# Supervised Quantile Normalization for Low-rank Matrix Approximation

**Marco Cuturi** [1]   **Olivier Teboul** [1]   **Jonathan Niles-Weed** [2]   **Jean-Philippe Vert** [1]

## Abstract

Low rank matrix factorization is a fundamental building block in machine learning, used for instance to summarize gene expression profile data or word-document counts. To be robust to outliers and differences in scale across features, a matrix factorization step is usually preceded by ad-hoc feature normalization steps, such as `tf-idf` scaling or data whitening. We propose in this work to learn these normalization operators jointly with the factorization itself. More precisely, given a $d \times n$ matrix $X$ of $d$ features measured on $n$ individuals, we propose to learn the parameters of quantile normalization operators that can operate row-wise on the values of $X$ and/or of its factorization $UV$ to improve the quality of the low-rank representation of $X$ itself. This optimization is facilitated by the introduction of a new differentiable quantile normalization operator built using optimal transport, providing new results on top of existing work by (Cuturi et al. 2019). We demonstrate the applicability of these techniques on synthetic and genomics datasets.

## 1. Introduction

The vast majority of machine learning problems start with a matrix $X \in \mathbb{R}^{d \times n}$ of measurements that keeps track of $d$ features measured on $n$ individuals. An important way to summarize the information contained in $X$ is to find a low-rank matrix factorization, namely two matrices $U$ and $V$ of sizes $d \times k$ and $k \times n$ such that $UV \approx X$, as quantified in a relevant matrix norm. While this problem is known to boil down to the truncated singular value decomposition of $X$ when the norm is Euclidean, the two most recent decades have succeeded in producing extremely useful variations on that problem, handling for instance the cases in which the entries of $X$ are non-negative (Lee & Seung, 1999; Hof-

mann, 2001; Févotte & Idier, 2011), binary (Slawski et al., 2013) or even describe rank values (Le Van et al., 2015); considering various forms of sparse priors on the factors themselves (d'Aspremont et al., 2005; Mairal et al., 2010; Jenatton et al., 2011; Witten et al., 2009); and extending these problems to cases where the matrices are incomplete (Koren et al., 2009; Candès & Recht, 2009).

**Low Rank Approximations.** Let $\mathbb{R}_k^{d \times n}$ be the set of $d \times n$ matrices of rank $k$. Choosing a divergence $\Delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined on a subset of matrices $\mathcal{X} \subset \mathbb{R}^{d \times n}$, and $\mathcal{Z}_k \subset \mathcal{X} \cap \mathbb{R}_k^{d \times n}$, one can introduce the operator

$$\Pi_k(X) := \underset{Z \in \mathcal{Z}_k}{\operatorname{argmin}} \Delta(X, Z).$$

While a very large literature has focused on considering various divergences $\Delta$, such as Frobenius, KL (Lee & Seung, 1999), Beta-divergences (Févotte & Idier, 2011) or Wasserstein (Rolet et al., 2016); and sets $\mathcal{Z}_k$ (sparse, non-negative), an important practical limitation of these approaches is that they perform well if the values described in $X$ have a distribution that is somewhat shared across features: Because the discrepancy $\Delta$ is usually additive, the loss $\Delta$ can be impacted by differences in ranges. This problem is addressed by "massaging" the entries of $X$ first, notably through ad-hoc normalization schemes carried out feature-by-feature, such as taking logarithms for gene expression data (Risso et al., 2018) or using `tf-idf` schemes for text data, before feeding this modified matrix $\tilde{X}$ to the projector $\Pi_k$.

**Increasing Feature Rescaling.** We propose in this paper to automatically learn such a renormalization, rather than leave to the user the arduous choice of selecting a suboptimal method. We also claim that we can gain interpretability by then finding out which features seem to be inflated / deflated to improve factorization. Our approach allows for an increasing map, defined for each of the $d$ features, to be applied to all $n$ values of the row of a matrix, either prior to and/or after the factorization step. The benefit of increasing maps is that they preserve the relative order of samples, which is an important point for interpretation. Denoting by $\mathcal{T}$ the set of increasing maps from $\mathbb{R}$ to $\mathbb{R}$, we consider a family of $d$ such maps $\mathbf{T} = (T_1, \ldots, T_d) \in \mathcal{T}^d$, to which we associate (using the same symbol) an operator applying each map $T_i$ to the corresponding row of a $d \times n$ matrix $W$:

$$\mathbf{T}(W) = [T_i(W_{ij})]_{ij} \in \mathbb{R}^{d \times n}.$$

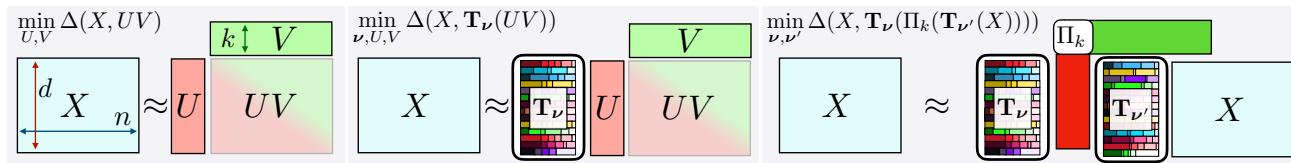[1]Google Research, Brain Team [2]New York University. Correspondence to: Marco Cuturi <cuturi@google.com>.

*Figure 1. (Left)* Schematic description of classic matrix factorization, with rank $k$ factor matrices $U, V$. *(Middle)* QMF, the approach we propose to re-normalize the values of each single row $i$ of a factorization $UV$ a quantile renormalization operator defined per feature, using a target measure $\nu_i$. Each measure $\nu_i$ is described by a probability vector $\mathbf{b}_i$ of size $m$ and a vector $\mathbf{q}_i$ of $m$ quantiles (increasing values) at those levels. $\boldsymbol{\nu}$ stores all of these distributions. *(Right)* Assuming one has access to a (differentiable) projector $\Pi_k$, we can require additionally that $X$ be itself the quantile normalization of a reconstruction of itself, after another quantile normalization operation. Here $\mathbf{T}_{\boldsymbol{\nu}'}$ acts as a "deflating" mapping done to facilitate factorization, "reinflated" by $\mathbf{T}_{\boldsymbol{\nu}}$ to yield the best possible reconstruction.

Our goal is to find one (or possibly two) maps $\mathbf{T}$ that can work hand-in-hand with matrix factorization to minimize reconstruction error. We consider first *quantile matrix factorization* (QMF), which minimizes $\Delta(X, \mathbf{T}(UV))$ jointly in $U, V$ and $\mathbf{T}$. We introduce an alternative approach involving two maps $\mathbf{T}, \mathbf{T}'$, to minimize $\Delta(X, \mathbf{T}(\Pi_k(\mathbf{T}'(X))))$ (QMFQ), see Fig. 1.

**Scaling with Quantile Normalization.** To define and optimize families of maps $T_i$, we need to represent monotonic invertible functions in a parametric form that is amenable to optimization. While several approaches have been proposed recently to parameterize such maps (You et al., 2017; Wehenkel & Louppe, 2019; Gupta et al., 2016, and references therein), we propose here a new approach that can be conveniently optimized with respect to both the parameters of $T$ as well as its inputs, with the added benefit that one can control *exactly* the row-wise distributions of the outputs of $\mathbf{T}$. This can be useful for instance to enforce similarities between the values taken jointly across one or more several rows, or to "pin" these values to lie in a segment, as we do in our experiments. In order to reach that property, we parameterize each map $T_i$ as a *quantile normalization* operator w.r.t to a target measure, written as $T_{\nu_i}$. Differentiation is achieved by extending the toolbox of Cuturi et al. (2019) to include soft-quantile normalisation operators.

**Contributions.** Our contributions are two-fold: **(i)** After introducing recent tools of Cuturi et al. (2019), we improve them in three ways in §2: we add to their operators a new differentiable quantile normalization operator; we prove the monotonicity of all these operators, putting these tools on a sound footing; and we derive the implicit differentiation of these operators, rather than unrolling Sinkhorn iterations. **(ii)** We introduce low-rank factorization models in §3 that employ this soft-quantile normalization layer, and propose various algorithmic approaches to train them (including stochastic schemes), either relying on implicit or explicit factorizations as in Fig.1. We test these approaches in §4 on synthetic datasets and on real multiomics cancer data.

