Random block coordinate descent methods for computing optimal transport and convergence analysis

Yue Xie* Zhongjian Wang † Zhiwen Zhang ‡

December 15, 2022

Abstract

The optimal transport (OT) problem is reduced to a linear programming (LP) problem through discretization. In this paper, we introduced the random block coordinate descent (RBCD) methods to directly solve this LP. In each iteration, we restrict the potential large-scale optimization problem to small LP subproblems constructed via randomly chosen working sets. Using an expected Gauss-Southwell-q rule to select these working sets, we equip a vanilla version ($\mathbf{RBCD}_0$) with almost sure convergence and linear convergence rate in expectation to solve a general LP problem. By further exploring the special structure of constraints in the OT problems and leveraging the theory of linear systems, we proposed several approaches to refine the random working set selection and accelerate the vanilla method. Preliminary numerical experiments verify the acceleration effects, solution sparsity and demonstrate several merits of an accelerated random block coordinate descent (**ARBCD**) over the Sinkhorn's algorithm when seeking highly accurate solutions.

AMS subject classification: 65C35, 68W20, 90C08, 90C25.

Keywords: Optimal transport, deep particle method, convex optimization, random block coordinate descent, convergence analysis.

1 Introduction

Background and motivation The optimal transport problem was first introduced by Monge in 1781, which aims to find the most cost-efficient way to transport mass from a set of sources to a set of sinks. Later, the theory was modernized and revolutionized by Kantorovich in 1942, who found a key link between optimal transport and linear programming. In recent years, optimal transport has become a popular and powerful tool in data science, especially in image processing, machine learning, and deep learning areas, where it provides a very natural way to compare and interpolate probability distributions. For instance, in generative models [2, 20, 46], a natural penalty function is the distance between the data and the generated distribution. The transport plan, which minimizes the transportation cost, provides solutions to image registration [16] and seamless copy [31]. Apart from data science, in the past three decades, there has been an explosion of research interest in the optimal transport because of the deep connections between the optimal transport problems with quadratic cost functions and a diverse class of partial differential equations (PDEs) arising in statistical mechanics and fluid mechanics; see e.g. [8, 5, 29, 19, 45] for just a few of the most prominent results and references therein.

Inspired by this research progress, we have developed efficient numerical methods for solving multi-scale PDE problems using the optimal transport approach. Specifically, in our recent paper, we proposed a deep particle method to learn and compute invariant measures of parameterized stochastic dynamical systems [46]. To achieve this goal, we develop a deep neural network (DNN) to map a uniform distribution (source) to an invariant measure (target), where the Péclet number is an input parameter for the DNN.

*Department of Mathematics and HKU Musketeers Foundation Institute of Data Science, The University of Hong Kong, Pokfulam Road, Hong Kong SAR, China yxie21@hku.hk

†Department of Statistics and CCAM, The University of Chicago, Chicago, IL 60637, USA. zhongjian@statistics.uchicago.edu

‡Department of Mathematics, The University of Hong Kong, Pokfulam Road, Hong Kong SAR, China zhangzw@hku.hk

The network is trained by minimizing the 2-Wasserstein distance (2-WD) between the measure of network output μ and target measure ν. We consider a discrete version of 2-WD for finitely many samples of μ and ν, which involves a linear program (LP) optimizing over doubly stochastic matrices [39].

Directly solving the LP by the interior point method [47] is too costly. Motivated by the domain decomposition method [41] in scientific computing, which solves partial differential equations using subroutines that solve problems on subdomains and has the advantage of saving memory (i.e., using the same computational resource, it can compute a larger problem), we devised a mini-batch interior point method by sampling smaller sub-matrices while preserving row and column sums. This turns out to be very efficient and integrated well with the stochastic gradient descent (SGD) method for the entire network training. However, we did not get the convergence analysis for this mini-batch interior point method in [46].

The aim of this paper is twofold. First, we want to equip the mini-batch interior point method in [46] with rigorous convergence analysis after minimal modification. Second, we desire to improve the mini-batch selection strategy and achieve a better and more robust performance in computing optimal transport problems. We realize that the mini-batch interior point method coincides with the random block coordinate descent (RBCD) method in optimization terminology. In particular, it applies the block coordinate descent (BCD) method to the LP problem directly, selects the working set randomly, and solves the subproblems using the primal-dual interior-point method [47] or any efficient linear programming solver. Encouraged by the proved efficiency of this approach, we will develop theoretical results on solving LP with RBCD methods and investigate various ways to choose working sets.

Theorectical contributions In this work, we first propose an expected Gauss-Southwell-q rule to guide selection of the working set. It enables almost sure convergence and linear convergence rate in expectation to solve a general LP. Based on this rule, we design a vanilla RBCD method - $\mathbf{RBCD}_0$, which chooses the working set with full randomness. Then we explore the special linear system in the LP formulation of OT. We characterize all the elementary vectors of the null space and provides a plan to find conformal realization of any given vector in the null space with low computational cost. Based on these findings, we propose various approaches to refine the working set selection and improve the performance of $\mathbf{RBCD}_0$. A better estimation of the constant in the linear convergence rate is shown. Moreover, we incorporate an acceleration technique inspired by the momentum concept.

Numerical experiments We conduct numerical experiments to illustrate the performance of the proposed methods. Synthetic data sets and invariant measures generated from IPM methods are utilized to create distributions ranging from 1D to 3D. First, we compare among different RBCD methods proposed in this article: we demonstrate the benefits by refining working set selection and verify the effect of an acceleration technique. We also illustrate the gap between theory and practice regarding convergence rate and sparse solutions generated by the proposed RBCD methods. Second, we compare the one with the best performance, **ARBCD**, with the Sinkhorn's algorithm. Preliminary numerical experiments show that **ARBCD** outperforms the Sinkhorn's algorithm in computation time when seeking solutions with relatively high accuracy.

Previous research on (R)BCD BCD and RBCD are well-studied for essentially unconstrained smooth optimization (sometimes allow separable constraints or nonsmooth objective functions): [4, 15, 40] investigate BCD with cyclic coordinate search; [28, 24, 35] study RBCD to address problems with possibly nonsmooth separable objective functions; other related works include theoretical speedup of RBCD ([36, 25]), second-order sketching ([34, 6]). However, much less is known for their convergence properties when applied to problems with nonseparable nonsmooth functions as summands or coupled constraints. To our best knowledge, no one has ever considered using the RBCD to solve general LP before and the related theoretical guarantees are absent. In [26], the authors studied the RBCD method to tackle problems with a convex smooth objective and coupled linear equality constraints $x_1 + x_2 + \ldots + x_N = 0$; a similar algorithm named random sketch descent method [27] is investigated to solve problems with general smooth objective and general coupled linear equality constraints $Ax = b$. However, after adding the simple bound constraints $x \geq 0$, the analysis in either [26, 27] may not work anymore, nor can it

be easily generalized. Beck [3] studied a greedy coordinate descent method but focus on a single linear equality constraint and bound constraints. In Paul Tseng and his collaborators' work [42, 43, 44], a block coordinate gradient descent method is proposed to solve linearly constrained optimization problems including general LP. In these works, a Gauss-Southwell-q rule is proposed to guide the selection of the working set in each iteration. Therefore, the working set selected in a deterministic fashion can only be decided after solving a quadratic program with a similar problem size as the original one. In contrast, the mini-batch interior point/RBCD method we propose chooses the working set with randomness and low computational cost. Another line of research to deal with separable functions, linear coupled constraints and other additional separable constraints considers the alternating direction method of multipliers (ADMM) [9, 17, 48, 49], which updates blocks of primal variables in Gauss-Seidal fashion and also involves multipliers update.

Existing algorithms for OT Encouraged by the success in applying the Sinkhorn's algorithm to the dual of entropy regularized OT [10], researchers conducted extensive studies along this line, including other types of regularization [7][12], acceleration [14][22] and numerical stability [38]. Other works significantly different from the entropy regularization framework include [21], which consider computing Schrödinger bridge problem (in fact equivalent to OT with fisher information regularization), and [13], a multiscale strategy suitable for OT between points intrinsically on low dimensional spaces. RBCD in this study is a regularization-free method. Therefore, it does not need to deal with inaccurate solution and numerical stability issues introduced by the regularization term. Moreover, each subproblem in RBCD is a small-size LP and allows flexible choices for resolution.

Organization The rest of the paper is organized as follows. In Section 2, we review the basic idea of optimal transport and Wasserstein distance. In Section 3, we introduce the expected Gauss-Southwell-q rule and a vanilla RBCD ($\mathbf{RBCD}_0$) for computing general LP problems. In Section 4, we explore the property of the linear system in OT and propose several approaches to refine and accelerate $\mathbf{RBCD}_0$. In Section 5, numerical results are presented to demonstrate the performance of our methods. Finally, concluding remarks are made in Section 6.

Notation. For any matrix X, denote $X(i,j)$ as its element in ith column and jth row, denote $X(:,j)$ as its jth row vector. For a vector v, we usually use superscripts to denote its copies (e.g., v^k in kth iteration of an algorithm) and use subscripts to denote its components (e.g., v_i); for a scalar, we usually use subscripts to denote its copies. Occasional inconsistent cases will be declared in context. $\mathrm{mod(k,n)}$ means k modulo n. For any vector v, $\mathrm{supp(v)} \triangleq \{\mathrm{i} \in \{1,\ldots,\mathrm{n}\} \mid \mathrm{v_i} \neq 0\}$. Given a matrix $X \in \mathbb{R}^{n\times n}$, we define its vectorization as follows,

$$\mathrm{vec(X)} \triangleq (\mathrm{X(:,1)^T, X(:,2)^T, ..., X(:,n)^T})^{\mathrm{T}}.$$

For any positive integer $k \geq 2$, denote $[1,k] \triangleq \{1,...,k\}$.

2 Optimal transport problems and Wasserstein distance

The Kantorovich formulation of optimal transport can be described as follows,

$$\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{X\times Y} C(x,y)\ \mathrm{d}\gamma(x,y) \tag{1}$$

where $\Gamma(\mu,\nu)$ is the set of all measures on $X \times Y$ whose marginal distribution on X is μ and marginal distribution on Y is ν, $C(x,y)$ is the transportation cost. In this article, we refer to the Kantorovich formulation when we mention optimal transport.

Wasserstein distances are metrics on probability distributions inspired by the problem of optimal mass transport. They measure the minimal effort required to reconfigure the probability mass of one distribution in order to recover the other distribution. They are ubiquitous in mathematics, especially in fluid mechanics, PDEs, optimal transport, and probability theory [45]. One can define the p-Wasserstein distance between probability measures μ and ν on a metric space Y with distance function $dist$ by

$$W_p(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{Y\times Y} dist(\tilde{y},y)^p \, \mathrm{d}\gamma(\tilde{y},y)\right)^{1/p} \tag{2}$$

where $\Gamma(\mu,\nu)$ is the set of probability measures γ on $Y\times Y$ satisfying $\gamma(A\times Y)=\mu(A)$ and $\gamma(Y\times B)=\nu(B)$ for all Borel subsets $A,B\subset Y$. Elements $\gamma\in\Gamma(\mu,\nu)$ are called couplings of the measures μ and ν, i.e., joint distributions on $Y\times Y$ with marginals μ and ν on each axis. p-Wasserstein distance is a special case of optimal transport when $X=Y$ and the cost function $c(\tilde{y},y)=dist(\tilde{y},y)^p$.

