Latitude: A Model for Mixed Linear–Tropical Matrix Factorization

Sanjar Karaev[*] James Hook[†] Pauli Miettinen[*]

Abstract
Nonnegative matrix factorization (NMF) is one of the most frequently-used matrix factorization models in data analysis. A significant reason to the popularity of NMF is its interpretability and the 'parts of whole' interpretation of its components. Recently max-times, or subtropical, matrix factorization (SMF) has been introduced as an alternative model with equally useful 'winner takes it all' interpretation. In this paper we propose a new mixed linear–tropical model and a new algorithm called Latitude that combines NMF and SMF, being able to smoothly alternate between the two. In our model the data is modeled using latent factors and additional latent parameters that control whether the factors are interpreted as NMF or SMF features or as mixtures of both. We present an algorithm for our novel matrix factorization. Our experiments show that it improves over both baselines and can yield interpretable results that reveal more of the latent structure than either NMF or SMF alone.

Keywords: matrix factorization; subtropical algebra; NMF

1 Introduction

Matrix factorizations are a popular method for extracting latent structure from the data. Different factorizations find different types of structure. For example, singular value decomposition (SVD) and principal component analysis (PCA) can be used to find the directions of the greatest variance in the data. In other cases, we might want to decompose the matrix into nonnegative components to gain so-called "parts-of-whole" interpretation. For that, we would use some nonnegative matrix factorization (NMF) algorithm. Or perhaps, instead of taking the sum of the nonnegative components, we are only interested in the largest elements, to gain "winner-takes-it-all" interpretation; for that, we would use subtropical matrix factorizations (SMF). Matrix factorizations are global models, meaning that they apply their structures, be that SVD, NMF, or something else, to the whole matrix. But it is not clear that any data is only a result of a single model. Indeed, it can be that parts of the data are formed using a sum of the rank-1 components, while other parts are formed by taking the element-wise maximums. Consider, for example, the classical example of movie ratings data, that is, people-by-movies matrix containing the movies' ratings. It is often assumed that these ratings have a latent low-rank structure, that there exists some k factors that dictate how much different people like different movies, and that the users' final rating is a linear combination of these factors. For example, Alice might like some movie because she likes the director, the lead actor, and the genre, though she's less keen on the supporting actress. But it is equally plausible that some factor is so dominant, that the rating is dictated by that factor alone. For example, Bob might like all Star Wars movies simply because they are Star Wars movies, and completely irrespective of their director, actors and actresses, or other factors. In this situation, taking the largest value instead of the sum would be a better model. In this paper we present a *mixed linear–tropical model* that allows us to mix NMF and subtropical matrix decompositions. This provides much more accurate decompositions than what we can achieve using either NMF or SMF – or even SVD – alone (see Section 5). That we can improve over the base models, NMF and SMF, indicates that our hypothesis of the data being of mixed structure is correct. In addition to giving a better reconstruction error, our model is also highly interpretable, and uncovers interesting novel structure from the data. Namely, we can study which elements are more NMF-style and which are more SMF-style. Our algorithm for finding the mixed linear–tropical structure is called `Latitude`, as it varies smoothly between the tropic (SMF) and pole (NMF). `Latitude` can be used to decompose relatively large data sets, as it scales linearly with the input data.

Main contributions. In this paper we present a novel matrix factorization model, called mixed linear–tropical model (Section 3) and a scalable algorithm for finding a decomposition in this model (Section 4). Our experiments (Section 5) show that our algorithm finds decompositions that have smaller reconstruction error than what NMF or SMF methods – or even SVD – can find, and that the results are also intuitive and reveal interesting structures from the data sets.

2 Notation and Basic Definitions

In this section we present the basic notation and briefly recall NMF and SMF. We postpone the discussion of the related work to Section 6.

[*]Max Planck Institute for Informatics, Germany.
[†]University of Bath, United Kingdom.
`{skaraev,pmiettin}@mpi-inf.mpg.de`, `j.l.hook@bath.ac.uk`

Basic notation. Throughout this paper we will denote the ith row of matrix $\boldsymbol{A}$ with $\boldsymbol{A}_i$ and the jth column with $\boldsymbol{A}^j$. Element (i,j) of $\boldsymbol{A}$ is $\boldsymbol{A}_{ij}$. We use $\mathbb{R}_+$ to denote the nonnegative real numbers $[0,\infty)$, and $\mathbb{N}$ to denote the natural numbers $\{1,2,\ldots\}$. The Frobenius norm of a matrix $\boldsymbol{A}$ is denoted by $\|\boldsymbol{A}\|_F = \left(\sum_{ij}\boldsymbol{A}_{ij}^2\right)^{1/2}$.

Nonnegative matrix factorization. In nonnegative matrix factorization (NMF), we are given a nonnegative matrix $\boldsymbol{A}\in\mathbb{R}_+^{n\times m}$ and target rank k, and our task is to find nonnegative factor matrices $\boldsymbol{B}\in\mathbb{R}_+^{n\times k}$ and $\boldsymbol{C}\in\mathbb{R}_+^{k\times m}$ that minimize $\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{C}\|_F$. Alternatively, we can write $\boldsymbol{B}\boldsymbol{C}=\sum_{i=1}^k \boldsymbol{B}^i\boldsymbol{C}_i$, where each $\boldsymbol{B}^i\boldsymbol{C}_i$ is a nonnegative rank-1 component matrix. Each component matrix contributes a nonnegative part to the total sum, and it is standard to interpret these rank-1 components as "parts of a whole." Over the years, many different algorithms have been proposed to solve NMF, with methods based on alternating least squares optimization or multiplicative update rules being the most prominent ones (see [4] for a comprehensive treatise).

Subtropical matrix factorization. Subtropical matrix factorization (SMF) is similar to NMF, but it replaces the sum with the maximum in the component formulation: $\boldsymbol{B}\boxtimes\boldsymbol{C}=\max_{i=1}^k\{\boldsymbol{B}^i\boldsymbol{C}_i\}$, where the maximum is taken element-wise. Equivalently, SMF is a matrix factorization over the *subtropical (or max-times) semi-ring*, that is, values $\mathbb{R}_+$ endowed with the addition operation max and the multiplication operation $\times$ (i.e. the standard multiplication). To recap, in SMF, we are given a nonnegative matrix $\boldsymbol{A}\in\mathbb{R}_+^{n\times m}$ and target rank k, and our task is to find nonnegative factor matrices $\boldsymbol{B}\in\mathbb{R}_+^{n\times k}$ and $\boldsymbol{C}\in\mathbb{R}_+^{k\times m}$ that minimize $\|\boldsymbol{A}-\boldsymbol{B}\boxtimes\boldsymbol{C}\|_F=\left\|\boldsymbol{A}-\max_{i=1}^k\{\boldsymbol{B}^i\boldsymbol{C}_i\}\right\|_F$.