## 2. Differentiable Quantile Normalization using Optimal Transport

We recall the approach proposed recently by Cuturi et al. (2019) to view ranking and sorting problems as optimal transport problems that can be turned into differentiable operators through regularization. We then proceed with three contributions: *(i)* We extend their operators to define a *quantile normalization operator* $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$ which takes an array of weighted values $\mathbf{x}$ and modifies them so that these values now follow a given target quantile distribution as described by $\mathbf{b}$ and $\mathbf{q}$. The parameter $\varepsilon > 0$ is a smoothing parameter to ensure differentiability. *(ii)* We prove the monotonicity of Sinkhorn-ranks, Sinkhorn-sort and of the newly introduced Sinkhorn-quantile normalization operators. This is an important result that was missing from Cuturi et al. (2019) (informally, proving the the curves in the middle plot of the Figure 2 can never cross) and that is also crucial to ground to work on solid footing, since we can thus guarantee that our functions $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$ are indeed increasing and therefore conserve ranks. *(iii)* We introduce an implicit differentiation scheme of the solutions of regularized OT, which offers an interesting alternative to the automatic differentiation of Sinkhorn iterations that was put forward by Cuturi et al. (2019).

**Notations.** We denote by $\Sigma_n = \{\mathbf{u} \in \mathbb{R}^n_+ | \mathbf{u}^T \mathbf{1}_n = 1\}$ the set of $n$-dimensional probability vectors. For any vector, $\mathbf{x} = (x_1, \dots)$, we write $\overline{\mathbf{x}}$ for its cumulative-sum vector, namely the vector with entries $[\sum_{j \leqslant i} x_j]_i$. When applied on a matrix $R$, the same operator $\overline{R}$ denotes the cumsum operation applied *row-wise*. A function $c : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is submodular if it is twice differentiable and $\partial^2 c / \partial x \partial y < 0$. For two probability vectors $\mathbf{a}, \mathbf{b}$ of size $n$ and $m$, we write $U(\mathbf{a}, \mathbf{b}) = \{P \in \mathbb{R}^{n \times m}_+ | P \mathbf{1}_m = \mathbf{a}, P^T \mathbf{1}_n = \mathbf{b}\}$ for the transportation polytope. Operations on matrices are to be understood elementwise, and we use $\circ$ for the elementwise product between matrices or vectors.

## 2.1. Background: Soft-Ranking/Sorting using OT

Suppose one is given an array $\mathbf{x} = (x_1, \ldots, x_n)$ of $n$ numbers, weighted by a positive probability vector $\mathbf{a} = (a_1, \ldots, a_n)$ of the same size. The idea of Cuturi et al. (2019) is to consider an auxiliary vector $\mathbf{y} = (y_1, \ldots, y_m)$ of ordered values—typically the regular grid of $m$ values in $[0, 1]$—to form a cost matrix $C_{\mathbf{xy}} := [c(x_i, y_j)]_{ij}$, with $c$ submodular. Along with probability vector $\mathbf{b} = (b_1, \ldots, b_m)$ for $\mathbf{y}$, one defines then a (primal) regularized OT problem:

$$P_\star^\varepsilon := \operatorname*{argmin}_{P \in U(\mathbf{a}, \mathbf{b})} \langle P, C_{\mathbf{xy}} \rangle - \varepsilon H(P), \qquad \text{(P-RegOT)}$$

where $H(P) = -\sum_{i,j} P_{ij} (\log P_{ij} - 1)$ denotes $P$'s entropy. This regularized OT problem has a factorized solution (Cuturi, 2013) $P_\star^\varepsilon$ which can be written as $\operatorname{diag}(\mathbf{u}) K \operatorname{diag}(\mathbf{v})$, where $K = \exp(-C_{\mathbf{xy}}/\varepsilon)$ and $\mathbf{u} \in \mathbf{R}^n$ and $\mathbf{v} \in \mathbf{R}^m$ are fixed points of the Sinkhorn iteration. Cuturi et al. (2019) proposed the following smoothed ranking and sorting operators

$$\widetilde{R}_\varepsilon (\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y}) := n \mathbf{a}^{-1} \circ (P_{\ell_+}^\varepsilon \overline{\mathbf{b}}) \in [0, n]^n,$$

$$\widetilde{S}_\varepsilon (\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y}) := \mathbf{b}^{-1} \circ (P_{\ell_-}^\varepsilon{}^T \mathbf{x}) \in \mathbb{R}^m.$$

obtained after running the Sinkhorn algorithm, and writing

$$P_{\ell_+}^\varepsilon = \operatorname{diag}(\mathbf{u}_\ell) K \operatorname{diag}(\mathbf{v}_\ell), P_{\ell_-}^\varepsilon = \operatorname{diag}(\mathbf{u}_{\ell-1}) K \operatorname{diag}(\mathbf{v}_\ell).$$

The Sinkhorn algorithm is described in simplified form in Alg.1. In practice the number of iterations $\ell$ can be set dynamically, to enforce convergence.

**Modification to Guarantee Marginals.** There is a small but important modification we have done to these operators, compared to Cuturi et al. (2019): We consider in the definition of $\widetilde{S}_\varepsilon$ the scaling $\mathbf{u}_{\ell-1}$ and not the scaling $\mathbf{u}_\ell$. This is done in order to take advantage of the the fact that after any iteration on $\mathbf{v}$ in algorithm Alg.1, the transport matrix estimate $\operatorname{diag}(\mathbf{u}_{i-1}) K \operatorname{diag}(\mathbf{v}_i)$ has column-sums exactly equal to $\mathbf{b}$ (but row-sums not necessarily equal to $\mathbf{a}$), whereas $\operatorname{diag}(\mathbf{u}_i) K \operatorname{diag}(\mathbf{v}_i)$ has the opposite property (equality of row-sums to $\mathbf{a}$ is ensured, but not of column-sums to $\mathbf{b}$). This modification ensures that the operators $\widetilde{R}_\varepsilon$ and $\widetilde{S}_\varepsilon$ effectively apply row-stochastic kernels to their inputs, so each of the entries of $\widetilde{R}_\varepsilon$ and $\widetilde{S}$ are convex combinations of $\mathbf{x}$ and $n\overline{\mathbf{b}}$. These modifications are particularly important small $\ell$.

---

**Algorithm 1** Sinkhorn with $\ell$ iterations

**Inputs:** $\mathbf{a}, \mathbf{b}, \mathbf{x}, \mathbf{y}, \varepsilon, c$
$C_{\mathbf{xy}} \leftarrow [c(x_i, y_j)]_{ij}, \ K \leftarrow e^{-C_{\mathbf{xy}}/\varepsilon}, \ \mathbf{u} = \mathbf{1}_n$
**for** $i = 1, \ldots, \ell$ **do**
$\quad | \quad \mathbf{v}_i \leftarrow \mathbf{b}/K^T \mathbf{u}_{i-1}, \ \mathbf{u}_i \leftarrow \mathbf{a}/K \mathbf{v}_i$
**end**
**Result:** $\mathbf{u}_\ell, \mathbf{v}_\ell, \mathbf{u}_{\ell-1}, K$

---

## 2.2. Differentiable Quantile Normalization

In the continuous world, a quantile normalization operator maps values distributed according to a measure $\mu$ to values described in a measure $\nu$. That map is increasing (Santambrogio, 2015, §2), and is the composition of maps $Q_\nu \circ F_\mu$: one computes first the CDF of the input value $x$ w.r.t. $\mu$, and then outputs the quantile of $\nu$ at that level. The challenge of that transformation when instantiated on two discrete measures $\nu = \sum_j b_j \delta_{q_j}$ and $\mu = \sum_i a_i \delta_{x_i}$, is that it is not differentiable because quantile and CDF functions $Q_\nu$ and $F_\mu$ are staircase-like functions, and the output of their composition, requiring ranking, sorting and lookup tables is not continuous. We adopt now a discrete perspective leveraging the operators defined above to define a differentiable quantile normalization operator that takes the values of $\mathbf{x}$ (with weights $\mathbf{a}$) as inputs, computes their soft-transport to a target measure $(\mathbf{b}, \mathbf{y})$ where $\mathbf{y}$ is an arbitrary increasing sequence in $\mathbb{R}^m$, and then computes a convex combination of the quantiles as described in $\mathbf{q}$:

**Definition 2.1** (Soft-Quantile Normalization Operators). For any increasing vector $\mathbf{q} \in \mathbb{R}^m$ paired with a vector of weights $\mathbf{b} \in \Sigma_m$, the following operator denotes the quantile renormalization with respect to $\mathbf{q}$ of the values in $\mathbf{x}$ weighted by $\mathbf{a}$:

$$\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}} (\mathbf{a}, \mathbf{x}; \mathbf{y}) := \mathbf{a}^{-1} \circ (P_{\ell_+}^\varepsilon \mathbf{q}) \in \mathbb{R}^n. \qquad (1)$$

*Remark* 1. Since $\operatorname{diag}(\mathbf{a}^{-1}) P_{\ell_+}^\varepsilon$ is row-stochastic, the entries of $\widetilde{T}$ are convex combinations of entries of $\mathbf{q}$. As we show below, the entries of this vector have the same relative ordering as the entries of $\mathbf{x}$.