In the discrete case, the definition (2) has a simple intuitive interpretation: given a $\gamma\in\Gamma(\mu,\nu)$ and any pair of locations $(\tilde{y},y)$, the value of $\gamma(\tilde{y},y)$ tells us what proportion of μ mass at $\tilde{y}$ should be transferred to y, in order to reconfigure μ into ν. Computing the effort of moving a unit of mass from $\tilde{y}$ to y by $dist(\tilde{y},y)^p$ yields the interpretation of $W_p(\mu,\nu)$ as the minimal effort required to reconfigure μ mass distribution into that of ν.

In a practical setting [32], referred to as a point cloud, the closed-form solution of μ and ν may be unknown, instead only n independent and identically distributed (i.i.d.) samples of μ and n i.i.d. samples of ν are available. We approximate the probability measures μ and ν by empirical distribution functions:

$$\mu=\frac{1}{n}\sum_{i=1}^{n}\delta_{\tilde{y}^i} \qquad \text{and} \qquad \nu=\frac{1}{n}\sum_{j=1}^{n}\delta_{y^j}, \tag{3}$$

where δ_x is the dirac measure. Any element in $\Gamma(\mu,\nu)$ can clearly be represented by a transition matrix, denoted as $\gamma=(\gamma_{i,j})_{i,j}$ satisfying:

$$\gamma_{i,j}\geq 0; \qquad \forall j,\ \sum_{i=1}^{n}\gamma_{i,j}=\frac{1}{n}; \qquad \forall i,\ \sum_{j=1}^{n}\gamma_{i,j}=\frac{1}{n}. \tag{4}$$

Then $\gamma_{i,j}$ means the mass of $\tilde{y}^i$ that is transferring to y^j.

We denote all matrices in $\mathbb{R}^{n\times n}$ satisfying (4) as Γ^n, then (2) becomes

$$\hat{W}(f) := \left(\inf_{\gamma\in\Gamma^n}\sum_{i,j=1}^{n,n} dist(\tilde{y}^i,y^j)^p\gamma_{i,j}\right)^{1/p}. \tag{5}$$

Remark. *Γ^n is in fact the set of $n\times n$ doubly stochastic matrix [39] divided by n.*

Another practical setting, which is commonly used in fields of computer vision [30, 23], is to compute Wasserstein distance between two histograms. To compare two grey-scale figure (2D, size $n_0\times n_0$), we first normalize the grey-scale such that the values of cells of each picture sum to one. We denote centers of the cell as $\{y^i\}_{i=1}^n$ and $\{\tilde{y}^i\}_{i=1}^n$, then we can use two probability measures to represent the two figures:

$$\mu=\sum_{i=1}^{n}r_{1,i}\delta_{\tilde{y}^i} \qquad \text{and} \qquad \nu=\sum_{j=1}^{n}r_{2,j}\delta_{y^j},$$

where $r_{1,i},r_{2,j}\geq 0, \forall 1\leq i,j\leq n$, $\sum_{i=1}^{n}r_{1,i}=\sum_{j=1}^{n}r_{2,j}=1$. The discrete Wasserstein distance (5) keeps the same form while the transition matrix follows different constraint:

$$\gamma_{i,j}\geq 0; \qquad \forall j,\ \sum_{i=1}^{n}\gamma_{i,j}=r_{2,j}; \qquad \forall i,\ \sum_{j=1}^{n}\gamma_{i,j}=r_{1,i}. \tag{6}$$

Note that in both setting, the computation of Wasserstein distance is reduced to an LP, i.e.,

$$\begin{aligned} \min \quad & \sum_{1\leq i,j\leq n} C_{i,j}\gamma_{i,j} \\ \text{subject to} \quad & \sum_{j=1}^{n}\gamma_{i,j}=r_{1,i}, \quad \sum_{j=1}^{n}\gamma_{i,j}=r_{2,i}, \quad \gamma_{i,j}\geq 0, \end{aligned} \tag{7}$$

where $r^1 \triangleq (r_{1,1}, ..., r_{1,n})^T$ and $r^2 \triangleq (r_{2,1}, ..., r_{2,n})^T$ are two probability distributions, and $C_{i,j} = dist(\tilde{x}^i, x^j)^p$. More generally, we can let r^1 and r^2 be two nonnegative vectors and $C_{i,j} = C(\tilde{y}^i, y^j)$ be any appropriate transportation cost from $\tilde{y}^i$ to y^j, so (7) also captures the discrete OT.

However when the number of particles n becomes large, the number of variables (entries of γ) scales like n^2, which leads to costly computation. Therefore, we will discuss random block coordinate descent methods to keep the computational workload in each iteration reasonable.

3 Random block coordinate descent for standard LP

In this section, we first generalize the LP problem (7) to a more general one (see Eq.(8)). Then we propose a random block coordinate descent algorithm for resolution. Its almost sure convergence and linear convergence rate in expectation are analyzed.

We consider the following standard LP problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^N} \quad & c^T x \\ \text{subject to} \quad & Ax = b,\ x \geq 0, \end{aligned} \tag{8}$$

where $A \in \mathbb{R}^{M \times N}$, $b \in \mathbb{R}^M$, $c \in \mathbb{R}^N$, hence M is the number of constraint and N is the total degree of freedom. Suppose that $\mathcal{N} \triangleq \{1, \ldots, N\}$ and denote $\mathcal{X} \triangleq \{x \in \mathbb{R}^N \mid Ax = b, x \geq 0\}$ as the feasible set. Assume that (8) is finite and has an optimal solution. For any $x \in \mathcal{X}$ and $\mathcal{I} \subseteq \mathcal{N}$, denote

$$\mathcal{D}(x; \mathcal{I}) \triangleq \text{argmin}_{d \in \mathbb{R}^N} \left\{ c^T d \mid x + d \geq 0, Ad = 0, d_i = 0, \forall i \in \mathcal{N} \setminus \mathcal{I} \right\}. \tag{9}$$

$$q(x; \mathcal{I}) \triangleq \min_{d \in \mathbb{R}^N} \left\{ c^T d \mid x + d \geq 0, Ad = 0, d_i = 0, \forall i \in \mathcal{N} \setminus \mathcal{I} \right\}. \tag{10}$$

Namely, $\mathcal{D}(x; \mathcal{I})$ is the optimal solution set of the linear program in (9) and $q(x; \mathcal{I})$ is the optimal function value. We have that $q(x; \mathcal{I}) = c^T d$ for any $d \in \mathcal{D}(x; \mathcal{I})$. Denote $\mathcal{X}^*$ as the optimal solution set of (8). Then the following equations hold for any $x \in \mathcal{X}$:

$$\mathcal{X}^* = x + \mathcal{D}(x; \mathcal{N}), \tag{11}$$

$$q(x; \mathcal{N}) = c^T x^* - c^T x, \quad \forall x^* \in \mathcal{X}^*. \tag{12}$$

Consider the block coordinate descent (BCD) method for (8):

$$\begin{aligned} &\text{find } d^k \in \mathcal{D}(x^k, \mathcal{I}_k), \\ &x^{k+1} := x^k + d^k, \end{aligned} \tag{13}$$

where $\mathcal{I}_k \subset \mathcal{N}$ is the working set chosen at iteration k. Next we describe several approaches to select the working set $\mathcal{I}_k$.

Gauss-Southwell-q rule Motivated by the *Gauss-Southwell-q* rule introduced in [43], we desire to select $\mathcal{I}_k$ such that

$$q(x^k; \mathcal{I}_k) \leq v q(x^k; \mathcal{N}), \tag{14}$$

for some constant $v \in (0, 1]$. Note that by (12), we have

$$q(x^k; \mathcal{N}) = c^T (x^* - x^k), \tag{15}$$

where x^* is an optimal solution of (8). Therefore, (10)-(15) imply that

$$\begin{aligned} & c^T d^k \leq v c^T (x^* - x^k) \\ \overset{(13)}{\Longrightarrow}\ & c^T (x^{k+1} - x^k) \leq v c^T (x^* - x^k) \\ \Longrightarrow\ & c^T (x^{k+1} - x^*) \leq (1 - v) c^T (x^k - x^*). \end{aligned} \tag{16}$$

(16) indicates that the gap of function value decays exponentially with rate $1-v$, as long as we choose $\mathcal{I}_k$ according to the Gauss-Southwell-q rule (14) at each iteration k. A trivial choice of $\mathcal{I}_k$ to satisfy (14) is $\mathcal{N}$ and $v=1$. However, this choice results in a potential large-scale subproblem in BCD method (13), contradicting the purpose of using BCD. Instead, we should set an upper bound on the cardinality of $\mathcal{I}_k$, namely, a reasonable batchsize to balance the computational effort in each iteration and covergence performance of BCD. Next we discuss the existence of such an $\mathcal{I}_k$ given an upper bound l on $|\mathcal{I}_k|$, which necessitates the following concept.

Definition 1. *A vector $\bar{d} \in \mathbb{R}^N$ is conformal to $d \in \mathbb{R}^N$ if*

$$\mathrm{supp}(\bar{\mathrm{d}}) \subseteq \mathrm{supp}(\mathrm{d}),\ \bar{\mathrm{d}}_{\mathrm{i}}\mathrm{d}_{\mathrm{i}} \geq 0, \forall \mathrm{i} \in \mathcal{N}.$$

The following Theorem confirms the existence of such an $\mathcal{I}_k$ that satisfies (14), the proof of which follows closely to [42, Proposition 6.1] and can be found in the appendix.

Theorem 1. *Given any $x \in \mathcal{X}$, $l \in \{\mathrm{rank}(A)+1, \ldots, N\}$ and $d \in \mathcal{D}(x;\mathcal{N})$. There exist a set $\mathcal{I} \in \mathcal{N}$ satisfying $|\mathcal{I}| \leq l$ and a vector $\bar{d} \in \mathrm{null(A)}$ conformal to d such that*

$$\mathcal{I} = \mathrm{supp}(\bar{\mathrm{d}}). \tag{17}$$

$$q(x;\mathcal{I}) \leq \frac{1}{N-l+1} q(x;\mathcal{N}). \tag{18}$$

Proof. If $d=0$, then let $\bar{d}=0$ and $\mathcal{I}=\emptyset$. We have $q(x;\mathcal{I}) = q(x;\mathcal{N}) = 0$. Therefore, both (17) and (18) are satisfied. If $d \neq 0$ and $|\mathrm{supp(d)}| \leq \mathrm{l}$, then let $\bar{d}=d$. Thus, $\mathcal{I} = \mathrm{supp}(\bar{\mathrm{d}})$ satisfies $|\mathcal{I}| \leq l$ and $q(x;\mathcal{I}) = q(x;\mathcal{N})$. If $|\mathrm{supp(d)}| > \mathrm{l}$, then similar to the discussion in [42, Proposition 6.1], we have that

$$d = d^{(1)} + \ldots + d^{(r)},$$

for some $r \leq |\mathrm{supp(d)}| - \mathrm{l} + 1$ and some nonzero $d^{(s)} \in \mathrm{null(A)}$ conformal to d with $|\mathrm{supp}(\mathrm{d}^{(\mathrm{s})})| \leq \mathrm{l}$, $s = 1, ..., r$. Since $|\mathrm{supp(d)}| \leq \mathrm{N}$, we have $r \leq N-l+1$. Since $Ad^{(s)} = 0$ and $x_i + d_i^{(s)} \geq x_i + d_i \geq 0, \forall s = 1, ..., r$ and $\forall i \in \{i \mid d_i < 0\}$, we have that $x + d^{(s)} \in \mathcal{X}, \forall s = 1, ..., r$. Therefore,

$$q(x;\mathcal{N}) = c^T d = \sum_{s=1}^{r} c^T d^{(s)} \geq r \min_{s=1,\ldots,r} \{c^T d^{(s)}\}.$$

Denote $\bar{s} \in \mathrm{argmin}_{s=1,\ldots,r}\{c^T d^{(s)}\}$ and let $\mathcal{I} = \mathrm{supp}(\mathrm{d}^{(\bar{\mathrm{s}})})$, then $|\mathcal{I}| \leq l$ and

$$q(x;\mathcal{N}) \geq r c^T d^{(\bar{s})} \geq r q(x;\mathcal{I}) \geq (N-l+1) q(x;\mathcal{I}).$$

Therefore (17) and (18) hold for this $\mathcal{I}$ and $\bar{d} = d^{(\bar{s})}$. ∎

However, it is not clear how to identify the set $\mathcal{I}$ described in Theorem 1 with little computational effort for a general A. Therefore, we introduced the following.