Since only the element-wise largest element has effect to the final product, SMF is said to exhibit the "winner-takes-it-all" structure. This tends to yield sparser factor matrices [9, 10]. Note also that $(\boldsymbol{B}\boxtimes\boldsymbol{C})_{ij}\le(\boldsymbol{B}\boldsymbol{C})_{ij}$ for all i and j. Since SMF is taken over the subtropical semiring, it is possible that the factorization obtains smaller reconstruction error than SVD.

It should be noted that the concept of a rank-1 matrix in NMF and SMF coincide, even though rank-k decompositions are generally different. This is a key feature for our model. In *tropical algebra* the summation operation is max and the multiplication is $+$. Hence, matrix $\boldsymbol{A}$ has *tropical rank-1* if there exists vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ such that $\boldsymbol{A}_{ij}=\boldsymbol{a}_i+\boldsymbol{b}_j$. For more on tropical algebra, see Section 6 and references therein.

3 The Mixed Linear–Tropical Model

Rather than describing the data using NMF or subtropical matrix factorization, we propose a hybrid model that incorporates them both and allows for a smooth transition between the two. Ideally, given an input matrix $\boldsymbol{A}\in\mathbb{R}_+^{n\times m}$, we want to be able to determine what elements $\boldsymbol{A}_{ij}$ are better represented using the standard algebra, and which ones require the subtropical one. Namely, we seek factor matrices $\boldsymbol{B}\in\mathbb{R}_+^{n\times k}$ and $\boldsymbol{C}\in\mathbb{R}_+^{k\times m}$ and parameters $\boldsymbol{\alpha}\in\mathbb{R}^{n\times m}$ such that

$$\boldsymbol{A}_{ij}\approx f(\boldsymbol{\alpha}_{ij})(\boldsymbol{B}\boxtimes\boldsymbol{C})+g(\boldsymbol{\alpha}_{ij})(\boldsymbol{B}\boldsymbol{C})_{ij}\ , \tag{3.1}$$

for some functions f and g that we will define below. By representing $\boldsymbol{A}$ as a "mixture" of the normal and subtropical products of the factor matrices, we allow for more flexibility in fitting the data. Since by altering the parameter matrix $\boldsymbol{\alpha}$ we can choose different mixing coefficients for different elements of $\boldsymbol{A}$, it is possible to better explain data that has piecewise NMF and piecewise subtropical structure. Moreover, since the functions $f(\boldsymbol{\alpha}_{ij})$ and $g(\boldsymbol{\alpha}_{ij})$ don't have to be restricted to binary values, we can also express some elements $\boldsymbol{A}_{ij}$ as a weighted sum of $(\boldsymbol{B}\boxtimes\boldsymbol{C})_{ij}$ and $(\boldsymbol{B}\boldsymbol{C})_{ij}$.

It is important to note that the equation (3.1) is quite general, and unless we impose restrictions on functions f and g, as well as the matrix $\boldsymbol{\alpha}$, our model will overfit the data. When it comes to choosing the proper functions f and g, there is a trade-off between fitting the data and keeping the model simple. In this paper we use the convex combination $f(\boldsymbol{\alpha}_{ij})=\boldsymbol{\alpha}_{ij}, g(\boldsymbol{\alpha}_{ij})=1-\boldsymbol{\alpha}_{ij}, \boldsymbol{\alpha}_{ij}\in[0,1]$, which is very simple, and at the same time provides an intuitive transition from the standard product at $\boldsymbol{\alpha}_{ij}=0$ to the subtropical product at $\boldsymbol{\alpha}_{ij}=1$. We obtain

$$\boldsymbol{A}_{ij}\approx\boldsymbol{\alpha}_{ij}(\boldsymbol{B}\boxtimes\boldsymbol{C})+(1-\boldsymbol{\alpha}_{ij})(\boldsymbol{B}\boldsymbol{C})_{ij} \tag{3.2}$$

for $\boldsymbol{\alpha}_{ij}\in[0,1]$.

When choosing $\boldsymbol{\alpha}$ we are faced with a similar trade-off. Indeed, without additional constraints, we can fit arbitrarily complex matrices with *constant* factor matrices, as the following proposition illustrates.

PROPOSITION 3.1. *Let $\boldsymbol{A}\in[1,2]^{n\times m}$ and let $k=4$. There exists $\boldsymbol{\alpha}\in[0,1]^{n\times m}$, $\boldsymbol{B}\in\mathbb{R}_+^{n\times k}$, and $\boldsymbol{C}\in\mathbb{R}_+^{k\times m}$ such that all entries of $\boldsymbol{B}$ and $\boldsymbol{C}$ are the same and that $\boldsymbol{A}_{ij}=\boldsymbol{\alpha}_{ij}(\boldsymbol{B}\boxtimes\boldsymbol{C})+(1-\boldsymbol{\alpha}_{ij})(\boldsymbol{B}\boldsymbol{C})_{ij}$ for all $i=1,\ldots,n$ and $j=1,\ldots,m$.*

Proof. Let all entries of $\boldsymbol{B}$ and $\boldsymbol{C}$ be $\sqrt{3}/2$. Then for any $1\le i\le n$ and $1\le j\le m$ we have $(\boldsymbol{B}\boxtimes\boldsymbol{C})_{ij}=3/4$ and $(\boldsymbol{B}\boldsymbol{C})_{ij}=3$. Now if we set

$$\boldsymbol{\alpha}_{ij}=\frac{\boldsymbol{A}_{ij}-(\boldsymbol{B}\boldsymbol{C})_{ij}}{(\boldsymbol{B}\boxtimes\boldsymbol{C})_{ij}-(\boldsymbol{B}\boldsymbol{C})_{ij}}\ , \tag{3.3}$$

then $0\le\boldsymbol{\alpha}_{ij}\le1$ holds. By plugging (3.3) into (3.2), we obtain $\boldsymbol{\alpha}_{ij}(\boldsymbol{B}\boxtimes\boldsymbol{C})_{ij}+(1-\boldsymbol{\alpha}_{ij})(\boldsymbol{B}\boldsymbol{C})_{ij}=\boldsymbol{A}_{ij}$, concluding the proof.

Being able to decompose essentially arbitrary matrices into constant factor matrices shows that unrestricted $\boldsymbol{\alpha}$ can have too much power. To constrain $\boldsymbol{\alpha}$, we enforce it to have essentially a tropical rank-1 structure:

$$\boldsymbol{\alpha}_{ij} = \sigma(\boldsymbol{\theta}_i + \boldsymbol{\phi}_j) \ , \tag{3.4}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{n\times 1}$ and $\boldsymbol{\phi} \in \mathbb{R}^{1\times m}$ are arbitrary vectors, and $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function.