*Remark* 2. As the regularization level $\varepsilon \to 0$ and if $m = n$, $\mathbf{a} = \mathbf{b} = 1/n$, then $P_\star^\varepsilon$ converges to the sorting permutation matrix of $x$, $\widetilde{R}_\varepsilon$ to the vector of ranks, and $\widetilde{T}_\epsilon$ to the usual quantile normalization operator (the entries of $\widetilde{T}_\epsilon$ are exactly those of $\mathbf{q}$ reindexed to agree with the arg-sort of $\mathbf{x}$), see Morvan & Vert (2017).

## 2.3. Monotonicity of Sinkhorn Operators

To be consistent as smoothed ranking, sorting and quantile normalization operators, $\widetilde{R}_\varepsilon$, $\widetilde{S}_\varepsilon$ and $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$ should possess basic monotonicity properties. The sorted vector $\widetilde{S}_\varepsilon (\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y})$ should be non-decreasing, and if $x_i \leqslant x_{i'}$ then the $i$th entry of $\widetilde{R}_\varepsilon$ (respectively, $\widetilde{T}_\varepsilon$) should be smaller than its $i'$th entry. As the following proposition shows, the smoothed ranking and sorting operators proposed here enjoy both of these properties, for any number of Sinkhorn iterations.

**Proposition 2.1.** *For any $\ell \geqslant 0$ and any submodular cost*

*c, the following relations hold:*

$$j \leqslant j' \implies [\widetilde{S}_\varepsilon(\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y})]_j \leqslant [\widetilde{S}_\varepsilon(\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y})]_{j'}$$

$$x_i \leqslant x_{i'} \implies [\widetilde{R}_\varepsilon(\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y})]_i \leqslant [\widetilde{R}_\varepsilon(\mathbf{a}, \mathbf{x}; \mathbf{b}, \mathbf{y})]_{i'}$$

$$x_i \leqslant x_{i'} \implies [\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}(\mathbf{a}, \mathbf{x}; \mathbf{y})]_i \leqslant [\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}(\mathbf{a}, \mathbf{x}; \mathbf{y})]_{i'}.$$

The proof is given in the supplementary, and relies on stochastic monotonicity of the rows of the iterations of the Sinkhorn algorithm, thanks to the crucial assumption that $c$ is submodular (Chiappori et al., 2017). A remarkable feature of this result is that it holds regardless of the number of iterations $\ell$.

### 2.4. Implicit Differentiation of Sinkhorn Operators

Cuturi et al. (2019) proposes to use a direct automatic differentiation of Sinkhorn iterations to obtain differentiability of the transports $P_{\ell_-}^\varepsilon$ and $P_{\ell_+}^\varepsilon$ that appear within $\widetilde{R}_\varepsilon, \widetilde{S}_\varepsilon, \widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$. Because storing all Sinkhorn iterations is required to use automatic differentiation, the RAM cost of this approach is heavy, totalling at least $O(\ell n + m + nm)$, notably when $\varepsilon$ is small, in which case $\ell$ can be typically several hundred. A possible workaround would be to use faster regularized OT solvers. However, most of the approaches investigated recently to speed up the Sinkhorn iterations yield non-differentiable computational graphs, since they involve conditional choices (Dvurechensky et al., 2018) or are not easy to parallelize (Altschuler et al., 2017). We have tried modified iterations (Thibault et al., 2017; Schmitzer, 2016) but they still prevent the use of these operators in larger scale settings. We propose to bypass this issue using implicit differentiation.

**Implicit Differentiation for Backpropagation.** Since our goal is to backpropagate through the transport we need a fast algorithm to apply the *transpose* of the Jacobian map of $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$ only w.r.t. inputs $\mathbf{x}, \mathbf{b}, \mathbf{q}$ (for reasons that will become clear in the next section). A variant of the computations below appears in (Luise et al., 2018), which we complement in several ways: Their goal was to compute the gradient of $\langle P_\star^\varepsilon, C_{\mathbf{xy}} \rangle$ w.r.t. $\mathbf{b}$ only, their method uses a Cholesky factorization of a $(n + m) \times (n + m)$ matrix, while we do away with this step using a Schur complement and generalize derivations to also inputs $\mathbf{x}$. Similar computations have also been carried out to prove statistical results by Klatt et al. (2018).

**Differentiating the OT Matrix.** The main challenge when computing the transpose-Jacobians of $\widetilde{R}_\varepsilon, \widetilde{S}_\varepsilon$ and $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$, is to differentiate the optimal transport matrix $P_\star^\varepsilon$ w.r.t any of the relevant inputs (both $P_{\ell_-}^\varepsilon$ and $P_{\ell_+}^\varepsilon$ are assumed to have converged to the solution $P_\star^\varepsilon$, which we must assume to do implicit calculus). Since $P_\star^\varepsilon$ does not change with $\mathbf{q}$ as apparent from (1) (a fact highlighted by Morvan & Vert, 2017), the transpose-Jacobian of $\widetilde{T}_{\varepsilon, \mathbf{b}, \mathbf{q}}$ w.r.t. $\mathbf{q}$ is simply

the map $H \to (H \circ P_\star^\varepsilon)^T \mathbf{1}_n$. The challenge is therefore to provide a fast way to apply the transpose-Jacobians of $P_\star^\varepsilon$ to any matrix $H$ of size $n \times m$ w.r.t. $\mathbf{x}$ and $\mathbf{b}$.

Given $H \in \mathbb{R}^{n \times m}$, and a variable $u$ that is either $\mathbf{x}$ or $\mathbf{b}$, we seek a linear operator $(\mathfrak{J}_u P_\star^\varepsilon)^T$ such that, assuming $P_\star^\varepsilon(u + du) - P_\star^\varepsilon(du) \approx \mathfrak{J}_u P_\star^\varepsilon(du)$, one has for any $H$ that $\langle H, \mathfrak{J}_u P_\star^\varepsilon(du) \rangle = \langle (\mathfrak{J}_u P_\star^\varepsilon)^T H, du \rangle$. We parameterize first $P_\star^\varepsilon$ as the solution of a dual regularized OT problem.

**Dual Formulation.** As discussed by Peyré & Cuturi (2019, §4), (P-RegOT) is equivalent to the dual problem below, where we use the notation $\mathbf{f} \oplus \mathbf{g} := \mathbf{f}\mathbf{1}_m^T + \mathbf{1}_n \mathbf{g}^T$ for the tensor addition of these two vectors,

$$\max_{\mathbf{f} \in \mathbb{R}^n, \mathbf{g} \in \mathbb{R}^m} \mathbf{f}^T \mathbf{a} + \mathbf{g}^T \mathbf{b} - \varepsilon \mathbf{1}_n^T e^{\frac{\mathbf{f} \oplus \mathbf{g} - C(u)}{\varepsilon}} \mathbf{1}_m. \quad \text{(D-RegOT)}$$

When $\mathbf{f}$ and $\mathbf{g}$ are optimal, one has $P_\star^\varepsilon = e^{\frac{\mathbf{f} \oplus \mathbf{g} - C(u)}{\varepsilon}}$.

