MATRIX FACTORIZATION IN TROPICAL AND MIXED TROPICAL-LINEAR ALGEBRAS

Ioannis Kordonis[1], *Emmanouil Theodosis*[2], *George Retsinas*[1], *Petros Maragos*[1]

[1]*School of Electrical and Computer Engineering National Technical University of Athens*, Greece
[2]*School of Engineering and Applied Sciences Harvard University Cambridge, MA 02138*
kordonis@central.ntua.gr, etheodosis@seas.harvard.edu, gretsinas@central.ntua.gr , maragos@cs.ntua.gr

ABSTRACT

Matrix Factorization (MF) has found numerous applications in Machine Learning and Data Mining, including collaborative filtering recommendation systems, dimensionality reduction, data visualization, and community detection. Motivated by the recent successes of tropical algebra and geometry in machine learning, we investigate two problems involving matrix factorization over the tropical algebra. For the first problem, Tropical Matrix Factorization (TMF), which has been studied already in the literature, we propose an improved algorithm that avoids many of the local optima. The second formulation considers the approximate decomposition of a given matrix into the product of three matrices where a usual matrix product is followed by a tropical product. This formulation has a very interesting interpretation in terms of the learning of the utility functions of multiple users. We also present numerical results illustrating the effectiveness of the proposed algorithms, as well as an application to recommendation systems with promising results.

Index Terms— Tropical Algebra and Geometry, Matrix Factorization, Dimensionality Reduction, Recommendation Systems

1. INTRODUCTION

Tropical geometry is a research field combining ideas and methods from max-plus algebra (e.g., [1]) with algebraic geometry (see for example [2]). In the last few years, there is a developing interest in the application of tropical geometric ideas and tools to machine learning problems. Some of the applications include the analysis and simplification of piece-wise linear neural networks and the modeling of graphical statistical models. For a review and some recent results see [3].

This paper proposes some ideas and algorithms for matrix factorization over the tropical algebra and over mixed tropical/linear algebras. Matrix Factorization (MF) is a classical topic in Machine Learning and Data mining and MF techniques (e.g. low-rank or nonnegative MF) have found numerous and diverse applications, such as collaborative filtering, dimensionality reduction, data visualization, community detection, blind source separation, and knowledge discovery, to name a few [4].

The contribution of this work is twofold. First, we propose some simple algorithms for Tropical Matrix Factorization (TMF) problem that manage to avoid a large number of locally optimal solutions and compare favorably with algorithms from the literature. Second, we introduce a new matrix factorization problem, that involves approximating a given matrix as a usual product of two matrices, followed by a tropical product with a third matrix. We refer to this problem as the Tropical Compression (TC) problem. This formulation has an interesting interpretation in terms of learning the utility function of multiple users. Particularly, utility functions are usually modeled as concave functions of their arguments (e.g. [5]). We will see that TC formulation can be used to approximate a vector of utility functions with unknown arguments. We will also present an application of the proposed matrix factorizations in recommendation systems.

Related Work: There is some prior work to the TMF problem, that is to approximate a matrix as max-plus product of two matrices with given dimensions. Early applications of TMF include the problem of state space realization of max-plus systems [6]. The exact formulation of TMF can be reduced to an Extended Linear Complementarity Problem (ELCP) [7]. ELCPs also describe the solution of sets of tropical polynomial equations [8]. Unfortunately, the general TMF problem is NP-hard [9]. An approximate technique for TMF was introduced in [6]. The algorithm was extended in [10–12], and some applications in data mining were presented. A closely related algorithm was proposed in [13], for approximating symmetric matrices as the max-plus product of a matrix with its transpose. Algorithms for the related problem of approximate sub-tropical matrix factorization, i.e., matrix factorization over the max-product semi-ring were proposed in [14–16]. For a review of several matrix factorization formulations over non-standard algebras

The research project was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the “2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers” (Project Number:2656, Acronym: TROGEMAL).

Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024), Seoul, Korea, 2024.

see [17].

2. PRELIMINARIES

In this section, we introduce some basic notions of max-plus or tropical algebra. The underlying space is $\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}$. This set is equipped with two binary operations $\vee$ and $+$, where $x \vee y = \max(x, y)$ and $+$ is the usual scalar addition. In this space, maximization has the role of the usual addition and addition the role of usual multiplication. We also consider the vector space $\mathbb{R}^p_{\max}$ where the internal operation $\boldsymbol{x} \vee \boldsymbol{y}$ is defined entry-wise, i.e., $[\boldsymbol{x} \vee \boldsymbol{y}]_i = \max(x_i, y_i)$ and the external operation $\lambda + \boldsymbol{x}$, for $\lambda \in \mathbb{R}_{\max}, \boldsymbol{x} \in \mathbb{R}^p_{\max}$, is defined as $[\lambda + \boldsymbol{x}]_i = \lambda + x_i$.

For a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times p}_{\max}$ and a vector $\boldsymbol{x} \in \mathbb{R}^p_{\max}$, we define the tropical matrix-vector multiplication as

$$[\boldsymbol{A} \boxplus \boldsymbol{x}]_i = \max_j (A_{ij} + x_j). \tag{1}$$

Similarly, for matrices $\boldsymbol{A} \in \mathbb{R}^{m \times p}_{\max}$ and $\boldsymbol{B} \in \mathbb{R}^{p \times n}_{\max}$, we define the tropical matrix multiplication as

$$[\boldsymbol{A} \boxplus \boldsymbol{B}]_{ij} = \max_l (A_{il} + B_{lj}). \tag{2}$$

Tropical polynomials are polynomials in the max-plus algebra. A tropical polynomial function $p : \mathbb{R}^n \to \mathbb{R}_{\max}$ is defined as

$$p(\boldsymbol{x}) = \bigvee_{i=1}^{m_p} (a_i + \boldsymbol{b}_i^T \boldsymbol{x}), \tag{3}$$

where $\boldsymbol{b}_i \in \mathbb{R}^n$, $a_i \in \mathbb{R}_{\max}$. A vector of tropical polynomials is called a tropical map. Observe that a tropical map can be expressed in the form $\boldsymbol{A} \boxplus (\boldsymbol{B}\boldsymbol{x})$, for appropriate matrices $\boldsymbol{A}, \boldsymbol{B}$.

For a matrix $\boldsymbol{A}$ the Frobenius norm is given by $\|\boldsymbol{A}\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$. Finally, we use $\mathbb{1}$ to describe an indicator function, i.e., $\mathbb{1}_{i=j} = 1$ if $i = j$ and zero otherwise.

3. TROPICAL MATRIX FACTORIZATION

Assume that $\boldsymbol{Y}$ is an $n \times p$ matrix. The approximate Tropical Matrix Factorization problem is to find $n \times r$ and $r \times p$ matrices $\boldsymbol{A}, \boldsymbol{B}$, with given $r < \min(n, p)$ that solve the optimization problem

$$\underset{\boldsymbol{A},\boldsymbol{B}}{\text{minimize}} \quad \|\boldsymbol{Y} - \boldsymbol{A} \boxplus \boldsymbol{B}\|_F^2, \tag{4}$$

where $\|\cdot\|_F$ is the Frobenius norm[1]

[1] We could also call the above problem as the Tropical Low Rank matrix approximation problem. However, tropical rank has at least three non-equivalent definitions (see for example [2]). This formulation corresponds to the 'Barvinok rank'. However, to avoid confusion we call it the TMF problem.

We start with a simple Gradient Descent (GD) formulation for the above problem. Observe that the function

$$\boldsymbol{f}(\boldsymbol{A}, \boldsymbol{B}) = \boldsymbol{A} \boxplus \boldsymbol{B}$$

is piecewise linear, and in the generic case, each entry of $[\boldsymbol{A} \boxplus \boldsymbol{B}]_{ij}$ depends on a single pair maximizing entries of $\boldsymbol{A}, \boldsymbol{B}$.

