A CONNECTION BETWEEN OT AND RWOC

Theorem 1. Denote $\Pi(a,b) = \{S \in \mathbb{R}^{n\times m} : S\mathbf{1}_m = a, S^\top \mathbf{1}_n = b, S_{ij} \geq 0\}$ for any $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. Then at least one of the optimal solutions of the following problem lies in $\mathcal{P}$.

$$\min_{S\in\mathbb{R}^{n\times n}} \langle C(w), S\rangle, \quad \text{s.t. } S \in \Pi(\mathbf{1}_n, \mathbf{1}_n). \tag{14}$$

Proof. Denote the optimal solution of (14) as Z^*. As we mentioned earlier, this is a direct corollary of Birkhoff–von Neumann theorem (Birkhoff, 1946; Von Neumann, 1953). Specifically, Birkhoff–von Neumann theorem claims that the polytope $\Pi(\mathbf{1}_n, \mathbf{1}_n)$ is the convex hull of the set of $n \times n$ permutation matrices, and furthermore that the vertices of $\Pi(\mathbf{1}_n, \mathbf{1}_n)$ are precisely the permutation matrices.

On the other hand, (14) is a linear optimization problem. There would be at least one optimal solutions lies at the vertices given the problem is feasible. As a result, there would be at least one Z^* being a permutation matrix. □

B TWO PERSPECTIVES OF THE MOTIVATIONS OF BILEVEL OPTIMIZATION

B.1 FASTER CONVERGENCE

The bilevel optimization formulation has a better gradient descent iteration complexity than alternating minimization. To see this, consider a quadratic function $F(a_1, a_2) = a^\top P a + b^\top a$, where $a_1 \in \mathbb{R}^{d_1}$, $a_2 \in \mathbb{R}^{d_2}$, $a = [a_1^\top, a_2^\top]^\top \in \mathbb{R}^{(d_1+d_2)}$, $P \in \mathbb{R}^{(d_1+d_2)\times(d_1+d_2)}$, $b \in \mathbb{R}^{(d_1+d_2)}$. To further simplify the discussion, we assume $P = \rho\mathbf{1}_{(d_1+d_2)}\mathbf{1}_{(d_1+d_2)}^\top + (1-\rho)I_{d_1+d_2}$, where $I_{d_1+d_2}$ is the identity matrix. Then we have the following proposition.

Proposition 1. Given F defined in (9), we have

$$\frac{\lambda_{\max}(\nabla^2 F(a_1))}{\lambda_{\min}(\nabla^2 F(a_1))} = 1 + \frac{1-\rho+\lambda}{1-\rho}\frac{d_1\rho}{d_2\rho-\rho+\lambda+1} \quad \text{and} \quad \frac{\lambda_{\max}(\nabla^2_{a_1a_1} L(a_1,a_2))}{\lambda_{\min}(\nabla^2_{a_1a_1} L(a_1,a_2))} = 1 + \frac{d_1\rho}{1-\rho}.$$

Proof. For alternating minimization, the Hessian for a_1 is a submatrix of P, i.e.,

$$H_{\text{AM}} = \rho\mathbf{1}_{d_1}\mathbf{1}_{d_1}^\top + (1-\rho)I_{d_1},$$

whose condition number is

$$C_{\text{AM}} = 1 + \frac{d_1\rho}{1-\rho}.$$

We now compute the condition number for ROBOT. Denote

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

where $P_{11} \in \mathbb{R}^{d_1\times d_1}$, $P_{12} \in \mathbb{R}^{d_1\times d_2}$, $P_{21} \in \mathbb{R}^{d_2\times d_1}$, $P_{22} \in \mathbb{R}^{d_2\times d_2}$, and $b_1 \in \mathbb{R}^{d_1}$, $b_2 \in \mathbb{R}^{d_2}$. ROBOT first minimize over a_2,

$$a_2^*(a_1) = \arg\min_{a_2} F(a_1, a_2) = -(P_{22} + \lambda I_{d_2})^{-1}(P_{21}a_1 + b_2/2).$$