**Variations in $P_\star^\varepsilon$.** Assuming all other variables fixed,

$$P_\star^\varepsilon(u + du) - P_\star^\varepsilon(u) \approx \mathfrak{J}_u P_\star^\varepsilon(du)$$
$$= \frac{1}{\varepsilon} P_\star^\varepsilon(u) \circ \left( \mathfrak{J}_u \mathbf{f}(du) \mathbf{1}_m^T + \mathbf{1}_n \mathfrak{J}_u \mathbf{g}(du)^T - \mathfrak{J}_u C(du) \right)$$
$$= \frac{1}{\varepsilon} \mathrm{diag}(\mathfrak{J}_u \mathbf{f}(du)) P_\star^\varepsilon(u) + P_\star^\varepsilon \mathrm{diag}(\mathfrak{J}_u \mathbf{g}(du))$$
$$- P_\star^\varepsilon(u) \circ \mathfrak{J}_u C(du).$$

Using an arbitrary $H$, one recovers that

$$\langle H, \mathfrak{J}_u P_\star^\varepsilon(du) \rangle = \tfrac{1}{\varepsilon} \langle (\mathfrak{J}_u \mathbf{f})^T (H \circ P_\star^\varepsilon) \mathbf{1}_m, du \rangle$$
$$+ \tfrac{1}{\varepsilon} \langle (\mathfrak{J}_u \mathbf{g})^T (H^T \circ P_\star^{\varepsilon T}) \mathbf{1}_n, du \rangle$$
$$- \tfrac{1}{\varepsilon} \langle (H \circ P \circ \mathfrak{J}_u C^T) \mathbf{1}_m, du \rangle.$$

Three (transposed) Jacobians are therefore needed w.r.t. $u$, those of $\mathfrak{J}_u \mathbf{f}, \mathfrak{J}_u \mathbf{g}$ and $\mathfrak{J}_u C$. Note that $\mathfrak{J}_\mathbf{b} C = \mathbf{0}_{(n \times m) \times m}$ and we will assume $(\mathfrak{J}_\mathbf{x} C)^T$ can be accessed. For instance when $c = (\cdot - \cdot)^2$, $(\mathfrak{J}_\mathbf{x} C)^T = \Delta := 2(\mathbf{y}\mathbf{1}_n^T - \mathbf{1}_m \mathbf{x}^T)$. Jacobians of $\mathbf{f}(u)$ and $\mathbf{g}(u)$ require more work.

**Variations in $\mathbf{f}, \mathbf{g}$.** We use the $\min_\varepsilon$ operator to simplify equations, defined on matrices of arbitrary size as

$$\text{For } A \in \mathbb{R}^{p \times q}, \min_\varepsilon(A) = -\varepsilon \log((e^A)^T \mathbf{1}_q) \in \mathbb{R}^p.$$

For a vector $\mathbf{z} \in \mathbb{R}^{n+m}$ we write $\mathbf{z}_f$ and $\mathbf{z}_g$ for the vectors of its first $n$ and last $m$ entries respectively, such that $\mathbf{z} = \begin{bmatrix} \mathbf{z}_f \\ \mathbf{z}_g \end{bmatrix}$. The first order conditions for $\mathbf{f}, \mathbf{g}$, boil down to

$$\tau\left( \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix} \right) = 0, \text{ where } \tau(\mathbf{z}) := \begin{bmatrix} \min_\varepsilon(C_{\mathbf{xy}} - \mathbf{z}_f \oplus \mathbf{z}_g) + \varepsilon \log \mathbf{a} \\ \min_\varepsilon(C_{\mathbf{xy}}^T - \mathbf{z}_g \oplus \mathbf{z}_f) + \varepsilon \log \mathbf{b} \end{bmatrix}.$$

A sequence of computations (provided in the supplement) yields that the Jacobian of $\tau$ is a $n \times m$ block matrix,

$$\mathfrak{J}_\mathbf{z} \tau(\mathbf{x}, \mathbf{z}) = - \begin{bmatrix} I_n & M_1 \\ M_2 & I_m \end{bmatrix},$$

where we have written $M = e^{(\mathbf{z}_f \oplus \mathbf{z}_g - C_{\mathbf{xy}})/\varepsilon}$ to define $M_1 = \mathrm{diag}(1/M\mathbf{1}_m)M$ and $M_2 = \mathrm{diag}(1/M^T\mathbf{1}_n)M^T$. Note that in the case where $\mathbf{z}$ coincides with a solution $\left[\begin{smallmatrix}\mathbf{f}\\\mathbf{g}\end{smallmatrix}\right]$ to (D-RegOT), $M$ is the optimal transportation plan. $M_1$ and $M_2$ can therefore be interpreted as Markov kernel (row-stochastic) matrices. Using the matrix inversion theorem we obtain that the inverse *transposed* Jacobian is

$$-\mathfrak{J}_{\mathbf{z}}\tau(\mathbf{z})^{-T} = \begin{bmatrix} I_n + M_2^T S^{-1} M_1^T & -M_2^T S^{-1} \\ -S^{-1} M_1^T & S^{-1} \end{bmatrix},$$

with the transpose-Schur complement $S = I_m - M_1^T M_2^T$.

**Differentiation w.r.t x or b.** The implicit mechanism linking variable $u$ (where $u$ is still either $\mathbf{x}$ or $\mathbf{b}$) with $\mathbf{f}(u)$ and $\mathbf{g}(u)$ is given by the implicit function theorem (here, instantiated in its transpose form) which states that, at optimality (here we overload $\tau$ to also consider values $\mathbf{x}$ or $\mathbf{b}$ as inputs to the first order equation for simplicity),

$$\left(\mathfrak{J}_u \begin{bmatrix} \mathbf{f}(u) \\ \mathbf{g}(u) \end{bmatrix}\right)^T = -(\mathfrak{J}_u\tau(u))^T(\mathfrak{J}_{\mathbf{z}}\tau(\mathbf{z}))^{-T},$$

With a few computations we have for $u$ equal to $\mathbf{x}$ or $\mathbf{b}$,

$$(\mathfrak{J}_{\mathbf{x}}\tau)(\mathbf{x})d\mathbf{x} = \begin{bmatrix} d\mathbf{x}\circ(M1\circ\Delta)\mathbf{1}_m \\ (M_2\circ\Delta^T)d\mathbf{x} \end{bmatrix}, (\mathfrak{J}_{\mathbf{b}}\tau(\mathbf{b}))d\mathbf{b} = \begin{bmatrix} \mathbf{0}_n \\ \varepsilon d\mathbf{b}/\mathbf{b} \end{bmatrix},$$

and therefore

$$(\mathfrak{J}_{\mathbf{x}}\tau)^T(d\mathbf{z}) = \mathbf{z}_f \circ (M_1 \circ \Delta)\mathbf{1}_m + (M_2^T \circ \Delta)d\mathbf{z}_g,$$
$$(\mathfrak{J}_{\mathbf{b}}\tau)^T(d\mathbf{z}) = \mathbf{b}/\varepsilon \circ d\mathbf{z}_g.$$

Applying these results backwards, we recover Alg. 2, which provides all "custom" gradients needed to incorporate Sinkhorn operators in end-to-end differentiable pipelines.