Expected Gauss-Southwell-q rule We introduce randomness in the selection of $\mathcal{I}_k$ to reduce the potential computation burden in identifying an $\mathcal{I}_k$ that satisfies (14). Consider an *expected Gauss-Southwell-q rule*:

$$\mathbb{E}[q(x^k;\mathcal{I}_k) \mid \mathcal{F}_k] \leq v q(x^k;\mathcal{N}), \tag{19}$$

where $v \in (0,1]$ is a constant, and $\mathcal{F}_k \triangleq \{x^0, \ldots, x^k\}$ denotes the history of the algorithm. Therefore, using the notations of LP (8) and BCD method (13):

$$\begin{aligned}
(10)(15)(19) &\implies \mathbb{E}[c^T d^k \mid \mathcal{F}_k] \leq v c^T (x^* - x^k) \\
&\implies \mathbb{E}[c^T (x^{k+1} - x^k) \mid \mathcal{F}_k] \leq v c^T (x^* - x^k) \\
&\implies \mathbb{E}[c^T (x^{k+1} - x^*) \mid \mathcal{F}_k] - c^T (x^k - x^*) \leq v c^T (x^* - x^k)
\end{aligned}$$

$$\implies \mathbb{E}[c^T(x^{k+1} - x^*) \mid \mathcal{F}_k] \leq (1 - v)c^T(x^k - x^*), \tag{20}$$

where x^* is an optimal solution of (8). According to [33, Lemma 10, page 49], $c^T(x^k - x^*) \to 0$ almost surely. Moreover, if we take expectations on both sides of (20),

$$\begin{aligned} &\mathbb{E}[c^T(x^{k+1} - x^*)] \leq (1 - v)\mathbb{E}[c^T(x^k - x^*)] \\ \implies &\mathbb{E}[c^T(x^k - x^*)] \leq (1 - v)^k\mathbb{E}[c^T(x^0 - x^*)]. \end{aligned}$$

i.e., the expectation of function value gap converges to 0 exponentially with rate $1 - v$.

Vanilla random block coordinate descent Based on the expected Gauss-Southwell-q rule, we formally propose a vanilla random block coordinate descent (**RBCD$_0$**) algorithm (Algorithm 1) to solve the LP (8). Specifically, we choose the working set $\mathcal{I}_k$ with full randomness, that is, randomly choose an index set of cardinality l out of $\mathcal{N}$. Then with probability at least $\frac{1}{\binom{N}{l}}$, the index set will be the same as or cover the working set suggested by Theorem 1. As a result, (19) will be satisfied with $v \geq \frac{1}{\binom{N}{l}(N-l+1)}$.

Algorithm 1 Vanilla random block coordinate descent (**RBCD$_0$**)

(Initialization) Choose feasible $x^0 \in \mathbb{R}^N$ and the batch size $l > 0$.
for $k = 0, 1, 2, \ldots$ **do**
  Step 1. Choose $\mathcal{I}_k$ uniformly randomly from $\mathcal{N}$ with $|\mathcal{I}_k| = l$.
  Step 2. Find $d^k \in \mathcal{D}(x^k; \mathcal{I}_k)$.
  Step 3. $x^{k+1} := x^k + d^k$.
end for

Therefore, according to the previous discussions, Algorithm 1 generates a sequence $\{x^k\}$ such that the value of $c^T x^k$ converges to the optimal with probability 1. Moreover, the expectation of the optimality gap converges to 0 exponentially. Note that $\frac{1}{\binom{N}{l}(N-l+1)}$ is only a loose lower bound of v. It can be very small when N grows large because of the binomial coefficient $\binom{N}{l}$. However, in the numerical experiments (c.f. Sec. 5), this lower bound is rarely met. In the following subsection, we will discuss further improvement of this bound given the specific structure of OT.

4 Random block coordinate descent and optimal transport

Denote the cost matrix $C \triangleq (C_{i,j})_{i,j}$ in (7). Then calculating the OT between two measures with finite support (problem (7)) is a special case of (8), where $c = \mathrm{vec(C)}$, and $N = n^2$. The constraint matrix A has the following structure:

$$A \triangleq \underbrace{\begin{pmatrix} I_n & I_n & \ldots & I_n \\ \mathbf{1}_n^T & & & \\ & \mathbf{1}_n^T & & \\ & & \ddots & \\ & & & \mathbf{1}_n^T \end{pmatrix}}_{n \text{ blocks}}, \tag{21}$$

where I_n is an $n \times n$ identity matrix, $\mathbf{1}_n$ is an n dimensional vector of all 1's (then $M = 2n$). Right hand side b in (8) has the form $b \triangleq ((r^1)^T, (r^2)^T)^T$, where $r^1, r^2 \in \mathbb{R}^n_+$ can be two discrete probability distributions. Next, we discuss the property of matrix A and null(A).

Property of matrix A A nonzero $d \in \mathbb{R}^N$ is an *elementary vector* of null(A) if $d \in$ null(A) and there is no nonzero $d' \in$ null(A) that is conformal to d and supp(d′) $\neq$ supp(d). According to the definition in (21), we say that a nonzero matrix X is an *elementary matrix* of null(A) if vec(X) is an elementary

vector of null(A). For simplicity, a matrix M^1 being conformal to M^2 means vec(M^1) being conformal to vec(M^2) for the rest of this paper. Now we define a set $\mathcal{E}_A$:

$X \in \mathcal{E}_A \subseteq \mathbb{R}^{n\times n} \iff X \neq 0$, and after row and column permutations, X is a multiple of one of the following matrices:

$$E^2 = \begin{pmatrix} 1 & -1 & & & \\ -1 & 1 & & & \\ & & 0 & & \\ & & & \ddots & \\ & & & & 0 \end{pmatrix}_{n\times n}, E^3 = \begin{pmatrix} 1 & & -1 & & \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{pmatrix}_{n\times n}, ...,$$

$$E^{n-1} = \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & & \ddots & \\ & & & -1 & 1 \\ & & & & & 0 \end{pmatrix}_{n\times n}, E^n = \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & & \ddots & \\ & & & -1 & 1 \end{pmatrix}_{n\times n}.$$

Lemma 2. *Every matrix in $\mathcal{E}_A$ is an elementary matrix of* null(A).

Theorem 3. *Given any $D \in \mathbb{R}^{n\times n}$, if* vec(D) $\in$ null(A)*, then D has a conformal realization [37, Section 10B], namely:*

$$D = D^{(1)} + D^{(2)} + \ldots + D^{(s)}, \tag{22}$$

where $D^{(1)}, \ldots, D^{(s)}$ are elementary matrices of null(A) *and $D^{(i)}$ is conformal to D, for all $i = 1, \ldots, s$. In particular, $D^{(i)} \in \mathcal{E}_A$, $\forall i = 1, ..., s$. Therefore, $\mathcal{E}_A$ includes all the elementary matrices of* null(A).

Proof. First, we show that for any nonzero D such that vec(D) $\in$ null(A), there exists $X \in \mathcal{E}_A$ such that X is conformal to D. We prove this by contradiction and induction.

Suppose that no $X \in \mathcal{E}_A$ is conformal to D. Note that vec(D) $\in$ null(A) is equivalent to $\sum_{i=1}^{m} D(i, \bar{j}) = \sum_{j=1}^{n} D(\bar{i}, j) = 0, \forall \bar{i}, \bar{j}$. WLOG, suppose that $D(1,1) \neq 0$ since we can permute row/column to let $D(1,1) \neq 0$. Further, suppose that $D(1,1) > 0$ since we can otherwise prove the same statement for $-D$. Since vec(D) $\in$ null(A), the first column of D must have one negative element. Suppose $D(2,1) < 0$ WLOG. The second row of D must have one positive element, so suppose $D(2,2) > 0$ WLOG. Since no $X \in \mathcal{E}_A$ is conformal to D, we must have $D(1,2) \geq 0$. Therefore, the 2×2 principal matrix of D has the following sign arrangement (after appropriate row/column permutations),

$$\begin{pmatrix} + & +/0 \\ - & + \end{pmatrix},$$

where we use $+$, $+/0$, $-$, and $-/0$ to indicate that the corresponding entry is positive, nonnegative, negative, and nonpositive respectively. If $n = 2$, then the above pattern is impossible, leading to a contradiction. Suppose that $n \geq 3$. For math induction, we assume that after appropriate row/column permutations, the $k \times k$ principal matrix of D has the following sign arrangement ($2 \leq k \leq n-1$),

$$\begin{pmatrix} + & +/0 & +/0 & \ldots & +/0 \\ - & + & +/0 & \ddots & \vdots \\ -/0 & - & + & \ddots & +/0 \\ \vdots & \ddots & \ddots & \ddots & +/0 \\ -/0 & \ldots & -/0 & - & + \end{pmatrix}, \tag{23}$$

i.e., $D(i,j) \geq 0$, $\forall i \leq j \leq k$; $D_{ij} \leq 0$, $\forall j < i \leq k$; $D(i,i) > 0$, $\forall 1 \leq i \leq k$; $D(i+1,i) < 0$, $\forall 1 \leq i \leq k-1$.

kth column of D needs to have at least one negative element, so suppose $D(k+1,k) < 0$ WLOG. No $X \in \mathcal{E}_A$ is conformal to D, so $D(k+1,i) \leq 0$, $\forall i = 1, ..., k-1$. Otherwise, let i_0 be the largest index $1, \cdots, k-1$ such that $D(k+1, i_0) > 0$. Then the submatrix $D(i_0+1 : k+1, i_0 : k)$ takes the form,

$$\begin{pmatrix} - & + & +/0 & \dots & +/0 \\ -/0 & - & + & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & +/0 \\ -/0 & \dots & -/0 & - & + \\ + & -/0 & \dots & -/0 & - \end{pmatrix}. \tag{24}$$

Moving the first column of (24) to the last (i.e., for D, move the i_0th column and insert it between k and $k+1$th column) and shift the resulting submatrix to the upper left corner through permutation operations, we can see E_{k-i_0+1} is conformal to it.

$(k+1)$th row of D needs to have at least one positive element, so suppose $D(k+1, k+1) > 0$ WLOG. Similar argument shows if there is no $X \in \mathcal{E}_A$ is conformal to D, so $D(i, k+1) \geq 0$, $\forall i = 1, ..., k$.