Thus, GD takes the form

$$\pi(i, j) \leftarrow \underset{l}{\text{argmax}} \{A_{il} + B_{lj}\}, \tag{5}$$

$$A_{il} \leftarrow A_{il} - \alpha \sum_j (A_{il} + B_{lj} - Y_{ij}) \mathbb{1}_{l=\pi_k(i,j)} \tag{6}$$

$$B_{lj} \leftarrow B_{lj} - \alpha \sum_i (A_{il} + B_{lj} - Y_{ij}) \mathbb{1}_{l=\pi_k(i,j)} \tag{7}$$

where α is the step-size. In case of many maximizers in (5), assume that one is chosen at random.

In this problem, there is a large number of local minima and stationary points. The partial derivatives with respect to all A_{il} such that $\mathbb{1}_{l=\pi(i,j)} = 0$ for all j, are zero. Thus, if the value of A_{il} is very small, the partial derivative will be always zero and local search would not be able to change it. We call the entries A_{il} of matrix $\boldsymbol{A}$ that do not contribute to any part of $\boldsymbol{A} \boxplus \boldsymbol{B}$ *ineffective*.

We then propose a simple modification of the gradient descent scheme to mitigate this issue

$$A_{il} \leftarrow A_{il} - \alpha \sum_j (A_{il} + B_{lj} - Y_{ij}) s_{i,l,j} \tag{8}$$

$$B_{lj} \leftarrow B_{lj} - \alpha \sum_i (A_{il} + B_{lj} - Y_{ij}) s_{i,l,j}, \tag{9}$$

where $s_{i,l,j} = 1$ if $l = \pi_k(i, j)$ and ϵ_k otherwise. We choose ϵ_k to be small positive constants. The idea behind this modification is that for all the ineffective entries of matrices $\boldsymbol{A}, \boldsymbol{B}$ change and thus have the opportunity in the next iteration to attain the maximum in (5). Note that similar optimization ideas were used in the context of neural network pruning in [18]. We call this method Gradient Decent with Multiplicative Noise (GDMN). We will also study a closely related modification, where we just add a stochastic value ε_k in (6), (7), which we call Gradient Decent with Additive Noise (GDAN). As we shall see in the numerical section, these modifications are surprisingly effective for avoiding bad local minima.

We then shift our attention to the case where $\boldsymbol{Y}$ is partially specified. That is, we do not have access to all the entries Y_{ij} but only for a subset $\mathcal{O} \subset \{1, .., n\} \times \{1, .., p\}$. Then, problem (4) becomes

$$\underset{\boldsymbol{A},\boldsymbol{B}}{\text{minimize}} \quad \|Z_{\mathcal{O}} \circ (\boldsymbol{Y} - \boldsymbol{A} \boxplus \boldsymbol{B})\|_F^2, \tag{10}$$

where $Z_{\mathcal{O}}$ is an $n \times p$ matrix with ones in the entries $(i, j) \in \mathcal{O}$ and zeros elsewhere, and '$\circ$' stands for the Hadamard (element-wise) product.

In this case (8), (9) become

$$A_{il} \leftarrow A_{il} - \alpha \sum_{j:(i,j)\in\mathcal{O}} (A_{il} + B_{lj} - Y_{ij}) s_{i,l,j}$$
$$B_{lj} \leftarrow B_{lj} - \alpha \sum_{i:(i,j)\in\mathcal{O}} (A_{il} + B_{lj} - Y_{ij}) s_{i,l,j},$$