---

**Algorithm 2** Sinkhorn and Jacobians Transpose Evaluations

---
**Inputs:** $\mathbf{a}, \mathbf{b}, \mathbf{x}, \mathbf{y}, \varepsilon, c, \rho > 0, H \in \mathbb{R}^{n\times m}$
$C \leftarrow [c(x_i, y_j)]_{ij}, \ \Delta \leftarrow (\mathfrak{J}_{\mathbf{x}}C)^T.$
**repeat**
  $\quad \mathbf{g} \leftarrow \varepsilon\log\mathbf{b} + \min_\varepsilon(C^T - \mathbf{g}\oplus\mathbf{f}) + \mathbf{g}$
  $\quad \mathbf{f} \leftarrow \varepsilon\log\mathbf{a} + \min_\varepsilon(C - \mathbf{f}\oplus\mathbf{g}) + \mathbf{f}$
**until** $\|e^{-\frac{1}{\varepsilon}(C^T - \mathbf{g}\oplus\mathbf{f})}\mathbf{1}_n - \mathbf{b}\| < \rho$;
$P \leftarrow e^{-\frac{1}{\varepsilon}(C^T - \mathbf{g}\oplus\mathbf{f})}, \mathbf{f} \leftarrow \mathbf{f} - \mathbf{f}[0], \mathbf{g} \leftarrow \mathbf{g} + \mathbf{f}[0]$
$M_1 \leftarrow \mathrm{diag}(1/P\mathbf{1}_m)P, M_1[0,:] = \mathbf{0}_{1\times m}$
$M_2 \leftarrow \mathrm{diag}(1/P^T\mathbf{1}_n)P^T$
$\mathbf{z}_f \leftarrow (H \circ P)\mathbf{1}_m, \mathbf{z}_g \leftarrow (H \circ P)^T\mathbf{1}_n$
$\mathbf{w}_g \leftarrow S^{-1}(M_1^T\mathbf{z}_f - \mathbf{z}_g).$
$\mathbf{z}_f \leftarrow \mathbf{z}_f + M_2^T\mathbf{w}_g, \mathbf{z}_g \leftarrow -\mathbf{w}_g$
$\mathfrak{J}_{\mathbf{x}}^T H = -\frac{1}{\varepsilon}(H\circ P\circ\Delta)\mathbf{1}_m + \mathbf{z}_f\circ(M_1\circ\Delta)\mathbf{1}_m + (M_2^T\circ\Delta)\mathbf{z}_g$
$\mathfrak{J}_{\mathbf{b}}^T H = \mathbf{b} \circ \mathbf{z}_g/\varepsilon$
**Result:** $P, \mathfrak{J}_{\mathbf{x}}^T H, \mathfrak{J}_{\mathbf{b}}^T H$

---

*Remark* 3. Alg.2 above contains two extra steps not appearing in our presentation. Those consist in setting to 0 the first
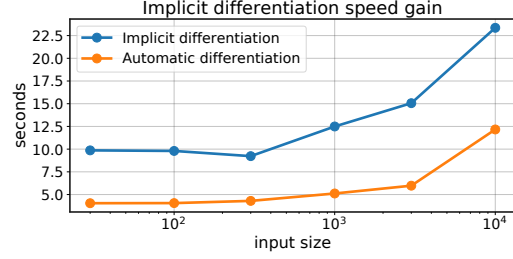


*Figure 2.* Comparison of raw computation times (including tensorflow instantiation) of transposed Jacobians using AD and the implicit approach outlined in Algo.2. Execution carried n batches, with 10 averages, $m = 10$ and $\varepsilon = 10^{-2}$

entry of $\mathbf{f}$ (and offsetting all other entries) and deleting the first row of $M_1$. This modification is due to the fact, also noticed by Luise et al. (2018), that $\mathbf{f}$ and $\mathbf{g}$ are determined up to a constant. Pinning the first variable of $\mathbf{f}$ to 0 helps lift this indeterminacy, and slightly modifies $M_1$ by removing its first row, ensuring $S$ is invertible.

*Remark* 4. The implicit approach outlined here is particularly well suited to the case where $m \ll n$, since the linear system to be solved is of size $m \times m$ and dominates the cost of the final iterations outlined in Alg. 2. As can be expected, we do observe in practice that the execution time of Alg. 2 is roughly half that of the backprop approach used in (Cuturi et al., 2019). The biggest improvement is of course in terms of memory. All of the experiments done next exploit this approach, and were stable numerically.

# 3. Matrix Factorization using Quantile Renormalization

We explore in this section the problem of finding a good low-rank approximation $Z \in \mathbb{R}_k^{d\times n}$ to a data matrix $X \in \mathbb{R}^{d\times n}$ using the tools introduced in §2.

## 3.1. Scaling and Factorization Models

The most standard way to express the matrix factorization problem is to search for a low-rank matrix $Z = UV$ that directly approximates $X$, where $Z$ is the product of a tall and slim matrix $U \in \mathcal{U}_k \subset \mathbb{R}^{d\times k}$ and a short and fat matrix $V \in \mathcal{V}_k \subset \mathbb{R}^{k\times n}$. That approximation is measured in terms of $\Delta(X, UV)$ for some divergence $\Delta$ (Figure 1, left). As mentioned in §1, we propose first to consider a family $\mathbf{T}$ of row-wise monotonic transform, such that $X$ is approximately equal to $\mathbf{T}(UV)$ for a product of factors $UV$ (Figure 1, middle). This suggests to consider the matrix-factorization using supervised quantile normalization:

$$\min_{\mathbf{T}\in\mathcal{T}^d, U\in\mathcal{U}_k, V\in\mathcal{V}_k} \Delta(X, \mathbf{T}(UV)). \qquad \text{(QMF)}$$

We propose a second formulation that provides a more fined grained control on the low-rank factorization itself, and

which requires the simultaneous optimization of two transformations $\mathbf{T}$ and $\mathbf{T}'$ to balance two different goals: *(i)* the transformation of $X$ under $\mathbf{T}'$ should facilitate obtaining a low-rank approximation, namely by minimizing the gap between $\mathbf{T}'(X)$ and $\Pi_k(\mathbf{T}'(X))$; *(ii)* the low-rank approximation should allow, up to another transformation $\mathbf{T}$, the approximate recovery of $X$, i.e., $X \approx \mathbf{T}(\Pi_k(\mathbf{T}'(X)))$. These two goals—ease of factorization without sacrificing the ability to reconstruct—can be modelled through the following bi-level optimization problem ($\Pi_k$ being itself a projection, the solution of an optimization)

$$\min_{\mathbf{T},\mathbf{T}'\in\mathcal{T}^d} \Delta(X, \mathbf{T}(\Pi_k(\mathbf{T}'(X)))). \qquad \text{(QMFQ)}$$

Note that if there is a transformation $\mathbf{T}_X \in \mathcal{T}^d$ such that $\mathbf{T}_X(X)$ is exactly low-rank, then taking $\mathbf{T}' = \mathbf{T}_X$, $\mathbf{T} = \mathbf{T}_X^{-1}$ solves (QMF) and (QMFQ). In the general case, though, there is no reason why we should have $\mathbf{T}' = \mathbf{T}^{-1}$ at optimum. Note that (QMFQ) is harder to minimize than (QMF), since any solution $\mathbf{T},\mathbf{T}'$ of (QMFQ) leads to a feasible point of (QMF) with the same value (by taking $UV = \Pi_k(\mathbf{T}'(X))$), so the optimal value of (QMF) is at least as low as that of (QMFQ). We now describe, building on the different results in §2, our approach to parameterize the families of maps $\mathbf{T}, \mathbf{T}'$.

### 3.2. Feature Scaling using Quantile Normalization.

Recall that given two distributions $\mu$ and $\nu$ over $\mathbb{R}$, a quantile normalization operator $T_{\mu\to\nu} : \mathbb{R} \to \mathbb{R}$ takes samples distributed according to $\mu$ and applies a non-decreasing transformation such that these samples, after this transformation, are distributed according to $\nu$. Writing $F_\mu : \mathbb{R} \to (0,1)$ for the cumulative distribution function (CDF) of $\mu$ and $Q_\nu : (0,1) \to \mathbb{R}$ for the quantile function of $\nu$, such a non-decreasing map can be written as

$$T_{\mu\to\nu} := Q_\nu \circ F_\mu.$$

Notice therefore that, using classic identities relating the CDF and quantile functions of a measure, one has

$$T_{\mu\to\nu}^{-1} = F_\mu^{-1} \circ Q_\nu^{-1} = Q_\mu \circ F_\nu = T_{\nu\to\mu},$$

**From Data and Variables to Measures.** In both (QMF) and (QMFQ) problems, the input measure $\mu$ will be the discrete measure of values extracted from the row of a matrix $W$, where $W$ will be either $X$ directly (as in QMFQ), or a low-rank reconstruction, either in explicit $UV$ or implicit $\Pi_k$ form. We parameterize the output measure $\nu$ as a discrete measure of finite support. To solve (QMF) or (QMFQ) in $\mathbf{T}$, one would then require that the outputs of $T_{\mu\to\nu}$ be differentiable according to *both* the input measures (notably when applied to a reconstruction) and the parameterized output measure $\nu$. The differentiation w.r.t

to quantiles themselves was investigated by Morvan & Vert (2017). The differentiability w.r.t. inputs can be obtained using the soft-quantile normalization operator introduced in §2.2. Note that our definition also has the added flexibility, compared to Morvan & Vert (2017), of introducing weighted quantiles with parameter $\mathbf{b}$ (this is equivalent to defining exactly the levels $\bar{\mathbf{b}}$ to which these quantiles correspond).