Now, given the factors $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$, $\boldsymbol{C} \in \mathbb{R}_+^{k\times m}$ and the parameter vectors $\boldsymbol{\theta} \in \mathbb{R}^{n\times 1}$, $\boldsymbol{\phi} \in \mathbb{R}^{1\times m}$, we can define their mixed linear–tropical product, $\boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C}$, elementwise as follows

$$(\boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C})_{ij} = \boldsymbol{\alpha}_{ij}(\boldsymbol{B} \boxtimes \boldsymbol{C})_{ij} + (1-\boldsymbol{\alpha}_{ij})(\boldsymbol{BC})_{ij} \ , \tag{3.5}$$

where $\boldsymbol{\alpha}_{ij} = \sigma(\boldsymbol{\theta}_i + \boldsymbol{\phi}_j)$.

It is trivial to see that when elements in both $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ tend to $-\infty$, we have $\boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C} \to \boldsymbol{BC}$. The greater the element $\boldsymbol{\alpha}_{ij} = \sigma(\boldsymbol{\theta}_i + \boldsymbol{\phi}_j)$ is, the closer the corresponding element in the mixed product is to the subtropical product. In the limit, when all elements of $\boldsymbol{\alpha}$ tend to ∞, we have $\boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C} \to \boldsymbol{B} \boxtimes \boldsymbol{C}$.

We can interpret the values in parameter vectors $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ to give the "typical" level of NMF or subtropical structure associated with the corresponding rows and columns. If, for example, $\boldsymbol{\theta}_i \ll 0$, it means that row i has strong NMF-type structure, while $\boldsymbol{\theta}_i \gg 0$ would mean strongly subtropical structure. If $\boldsymbol{\theta}_i \approx 0$, then the structure is an even mixture of the two. Similarly, if $\boldsymbol{\theta}_i + \boldsymbol{\phi}_j \gg 0$, then the element $\boldsymbol{A}_{ij}$ has subtropical structure, and vice versa for $\boldsymbol{\theta}_i + \boldsymbol{\phi}_j \ll 0$. This interpretation also explains why we use the tropical rank-1 model, that is, summation, instead of the standard rank-1 model $\boldsymbol{\theta}\boldsymbol{\phi}^T$: if we calculate the product, we cannot interpret negative values of $\boldsymbol{\theta}_i$ or $\boldsymbol{\phi}_j$ as indicative of "typically NMF" structure, as if both $\boldsymbol{\theta}_i, \boldsymbol{\phi}_j < 0$, then $\boldsymbol{\theta}_i\boldsymbol{\phi}_j > 0$, indicating subtropical structure.

Now we can define the main problem considered in this paper.

Problem 3.1. *Given an input matrix $\boldsymbol{A} \in \mathbb{R}_+^{n\times m}$ and an integer $k > 0$, find two factor matrices $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$ and $\boldsymbol{C} \in \mathbb{R}_+^{k\times m}$ and parameter vectors $\boldsymbol{\theta} \in \mathbb{R}^{n\times 1}$ and $\boldsymbol{\phi} \in \mathbb{R}^{1\times m}$ such that*

$$E(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{\theta},\boldsymbol{\phi}) = \|\boldsymbol{A} - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C}\|_F \tag{3.6}$$

is minimized.

Unfortunately it seems that the optimization of the above problem is hard:

Proposition 3.2. *Given $\boldsymbol{A} \in \mathbb{R}_+^{n\times m}$, k, $\boldsymbol{\theta}$, and $\boldsymbol{\phi}$, finding $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$ and $\boldsymbol{C} \in \mathbb{R}_+^{k\times m}$ that minimize $E(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{\theta},\boldsymbol{\phi})$ is* NP*-hard. It is also* NP*-hard to find $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$ and $\boldsymbol{C} \in \mathbb{R}_+^{k\times m}$ that approximate $E(\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{\theta},\boldsymbol{\phi})$ to within any polynomially computable factor.*

The proposition is a direct consequence of the NP-hardness of computing or approximating NMF [16] or subtropical matrix factorization [10].

4 The Algorithm

The algorithm `Latitude` (Algorithm 1) finds a mixed linear–tropical matrix decomposition of the given input data.[1] As input it accepts the data matrix $\boldsymbol{A} \in \mathbb{R}_+^{n\times m}$, the rank of the sought decomposition $k \in \mathbb{N}$, and an integer parameter $N \in \mathbb{N}$, that determines the number of iterations of the algorithm. It returns the computed factors $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$ and $\boldsymbol{C} \in \mathbb{R}_+^{k\times m}$ and parameter vectors $\boldsymbol{\theta} \in \mathbb{R}^{n\times 1}$ and $\boldsymbol{\phi} \in \mathbb{R}^{1\times m}$. `Latitude` has also one parameter, $M \in \mathbb{R}_+$. Each element in $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ must belong to the $[-M, M]$ interval. In practice very high values in the parameter vectors do not make sense due to the use of the sigmoid function (see (3.4)) – they would get "smoothed out" and make only marginal changes to the parameter matrix $\boldsymbol{\alpha}$. For this reason for all experiments in this paper we used $M = 5$, at which point $\sigma(M) = 0.9933$, and there is almost nothing to be gained by increasing M further.

The main idea of `Latitude` is to repeatedly use a routine that solves the linear–tropical regression problem to alternatingly update the factor matrices and the parameter vectors. Namely, when the factor matrix $\boldsymbol{B}$ and the parameter vector $\boldsymbol{\theta}$ are fixed, finding the other factor matrix $\boldsymbol{C}$ and parameter vector $\boldsymbol{\phi}$ reduces to solving the problem

$$[\boldsymbol{C}^j, \boldsymbol{\phi}_j] \leftarrow \mathop{\arg\min}_{\boldsymbol{c} \in \mathbb{R}_+^{k\times 1},\, s\in[-M,M]} \|\boldsymbol{A}^j - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},s} \boldsymbol{c}\|_F \tag{4.7}$$

m times (once per column of $\boldsymbol{C}$). Then we fix $\boldsymbol{C}$ and $\boldsymbol{\phi}$ and do the same for $\boldsymbol{B}$ and $\boldsymbol{\theta}$. This process is repeated M times. The algorithm starts by initializing the factor matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ (line 2). This can be done by using random matrices, or, for example, by using some NMF algorithm. Starting with a "pure" NMF solution gives us a reasonable initial solution, and we use that initialization in our experiments. The updates to the factors and parameters are done inside the main loop (lines 12–20), where line 14 updates $\boldsymbol{C}$ and $\boldsymbol{\phi}$, and line 16 updates $\boldsymbol{B}$ and $\boldsymbol{\theta}$. On each iteration we check if the current solution $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{\theta}$, $\boldsymbol{\phi}$ improves on the best one found before that (line 17), and if it does, then we update the best solution and the best error (lines 18–20).