Therefore, the $(k+1) \times (k+1)$ principal matrix of D has exactly the same sign pattern as indicated by (23), after appropriate row/column permutations. Note that this is true when $k+1 = n$. However, D itself cannot have the sign pattern as (23) after row/column permutations since the summation of each column/row of D is 0. Contradiction.

Suppose that $X^{(1)} \in \mathcal{E}_A$ and $X^{(1)}$ is conformal to D. Then $X^{(1)}$ can be scaled properly by $\alpha_1 > 0$ such that $|\text{supp}(\text{D} - \alpha_1 \text{X}^{(1)})| < |\text{supp}(\text{D})|$ and $D - \alpha_1 X^{(1)}$ is conformal to D. Denote $D^{(1)} \triangleq \alpha_1 X^{(1)}$ and $\bar{D}^{(1)} = D - D^{(1)}$. $\bar{D}^{(1)}$ is the new D and we repeat this process. Eventually, we have that the conformal realization (22) holds since $|\text{supp}(\text{D})| \leq \text{n}^2$. If D is an elementary matrix, by the conformal realization of D as in (22), D must have the same support with all $D^{(i)} \in \mathcal{E}_A$, $i = 1, ..., s$. Therefore, by definition of $\mathcal{E}_A$, D must be a multiple of the special matrix in the description of $\mathcal{E}_A$ after a certain row/column permutation, and itself is in $\mathcal{E}_A$. Thus $\mathcal{E}_A$ describes all the elementary matrices of null(A). ∎

Remark. *For a given D such that* vec(D) ∈ null(A)*, a simple algorithm following the proof in Theorem 3 to find an elementary matrix X of $D \neq 0$ will cost at most $O(n^2)$ operations. Select appropriate $\alpha > 0$ such that $D - \alpha X$ is conformal to D and* $|\text{supp}(\text{D} - \alpha\text{X})| < |\text{supp}(\text{D})|$*. Repeat this process and we can find the conformal realization in* $\text{supp}(\text{D}) \leq \text{n}^2$ *steps. Therefore, the total operation to find the conformal realization is $O(n^4)$. In contrast, the approach proposed by [44] finds a conformal realization with support cardinality less than l (usually l is much smaller than n^2) is $O(n^3(n^2 - l)^2)$.*

Working set selection By analyzing the structure of elementary matrices of null(A), we will have a better idea of potential directions along which the transport cost is minimized by a large amount. This is supported by the following theorem, where we continue using notations introduced in Section 3.

Theorem 4. *Consider the linear program* (8) *where $A \in \mathbb{R}^{M \times N}$ and $b \in \mathbb{R}^M$ are defined as in* (21) *($M = 2n$, $N = n^2$). Given any $X \in \mathbb{R}^{n \times n}$ and $D \in \mathbb{R}^{n \times n}$ such that* vec(X) ∈ $\mathcal{X}$*, and* vec(D) ∈ $\mathcal{D}(\text{vec(X)}; \mathcal{N})$*. There exists an elementary matrix $\bar{D}$ of* null(A) *conformal to D such that for any set $\mathcal{I} \in \mathcal{N}$ satisfying*

$$\mathcal{I} \supseteq \text{supp}(\text{vec}(\bar{\text{D}})),$$

We have

$$q(\text{vec(X)}; \mathcal{I}) \leq \left(\frac{1}{\text{n}^2 - 3}\right) \text{q}(\text{vec(X)}; \mathcal{N}). \tag{25}$$

Proof. Since vec(D) ∈ $\mathcal{D}(\text{vec(X)}; \mathcal{N})$, vec(D) ∈ null(A). Then based on Theorem 3, we have the conformal realization:

$$D = D^{(1)} + D^{(2)} + ... + D^{(s)}.$$

Moreover, proof of Theorem 3 indicates that we can construct this realization with $s \leq n^2 - 3$, because the support of $D^{(i)}$ has cardinality at least 4. Then similar to discussion in Theorem 1, we may find $\bar{s} \in \{1, \ldots, s\}$ such that $\bar{D} = D^{(\bar{s})}$, $\mathcal{I} \supseteq \mathrm{supp}(\mathrm{vec}(\mathrm{D}^{(\bar{s})}))$, and

$$q(\mathrm{vec(X)}; \mathcal{N}) \geq (\mathrm{n}^2 - 3)\mathrm{q}(\mathrm{vec(X)}; \mathcal{I}).$$

∎

Now we discuss two approaches to carefully select the support set $\mathcal{I}_k$ at iteration k of the block coordinate descent method (13):

1. *Diagonal band.* Given $3 \leq p < n$, denote

$$\mathcal{G} \triangleq \left\{ (i,j) \in \mathbb{Z}^2 \middle| \begin{array}{lll} i \in [j, j+p-1] & \text{if} & j \in [1, n-p+1] \\ i \in [1, ..., j+p-n-1] \cup [j, n] & \text{if} & j \in [n-p+2, n] \end{array} \right\},$$

and construct matrix $G \in \mathbb{R}^{n \times n}$ such that

$$G(i,j) = \begin{cases} 1, & \text{if} \quad (i,j) \in \mathcal{G}, \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

Therefore, G has the following structure:

$$p \left\{ \begin{pmatrix} 1 & & & & 1 & \ldots & 1 \\ \vdots & 1 & & & & \ddots & \vdots \\ 1 & \vdots & \ddots & & & & 1 \\ 1 & 1 & & 1 & & & \\ & 1 & \ddots & \vdots & 1 & & \\ & & \ddots & 1 & \vdots & \ddots & \\ & & & 1 & 1 & \ldots & 1 \end{pmatrix}_{n \times n} \right\} (p-1)$$

It is like a band of width p across the diagonal, hence the name. Then we may construct $\bar{D}^k \in \mathbb{R}^{n \times n}$ and $\mathcal{I}_k$ as follows:

$$\begin{array}{l} \text{Obtain } \bar{D}^k \text{ by uniformly randomly permuting all columns and rows of } G. \\ \text{Let} \quad \mathcal{I}_k \triangleq \mathrm{supp}(\mathrm{vec}(\bar{\mathrm{D}}^{\mathrm{k}})). \end{array} \tag{27}$$

Note that $|\mathcal{I}_k| = np$.

2. *Submatrix.* Given $m < n$, obtain $\bar{D}^k$ and $\mathcal{I}_k$ such that

$$\begin{array}{l} \text{Uniformly randomly pick two sets of } m \text{ different numbers out of } \{1, ...n\}: \\ i_1, ..., i_m \quad \text{and} \quad j_1, ..., j_m. \\ \text{Let} \quad \bar{D}^k(i,j) = \begin{cases} 1 & \text{if} \quad i \in \{i_1, ..., i_m\} \quad \text{and} \quad j \in \{j_1, ..., j_m\}, \\ 0 & \text{otherwise.} \end{cases} \\ \text{Let} \quad \mathcal{I}_k \triangleq \mathrm{supp}(\mathrm{vec}(\bar{\mathrm{D}}^{\mathrm{k}})). \end{array} \tag{28}$$

In this case, the support of $\bar{D}^k$ is a submatrix of size $m \times m$. Therefore, $\mathcal{I}_k = m^2$.

Next we discuss two random block coordinate descent algorithms to solve (8)(21) whose working set selections are based on the two approaches discussed above.

Algorithm 2 Random block coordinate descent - diagonal band (**RBCD-DB**)

(Initialization) Choose feasible $X^0 \in \mathbb{R}^{n\times n}$ and band width $p \in [3, n]$. Let $x^0 = \mathrm{vec}(\mathrm{X}^0)$.
for $k = 0, 1, 2, \ldots$ **do**
  Step 1. Choose $\mathcal{I}_k$ according to (27).
  Step 2. Find $d^k \in \mathcal{D}(x^k; \mathcal{I}_k)$.
  Step 3. $x^{k+1} := x^k + d^k$.
end for

The following result describes the convergence property of Algorithm 2.

Theorem 5. *Consider* (8)(21). *Then sequence* $\{x^k\}$ *and* $\{\mathcal{I}_k\}$ *generated by Algorithm 2 satisfies the expected Gauss-Southwell-q rule* (19), *i.e.,*

$$\mathbb{E}[q(x^k; \mathcal{I}_k) \mid \mathcal{F}_k] \leq v q(x^k; \mathcal{N}),$$

with $v \geq \frac{n(p-2)}{(n^2-3)(n!)^2}$. *Therefore,* $c^T(x^k - x^*) \to 0$ *almost surely and* $\mathbb{E}[c^T(x^k - x^*)]$ *converges to* 0 *exponentially with rate* $1 - v$.

Proof. Given x^k, Theorem 4 guarantees that there exists $D^k \in \mathcal{E}_A$ such that if $\mathcal{I}_k \supseteq \mathrm{supp}(\mathrm{vec}(\mathrm{D^k}))$, then (25) holds for $\mathcal{I} = \mathcal{I}_k$ and $\mathrm{vec(X)} = \mathrm{x^k}$, i.e.,

$$q(x^k; \mathcal{I}_k) \leq \left(\frac{1}{n^2 - 3}\right) q(x^k; \mathcal{N}). \tag{29}$$

Next, we will estimate the probability that $\mathcal{I}_k \supseteq \mathrm{supp}(\mathrm{vec}(\mathrm{D^k}))$ holds.

Suppose that after row/column permutations and scaling of D^k, we obtain E^t, $2 \leq t \leq n$. Then after appropriate row and column swapping, D^k can be written as

$$t\left\{\begin{pmatrix} 0 & & & & & & & & & \\ * & * & & & & & & & & \\ * & & * & & & & & & & \\ & * & & \ddots & & & & & & \\ & & * & & * & & & & & \\ & & & \ddots & & * & & & & \\ & & & & * & * & 0 & & & \\ & & & & & & & \ddots & & \\ & & & & & & & & 0 & \end{pmatrix}_{n\times n}\right. . \tag{30}$$

That is, elements $(2, 1)$ and $(3, 1)$ are nonzeros; elements (j, j) and $(\mathrm{mod(j+2, n)}, \mathrm{j})$ are nonzeros, for all $j = 2, ..., t-1$; elements (t, t) and $(\mathrm{mod(t+1, n)}, \mathrm{t})$ are nonzeros; all other elements are zeros. Obviously, support of this matrix is covered by the support of G in (26). Moreover, by moving the whole support in matrix (30) downwards or to the bottom right corner, we can create at least $n(p-2)-1$ more different matrices whose support are all covered by G. These $n(p-2)$ matrices can be obtained by permuting rows and columns of D^k in $n(p-2)$ different ways. Therefore, the probability that $\mathcal{I}_k$ will cover the support of D^k is at least $\frac{n(p-2)}{(n!)^2}$, and we have that

$$\begin{aligned}
\mathbb{E}[q(x^k; \mathcal{I}_k) \mid x^k] &= \sum_{\mathrm{supp(vec(D^k))} \subseteq \mathcal{I}} q(x^k; \mathcal{I}) P(\mathcal{I}_k = \mathcal{I}) + \sum_{\mathrm{supp(vec(D^k))} \not\subseteq \mathcal{I}} q(x^k; \mathcal{I}) P(\mathcal{I}_k = \mathcal{I}) \\
&\leq \left(\frac{1}{n^2 - 3}\right) q(x^k; \mathcal{N}) P(\mathrm{supp(vec(D^k))} \subseteq \mathcal{I}_\mathrm{k}) + 0 \\
&\leq \left(\frac{n(p-2)}{(n^2-3)(n!)^2}\right) q(x^k; \mathcal{N})
\end{aligned}$$

Therefore, the expected Gauss-Southwell-q rule (19) holds with v at least $\frac{n(p-2)}{(n^2-3)(n!)^2}$. ∎

Remark. *It can be shown that if n is large enough and p is chosen between $O(log(n))$ and $O(n)$, then the lower bound for constant v derived in Theorem 5 is better than the one estimated for Algorithm 1, i.e., $\frac{1}{\binom{N}{l}(N-l+1)}$. In fact, we have the following results.*

Lemma 6. *Suppose that $\bar{K} \geq 2$ and $\eta > 0$ satisfies*

$$\frac{2\bar{K}-3}{2(\bar{K}-1)} + \log\left(\frac{\bar{K}}{2}\right) > 2/\eta,$$

and n satisfies

$$n \geq \frac{4}{\left(\frac{2\bar{K}-3}{2(\bar{K}-1)} + \log\left(\frac{\bar{K}}{2}\right)\right)\eta - 2}, \quad \frac{n}{\log(n)} \geq \eta\bar{K}.$$

Then for any $p \in [\eta\log(n), \frac{n}{\bar{K}}]$, and $p \geq 3$, we have $\frac{n(p-2)}{(n^2-3)(n!)^2} \geq \frac{1}{\binom{n^2}{np}(n^2-np+1)}$.