### 3.3. Row-Wise Soft-Quantile Transformation

**Input Measures** $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_d)$. $\mu_i$ is the distribution of values of the $i$-th feature of a matrix, written here $W$. $W$ can be the original data matrix $X$ or, more to the point for optimization, its explicit ($UV$) or implicit ($\Pi_k(X)$) reconstructions. The measure for feature $i$ is therefore $\mu_i(W) := \frac{1}{n}\sum_{j=1}^{n} \delta_{W_{ij}}$.

**Target Measures** $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_d)$. We store the values of vectors $\mathbf{b}_i$ and $\mathbf{q}_i$ row-wise in matrices $B, Q$ to obtain $\nu_i(B,Q) := \frac{1}{n}\sum_{j=1}^{m} B_{ij}\delta_{Q_{ij}}$, where $B, Q \in \mathbb{R}^{d\times m}$, $B$ is row-stochastic and $Q$ is row increasing. Notice $B$ and $Q$ are simultaneously pictured in Fig.1 underneath the labels $\mathbf{T}_{\boldsymbol{\nu}}$ : the colors varying from dark to light denote increasing values for $Q$ row-wise, whereas the varying sizes of buckets in each row stand for probability weights $B$. Here, parameter $m$ effectively controls the complexity / size of $\nu_i$. We argue that in most cases, the budget of target quantiles $m$ should be much lower that $n$, as discussed as well by Cuturi et al. (2019). In applications where $n$ is a few hundreds, we find that choosing small $m$, such as 8 or 16, works very well (See Fig. 6).

**From Discountinuous to Everywhere Differentiable.** Given an arbitrary sorted sequence $\mathbf{y} = (y_1, \ldots, y_m)$, our approach relies on the following identity, valid only when $n = m$ and $\mathbf{b} = \mathbf{a} = \frac{\mathbf{1}_n}{n}$ (and therefore $B = \mathbf{1}_{d\times n}/n$), and for each $i \leqslant d$, $\mathbf{w} = W_{i\cdot}, \mathbf{q} = Q_{i\cdot}$.

$$\widetilde{T}_{\varepsilon,\mathbf{b},\mathbf{q}_i}(\mathbf{a}, \mathbf{w}; \mathbf{y}) \underset{\varepsilon\to 0}{\to} T_{\mu_i\to\nu_i}(\mathbf{w}).$$

The main interest in using the expression on the left is that, either through back-propagation or implicit differentiation as proposed in §2.4, the $\widetilde{T}_{\varepsilon,\mathbf{b},\mathbf{q}_i}$ operator is differentiable in all of its inputs and parameters (crucially $W, B, Q$ here as soon as $\varepsilon > 0$, while $T_{\mu_i\to\nu_i}$ is not.

**Row-Stochasticity of** $B$**.** The constraints that $B$ is row-wise stochastic can be taken into account by introducing precursors under the action of a soft-max. More specifically, we will write $B$ as the row-wise softmax of a precursor matrix of $F \in \mathbb{R}^{d\times m}$,

$$B(F) := \sigma(F) = \text{diag}(1/(e^F\mathbf{1}_m)\,e^F.$$

**Monotonicity and Range of** $Q$**.** The constraint that $Q$ is increasing can be enforced by considering cumulative sums

of non-negative values, possibly offset by a constant. In its most direct form, notably when deflating $X$ to obtain a matrix easy to factorize as in (QMFQ), $Q$ can be cast as the row-wise cumulative sum of the exponentials of an arbitrary precursor matrix $R \in \mathbb{R}^{d \times m}$,

$$Q'(R) := \overline{e^R},$$

where the cumsum operator is applied row-wise. When a quantile operator is carried out to inflate back the results of low-rank factorizations $UV$ or $\Pi_k$, we can directly "pin" the quantiles to lie in a range of values known beforhand. Indeed, since the goal is to reconstruct a known data matrix $X$, one can set those segments to be $[s_i, t_i]$ where $s_i$ and $t_i$ are the minimums and maximums of row $i$. Therefore, for a slightly slimmer matrix $R \in \mathbb{R}^{d \times (m-1)}$, and storing the ranges of values in $\mathbf{s} = (s_1, \ldots, s_d)$ and $\mathbf{t} = (t_1, \ldots, t_d)$, we recover the following map to define suitable quantiles from a precursor $R$,

$$Q(R, \mathbf{s}, \mathbf{t}) := \operatorname{diag}(\mathbf{t} - \mathbf{s})[\mathbf{0}_d, \overline{\sigma(R)}] + \mathbf{s}\mathbf{1}_m^T,$$

therefore recovering increasing quantiles that are pinned down to lie exactly in the desired ranges.

### 3.4. Quantile Matrix Factorization

We treat the first problem outlined in (QMF) as an optimization problem with two precursor variables $F, R \in \mathbb{R}^{d \times m}$ that characterize target measures through quantile values $Q(R)$ and probability weights $B(F)$. We assume that $\Delta$ is separable along rows, and use the following notations for space: Given two precursor matrices $F$ and $R$, the map $\tilde{\mathbf{T}}_{\varepsilon, F, R}$ applies to each row $i$ the soft quantile operator defined using weights $\mathbf{b}_i = B(F)_{i\cdot}$ and quantiles $\mathbf{q}_i = Q'(R)_{i\cdot}$.

$$\min_{\substack{R \in \mathbb{R}^{n \times m-1} \\ F \in \mathbb{R}^{n \times m} \\ U \in \mathbb{R}^{d \times k}, V \in \mathbb{R}^{k \times n}}} \sum_i \Delta\left(X, \tilde{\mathbf{T}}_{\varepsilon, F, R}(UV)\right).$$

Note that the main computational effort here consists in applying $d$ quantile normalization operators. When suitable, we therefore use mini-batch sampling on the $d$ features to perform SGD on all parameters. When used on non-negative matrix factorization problems, as demonstrated in §4, we also parameterize $U, V$ as exponential maps of precursor matrices of the same size to enfore non-negativity.

### 3.5. Quantiles Matrix Factorization Quantiles

The optimization problem outlined in (QMFQ) is a bilevel programming problem, and therefore less scalable than QMF. We consider it nonetheless because of its interest as a modeling tool: The result of QMFQ can be used to normalize first a new incoming point, project it on the dictionary $U$ resulting from $\Pi_k(\mathbf{T}'(X))$, and then project it back using

$\mathbf{T}$. To optimize this bilevel problem, we consider here the case in which $\Pi_k$, the (approximate) projection operator can be computed with an accesss to an operator computing the transpose of the Jacobian applied to an input matrix. This is notably the case when using SVD, with the analytic formulas provided for truncated SVD (Feppon & Lermusiaux, 2018, Thm. 25), or by unrolling a fixed point iteration, such as the multiplicative updates proposed in (Lee & Seung, 1999) to minimize the Kullback-Leibler loss between two non-negative matrices. We consider here the latter approach to consider for $F, F', R' \in \mathbb{R}^{d \times m}$ and $R \in \mathbb{R}^{d \times (m-1)}$.

$$\min_{\substack{R \in \mathbb{R}^{n \times m-1} \\ F, F', R' \in \mathbb{R}^{n \times m}}} \sum_i \Delta\left(X, \tilde{\mathbf{T}}_{\varepsilon, F, R}(\Pi_k(\tilde{\mathbf{T}}_{\varepsilon, F', R'}(X)))\right).$$

## 4. Experiments

**Toy Illustrations.** We consider in this section the following dimensions: $d = 160$, $n = 80$, $k = 8$. We generate two ground truth factors $U_*$ and $V_*$ randomly, $U_*$ is a table of i.i.d Poisson realizations with parameter $\lambda = 2$, whereas each column of $V_*$ is drawn according to a Dirichlet prior with parameters $\alpha = 1/2$. We then apply a "ground truth" quantile normalization to these entries, $\mathbf{T}_{0, F_*, R_*}(U_* V_*)$, where the precursors $R$ are sampled as a standard Gaussian multivariate distribution, and $F_*$ is a vector of zeros of size $m_* = n$ (using here standard quantile renormalization, not regularized). We then run NMF, QMF and QMFQ using the same $k = 8$. In all experiments reported here, we set $\varepsilon$ and learning rates to $0.01$.