[1]Code is available at `https://people.mpi-inf.mpg.de/~pmiettin/linear-tropical/`

Algorithm 1 Latitude

```
    Input: $\boldsymbol{A} \in \mathbb{R}_+^{n\times m}$, $k \in \mathbb{N}$, $N \in \mathbb{N}$
    Output: $\boldsymbol{B}^* \in \mathbb{R}_+^{n\times k}$, $\boldsymbol{C}^* \in \mathbb{R}_+^{k\times m}$, $\boldsymbol{\theta}^* \in \mathbb{R}^{n\times 1}$, $\boldsymbol{\phi}^* \in \mathbb{R}^{1\times m}$
    Parameters: $M$    ▷ The maximum possible value of parameter vectors. In practice 5 is a good choice
 1: function LATITUDE($\boldsymbol{A}$, $k$, $N$)
 2:     initialize $\boldsymbol{B}$ and $\boldsymbol{C}$
 3:     $\boldsymbol{D} \leftarrow \boldsymbol{BC} - \boldsymbol{A}$
 4:     $\boldsymbol{f}_i \leftarrow \sum_{j=1}^m \boldsymbol{D}_{ij}, \boldsymbol{g}_j \leftarrow \sum_{i=1}^n \boldsymbol{D}_{ij}$
 5:     $\boldsymbol{s}_i \leftarrow$ index of the $i$-th smallest element of $\boldsymbol{f}$
 6:     $\boldsymbol{t}_j \leftarrow$ index of the $j$-th smallest element of $\boldsymbol{g}$
 7:     $\boldsymbol{\theta}_i \leftarrow \frac{i-n}{n-1}M$
 8:     $\boldsymbol{\phi}_j \leftarrow \frac{j-m}{m-1}M$
 9:     $\boldsymbol{B}^* \leftarrow \boldsymbol{B}, \boldsymbol{C}^* \leftarrow \boldsymbol{C}$    ▷ Initialize best factors.
10:     $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}, \boldsymbol{\phi}^* \leftarrow \boldsymbol{\phi}$    ▷ Initialize best parameters.
11:     $bestError \leftarrow \|\boldsymbol{A} - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C}\|_F$
12:     for $iter \leftarrow 1$ to $N$ do
13:         for $j \leftarrow 1$ to $m$ do
14:             $[\boldsymbol{C}^j, \boldsymbol{\phi}_j] \leftarrow$ MixReg($\boldsymbol{A}^j, \boldsymbol{B}, \boldsymbol{C}^j, \boldsymbol{\theta}, \boldsymbol{\phi}_j, M$)
15:         for $i \leftarrow 1$ to $n$ do
16:             $[\boldsymbol{B}_i, \boldsymbol{\theta}_i] \leftarrow$ MixReg($\boldsymbol{A}_i^T, \boldsymbol{C}^T, \boldsymbol{B}_i^T, \boldsymbol{\phi}, \boldsymbol{\theta}_i, M$)
17:         if $\|\boldsymbol{A} - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C}\|_F < bestError$ then
18:             $\boldsymbol{B}^* \leftarrow \boldsymbol{B}, \boldsymbol{C}^* \leftarrow \boldsymbol{C}$
19:             $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}, \boldsymbol{\phi}^* \leftarrow \boldsymbol{\phi}$
20:             $bestError \leftarrow \|\boldsymbol{A} - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},\boldsymbol{\phi}} \boldsymbol{C}\|_F$
21:     return $\boldsymbol{B}^*$, $\boldsymbol{C}^*$, $\boldsymbol{\theta}^*$, $\boldsymbol{\phi}^*$
```

The function MixReg (Algorithm 2) solves problem (4.7), and is where the actual updates to the factors and parameters are performed. It takes as input vector $\boldsymbol{a} \in \mathbb{R}_+^{n\times 1}$, the first factor matrix $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$, an initial solution for the output vector $\boldsymbol{c} \in \mathbb{R}_+^{k\times 1}$, the column parameter vector $\boldsymbol{\theta}$, the starting value for the row parameter element t, and the number $M > 0$ that defines the range of the values in the parameter vectors. It returns the updated versions of the vector $\boldsymbol{c}$ and the element t. Finding the global minimum of (4.7) with respect to both $\boldsymbol{c}$ and t is hard, and hence we update them separately. In fact, even when the parameter t is fixed, optimizing (4.7) with respect to $\boldsymbol{c}$ is problematic. To see that, let us first rewrite (4.7) for a fixed value of t. It becomes

$$\arg\min_{\boldsymbol{c}\in\mathbb{R}_+^{k\times 1}} \|\boldsymbol{a}-(\sigma(\boldsymbol{\theta}+t)\boldsymbol{B}\boxtimes\boldsymbol{C}+(1-\sigma(\boldsymbol{\theta}+t))\boldsymbol{Bc})\|_F \ . \tag{4.8}$$

For every $1 \le i \le n$ denote by $\varphi(i, \boldsymbol{c})$ the index of the largest element in the vector $\boldsymbol{B}_i \Box \boldsymbol{c}^T$, where $\Box$ is the

Algorithm 2 MixReg

```
   Input: $\boldsymbol{a} \in \mathbb{R}_+^{n\times 1}$, $\boldsymbol{B} \in \mathbb{R}_+^{n\times k}$, $\boldsymbol{c} \in \mathbb{R}_+^{k\times 1}$, $\boldsymbol{\theta} \in \mathbb{R}^{n\times 1}$, $t \in \mathbb{R}$, $M > 0$
   Output: $\boldsymbol{c} \in \mathbb{R}_+^{k\times 1}$, $t \in \mathbb{R}$
1: function MIXREG($\boldsymbol{a}$, $\boldsymbol{B}$, $\boldsymbol{c}$, $\boldsymbol{\theta}$, $t$, $M$)
2:     $\boldsymbol{X}_i \leftarrow \boldsymbol{B}_i \Box \boldsymbol{c}^T$
3:     $\boldsymbol{\alpha} \leftarrow \sigma(\boldsymbol{\theta} + t)$
4:     $\boldsymbol{T}_{ij} \leftarrow \begin{cases} 1 & j = \arg\max_{1\le s\le k} \boldsymbol{X}_{is} \\ 1-\boldsymbol{\alpha}_i & \text{otherwise} \end{cases}$
5:     $\boldsymbol{Y} \leftarrow \boldsymbol{B} \Box \boldsymbol{T}$
6:     $\boldsymbol{c} \leftarrow \arg\min_{\boldsymbol{p}\in\mathbb{R}_+^{k\times 1}} \|\boldsymbol{a} - \boldsymbol{Bp}\|_F$
7:     $t \leftarrow \arg\min_{s\in[-M,M]} \|\boldsymbol{a} - \boldsymbol{B} \boxtimes_{\boldsymbol{\theta},s} \boldsymbol{c}\|_F$
8:     return $\boldsymbol{c}$, $t$
```

