, "BEAVR: A browser-based tool for the exploration and visualization of RNA-seq data," *BMC Bioinf.*, vol. 21, no. 1, 2020, Art. no. 221.

[14] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," *Proc. Nat. Acad. Sci.*, vol. 101, no. 12, pp. 4164–4169, 2004.

[15] L. Cantini et al., "Benchmarking joint multi-omics dimensionality reduction approaches for the study of cancer," *Nature Commun.*, vol. 12, 2021, Art. no. 124.

[16] Z. Zhang et al., "Molecular subtyping of serous ovarian cancer based on multi-omics data," *Sci. Rep.*, vol. 6, no. 1, 2016, Art. no. 26001.

[17] X. Zhu, T. Ching, X. Pan, S. M. Weissman, and L. Garmire, "Detecting heterogeneity in single-cell RNA-Seq data by non-negative matrix factorization," *PeerJ*, vol. 5, 2017, Art. no. e2888.

[18] R. Gaujoux and C. Seoighe, "A flexible R package for nonnegative matrix factorization," *BMC Bioinf.*, vol. 11, no. 1, 2010, Art. no. 367.

[19] A. Quintero et al., "ShinyButchR: Interactive NMF-based decomposition workflow of genome-scale datasets," *Biol. Methods Protoc.*, vol. 5, no. 1, 2020, Art. no. bpaa022.

[20] X. L. Peng, R. A. Moffitt, R. J. Torphy, K. E. Volmar, and J. J. Yeh, "De novo compartment deconvolution and weight estimation of tumor samples using DECODER," *Nature Commun.*, vol. 10, no. 1, 2019, Art. no. 4729.

[21] M. I. Love, W. Huber, and S. Anders, "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2," *Genome Biol.*, vol. 15, no. 12, 2014, Art. no. 550.

[22] R. A. Moffitt et al., "Virtual microdissection identifies distinct tumor-and stroma-specific subtypes of pancreatic ductal adenocarcinoma," *Nature Genet.*, vol. 47, no. 10, pp. 1168–1178, 2015.

[23] A. Cichocki, M. Mørup, P. Smaragdis, W. Wang, and R. Zdunek, "Advances in nonnegative matrix and tensor factorization," *Comput. Intell. Neurosci.*, vol. 2008, 2008, Art. no. 852187.

[24] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. 13th Int. Conf. Neural Inf. Process. Syst.*, 2000, pp. 535–541.

[25] M. Vancamelbeke et al., "Genetic and transcriptomic bases of intestinal epithelial barrier dysfunction in inflammatory bowel disease," *Inflammatory Bowel Dis.*, vol. 23, pp. 1718–1729, Oct. 2017.

[26] S. S. Shen-Orr et al., "Cell type–specific gene expression differences in complex tissues," *Nature Methods*, vol. 7, pp. 287–289, 2010.

[27] D. Aran, M. Sirota, and A. J. Butte, "Systematic pan-cancer analysis of tumour purity," *Nature Commun.*, vol. 6, Dec. 2015, Art. no. 8971.

[28] A. Calon et al., "Stromal gene expression defines poor-prognosis subtypes in colorectal cancer," *Nature Genet.*, vol. 47, no. 4, pp. 320–329, Apr. 2015.

[29] A. A. D. H. E. Cancer Genome Atlas Research Network. Electronic address, and N. Cancer Genome Atlas Research, "Integrated genomic characterization of pancreatic ductal adenocarcinoma," *Cancer Cell*, vol. 32, no. 2, pp. 185–203, 2017.

[30] M. J. Muraro et al., "A single-cell transcriptome atlas of the human pancreas," *Cell Syst.*, vol. 3, no. 4, pp. 385–394, 2016.

[31] K. Oh, L. A. Torre-Healy, and R. A. Moffitt, "Integrated analysis of the tumor microenvironment," *GitHub Repository*, 2023. [Online]. Available: https://github.com/rmoffitt/scOh

[32] S. L. Salzberg, "Open questions: How many genes do we have?," *BMC Biol.*, vol. 16, no. 1, 2018, Art. no. 94.

[33] M. Pertea et al., "CHESS: A new human gene catalog curated from thousands of large-scale RNA sequencing experiments reveals extensive transcriptional noise," *Genome Biol.*, vol. 19, no. 1, 2018, Art. no. 208.

Michael D. Sweeney received the BS degree in applied mathematics and statistics and computer science from Stony Brook University in 2021. He is currently working toward the PhD degree with the Department of Biostatistics, University of Michigan School of Public Health. He is a predoctoral Genome Science Training Program trainee under support from the National Human Genome Research Institute of the NIH. His current research interests include bulk RNA-seq cell type deconvolution and droplet-based snRNA-seq data analysis.

Luke A. Torre-Healy received the BA degree in neuroscience from Vanderbilt University, in 2015. He is/was a former research technician with the Lerner Research Institute, Cleveland Clinic Foundation. He is currently working toward the MSTP degree (5th year) with Renaissance School of Medicine, Stony Brook University. He has authored or coauthored 12 peer-reviewed publications, one editorial, one accepted (ISBI 2022), and three in review. His research interests include bringing computational features to clinical impact through histopathology and genomics in cancer. He is/was the National vice president of internal affairs for The Latino Medical Student Association, member and mentor of the American Physician-Scientist Association, and member of the American Association of Cancer Research.

Virginia L. Ma currently working toward the bachelor's in mathematics and the master's degree in statistics with Harvard University. She was a 2018 Simons Summer Research Program Fellow with Stony Brook University. Her research interests include high-dimensional inference, empirical bayes, and applications to genetics and genomics.

Margaret A. Hall received the BS degree in biology from Binghamton University in 2019, and the MS degree in biomedical informatics from Stony Brook University (SBU) in 2021. She is/was a Former Graduate Student Researcher with Stony Brook University. She is currently a Software Developer Programmer/Analyst with SBU.

Lucie Chrastecka received the BS degree in biology from Long Island University in 2016. She is currently working toward the PhD degree (6th Year) in molecular and cellular pharmacology with Stony Brook University. Her research interests include building computational tools for analysis and interpretations of omic data for cancer research.

Alisa Yurovsky received the BS degree in computer science from Carnegie Mellon University, in May 2001, the MS degree in computer science from École Polytechnique Fédérale de Lausanne, in December 2010, and the PhD degree in computer science from Stony Brook University, in December 2020. She is/was a Postdoctoral Fellow with the School of Medicine, Stony Brook University. She has authored or coauthored 15 peer reviewed publications. Alisa's research interests include algorithms in computational biology, statistical analysis of RNA-seq data, Precision Medicine, and mixture modeling. She was the recipient of 2016 NSF Graduate Research Fellowship (GRFP) and 2020 NSF/CRA/CCC Computing Innovation Postdoctoral Fellowship (CIFellows 2020).

Richard A. Moffitt received the BS degree in biomedical engineering and the PhD degree in bioengineering from the Georgia Institute of Technology in 2004 and 2009, respectively. He is currently an assistant professor with the Department of Biomedical Informatics, and Stony Brook Cancer Center, Stony Brook University. He has coauthored more than 50 peer-reviewed manuscripts with an h-index of 28. His research interests include application of deconvolution to admixed biological data, single cell and spatial transcriptomics, and electronic health record analytics. He was the recipient of Postdoctoral Fellowships at both the Center for Cancer Nanotechnology Excellence at Georgia Tech and Emory University, and later at the University of North Carolina Lineberger Comprehensive Cancer Center.