Proof. See Appendix. ∎

Let $n \geq 30$, $\eta = 1$, $\bar{K} = 8$. Then according to Lemma 6, for $log(n) \leq p \leq n/8$, the lower bound $\frac{n(p-2)}{(n^2-3)(n!)^2}$ is larger. We believe that this is a fairly reasonable range of p when n grows large. This lower bound is improved because we have knowledge of the structure of the elementary matrix when solving OT.

As for the submatrix approach, we often find it quite efficient in numerical experiments. However, its global convergence property is not guaranteed. In fact, there is a counterexample (see B). Therefore, we design an algorithm that combines these two approaches together.

Algorithm 3 Random block coordinate descent - submatrix and diagonal Band (**RBCD-SDB**)

(Initialization) Choose feasible $X^0 \in \mathbb{R}^{n\times n}$, submatrix row/column dimension m, band width $p \in [3, n]$ and selection parameter $s \in [0, 1]$. Let $x^0 = \mathrm{vec}(\mathrm{X}^0)$.
for $k = 0, 1, 2, \ldots$ **do**
Step 1. With probability s, choose $\mathcal{I}_k$ according to (28); otherwise, choose $\mathcal{I}_k$ according to (27).
Step 2. Find $d^k \in \mathcal{D}(x^k; \mathcal{I}_k)$.
Step 3. $x^{k+1} := x^k + d^k$.
end for

The convergence of Algorithm 3 is guaranteed by the next theorem.

Theorem 7. *Consider* (8)(21). *Then sequence $\{x^k\}$ and $\{\mathcal{I}_k\}$ generated by Algorithm 2 satisfies the expected Gauss-Southwell-q rule* (19), *with $v \geq \frac{sn(p-2)}{(n^2-3)(n!)^2}$. Therefore, $c^T(x^k - x^*) \to 0$ almost surely and $\mathbb{E}[c^T(x^k - x^*)]$ converges to 0 exponentially with rate $1 - v$.*

Proof. Given x^k, Theorem 4 shows that there exists $D^k \in \mathcal{E}_A$ such that if $\mathcal{I}_k \supseteq \mathrm{supp}(\mathrm{vec}(\mathrm{D}^k))$, then (25) holds with $\mathcal{I} = \mathcal{I}_k$ and $\mathrm{vec}(\mathrm{X}) = \mathrm{x}^k$. We estimate the probability that $\mathcal{I}_k \supseteq \mathrm{supp}(\mathrm{vec}(\mathrm{D}^k))$.

First, consider the case that after row/column permutations and scaling of D^k, we obtain E^t, $2 \leq t \leq m$. If $\mathcal{I}_k$ is chosen according to (27), then similar to discussion in Theorem 5, $\mathcal{I}_k$ will cover the support of D^k with probability at least $\frac{n(p-2)}{(n!)^2}$. If $\mathcal{I}_k$ is chosen according to (28), then $\mathcal{I}_k$ will cover the support of D^k with probability

$$\frac{\binom{n-t}{m-t}^2}{\binom{n}{m}^2} = \left(\frac{(n-t)!/((m-t)!(n-m)!)}{n!/(m!(n-m)!)}\right)^2 = \left(\frac{m!/(m-t)!}{n!/(n-t)!}\right)^2.$$

Therefore, in this case, the probability p_t that $\mathcal{I}_k$ cover the support of D^k is:

$$p_t \geq \frac{sn(p-2)}{(n!)^2} + (1-s)\left(\frac{m!/(m-t)!}{n!/(n-t)!}\right)^2.$$

Then we consider the case that when we get E^t, $m+1 \leq t \leq n$ after row/column permutations and rescaling of D^k. In this case, if $\mathcal{I}_k$ is chosen according to (27), $\mathcal{I}_k$ will cover the support of D^k with probability at least $\frac{n(p-2)}{(n!)^2}$; if $\mathcal{I}_k$ is chosen according to (28), this probability is 0. Therefore, in this case we have $p_t \geq \frac{sn(p-2)}{(n!)^2}$. In general, the probability that $\mathcal{I}_k$ cover the support of D^k is at least $\min\{p_t\} \geq \frac{sn(p-2)}{(n!)^2}$. Similar to discussion in Theorem 4, (19) will hold with $v \geq \frac{sn(p-2)}{(n^2-3)(n!)^2}$. ∎

Accelerated random block coordinate descent Algorithm 4 is an accelerated random block coordinate descent (**ARBCD**) algorithm. It selects the working set $\mathcal{I}_k$ in a different way from Algorithm 3 intermittently for acceleration. At times, we build $\mathcal{I}_k$ based on the iterates generated by the algorithm in the past, i.e., $x^{end} - x^{start}$. This vector reflects the progress achieved by running the **RBCD-SDB** for a few iterations. It predicts the direction in which the algorithm potentially makes further improvements. Such a choice is analogous to the momentum concept and often employed acceleration techniques in optimization, such as in the heavy ball method and Nesterov acceleration. Algorithm 4 has a similar convergence rate as Algorithm 4 (note that acceleration iteration happens occasionally). We will verify its improved performance in numerical experiments.

Algorithm 4 Accelerated random block coordinate descent (**ARBCD**)

(Initialization) Choose feasible $X^0 \in \mathbb{R}^{n\times n}$, submatrix row/column dimension m, band width $p \in [3, n]$, selection parameter $s \in [0, 1]$, and acceleration interval T. Let $x^0 = \mathrm{vec}(\mathrm{X}^0)$, $x^{start} = x^{end} = x^0$. Binary variable *acc*.
for $k = 0, 1, 2, \ldots$ **do**
  Step 1. Choose $\mathcal{I}_k$ as following.
  if $\mathrm{mod}(\mathrm{k}+1, \mathrm{T}) \neq 0$ or $|\mathrm{supp}(\mathrm{x}^{\mathrm{end}} - \mathrm{x}^{\mathrm{start}})| \leq \mathrm{m}^2$ **then**
    $acc =$ `false`. With probability s, choose $\mathcal{I}_k$ according to (28); otherwise, choose $\mathcal{I}_k$ according to (27).
  else
    $acc =$ `true`. Choose $\mathcal{I}_k$ uniformly randomly from $\mathrm{supp}(\mathrm{x}^{\mathrm{end}} - \mathrm{x}^{\mathrm{start}})$ so that $|\mathcal{I}_k| = m^2$.
  end if
  Step 2. Find $d^k \in \mathcal{D}(x^k; \mathcal{I}_k)$.
  Step 3. Update $x^{k+1} := x^k + d^k$;
  Step 4. Update $x^{end} = x^{k+1}$.
  if $acc =$ `true`. **then**
    Update $x^{start} = x^{k+1}$.
  end if
end for

5 Numerical experiments

In this section, we conduct numerical experiments on various examples of optimal transport problems. First, we compare various random block coordinate descent methods with different working set selection proposed in this article; then we compare the one with the best performance - **ARBCD** with **Sinkhorn** and demonstrate several advantages of **ARBCD**.

5.1 Comparison between various random block coordinate descent methods with different working set selection rules

In this subsection, we apply the proposed random block coordinate descent methods (Alg. 1 - Alg. 4) to calculate the Wasserstein distance between three pairs of distributions. We compare these algorithms to illustrate the difference between various working set selection rules. We also inspect the difference between theoretical and actual convergence rates, as well as the solution sparsity.

Experiment settings We compute the Wasserstein distance between a pair of 1-dim probability distributions (standard normal and uniform over $[-1,1]$), a pair of 2-dim probability distributions (uniform over $[-\pi,\pi]^2$ and an empirical invariant measure obtained from IPM simulation of reaction-diffusion particles in advection flows, detail configurations see [46], Section 4.2, 2D cellular flow, $\kappa = 2^{-4}$), and a pair of 3-dim distributions (uniform over $[-1,1]^3$ and 3 dimensional multivariate normal distribution). When computing Wasserstein dist. between the pair of 1-dim probability distributions, we utilize their histograms (c.f. Section 2): Let $n = 1001$. Centers of the cells are $x^i = \tilde{x}^i = \frac{i-501}{500}$, $i = 1, .., 1001$; $C_{i,j} = dist(\tilde{x}^i, x^j)^2$, $1 \le i, j \le 1001$; $r_{1,i} = \frac{\phi(x^i)}{\sum_{i=1}^{1001}\phi(x^i)}$, $i = 1, ..., 1001$, where $\phi(x)$ is the pdf of standard normal; $r^2 = (1/1001, ..., 1/1001)^T \in \mathbb{R}^{1001}$. When calculating the Wasserstein dist. between the second and third pairs, we apply the point cloud setting (c.f. Section 2): Let $n = 1000$. For each pair, use i.i.d. samples $\{\tilde{y}^i\}$ and $\{y^j\}$, $1 \le i, j \le 1000$ to approximate the two continuous probability measure respectively. Let $C_{i,j} = dist(\tilde{y}^i, y^j)^2$ and $r^1 = r^2 = (1/1000, ..., 1/1000)^T \in \mathbb{R}^{1000}$. Figure 1 captures these three pairs of distributions. For all cases, we first use the `linprog` in `Matlab` to find a solution with high precision (dual-simplex, constraint tolerance 1e-9, optimality tolerance 1e-10).