element-wise (Hadamard) product. We have

$$\begin{aligned} &\boldsymbol{\alpha}_i\boldsymbol{B}_i \boxtimes \boldsymbol{c} + (1-\boldsymbol{\alpha}_i)\boldsymbol{B}_i\boldsymbol{c} \\ &= \boldsymbol{\alpha}_i \max_s\{\boldsymbol{B}_{is}\boldsymbol{c}_s\} + (1-\boldsymbol{\alpha}_i)\sum_s \boldsymbol{B}_{is}\boldsymbol{c}_s \\ &= \boldsymbol{B}_{i\varphi(i,\boldsymbol{c})}\boldsymbol{c}_{\varphi(i,\boldsymbol{c})} + (1-\boldsymbol{\alpha}_i)\sum_{s\neq\varphi(i,\boldsymbol{c})} \boldsymbol{B}_{is}\boldsymbol{c}_s \ , \end{aligned} \tag{4.9}$$

and hence the problem (4.8) is transformed into

$$\arg\min_{\boldsymbol{c}\in\mathbb{R}_+^{k\times 1}} \|\boldsymbol{a}-\boldsymbol{Y}(\boldsymbol{c})\boldsymbol{c}\|_F,\ \boldsymbol{Y}(\boldsymbol{c})_{ij} = \begin{cases} 1 & j = \varphi(i,\boldsymbol{c}) \\ 1-\boldsymbol{\alpha}_i & \text{otherwise} \end{cases} . \tag{4.10}$$

If the coefficient matrix $\boldsymbol{Y}(\boldsymbol{c})$ did not depend on $\boldsymbol{c}$, (4.10) would become a standard nonnegative linear regression problem. Unfortunately, the dependence of $\varphi(i, \boldsymbol{c})$ on $\boldsymbol{c}$ is very complex, and hence it is hard to solve (4.10) directly. In order to overcome this obstacle, we use another heuristic, that is we fix the coefficient matrix $\boldsymbol{Y}(\boldsymbol{c})$, and assume it to be independent from $\boldsymbol{c}$. Under these assumptions $\boldsymbol{c}$ can be found using a standard nonnegative linear regression algorithm. We use the MATLAB built-in lsqnonneg. The matrix $\boldsymbol{Y}$ is built on lines 2–5, and the vector $\boldsymbol{c}$ is found on line 6. Finally, on line 7 we update the parameter t. This is done using the binary search on the interval $[-M, M]$ for the point where the derivative with respect to t is close to 0.

Time complexity. Running Latitude comprises of executing NMF to initialize the factors, and then repeatedly updating them, as well as the parameters, using the MixReg routine. For each $i = 1, \ldots, n$ and $j = 1, \ldots, m$, MixReg is called N times. In order to estimate the complexity of MixReg, it suffices to consider the case when it is called to update $\boldsymbol{C}$ and $\boldsymbol{\phi}$ as the alternate case is analogous (one just needs to replace n by m). Computing the matrix $\boldsymbol{Y}$ (lines 2–5) and finding t (line 7) take time $O(nk)$ each; the latter one because it is enough to make a finite number of steps of the binary search. Thus, if we denote by $\Gamma(n,k)$ the complexity of solving the nonnegative linear regression problem, then the running time of `MixReg` would be given by $O(nk) + \Gamma(n,k)$. Since we use `NMF` to initialize the factors, the runtime of `Latitude` depends on what NMF algorithm is called. If we denote the complexity of `NMF` by $\Pi(n,m,k)$, then the total complexity of `Latitude` is $Nm(O(nk) + \Gamma(n,k)) + Nn(O(mk) + \Gamma(m,k)) + \Pi(n,m,k) = O(Nnmk) + Nm\Gamma(n,k) + Nn\Gamma(m,k) + \Pi(n,m,k)$.

Using `lsqnonneg` for the nonnegative regression and denoting its average number of iterations by ℓ as above, we have that $\Gamma(n,k) = O(\ell nk^2)$. Using projected ALS algorithm [4] for the NMF, each iteration takes $O(nk^2 + mk^2 + nmk)$ time, and we denote the expected number of iterations of the `NMF` algorithm by t. With these choices, the total time complexity becomes $O\big(Nk(nm + lnmk) + tk(nk + mk + nm)\big)$. Importantly, this is linear in the dimensions of the input matrix.

For actual runtime on various real-world datasets see the full version of the paper [8].

5 Experimental Evaluation

In this section we test `Latitude` on both synthetic and real-world data, in order to verify how well it can recover the mixed tropical-linear structure. We also compare it against various benchmark matrix factorization methods.

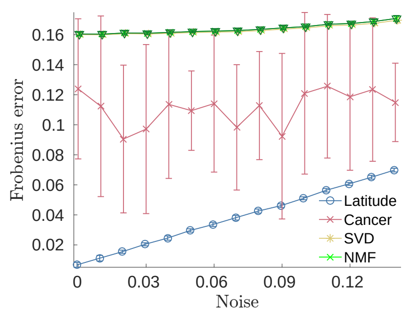
5.1 Other methods. Since `Latitude` is designed to work with data that has a mixture of NMF and SMF structures, it is important to compare against algorithms that target them both. There is a multitude of NMF algorithms, but in this paper we use MATLAB's default implementation `nnmf`, to which we will refer simply as `NMF`. We will also compare against `SVD` in order to get a comparison to optimal rank-k decomposition. Unlike `NMF` or `SVD`, the subtropical matrix factorization is a quite new direction of research, and to the best of our knowledge there are only two available algorithms: `Cancer` [9] and `Capricorn` [10]. Of these, `Cancer` is more suitable due to its ability to handle Gaussian noise, and hence we chose it over `Capricorn`.

5.2 Synthetic Experiments. The purpose of the synthetic experiments is to verify that the proposed algorithms are actually capable of recovering the sought structure when the data conforms to the mixed tropical-linear model. First we generate the data using the mixed tropical-linear structure, then add some Gaussian noise, and finally run the methods to see how much structure they can recover. Unless stated otherwise, the matrices are of size 1000×800 with true rank 10 and values drawn uniformly at random from the $[0,1]$ interval. The factor density is by default 20%, and the standard deviation of the Gaussian noise is 0.01. In order to make sure that after applying the noise the data remains nonnegative, we truncate all values below 0. The parameter vectors $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are drawn uniformly at random from the $[-5,5]$ interval. For the pure subtropical and NMF structure experiments we did not use parameters, but rather multiplied the factors directly. The reconstruction error is always measured against the original, noise-free matrix.