4. THE TROPICAL COMPRESSION PROBLEM

We first define the Tropical Compression (TC) problem. Assume that $\boldsymbol{y}_1, \dots, \boldsymbol{y}_N$ are datapoints in $\mathbb{R}^n$, with $N \geq n$. The tropical compression problem is to find a description of the given dataset as the output of a tropical map. That is, we search for datapoints $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ in $\mathbb{R}^p$ with $p < n$ and matrices $\boldsymbol{B} \in \mathbb{R}^{m\times p}$ and $\boldsymbol{A} \in \mathbb{R}^{n\times m}_{\max}$ that solve the following problem

$$\underset{\boldsymbol{A},\boldsymbol{B},\boldsymbol{X}}{\text{minimize}} \|\boldsymbol{Y} - \boldsymbol{A} \boxplus (\boldsymbol{B}\boldsymbol{X})\|_F^2, \tag{11}$$

where $\boldsymbol{X} = [\boldsymbol{x}_1 \ \dots, \boldsymbol{x}_N]$, $\boldsymbol{Y} = [\boldsymbol{y}_1 \ \dots, \boldsymbol{y}_N]$, and $\boldsymbol{B} = [\boldsymbol{b}_1^T \ \dots \ \boldsymbol{b}_m^T]^T$. This is a factorization problem in a mixed tropical-linear algebras, since a matrix is written as a linear algebraic product of two matrices, followed by a tropical product with a third.

We then present a motivating example. Assume that there is a set of n persons and a set of N items and that the preference of each person towards an item is described by a utility function. Each item has several features and the utility of each user if they receive that item is a piece-wise linear concave function of its features[2]. Assume also that the features of each item i are described by an unknown $p-$dimensional vector $\boldsymbol{x}_i$.

If $\bar{\boldsymbol{Y}}$ is the matrix describing the utility of each person from each item, then $\bar{Y}_{ij}$ can be written as

$$\bar{Y}_{ij} = \min(-\boldsymbol{b}_1^T \boldsymbol{x}_j - a_{i,1}, \dots, -\boldsymbol{b}_m^T \boldsymbol{x}_j - a_{i,m}),$$

where $\boldsymbol{x}_j$ is the vector of characteristics of object j, and $-\boldsymbol{b}_l$'s the slopes of the piecewise linear utility function. Then, $\boldsymbol{Y} = -\bar{\boldsymbol{Y}}$ can be written as

$$\boldsymbol{Y} = \boldsymbol{A} \boxplus (\boldsymbol{B}\boldsymbol{X}),$$

for appropriate matrices $\boldsymbol{A}, \boldsymbol{B}$. Particularly, $\boldsymbol{B}$ contains as rows the slopes of all the different users[3]. In the case where both the features $\boldsymbol{x}_j$ of the objects and the slopes $\boldsymbol{b}_{i,l}$ are unknown, the description of $\boldsymbol{Y}$ reduces to a tropical compression problem.

[2] Let us note that utility functions are very often modeled as concave functions (e.g. [5]). An intuitive reason for this choice is the principle of diminishing marginal utility. Furthermore, piece-wise linear concave functions can approximate arbitrarily well any concave function.

[3] In case where the utility function of some user i does not include a slope $\boldsymbol{b}_l$, then $a_{il} = -\infty$.

4.1. A Numerical Algorithm for the TC Problem

Let us transform (11) into

$$\begin{array}{ll} \underset{\boldsymbol{A}\in\mathbb{R}^{n\times m}_{\max}, \boldsymbol{C}\in\mathbb{R}^{m\times N}}{\text{minimize}} & \|\boldsymbol{Y} - \boldsymbol{A} \boxplus \boldsymbol{C}\|_F^2, \\ \text{subj. to} & \text{rank}(\boldsymbol{C}) \leq p \end{array}. \tag{12}$$

It is not difficult to see that a solution $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{X}$ of (11) corresponds to a solution $\boldsymbol{A}, \boldsymbol{C}$ of (12), with $\boldsymbol{C} = \boldsymbol{B}\boldsymbol{X}$. On the other hand, for a solution $\boldsymbol{A}, \boldsymbol{C}$ of (12), we can perform a rank factorization on matrix $\boldsymbol{C} = \boldsymbol{B'}\boldsymbol{X'}$ where $\boldsymbol{B'} \in \mathbb{R}^{m\times p'}, \boldsymbol{X'} \in \mathbb{R}^{p'\times N}$, and $p' = rank(\boldsymbol{C}) \leq p$. By adding an appropriate number of zero columns in $\boldsymbol{B'}$ and zero rows in $\boldsymbol{X'}$, we obtain a set of matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{X}$ solving (11).