The factors $U, V$ used in NMF, QMF and each inner evaluation of $\Pi_k$ in QMFQ are initialized with random uniform values (to retain consistency across outer iterations, the seed of QMFQ is always the same). We plot in Fig. 5 the KL divergence of these three different methods. We plot in Fig. 6 the two quantile distributions quantiles learned by QMFQ for the first feature, as well as the learned quantile for QMF.

**Error Bars on Larger Experiments.** We consider the following dimensions, $d = 500$, $n = 256$, $k = 10$, and run the algorithms across various setups, including mini-batches for QMF, various inner loop iterations for QMFQ and various values for targets $m$. We monitor the KL decrease averaged over 8 repeats of the data generation process outlined above (quantile normalization of $U_*, V_*$) to which we add a truncated Gaussian noise (non-negative values) of standard deviation 10. All of our results (see supplementary for more exhaustive explorations of the parameter space) agree with intuition and show the robustness of the two approaches presented here, and are summerized in Fig. 4.

**Genomics.** As an illustration on real-world data we consider the problem of multiomics data integration, a domain where NMF has been shown to be a relevant approach to capture
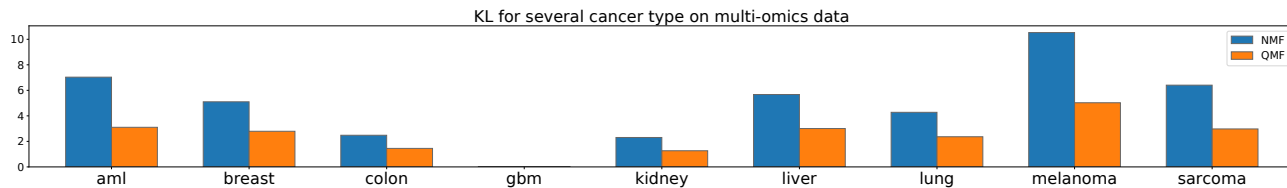
*Figure 3.* KL losses for various cancer datasets as described in (Chalise & Fridley, 2017), in which dimensions are a few hundreds patients for $n$ and $d \approx 11,500$ per dataset. Here $QMF$ is computed with small batch size 64 for QMF along the $d$ dimension and $m = 16$.
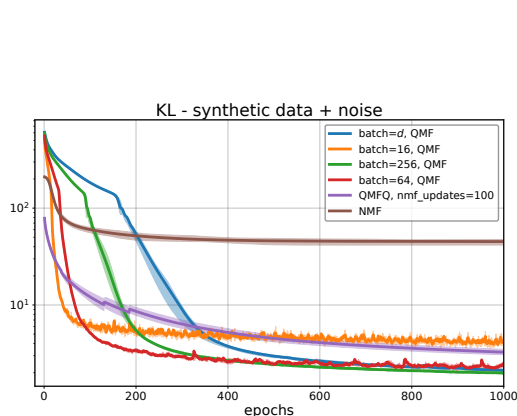


*Figure 4.* KL decrease in a synthetic $\mathbf{T}_*(UV) + \max(10\,\mathcal{N}, 0)$ model with noise. As can be expected, the size of the batch size for QMF influences early/late convergence.



*Figure 6.* (left) In blue, the true quantile distribution used to modify the values of the first row of the ground truth matrix $U_*V_*$. Both QMFQ and QMF methods are able to recover almost exactly lighter representations (here $m = 8$) of the true quantile distributions. (right) In addition, QMFQ also produces a transformation able to deflate the values of $X$ so that they can be easily factorized. Note the difference in ranges ($y$-axis) between the values of $\mathbf{q}$ and $\mathbf{q}'$.
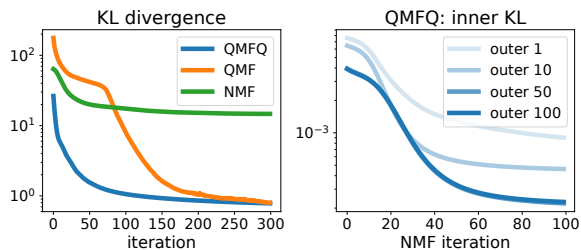


*Figure 5.* (left) Decrease of the KL loss across iterations for all three methods. The NMF quickly saturates (the ground truth is indeed *not* low rank), while QNMF and QNMFQ quickly reach almost perfect reconstruction, despite going through a quantile normalization step that only has a budget of $m = 8$ values. Note that the plot only describes outer loop iterations. In that sense QMFQ is far slower than QMF or NMF, since it requires a 100 iterations of NMF as an inner loop to correctly approximate $\Pi_k(\mathbf{T}'(X))$. (right) decrease of these inner loops as a function of the outer loop. As expected, the KL decreases faster as the algorithm progresses. Note that the KL scales on both plots are not comparable, since the right KL is computed using values taking different ranges of values (see also Fig. 6)
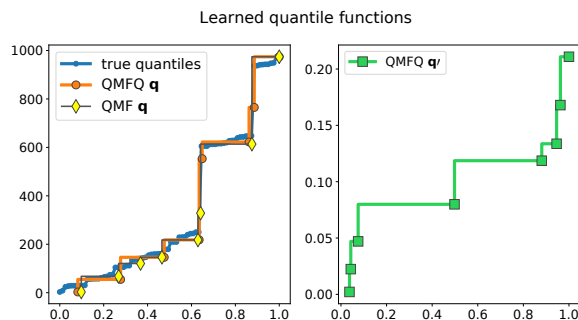
low-rank representations of cancer patients using multiple omics datasets (Chalise & Fridley, 2017). Following the recent benchmark of Cantini et al. (2020), we collected from The Cancer Genome Atlas (TCGA) three types of genomic data (gene expression, miRNA expression and methylation) for thousands of cancer samples from 9 cancer types, and compare a standard NMF to QMF in their ability to find a good low-rank approximation of the concatenated genomic matrices. Figure 3 confirms that on all cancers, QMF finds a factorization much closer to the original data than NMF does. We provide more detailed results in Annex B.2.

To further explore the biological relevance of the factorization found by QMF, we follow the protocol of Cantini et al. (2020) and compute the significance of association between each of the top 10 factors and survival, using a Cox regression-based survival analysis. For this experiment, we therefore compute a factorization in rank 10 with NMF and QMF after log-transforming the expression and methylation matrices, and keeping the top 6,000 genes with largest standard deviation in each data type. We train each model for 1,000 epochs, with a batch size of 64 and learning rate of 0.001. For QMF, we set the number of target quantiles to $m = 16$, and the regularization factor to 0.001. Figure 7 summarizes the results, showing for each cancer the negative base-10 logarithm of the Bonferroni-corrected P-values
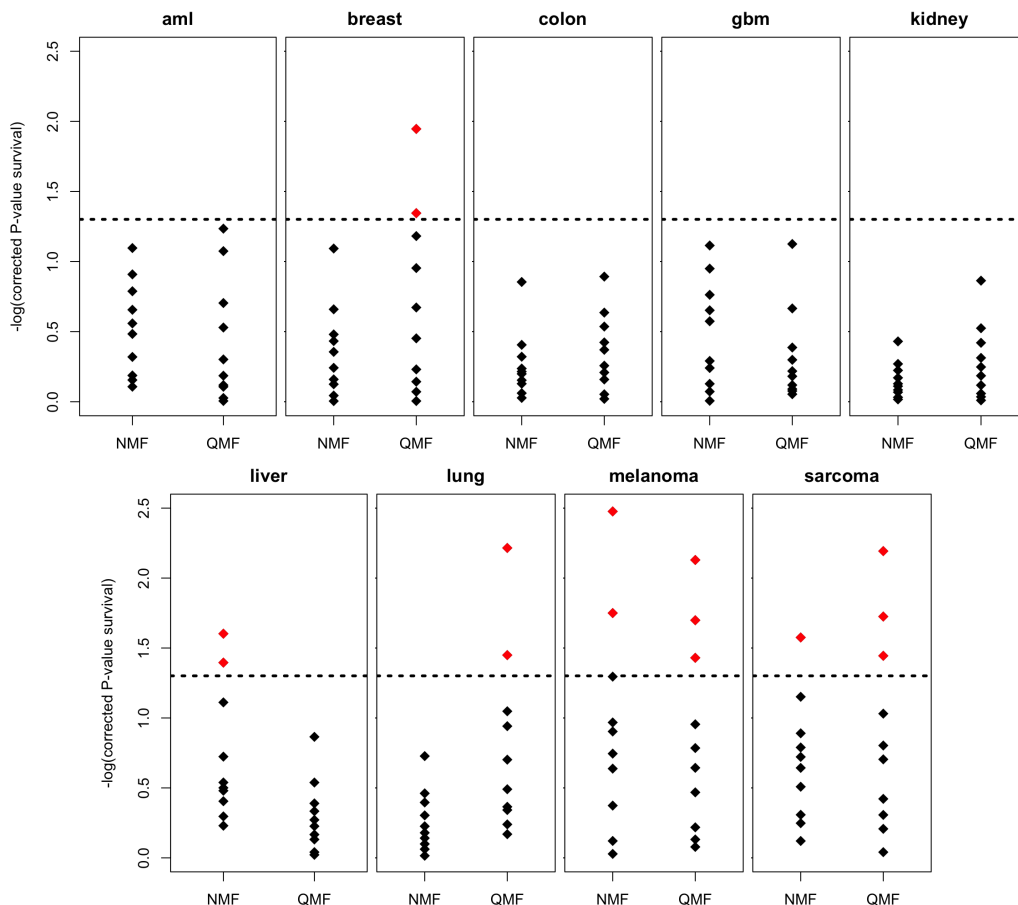
*Figure 7.* Identification of factors predictive of survival in cancers. For each method (NMF or QMF) and each cancer type, the dots show the negative base-10 logarithm of the Bonferroni-corrected P-values associating each of the 10 factors to survival (Cox regression-based survival analysis). The dot lines correspond to a corrected p-value of 0.05. Red dots indicate factors significantly associated with survival.