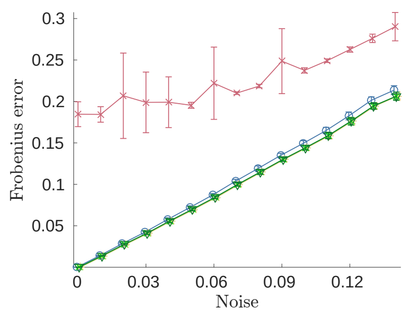
Varying noise with pure subtropical data. (Fig 1a) This experiment tests how well various methods can recover the pure subtropical structure, that is, the extreme case of all parameters being set to ∞. The data is generated by multiplying the factors using the subtropical matrix product. We varied the standard deviation of the Gaussian noise from 0 to 0.14 with increments of 0.01. `Latitude` is clearly the best method, followed by `Cancer`, and `NMF` and `SVD` come close together in the last place. The reason why `Latitude` beats `Cancer` on its own kind of data is that it has more leeway in choosing what structure to use, thus being able to fit everywhere where `Cancer` approximates the data well, but also deviate from the pure subtropical model when needed. `NMF` and `SVD` do not seem to find much structure in this experiment. In this and some other experiments `SVD` and `NMF` produce similar reconstruction errors, which sometimes makes their lines hard to distinguish.

Varying noise with pure NMF data. (Fig. 1b) This setup is analogous to the previous one, except now the data was generated using the pure NMF structure. Here, `NMF` and `SVD` are performing very well, as is expected as the data is generated with the NMF structure. `Latitude`, although having been initialized by `NMF`, only achieves the same results for zero level of noise – then its results start to slowly deteriorate. The cause of this is that it overfits to the noise. Nevertheless, `Latitude`'s results are not much worse than `NMF` or `SVD`, and hence it is definitely applicable to datasets that exhibit the pure NMF structure. Meanwhile `Cancer` is the worst of the methods, which is expected given that the data has pure NMF rather than subtropical structure.

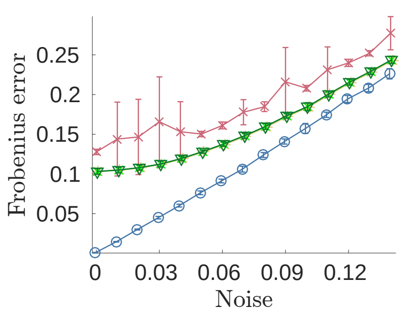
Varying noise with mixed data. (Fig. 1c) Here we test the actual mixed model by using parameters drawn uniformly at random from the $[-5,5]$ interval. This means that the expected value of $\boldsymbol{\theta}_i + \boldsymbol{\phi}_j$ is 0, which corresponds to the midpoint between the NMF and subtropical structures. The randomness ensures that both structures are present in the data. Here `NMF` and `SVD` perform much better than for the pure subtropical case, but `Latitude` is nevertheless the best method by



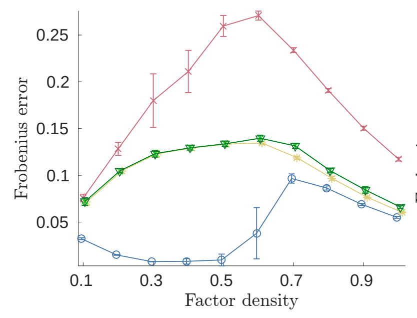
(a) Varying noise with pure subtropical data.



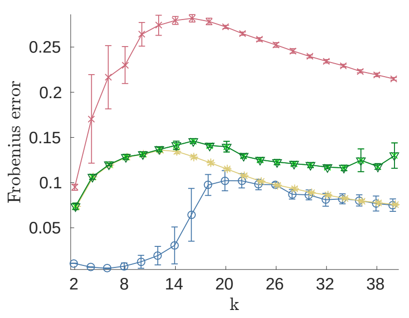
(b) Varying noise with pure NMF data.



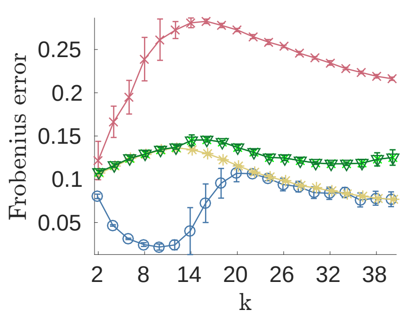
(c) Varying noise with mixed data.



(d) Varying factor density with mixed data.



(e) Varying rank with mixed data.



(f) Varying rank with mixed data a high level of noise.

Figure 1: **Reconstruction errors on synthetic data.** The x-axis represents the varying parameter and the y-axis the Frobenius error. All results are averages over 10 random matrices and the width of the error bars is twice the standard deviation.

a big margin, which demonstrates the advantages of combining both models.

Varying factor density with mixed data. (Fig. 1d) Here we varied the factor density from 10 % to 100 % with increments of 10 %. Again, we have `Latitude` as the best method. There is a peculiar bump on its curve at the very low density level. It can be explained by noise having more influence on sparse data, since then the data/noise ratio is worse.

Varying rank with mixed data. (Fig. 1e) Here we varied the actual rank of the data from 2 to 40 with increments of 2. The factor density was kept at 50 %. As in previous experiments, `Latitude` performs significantly better, especially for lower ranks. The full version of the paper [8] contains another variation of this setup.

Varying rank with mixed data with a high level of noise. (Fig 1f) Same setup as above, but with a higher level of noise (standard deviation 0.07). `Latitude` again performs much better than other methods, albeit having a weird bump for lower ranks. Here again it is explained by lower rank data having also lower density, which exacerbates the effect of the noise. It is worth mentioning that, with the exception of the subtropical data test (Fig 1a), `Cancer` gives the highest reconstruction error. This is not surprising since it aims at recovering the subtropical structure, which is no more present in the data in its pure form.

5.3 Real-World Experiments.

Now that we have evidence that `Latitude` can extract the mixed tropical-linear structure when it is present in the data, we want to see if this kind of structure is also present "in the wild". For that we ran all the competing algorithms on various real-world datasets. First we briefly describe the data, then provide the numerical comparison of the results of the algorithms, followed by some example results.

Datasets. Rather than using raw data, we did some common preprocessing for the real-world datasets. To ensure nonnegativity, we subtract from each column its smallest element. In addition, to make the data more uniform, we divide each column by its standard deviation. These steps are performed on all matrices except 4News, for which we use the TF-IDF model. Climate was obtained from the global climate data repository.[2] It describes historical climate data across different

[2]The raw data is available at `http://www.worldclim.org/`, accessed 18 July 2017.

geographical locations in Europe. Columns represent minimum, maximum, and average temperatures and precipitation in different months, and rows (2575) are 50-by-50 kilometer squares of land where measurements were made. Although temperatures and precipitation are seemingly heterogeneous and have different numeric scales, they are equally important in determining the climate type. To be able to use both of them together, prior to performing the standard preprocessing as with other matrices, we subtract from every column its mean. NPAS is a nerdiness personality test that uses different attributes to determine the level of nerdiness of a person.[3] It contains answers by 1418 respondents to a set of 36 questions that asked them to self-assess various statements about themselves on a scale of 1 to 7. We preprocessed NPAS analogously to Climate. Face is a subset of the Extended Yale Face collection of face images [5]. It consists of 222 32-by-32 pixel images under different lighting conditions. We used a preprocessed data by Xiaofei He et al.[4] We selected a subset of pictures with lighting from the left. 4News is a subset of the 20 Newsgroups dataset,[5] containing the usage of 800 words over 400 posts for 4 newsgroups.[6] Before running the algorithms we transformed the data to TF-IDF values, and scaled by dividing each entry by the greatest entry in the matrix. HPI is a land registry house price index.[7] Rows (253) represent months, columns (177) are locations, and entries are residential property price indices. Further information about these datasets is available in the full version of this paper [8].