Based on formulation (12), we propose a projected gradient descent type algorithm. The constraint set is non-convex. However, the projection on the set of rank-p matrices can be easily performed using singular value decomposition (see e.g. [19]). The projected version of (5)-(7) becomes

$$\pi(i,j) \leftarrow \underset{l}{\text{argmax}}\{A_{il} + C_{lj}\}, \tag{13}$$

$$A_{i,l} \leftarrow A_{i,l} - \alpha \sum_j (A_{il} + C_{lj} - Y_{ij}) s_{i,l,j}, \tag{14}$$

$$\tilde{C}_{i,l} \leftarrow C_{l,j} - \alpha \sum_i (A_{il} + C_{lj} - Y_{ij}) s_{i,l,j}, \tag{15}$$

$$\boldsymbol{C} \leftarrow \Pi_{\text{rank} \leq p}\left(\tilde{\boldsymbol{C}}\right) \tag{16}$$

where $\Pi_{\text{rank} \leq p}$ is the projection onto the set of matrices with rank less than or equal to p.

Remark 1 *Let us note that if $\boldsymbol{X}$ is known and treated as input, and $n = 1$ the problem reduces to a tropical regression problem [20].*

5. NUMERICAL EXAMPLES

5.1. Synthetic Data

At first we implement the proposed schemes in Python package CuPy (a version of NumPy that allows for GPU acceleration).

5.1.1. Tropical Matrix Factorization

We first present an example that illustrates the usefulness of the proposed modifications GDMN and GDAN. We chose a matrix $\boldsymbol{Y}$ given as

$$\boldsymbol{Y} = \bar{\boldsymbol{A}} \boxplus \bar{\boldsymbol{B}} + a\boldsymbol{R}, \tag{17}$$

where $\bar{\boldsymbol{A}}, \bar{\boldsymbol{B}}, \boldsymbol{R}$ are 10×5, 5×11, and 10×11 matrices the entries of which are chosen as i.i.d. random variables following the uniform distribution on $[0, 1]$, and $a = 0.1$.

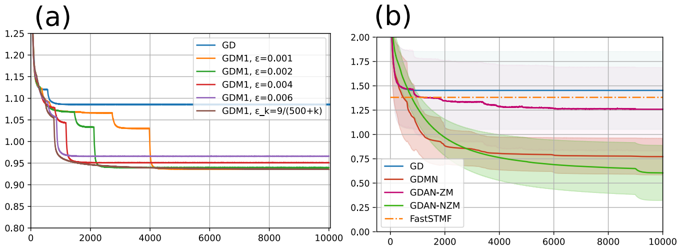

Fig. 1. *(a) The approximation error norm for Algorithm* (8)-(9). *(b) The comparison of the GD scheme and the proposed modifications, with FastSTMF. The results for (b) have been averaged over 10 runs. FastSTMF converges very fast (in* 2 *to* 5 *steps) but does not improve further. Since it uses a different kind of iteration, we included only its final value.*

Figure 1.a compares the norm of the error $\|\boldsymbol{Y} - \boldsymbol{A} \boxplus \boldsymbol{B}\|_F$, where the matrices $\boldsymbol{A}, \boldsymbol{B}$ are computed using Algorithm (8)-(9), with different values of ϵ (recall that ϵ represents the contribution of non-maximizing entries to the algorithm). The modification allows the algorithm to overcome some local optima. Observe that there is a trade-off between convergence speed and quality of solution. With a large value of ϵ, we have faster convergence to a worse solution. Additionally, we used a diminishing scheme for ϵ in the form $\epsilon_k = 9/(500 + k)$, where k is the iteration count. All the results are normalized, that is, we divide the error with the norm $a\|R\|_F$. Figure 1.b compares the modified versions of GD.