associating each of the 10 factors to survival, for both NMF and QMF. Red dots indicate factors which are significantly associated with survival (i.e., p-value<0.05 after Bonferroni correction). Out of 9 cancers, we see that QMF finds more factors associated with survival than NMF in 4 cases (breast, lung, melanoma and sarcoma), while NMF outperforms QMF in one case (liver), suggesting that the factorization found by QMF is not only more accurate in terms of KL loss than the one found by NMF, but also generally more biologically relevant. Interestingly, comparing these results with the benchmark of Cantini et al. (2020) which compares 8 different methods to extract relevant factors from genomic data (not necessarily based on matrix factorization), QMF matches or outperforms the best method of the benchmark in 4 cancers out of 9 (breast: 2 significant factors for QMF vs 2 for JIVE; colon: 0 for QMF vs 0 for all methods; lung: 2 for QMF vs 0 for all methods; sarcoma: 3 for QMF vs 2 for RGCCA, MCIA, Sckit and JIVE). The fact that QMF is the only method to find factors significantly associated with survival in lung cancers is particularly promising.

## Conclusion.

We have proposed in this work an extension of low-rank matrix factorization. Our model posits that matrix factorization can be carried out, while still being reconstructive, using quantile normalization operators. These proposals are grounded on several extensions of soft-ranking and sorting operators using OT laid out in §2. Our models in §3 do not assume relations between the several $\mathbf{q}_i$ and levels $\mathbf{b}_i$, nor regularize them: this is an important future research direction. Finally, our experiments suggest that despite the non-convexity of both QMFQ and QMF, out-of-the-shelf minimizers with minimal parameter tuning provided consistently far better results than vanilla NMF. As with matrix factorization, the non-convexity of this model seems well behaved, likely to be facilitated by smoothing $\varepsilon$.

# References

Altschuler, J., Weed, J., and Rigollet, P. Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration. *arXiv preprint arXiv:1705.09634*, 2017.

Candès, E. J. and Recht, B. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

Cantini, L., Zakeri, P., Hernandez, C., Naldi, A., Thieffry, D., Remy, E., and Baudot, A. Benchmarking joint multiomics dimensionality reduction approaches for cancer study. *bioRxiv*, 2020. doi: 10.1101/2020.01.14.905760.

Chalise, P. and Fridley, B. L. Integrative clustering of multilevel omic data based on non-negative matrix factorization algorithm. *PLoS One*, 12(5):e0176278, 2017.

Chiappori, P.-A., McCann, R. J., and Pass, B. Multi-to one-dimensional optimal transport. *Communications on Pure and Applied Mathematics*, 70(12):2405–2444, 2017.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems 26*, pp. 2292–2300, 2013.

Cuturi, M., Teboul, O., and Vert, J.-P. Differentiable ranking and sorting using optimal transport. In *Advances in Neural Information Processing Systems 32*, pp. 6858–6868, 2019.

d'Aspremont, A., Ghaoui, L. E., Jordan, M. I., and Lanckriet, G. R. A direct formulation for sparse PCA using semidefinite programming. In *Advances in neural information processing systems*, pp. 41–48, 2005.

Dvurechensky, P., Gasnikov, A., and Kroshnin, A. Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm. *arXiv preprint arXiv:1802.04367*, 2018.

Feppon, F. and Lermusiaux, P. F. A geometric approach to dynamical model order reduction. *SIAM Journal on Matrix Analysis and Applications*, 39(1):510–538, 2018.

Févotte, C. and Idier, J. Algorithms for nonnegative matrix factorization with the $\beta$-divergence. *Neural computation*, 23(9):2421–2456, 2011.

Gupta, M., Cotter, A., Pfeifer, J., Voevodski, K., Canini, K., Mangylov, A., Moczydlowski, W., and van Esbroeck, A. Monotonic calibrated interpolated look-up tables. *Journal of Machine Learning Research*, 17(109):1–47, 2016.

Hofmann, T. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196, 2001.

Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12(Jul):2297–2334, 2011.

Klatt, M., Tameling, C., and Munk, A. Empirical regularized optimal transport: Statistical theory and applications. *arXiv preprint arXiv:1810.09880*, 2018.

Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 42(8): 30–37, Aug 2009.

Le Van, T., van Leeuwen, M., Nijssen, S., and De Raedt, L. Rank matrix factorisation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 734–746. Springer, 2015.

Lee, D. D. and Seung, H. S. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755): 788–791, 1999.

Luise, G., Rudi, A., Pontil, M., and Ciliberto, C. Differential properties of Sinkhorn approximation for learning with Wasserstein distance. In *Advances in Neural Information Processing Systems 31*, pp. 5859–5870. 2018.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(Jan):19–60, 2010.

Morvan, M. L. and Vert, J.-P. Supervised quantile normalisation. *arXiv preprint arXiv:1706.00244*, 2017.

Peyré, G. and Cuturi, M. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning*, 11(5-6):355–607, 2019.

Risso, D., Perraudeau, F., Gribkova, S., Dudoit, S., and Vert, J.-P. A general and flexible method for signal extraction from single-cell RNA-seq data. *Nature Communications*, 9(284):1–17, 2018.

Rolet, A., Cuturi, M., and Peyré, G. Fast dictionary learning with a smoothed Wasserstein loss. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 630–638, 2016.

Santambrogio, F. *Optimal transport for applied mathematicians*. Birkhauser, 2015.

Schmitzer, B. Stabilized sparse scaling algorithms for entropy regularized transport problems. *arXiv preprint arXiv:1610.06519*, 2016.

Slawski, M., Hein, M., and Lutsik, P. Matrix factorization with binary components. In *Advances in Neural Information Processing Systems 26*, pp. 3210–3218. 2013.

Thibault, A., Chizat, L., Dossal, C., and Papadakis, N. Over-relaxed Sinkhorn-Knopp algorithm for regularized optimal transport. *arXiv preprint arXiv:1711.01851*, 2017.

Wehenkel, A. and Louppe, G. Unconstrained monotonic neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 1543–1553. 2019.

Witten, D. M., Tibshirani, R., and Hastie, T. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

You, S., Ding, D., Canini, K., Pfeifer, J., and Gupta, M. Deep lattice networks and partial monotonic functions. In *Advances in Neural Information Processing Systems 30*, pp. 2981–2989. Curran Associates, Inc., 2017.