Numerical experiments. The reconstruction errors for all the real-world experiments are shown in Table 1. `Latitude` and `SVD` are competing for the first place, with `Latitude` having the best reconstruction error in 2 datasets and `SVD` in 3. All other methods fall significantly behind. It is worth mentioning that `SVD` has an advantage in that it its factors are not restricted to nonnegative values. One can also argue that `Latitude` has more degrees of freedom due to having one additional dimension of parameters. For this reason we also test the truncated version, called `Lat.trunc.`, that was run with $k-1$ dimensions. It is still the third best method (after `SVD` and `Latitude`), beating both `NMF` and `Cancer` by a wide margin. Given these results we can conclude that the mixed tropical-linear structure is present in the datasets that we tested, and that `Latitude` is an appropriate algorithm to extract this structure.

Table 1: Reconstruction error for real-world datasets.

	Climate	NPAS	Face	4News	HPI
$k =$	10	10	40	20	15
`Latitude`	**0.023**	**0.207**	0.157	0.536	0.016
`Lat.trunc.`	0.025	0.213	0.158	0.541	0.017
`SVD`	0.025	0.209	**0.140**	**0.533**	**0.015**
`NMF`	0.080	0.223	0.302	0.541	0.124
`Cancer`	0.066	0.237	0.205	0.554	0.026

Interpretation. In order to validate that our approach also provides interpretable results, we study the results with Climate and Face in more detail. We used the ranks from Table 1. NMF is used in climate models [14], so we would expect this data to have mostly NMF structure, but certain phenomena, such as rainfall, and certain areas, such as mountains or coastal sites, can well have more subtropical structure. To validate this intuition, we can study the parameter vectors $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ and matrix $\boldsymbol{\alpha} = \big(\sigma(\boldsymbol{\theta}_i + \boldsymbol{\phi}_j)\big)_{ij}$. For the Climate data, these are depicted in Figure 2. Recall that for the parameters, negative values indicate NMF-type structure, while positive values indicate subtropical-type structure. Vector $\boldsymbol{\theta}$ corresponds to the geographical locations, and its values are plotted in a map in Figure 2a. As we expected, most of the data has NMF-type structure (depicted as blue), but especially Lapland, Portugal, and some mediterranean coastlines have more subtropical-type structure. These areas probably have some dominating climate phenomena, for example, heavy rainfall or low temperatures, that is best explained using subtropical structure. Vector $\boldsymbol{\phi}$ corresponds to the climate variables. The values in $\boldsymbol{\phi}$ are shown in Figure 2b, where we can see that most variables are negative, that is, they have NMF-type structure. Precipitation is an exception, as the precipitation variables for January and May are in fact positive, indicating more subtropical-type structure. The complete parameter matrix $\boldsymbol{\alpha}$ is shown in Figure 2c. Most elements in the factorization have medium to strong NMF-type structure, but there exist also elements with more subtropical-type structure. The vector $\boldsymbol{\theta}$ for the Face data corresponds to the pixels and is depicted in Figure 3a. It is clear that the dominating features of faces – eyes, nose, and mouth, are best expressed using subtropical-type structure, while the other parts are better explained using NMF-type structure. This is to be expected, as the subtropical areas are those where

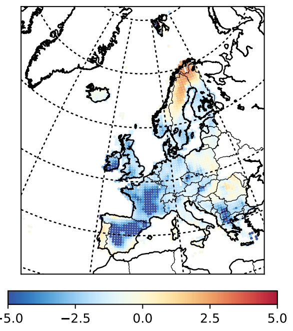
[3]The dataset can be obtained on the online personality website `http://personality-testing.info/_rawdata/NPAS-data.zip`, accessed 18 July 2017.

[4]`http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html`, accessed 18 July 2017

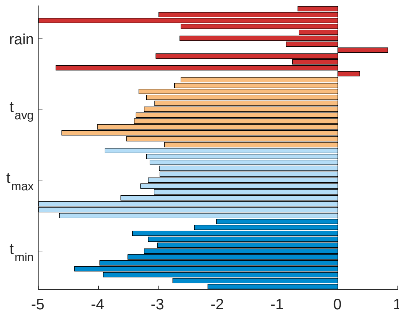
[5]`http://qwone.com/~jason/20Newsgroups/`, accessed 18 July 2017

[6]The authors are grateful to Ata Kabán for pre-processing the data, see [13].

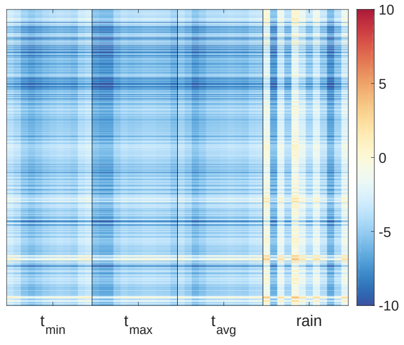
[7]Available at `https://data.gov.uk/dataset/land-registry-house-price-index-background-tables/`, accessed 18 July 2017



(a) Vector $\boldsymbol{\theta}$



(b) Vector $\boldsymbol{\phi}$



(c) Matrix $\boldsymbol{\alpha}$

Figure 2: Visualizations for the parameters in the decomposition of Climate. (a) Values in vector $\boldsymbol{\theta}$ plotted in a map. (b) Values in vector $\boldsymbol{\phi}$ shown as a bar plot. The variables are divided in four groups of twelve months corresponding to minimum, maximum, and average temperature, and precipitation ($\mathrm{t_{min}}$, $\mathrm{t_{max}}$, $\mathrm{t_{avg}}$, and rain, respectively). January is always at the bottom. (c) The matrix $\boldsymbol{\alpha} = \left(\sigma(\boldsymbol{\theta}_i + \boldsymbol{\phi}_j)\right)_{i,j}$. Columns are divided in four groups of twelve months, as in (b). January is always at the left.

lighting has the largest effects (either as bright areas, or areas in shadows, depending on the direction of the light). These extremes are often easiest to describe using the subtropical structure.

Similarly to Climate, we can also plot the matrix $\boldsymbol{\alpha}$.[8] There we notice that there are some faces that have a strong subtropical structure, and again, most of the structure is mostly NMF. To validate that also the factors are interpretable, we present examples from the left factor matrix $\boldsymbol{B}$ for the Face data in Figure 3c. We see that factors mostly depict facial features, except the one at the bottom right, which can be used to add lighting effects to the bottom left part of the figures.