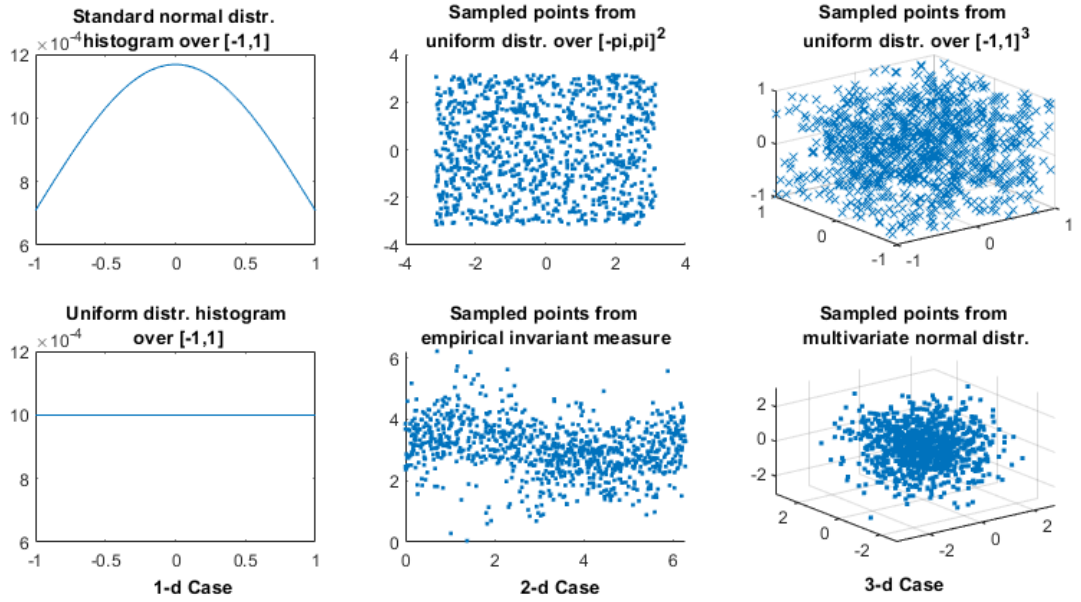

Figure 1: Three pairs of distributions
In 1-d case, we compare the histograms of two distributions; in 2-d and 3-d settings, we compare the samples/point clouds of two distributions.

Methods We specify the settings of the four algorithms. All algorithms are started at the same feasible $x^0 = \mathrm{vec}(\mathrm{r}^1(\mathrm{r}^2)^{\mathrm{T}})$ in each experiment. We solve the LP subproblems via `linprog` in `Matlab` with high precision (dual-simplex, constraint tolerance 1e-9).

RBCD$_0$. Algorithm 1: Vanilla random block coordinate descent. Let $l = 100^2$. Stop the algorithm after 5000 iterations.

RBCD-DB. Algorithm 2: Random block coordinate descent - diagonal band. Let $p = \lfloor 100^2/n \rfloor$. Stop the algorithm after 5000 iterations.

RBCD-SDB. Algorithm 3: Random block coordinate descent - submatrix and diagonal band. Let $m = 100$, $p = \lfloor m^2/n \rfloor$ and $s = 0.1$. Stop the algorithm after 5000 iterations.

ARBCD. Algorithm 4: Accelerated random block coordinate descent. Let $m = 100$, $p = \lfloor m^2/n \rfloor$, $s = 0.1$ and $T = 10$. Stop the algorithm after 5000 iterations. Note that the dimension of the subproblem per iteration is 100^2, 1/100 the size of the original one.

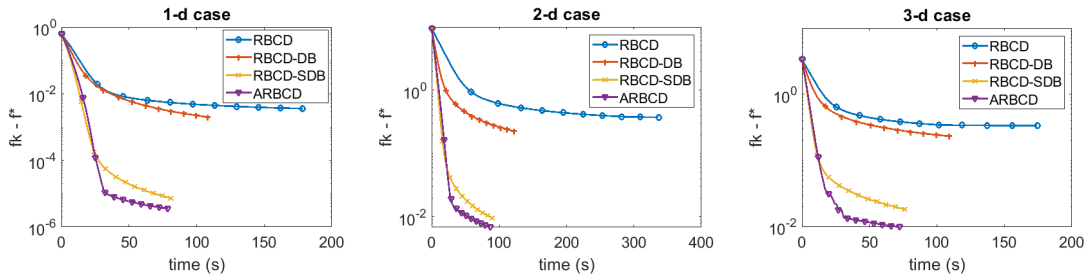

Figure 2: Comparison of algorithms to compute Wasserstein distance I

X-axis is the wall-clock time in seconds. Y-axis is the optimality gap $f_k - f^* = c^T x^k - c^T x^*$. This figure shows the trajectory/progress of Alg. 1 - Alg. 4 when computing the Wasserstein distance between the three pairs of prob. in 1-d, 2-d, and 3-d respectively. Each algorithm is run 5 times and the curves showcase the average behavior.

Table 1: Data of $\mathbf{RBCD}_0$

	1-d case			2-d case			3-d case	
iter.	$\bar{f}_k - f^*$	$\hat{v}$	iter.	$\bar{f}_k - f^*$	$\hat{v}$	iter.	$\bar{f}_k - f^*$	$\hat{v}$
0	0.6237	N/A	0	9.3538	N/A	0	3.3456	N/A
1000	0.0083	4.3e-3	1000	0.6254	2.7e-3	1000	0.4746	2.0e-3
2000	0.0054	4.3e-4	2000	0.4773	2.7e-4	2000	0.3821	2.2e-4
3000	0.0045	1.8e-4	3000	0.4183	1.3e-4	3000	0.3438	1.1e-4
4000	0.0039	1.4e-4	4000	0.3846	8.3e-5	4000	0.3371	2.0e-5
5000	0.0036	8.0e-5	5000	0.3762	2.2e-5	5000	0.3350	6.2e-6

Interpretation of Figure 2 We can see from Figure 2 that different approaches to choosing the working set of the same size may significantly affect the performance of random BCD types of methods. Curves of **RBCD-DB** are below those of $\mathbf{RBCD}_0$, showing that **RBCD-DB** has a better average performance. The reason is that $\mathbf{RBCD}_0$ generates the working set with full randomness, while **RBCD-DB** takes the structure of the elementary matrices into account. The latter makes an educated guess at the working set that decreases the objective function by a large amount. The submatrix approach (28) works very well in practice, and this is illustrated by the better performances of **RBCD-SDB** and **ARBCD** than **RBCD-DB**. In the long run, **ARBCD** dominates **RBCD-SDB**, verifying the acceleration effect. Note that the settings of algorithms are by default. We expect and observed similar behaviors of algorithms when changing the algorithm settings. On the other hand, the curves in these numerical experiments suggest sublinear convergence rates. Note that this observation does not contradict the theoretical linear convergence rate as long as v is small enough. We will verify that the numerical experiments do not violate the lower bounds we derived for the constant v in the linear convergence rates.

About Table 1 & 2 In these two tables we record the optimality gap $\bar{f}_k - f^*$ every 1000 iterations for both $\mathbf{RBCD}_0$ and **RBCD-DB**. $\bar{f}_k$ is the average function value at iteration k since we run the algorithms repeatedly for 5 times. Column $\hat{v}$ denotes the estimation of the constant v in the expected Gauss-Southwell-q rule (19). It is calculated by the formula: $\hat{v} = \sqrt[1000]{\bar{f}_k - \bar{f}_{k-1000}}$. Values of $\hat{v}$ in both

Table 2: Data of **RBCD-DB**

	1-d case			2-d case			3-d case	
iter.	$\bar{f}_k - f^*$	$\hat{v}$	iter.	$\bar{f}_k - f^*$	$\hat{v}$	iter.	$\bar{f}_k - f^*$	$\hat{v}$
0	0.6237	N/A	0	9.3538	N/A	0	3.3456	N/A
1000	0.0130	3.9e-3	1000	0.6002	2.7e-3	1000	0.4601	2.0e-3
2000	0.0057	8.2e-4	2000	0.3920	4.3e-4	2000	0.3414	3.0e-4
3000	0.0036	4.6e-4	3000	0.3074	2.4e-4	3000	0.2878	1.7e-4
4000	0.0026	3.3e-4	4000	0.2592	1.7e-4	4000	0.2544	1.2e-4
5000	0.0020	2.6e-4	5000	0.2276	1.3e-4	5000	0.2316	9.4e-5

Table 1 & 2 are far larger than the lower bounds for v: $\frac{1}{\binom{N}{l}(N-l+1)}$ and $\frac{(n-1)(p-2)}{(n^2-3)(n!)^2}$, corresponding to **RBCD$_0$** and **RBCD-DB** respectively, where $N = n^2$. They also decrease as we run more iterations, showing that the optimality gap shrinkage becomes less when the iterate is closer to the solution. We intend to study this phenomenon in our future work.

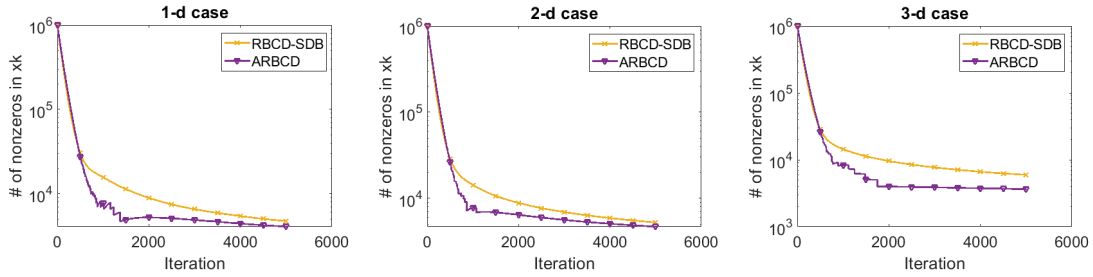

Figure 3: Sparsity of solutions
Y-axis records $\|x_k\|_0$, i.e., the number of nonzero elements in x_k. This figure shows the sparsity of x_k in **RBCD-SDB** and **ARBCD** when computing the Wasserstein distance given the three pairs of probability distributions. Each curve represents the average over 5 repetitions.

Sparse solutions We can observe from Figure 3 that the iterates in **RBCD-SDB** and **ARBCD** become sparse quickly and stay that way. The reason is that the solutions of optimal transport problems are usually sparse (for the point cloud setting, at least one of the optimal solutions satisfies $\|x^*\| = n$ because extreme points of LP in this setting are permutation matrices divided by n), and these two algorithms can locate solutions with high accuracy relatively fast. Therefore, the storage need for these two algorithms is reduced considerably after they are run for a while. In the point cloud setting, storage complexity is typically expected to decrease from $O(n^2)$ to $O(n)$ (note that the dimension of the subproblem per iteration is typically chosen as $O(n)$ because of the diagonal band approach with $p \geq 3$).

5.2 Comparison between ARBCD and Sinkhorn

In this subsection, we compare **ARBCD** with **Sinkhorn** and demonstrate several advantages **ARBCD** has over **Sinkhorn**.

Experiment settings Same as in Subsection 5.1.

Methods Implementation of **Sinkhorn** and **ARBCD** are specified as follows.
Sinkhorn. The algorithm proposed in [10] to compute Wasserstein distance. Let γ be the coefficient of the entropy term. We let $\gamma = \epsilon/(4 \log n)$ as suggested in [11]. In 1-d case, we consider the settings $\epsilon = 10^{-4}, 10^{-3}, 0.01, 0.1$ and 1; in 2-d case, we let $\epsilon = 10, 100$; in 3-d case, let $\epsilon = 1, 10$ (in 2-d and 3-d cases, smaller choices of ϵ cause numerical instability). The implementation follows Algorithm 1 in [11]. Iterations of **Sinkhorn** are projected onto the feasible region using a rounding procedure: Algorithm 2 in [1]. We stop **Sinkhorn** after 300000 (when $n \leq 201$) or 100000 (when $n \geq 1000$) iterations.
ARBCD. Algorithm 4: Accelerated random block coordinate descent. Let $m = 40$ when $n \leq 201$ and $m = 100$ when $n \geq 1000$. Let $p = \lfloor m^2/n \rfloor$, $s = 0.1$ and $T = 10$. Stop the algorithm after 5000 iterations. To be fair, we also project the solution in each iteration onto the feasible region via the rounding procedure.