We then compare the proposed algorithms with the FastSTMF algorithm from [12]. We factorize multiple 10×11 matrices with $r = 5$. To have a fair comparison, we use as an initial estimate in both algorithms the matrices proposed in [12]. Table 1 shows the matrix factorization error of FastSTMF and compares it with the error of the GD and the proposed variations. We have two implementations of GDAN, with zero mean (GDAN-ZM) and non-zero mean (GDAN-NZM). GDAN-NZM has the best performance between the variants examined. This is probably because it promotes competition between the different non-maximizing entries of matrices $\boldsymbol{A}, \boldsymbol{B}$.

Algorithm	Parameter a		
	0.01	0.1	0.5
FastSTMF [12]	11.2 ± 4.9	1.38 ± 0.52	$.52 \pm .06$
GD	13.2 ± 3.5	1.45 ± 0.40	$.43 \pm .04$
GDMN	06.3 ± 4.1	0.77 ± 0.19	$.39 \pm .04$
GDAN-ZM	11.4 ± 2.6	1.26 ± 0.43	$.42 \pm .04$
GDAN-NZM	$\mathbf{4.5 \pm 2.1}$	$\mathbf{0.60 \pm 0.28}$	$\mathbf{.34 \pm .05}$

Table 1. Comparison of the Algorithms: The normalized Frobenius error, for various values of a.

5.2. Real Data

5.2.1. Movielens 100k Dataset

We use the Movielens 100k Dataset [21], consisting of the ratings of 943 users to 1682 movies. There are in total 100000 ratings. Here we use the implicit feedback formulation. That is, we consider a matrix $\boldsymbol{Y}$ with a value of -1 if the person has watched a movie and $+1$ if they haven't.

We then use a factorization of matrix $\boldsymbol{Y}$. We split the data into 80% training, 10% validation, and 10% test, and apply a stochastic version of GD, and early stopping (note that we observe only a part of the ratings). We use two metrics, the RMS error and the Hit Rate at 10 (HR@10)[4].

The best approximation comes for an intermediate dimension $r = 35$ and has an RMS error equal to 0.396 in the test set and HR@10 is 0.755. We then consider the TC formulation of the problem (11), with $m = 40$ and $p = 25$. Using the modified projected gradient descent algorithm (13)-(16) we get an RMS error equal to 0.391 and HR@10 equal to 0.77. Compared to the TMF formulation, TC performs slightly better. It has also a smaller number of parameters and an intuitive interpretation.

5.2.2. Movielens 1M Dataset

We then turn to a larger dataset, Movielens 1M, with 1 million ratings from 6000 users on 4000 movies. We formulate matrix $\boldsymbol{Y}$, as in the previous subsection. Then, using the same train/validation/test split, we compute an approximate tropical factorization for matrix $\boldsymbol{Y}$, with $r = 40$. Then, the RMS error becomes 0.328 and the HR@10 becomes 0.742. For comparison, a carefully optimized and regularized linear factorization gives HR@10 equal to 0.731 [22]. For a TC formulation with $m = 100$ and $p = 35$, the RMS error becomes 0.327 and HR@10 becomes 0.743.

6. CONCLUSION AND FUTURE WORK

This paper formulates two matrix factorization problems, over the tropical algebra and over mixed linear tropical algebras respectively. For the first problem, we proposed some variations of Gradient Descent that lead to improved performance and compare favorably with an algorithm from the literature. For the second problem which, has an interesting interpretation, in terms of learning the utility function of a set of users, we proposed a non-convex projection gradient descent algorithm. The proposed algorithms were applied to a recommendation problem, using datasets MovieLens 100k and 1M, with promising results.