6 Related Work

Nonnegative matrix factorization is a well-studied data analysis method, and over time, many algorithms have been proposed (e.g. [12,14]; see [4] for a comprehensive treatise). NMF has been applied to many data mining problems (e.g. [2,15,17]) and algorithms for computing it are included in all major data analysis packages.

Subtropical (or max-times) matrix factorizations are less commonly used in data analysis – the first such use of approximate low-rank SMF was presented by [10] together with the `Capricorn` algorithm. `Capricorn` is designed for subtropical noise, and later [9] presented the `Cancer` algorithm that deals with Gaussian noise. Recently, `Capricorn` and `Cancer` have been unified under the `Equator` framework [11], that also provided other quality functions than the squared error.

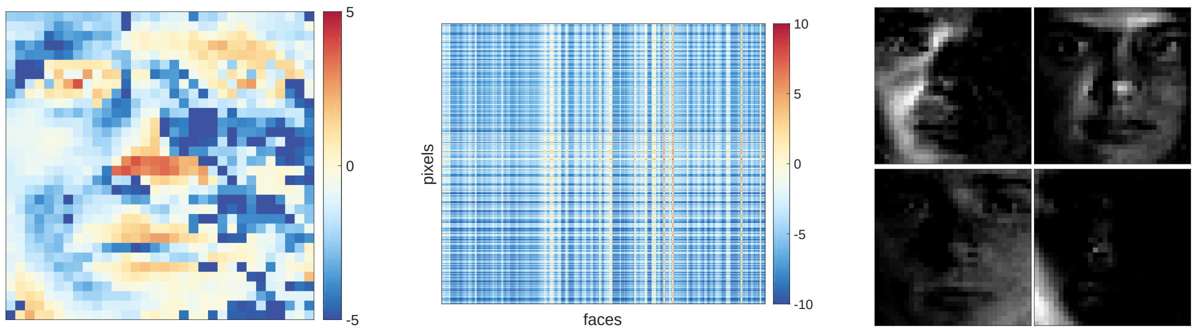
In general a tropical semiring is any semiring in which the addition operation is max or min. Other than max-times two well studied examples are the max-plus and min-plus semirings [3, 6]. Note that the max-plus and min-plus semirings are isomorphic via the map $h(x) = -x$ and that this transformation preserves the norm $d(x, y) = |x - y|$. Note also that max-plus (tropical) and max-times (subtropical) are also isomorphic via $h(x) = \exp(x)$, but that the norm is not preserved by this transformation [10]. This means that whilst the algebraic structures of max-plus and max-times are the same, approximation in max-plus works differently to approximation in max-times. Intuitively max-times gives a lower weight to relative perturbations of smaller numbers. Approximation of network structures by low-rank min-plus matrix factorization is explored in [7]. Possible applications of max-plus low-rank matrix factorizations to non-linear image processing are discussed in [1].

7 Conclusions

Mixed linear–tropical factorization is an interesting novel model for matrix factorization. By smoothly combining factorizations over two algebras, it allows us to model complex structure in an interpretable way. Our algorithm, `Latitude`, was able to consistently obtain better reconstruction errors than either NMF or SMF algorithms. Indeed, `Latitude` was often better than even SVD. And while SVD comes with well-known limitations to the interpretability, `Latitude`'s factorization is easier to interpret due to the nonnegative factor matrices and intuitive interpretation of the parameter vectors.

While `Latitude` generally showed superior performance compared to NMF or SMF, there were a few instances where it performed slightly worse, which was

[8]Plots of the $\boldsymbol{\alpha}$ matrix for the other data sets are in [8].



(a) Vector $\boldsymbol{\theta}$ (b) Matrix $\boldsymbol{\alpha}$ (c) Columns of $\boldsymbol{B}$

Figure 3: (a) Vector $\boldsymbol{\theta}$ for the Face data as an image. (b) Matrix $\boldsymbol{\alpha}$ for the Face data. (c) Four columns of $\boldsymbol{B}$ for the Face data.

due to overfitting to the noise. This raises the question of the use of regularization in mixed linear–tropical factorization and is left for future studies.

`Latitude` has running time which is linear in the input matrix's dimensions, making it a rather scalable method. Its reliance on nonnegative least-squares optimization, however, can be a limiting factor in scaling `Latitude` to big data and distributed systems. Our goal in this paper was to establish the feasibility and usability of mixed linear–tropical models, and developing more scalable algorithms is a natural next step.

In this work we constrained the parameter matrix $\boldsymbol{\alpha}$ to tropical rank-1 (before the logistic transformation). As we saw in Proposition 3.1, some constraints are mandatory for sensible decompositions. It is an interesting open question how much more power would a tropical rank-2 parameter matrix give. Also, the relationship between the rank of the factorization and the rank of the parameter matrix is currently unknown.

References

[1] J. Angulo and S. Velasco-Forero. Non-negative sparse mathematical morphology. *Advances in Imaging and Electron Physics*, 2017.

[2] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proc. Natl. Acad. Sci. U.S.A.*, 101(12):4164–4169, 2004.

[3] P. Butkovič. *Max-Linear Systems: Theory and Algorithms*. Springer, 2010.

[4] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. John Wiley & Sons, Chichester, 2009.

[5] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Generative models for recognition under variable pose and illumination. In *FG '00*, pages 277–284, 2000.

[6] B. Heidergott, G. J. Olsder, and J. van der Woude. *Max Plus at Work: Modeling and Analysis of Synchronized Systems: A Course on Max-Plus Algebra and Its Applications*. Princeton University Press, 2005.

[7] J. Hook. Linear regression over the max-plus semiring: algorithms and applications. *arXiv:1712.03499.*, 2017.

[8] S. Karaev, J. Hook, and P. Miettinen. Latitude: A model for mixed linear–tropical matrix factorization. Technical Report 1801.06136, arXiv, 2018.

[9] S. Karaev and P. Miettinen. Cancer: Another Algorithm for Subtropical Matrix Factorization. In *ECML-PKDD '16*, pages 576–592, 2016.

[10] S. Karaev and P. Miettinen. Capricorn: An Algorithm for Subtropical Matrix Factorization. In *SDM '16*, pages 702–710, 2016.

[11] S. Karaev and P. Miettinen. Algorithms for approximate subtropical matrix factorization. Technical Report 1707.08872, arXiv, July 2017.

[12] D. D. Lee and H. S. Seung. Algorithms for Non-negative Matrix Factorization. In *NIPS '01*, pages 556–562, 2001.

[13] P. Miettinen. *Matrix decomposition methods for data mining: Computational complexity and algorithms*. PhD thesis, University of Helsinki, 2009.

[14] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[15] V. P. Pauca, F. Shahnaz, M. W. Berry, and R. J. Plemmons. Text mining using nonnegative matrix factorizations. In *SDM '04*, pages 22–24, 2004.

[16] S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM J. Optim.*, 20(3):1364–1377, 2009.

[17] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *SIGIR '03*, pages 267–273, 2003.