Interpretation of Figure 4 We can observe the following from Figure 4: although **Sinkhorn** with larger ϵ may converge fast, the solution accuracy is also lower. In fact, this is true for all the Sinkhorn-based algorithms because the optimization problem is not exact - it has an extra entropy term. Therefore, the larger γ or ϵ is chosen, the less accurate the solution becomes. On the other hand, when ϵ is set

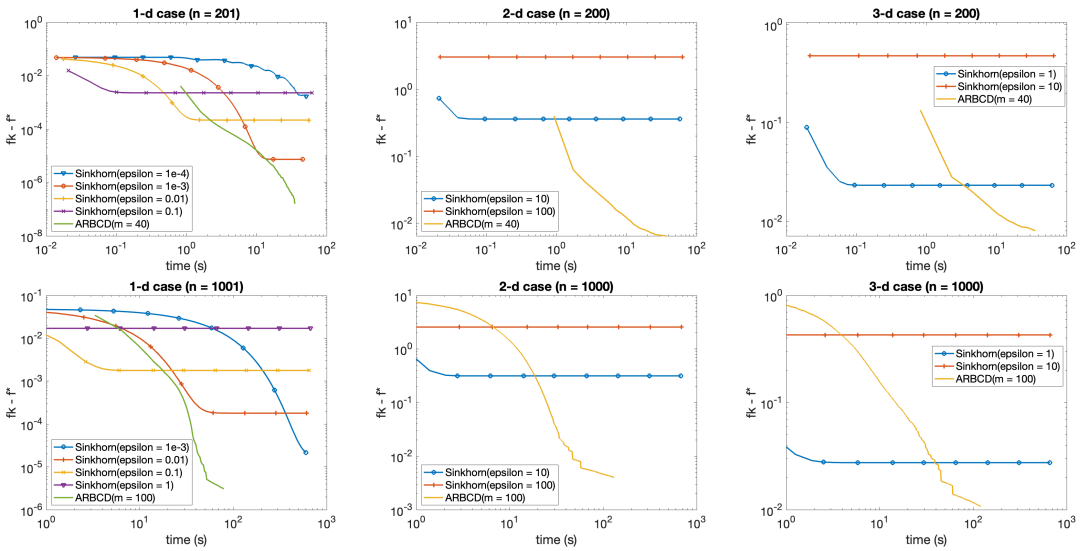

Figure 4: Comparison of algorithms to compute Wasserstein distance II
X-axis is the wall-clock time in seconds. Y-axis is the optimality gap $f_k - f^* = c^T x^k - c^T x^*$. This figure shows the trajectory/progress of Algorithm 4: **ARBCD** and **Sinkhorn** with different settings when computing the Wasserstein distance between the three pairs of prob. in 1-d, 2-d, and 3-d respectively. **ARBCD** is run 5 times and the curves showcase the average behavior.

smaller, the convergence of **Sinkhorn** may be quite slow, as is shown in the 1-d case. We also found that if a small ϵ is chosen, there may exist numerical instability issues. This is why we cannot show a curve with smaller ϵ in the 2-d and 3-d cases. This issue is also observed and mentioned in other literature [14, 18]. For the 1-d case ($n = 201$), we further inspect the solution quality in Figure 5. Transport plans in 1-d are given by different algorithms and we can see **ARBCD** recovers the optimal transport plan most accurately. In conclusion, if relatively higher precision is desired, **Sinkhorn** may not be a better choice than **ARBCD** to compute Wasserstein distance. Moreover, note that here we solve the subproblems in **ARBCD** using `Matlab` built-in solver `linprog`, which is not really considered the state-of-art. **ARBCD** can be faster if more efficient subproblem solvers are applied.

6 Conclusion

In this paper, we investigate RBCD to solve LP including OT problems. In particular, an expected Gauss-Southwell-q rule is proposed to select the working set $\mathcal{I}_k$ at iteration k. It guarantees almost sure convergence and linear convergence rate and is satisfied by all algorithms proposed in this work. We first consider a vanilla RBCD named $\mathbf{RBCD}_0$ to address a general LP. Then by exploring the structure of the matrix A in the linear system of OT, characterizing elementary matrices of null(A) and identifying conformal realization of any matrix $D \in$ null(A), we are able to refine the working set selection. We employ two approaches - diagonal band and submatrix for constructing $\mathcal{I}_k$ and an acceleration technique motivated by the momentum concept to enhance performance of $\mathbf{RBCD}_0$. In numerical experiments, we compare all the proposed RBCD methods and verify the acceleration effects as well as sparsity of solutions. We also illustrate the gap between theoretical convergence rate and the practical one. Last but not least, we run **ARBCD**, the best among all others, against the Sinkhorn's algorithm and showcase its advantages in seeking relatively accurate solutions of the OT problems.

Acknowledgements

The research of YX is partially supported by start-up fund of HKU-IDS. The research of ZZ is supported by Hong Kong RGC grant (Projects 17300318 and 17307921), the National Natural Science Foundation

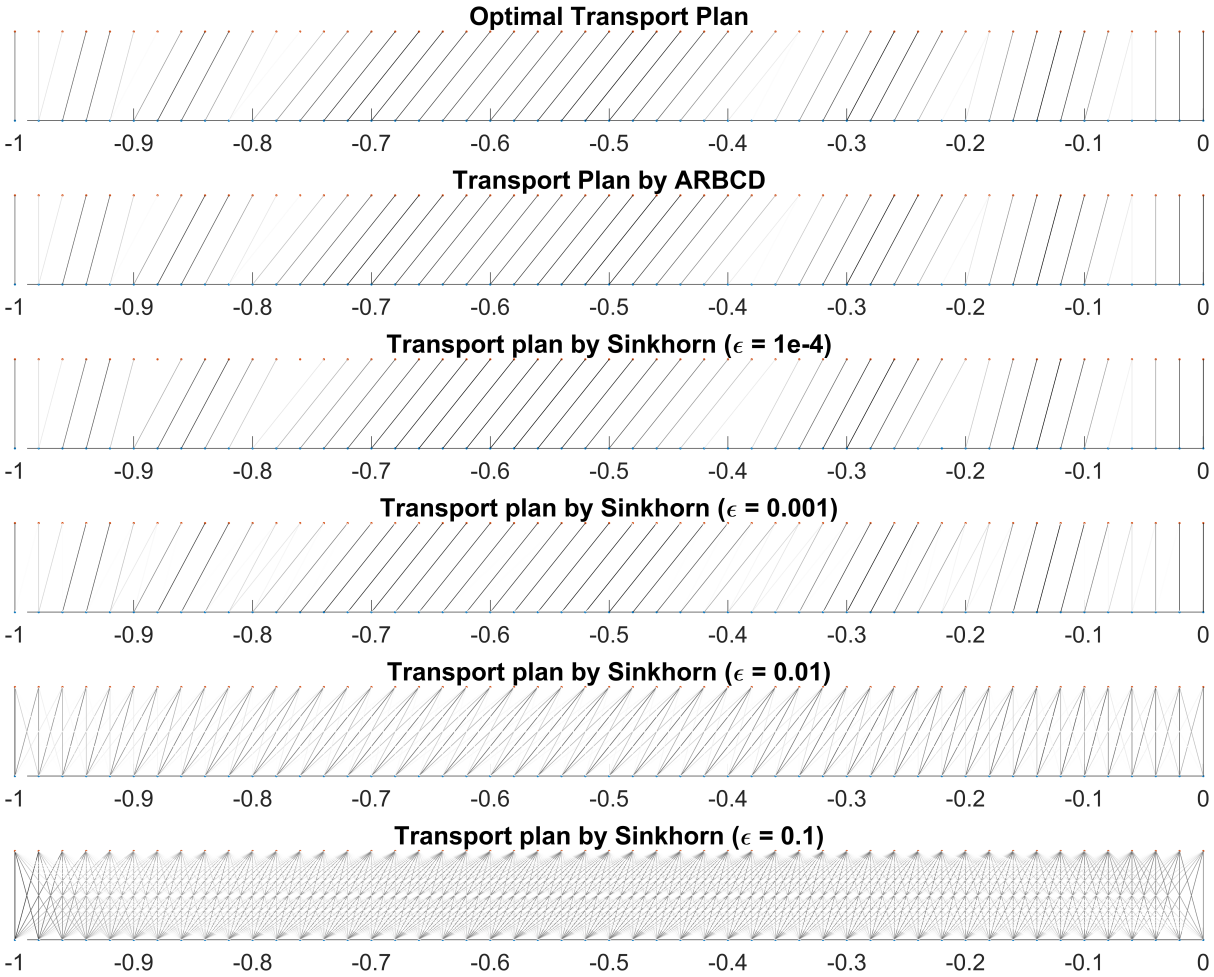

Figure 5: Transport plan given by different algorithms (1-d case, $n = 201$)
This figure shows the transport plan in the 1-d case. In each plot, the bottom distribution is uniform and the top is standard normal. Each line segment in between represents mass transported between a pair of points. The darker the line is, the more mass is transported. To plot the plans more clearly, we select every other mass point from -1 to 0 (so only include 51 mass points). The overall transport plans from -1 to 1 are symmetric plots so we only show half of them due to presentation clarity.

of China (Project 12171406), an R&D Funding Scheme from the HKU-SCF FinTech Academy, and Seed Funding for Strategic Interdisciplinary Research Scheme 2021/22 (HKU).

A Proof of Lemma 6

Proof. Suppose that

$$\log((n^2)!/(n^2-np)!) \geq 2\log(n!) + \log(np)! \tag{31}$$

Then

$$\begin{aligned}
& (n^2)!/(n^2-np)!)/(np)! && \geq (n!)^2 \\
\Longrightarrow \quad & \tbinom{n^2}{np}/(n!)^2 && \geq 1 \\
\Longrightarrow \quad & \tfrac{\binom{n^2}{np}}{(n!)^2} \cdot \tfrac{n(p-2)(n^2-np+1)}{n^2-3} && \geq 1 \\
\Longrightarrow \quad & \tfrac{n(p-2)}{(n^2-3)(n!)^2} && \geq \tfrac{1}{\binom{n^2}{np}(n^2-np+1)},
\end{aligned}$$

where the third inequality holds because $p \le n/2$ and $n \ge p\bar{K} \ge 6$. So we only need to prove (31). Note that

$$\begin{aligned}
\log \frac{(n^2)!}{(n^2-np)!} &= \sum_{x=n^2-np+1}^{n^2} \log(x) \ge \int_{n^2-np}^{n^2} (\log x)dx \\
&= n^2\log(n^2) - n^2 - \big((n^2-np)\log(n^2-np) - n^2 + np\big) \\
&= n^2\log(n^2) - (n^2-np)\log(n^2-np) - np \\
&\overset{(p=n/K)}{=} 2np\log n + \frac{K-1}{K}\cdot n^2 \cdot \log\frac{K}{K-1} - np \\
&\ge 2np\log n + \frac{2K-3}{2K-2}\cdot np - np. \qquad (32)
\end{aligned}$$