Some interesting directions for further research are the use of appropriate regularization techniques and the study of sparse approximate solutions for the TMF problem.

[4]HR@10 is defined as follows. For each user, form a list of 101 items choosing randomly from the test set 1 positive and 100 negative items. Then, count the number of users for which the positive item is ranked among the first 10 of the list [22].

7. REFERENCES

[1] P. Butkovič, *Max-linear systems: theory and algorithms*. Springer Science & Business Media, 2010.

[2] D. Maclagan and B. Sturmfels, "Introduction to tropical geometry," *Graduate Studies in Mathematics*, vol. 161, pp. 75–91, 2009.

[3] P. Maragos, V. Charisopoulos, and E. Theodosis, "Tropical geometry and machine learning," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 728–755, 2021.

[4] K.-L. Du, M. Swamy, Z.-Q. Wang, and W. H. Mow, "Matrix factorization techniques in machine learning, signal processing, and statistics," *Mathematics*, vol. 11, no. 12, p. 2674, 2023.

[5] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica: Journal of the Econometric Society*, pp. 520–534, 1965.

[6] B. De Schutter and B. De Moor, "Matrix factorization and minimal state space realization in the max-plus algebra," in *Proceedings of the 1997 American control conference (Cat. No. 97CH36041)*, vol. 5. IEEE, 1997, pp. 3136–3140.

[7] ——, "Minimal realization in the max algebra is an extended linear complementarity problem," *Systems & Control Letters*, vol. 25, no. 2, pp. 103–111, 1995.

[8] ——, "The extended linear complementarity problem," *Mathematical Programming*, vol. 71, no. 3, pp. 289–325, 1995.

[9] Y. Shitov, "The complexity of tropical matrix factorization," *Advances in Mathematics*, vol. 254, pp. 138–156, 2014.

[10] A. Omanović, P. Oblak, and T. Curk, "Application of tropical semiring for matrix factorization," *Uporabna informatika*, vol. 28, no. 4, 2020.

[11] A. Omanović, H. Kazan, P. Oblak, and T. Curk, "Sparse data embedding and prediction by tropical matrix factorization," *BMC bioinformatics*, vol. 22, no. 1, pp. 1–18, 2021.

[12] A. Omanović, P. Oblak, and T. Curk, "Faststmf: Efficient tropical matrix factorization algorithm for sparse data," *arXiv preprint arXiv:2205.06619*, 2022.

[13] J. Hook, "Min-plus algebraic low rank matrix approximation: a new method for revealing structure in networks," *arXiv preprint arXiv:1708.06552*, 2017.

[14] S. Karaev and P. Miettinen, "Capricorn: An algorithm for subtropical matrix factorization," in *Proceedings of the 2016 SIAM international conference on data mining*. SIAM, 2016, pp. 702–710.

[15] ——, "Cancer: Another algorithm for subtropical matrix factorization," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part II 16*. Springer, 2016, pp. 576–592.

[16] ——, "Algorithms for approximate subtropical matrix factorization," *Data Mining and Knowledge Discovery*, vol. 33, pp. 526–576, 2019.

[17] S. Karaev, "Matrix factorization over dioids and its applications in data mining," 2019.

[18] G. Retsinas, A. Elafrou, G. Goumas, and P. Maragos, "Weight pruning via adaptive sparsity loss," *arXiv preprint arXiv:2006.02768*, 2020.

[19] P. Jain, P. Kar *et al.*, "Non-convex optimization for machine learning," *Foundations and Trends® in Machine Learning*, vol. 10, no. 3-4, pp. 142–363, 2017.

[20] P. Maragos and E. Theodosis, "Multivariate tropical regression and piecewise-linear surface fitting," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3822–3826.

[21] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[22] V. W. Anelli, A. Bellogín, T. Di Noia, and C. Pomo, "Reenvisioning the comparison between neural collaborative filtering and matrix factorization," in *Proceedings of the 15th ACM Conference on Recommender Systems*, 2021, pp. 521–529.