The last inequality holds because $\log(1+x) \ge x - x^2/2$ for $x \in (0,1)$ and $p = n/K$. Meanwhile, right hand side of (31) satisfies the following:

$$\begin{aligned}
&2\log(n!) + \log(np)! \\
&\le 2(n+1)\log(n+1) - 2n + (np+1)\log(np+1) - np \\
&\le 2(n+1)(\log n + \log 2) - 2n + (np+1)(\log(np) + \log 2) - np \\
&= (np+2n+3)\log n + (np+1)\log p + 2(n+1)\log 2 - 2n - np + (\log 2)(np+1) \\
&\overset{(p=\frac{n}{K})}{=} 2np\log n + (2n+4)\log n + (\log 4)n + (\log 2)np + \log 8 - 2n - (1+\log K)np - \log K \\
&\overset{(K\ge\bar{K}\ge 2, n\ge 6)}{\le} 2np\log n + (2n+4)\log n + (\log 2)np - (1+\log K)np \qquad (33)
\end{aligned}$$

In order to show (31), we only need to confirm (33) $\le$ (32). By observation, this is equivalent to

$$\begin{aligned}
&\left(\tfrac{2K-3}{2K-2} + \log\left(\tfrac{K}{2}\right)\right)np &&\ge (2n+4)\log n \\
\overset{(p\ge\eta\log n, \bar{K}\le K)}{\Longleftarrow}\quad &\left(\tfrac{2\bar{K}-3}{2\bar{K}-2} + \log\left(\tfrac{\bar{K}}{2}\right)\right)\eta n &&\ge 2n+4 \\
\Longleftrightarrow\quad &\frac{4}{\left(\frac{2\bar{K}-3}{2\bar{K}-2} + \log\left(\frac{\bar{K}}{2}\right)\right)\eta - 2} &&\le n.
\end{aligned}$$

The last inequality is assumed. ∎

B A counterexample of interest

Example 1. *Consider LP problem* (7) *with* $n = 3$, $r^1 = r^2 = (1/3, 1/3, 1/3)^T$. *Let*

$$C = \begin{pmatrix} (1+\epsilon_1)^2 & (2-\epsilon_3)^2 & (1-\epsilon_2)^2 \\ (1-\epsilon_2)^2 & (1+\epsilon_1)^2 & (2-\epsilon_3)^2 \\ (2-\epsilon_3)^2 & (1-\epsilon_2)^2 & (1+\epsilon_1)^2 \end{pmatrix},$$

where $0 < \epsilon_i \ll 1$, $i = 1,2,3$ *such that* $2(1+\epsilon_1)^2 < (1-\epsilon_2)^2 + (2-\epsilon_3)^2$. *It can be easily seen that the optimal solution is* $\gamma^* = \frac{1}{3}\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$. *Suppose that* $\gamma^0 = \frac{1}{3}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. *If we use the submatrix approach* (28) *with* $m = 2$ *(largest number less than* n*) to select a working set* $\mathcal{I}_k$, *then the algorithm will be stuck at* γ^0. *It is not globally convergent. This cost matrix corresponds to the following case of transporting a three-point distribution to another one:*

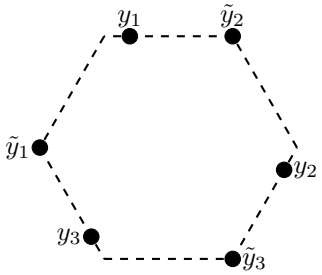

Figure 6: A counterexample of interest
The dashed hexagon has an edge length 1. Transport point distribution $\tilde{y}_1$, $\tilde{y}_2$ and $\tilde{y}_3$ to that of y_1, y_2 and y_3.

References

[1] J. Altschuler, J. Niles-Weed, and P. Rigollet, *Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration*, Advances in Neural Information Processing Systems, 30 (2017).

[2] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein generative adversarial networks*, in International Conference on Machine Learning, PMLR, 2017, pp. 214–223.

[3] A. Beck, *The 2-coordinate descent method for solving double-sided simplex constrained minimization problems*, Journal of Optimization Theory and Applications, 162 (2014), pp. 892–919.

[4] A. Beck and L. Tetruashvili, *On the convergence of block coordinate descent type methods*, SIAM Journal on Optimization, 23 (2013), pp. 2037–2060.

[5] J. Benamou and Y. Brenier, *A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem*, Numerische Mathematik, 84 (2000), pp. 375–393.

[6] A. S. Berahas, R. Bollapragada, and J. Nocedal, *An investigation of Newton-sketch and subsampled Newton methods*, Optimization Methods and Software, 35 (2020), pp. 661–680.

[7] M. Blondel, V. Seguy, and A. Rolet, *Smooth and sparse optimal transport*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2018, pp. 880–889.

[8] Y. Brenier, *Polar factorization and monotone rearrangement of vector-valued functions*, Communications on Pure and Applied Mathematics, 44 (1991), pp. 375–417.

[9] C. Chen, B. He, Y. Ye, and X. Yuan, *The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent*, Mathematical Programming, 155 (2016), pp. 57–79.

[10] M. Cuturi, *Sinkhorn distances: Lightspeed computation of optimal transport*, Advances in Neural Information Processing Systems, 26 (2013).

[11] P. Dvurechensky, A. Gasnikov, and A. Kroshnin, *Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm*, in International Conference on Machine Learning, PMLR, 2018, pp. 1367–1376.

[12] A. V. Gasnikov, E. Gasnikova, Y. E. Nesterov, and A. Chernov, *Efficient numerical methods for entropy-linear programming problems*, Computational Mathematics and Mathematical Physics, 56 (2016), pp. 514–524.

[13] S. Gerber and M. Maggioni, *Multiscale strategies for computing optimal transport*, Journal of Machine Learning Research, 18 (2017).

[14] S. Guminov, P. Dvurechensky, N. Tupitsa, and A. Gasnikov, *On a combination of alternating minimization and Nesterov's momentum*, in International Conference on Machine Learning, PMLR, 2021, pp. 3886–3898.

[15] M. Gurbuzbalaban, A. Ozdaglar, P. A. Parrilo, and N. Vanli, *When cyclic coordinate descent outperforms randomized coordinate descent*, Advances in Neural Information Processing Systems, 30 (2017).

[16] S. Haker, L. Zhu, A. Tannenbaum, and S. Angenent, *Optimal mass transport for registration and warping*, International Journal of Computer Vision, 60 (2004), pp. 225–240.

[17] B. He and X. Yuan, *On the $\mathcal{O}(1/n)$ convergence rate of the Douglas–Rachford alternating direction method*, SIAM Journal on Numerical Analysis, 50 (2012), pp. 700–709.

[18] A. Jambulapati, A. Sidford, and K. Tian, *A direct $\tilde{O}(1/\epsilon)$ iteration parallel algorithm for optimal transport*, Advances in Neural Information Processing Systems, 32 (2019).

[19] R. Jordan, D. Kinderlehrer, and F. Otto, *The variational formulation of the Fokker–Planck equation*, SIAM Journal on Mathematical Analysis, 29 (1998), pp. 1–17.

[20] N. Lei, K. Su, L. Cui, S.-T. Yau, and X. D. Gu, *A geometric view of optimal transportation and generative model*, Computer Aided Geometric Design, 68 (2019), pp. 1–21.

[21] W. Li, P. Yin, and S. Osher, *Computations of optimal transport distance with Fisher information regularization*, Journal of Scientific Computing, 75 (2018), pp. 1581–1595.

[22] T. Lin, N. Ho, and M. I. Jordan, *On the efficiency of entropic regularized algorithms for optimal transport*, Journal of Machine Learning Research, 23 (2022), pp. 1–42.

[23] H. Ling and K. Okada, *An efficient earth mover's distance algorithm for robust histogram comparison*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (2007), pp. 840–853.

[24] Z. Lu and L. Xiao, *On the complexity analysis of randomized block-coordinate descent methods*, Mathematical Programming, 152 (2015), pp. 615–642.

[25] I. Necoara and D. Clipici, *Parallel random coordinate descent method for composite minimization: Convergence analysis and error bounds*, SIAM Journal on Optimization, 26 (2016), pp. 197–226.

[26] I. Necoara, Y. Nesterov, and F. Glineur, *Random block coordinate descent methods for linearly constrained optimization over networks*, Journal of Optimization Theory and Applications, 173 (2017), pp. 227–254.

[27] I. Necoara and M. Takáč, *Randomized sketch descent methods for non-separable linearly constrained optimization*, IMA Journal of Numerical Analysis, 41 (2021), pp. 1056–1092.

[28] Y. Nesterov, *Efficiency of coordinate descent methods on huge-scale optimization problems*, SIAM Journal on Optimization, 22 (2012), pp. 341–362.

[29] F. Otto, *The geometry of dissipative evolution equations: the porous medium equation*, Taylor & Francis, (2001).

[30] S. Peleg, M. Werman, and H. Rom, *A unified approach to the change of resolution: Space and gray-level*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 11 (1989), pp. 739–742.

[31] M. Perrot, N. Courty, R. Flamary, and A. Habrard, *Mapping estimation for discrete optimal transport*, Advances in Neural Information Processing Systems, 29 (2016).

[32] G. Peyré and M. Cuturi, *Computational optimal transport*, Foundations and Trends in Machine Learning, 11 (2019), pp. 355–607.

[33] B. T. Polyak, *Introduction to optimization*, Optimization Software, Inc., Publications Division, New York, (1987).

[34] Z. Qu, P. Richtárik, M. Takác, and O. Fercoq, *SDNA: stochastic dual Newton ascent for empirical risk minimization*, in International Conference on Machine Learning, PMLR, 2016, pp. 1823–1832.

[35] P. Richtárik and M. Takáč, *Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function*, Mathematical Programming, 144 (2014), pp. 1–38.

[36] ——, *Parallel coordinate descent methods for big data optimization*, Mathematical Programming, 156 (2016), pp. 433–484.

[37] R. T. Rockafellar, *Network flows and monotropic optimization*, vol. 9, Athena scientific, 1999.

[38] B. Schmitzer, *Stabilized sparse scaling algorithms for entropy regularized transport problems*, SIAM Journal on Scientific Computing, 41 (2019), pp. A1443–A1481.

[39] R. Sinkhorn, *A relationship between arbitrary positive matrices and doubly stochastic matrices*, The annals of mathematical statistics, 35 (1964), pp. 876–879.

[40] R. Sun and Y. Ye, *Worst-case complexity of cyclic coordinate descent: $\mathcal{O}(n^2)$ gap with randomized version*, Mathematical Programming, 185 (2021), pp. 487–520.

[41] A. Toselli and O. Widlund, *Domain decomposition methods-algorithms and theory*, vol. 34, Springer Science & Business Media, 2004.

[42] P. Tseng and S. Yun, *Block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization*, Journal of Optimization Theory and Applications, 140 (2009), pp. 513–535.

[43] ——, *A coordinate gradient descent method for nonsmooth separable minimization*, Mathematical Programming, 117 (2009), pp. 387–423.

[44] ——, *A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training*, Computational Optimization and Applications, 47 (2010), pp. 179–206.

[45] C. Villani, *Topics in optimal transportation*, vol. 58, American Math. Soc., 2021.

[46] Z. Wang, J. Xin, and Z. Zhang, *DeepParticle: learning invariant measure by a deep neural network minimizing Wasserstein distance on data generated from an interacting particle method*, Journal of Computational Physics, (2022), p. 111309.

[47] S. Wright, *Primal-dual interior-point methods*, SIAM, 1997.

[48] Y. Xie and U. V. Shanbhag, *SI-ADMM: A stochastic inexact ADMM framework for stochastic convex programs*, IEEE Transactions on Automatic Control, 65 (2019), pp. 2355–2370.

[49] ——, *Tractable ADMM schemes for computing KKT points and local minimizers for ℓ_0-minimization problems*, Computational Optimization and Applications, 78 (2021), pp. 43–85.