Substituting $a_2^*(a_1)$ into $F(a_1, a_2)$, we can obtain the Hessian for a_1 is

$$H_{\text{ROBOT}} = P_{11} - P_{12}(P_{22} + \lambda I_{d_2})^{-1}P_{21}.$$

Using Sherman–Morrison formula, we can explicitly express P_{22}^{-1} as

$$P_{22}^{-1} = \frac{1}{1-\rho+\lambda} I_{d_2} - \frac{\rho}{(1-\rho+\lambda)(1-\rho+\lambda+\rho d_2)}\mathbf{1}_{d_2}\mathbf{1}_{d_2}^\top.$$

Substituting it into H_{ROBOT},

$$H_{\text{ROBOT}} = P_{11} - P_{12}P_{22}^{-1}P_{21} = (1-\rho)I_{d_1} + \left(\rho - \frac{d_2\rho^2}{d_2\rho-\rho+\lambda+1}\right)\mathbf{1}_{d_1}\mathbf{1}_{d_1}^\top.$$

Therefore, the condition number is

$$C_{\text{ROBOT}} = 1 + \frac{1-\rho+\lambda}{1-\rho}\frac{d_1\rho}{d_2\rho-\rho+\lambda+1}.$$

□

Note that C_{AM} increases linearly with respect to d_1. Therefore, the optimization problem inevitably becomes ill-conditioned as dimension increase. In contrast, C_{ROBOT} can stay in the same order of magnitude when d_1 and d_2 increase simultaneously.

Since the iteration complexity of gradient descent is proportional to the condition number (Bottou et al., 2018), ROBOT needs fewer iterations to converge than AM.

C DIFFERENTIABILITY

Theorem 2. For any $\epsilon > 0$, $S^*_\epsilon(w)$ is differentiable, as long as the cost $C(w)$ is differentiable with respect to w. As a result, the objective $\mathcal{L}_\epsilon(w) = \langle C(w), S^*_\epsilon(w)\rangle$ is also differentiable.

Proof. The proof is analogous to Xie et al. (2020).

We first prove the differentiability of $S^*_\epsilon(w)$. This part of proof mirrors the proof in Luise et al. (2018). By Sinkhorn's scaling theorem (Sinkhorn & Knopp, 1967),

$$S^*_\epsilon(w) = \text{diag}(e^{\frac{\xi^*(w)}{\epsilon}})e^{-\frac{C(w)}{\epsilon}}\text{diag}(e^{\frac{\zeta^*(w)}{\epsilon}}).$$

Therefore, since $C_{ij}(w)$ is differentiable, $\Gamma^{*,\epsilon}$ is differentiable if $(\xi^*(w), \zeta^*(w))$ is differentiable as a function of w.

Let us set

$$\mathcal{L}(\xi, \zeta; \mu, \nu, C) = \xi^T\mu + \zeta^T\nu - \epsilon\sum_{i,j=1}^{n,m} e^{-\frac{C_{ij}-\xi_i-\zeta_j}{\epsilon}}.$$

and recall that $(\xi^*, \zeta^*) = \arg\max_{\xi,\zeta} L(\xi, \zeta; \mu, \nu, C)$. The differentiability of (ξ^*, ζ^*) is proved using the Implicit Function theorem and follows from the differentiability and strict convexity in (ξ^*, ζ^*) of the function $\mathcal{L}$. □

Theorem 3. Denoting $\mathcal{L}_\epsilon = \langle C(w), S^*_\epsilon(w)\rangle$. The gradient of $\mathcal{L}_\epsilon$ with respect to w is

$$\nabla_w\mathcal{L}_\epsilon = \frac{1}{\epsilon}\sum_{i,j=1}^{n,n}\left((1-C_{ij})S^*_{\epsilon,ij} + \sum_{h,\ell=1}^{n,n} C_{h\ell}S^*_{\epsilon,h\ell}\frac{d\xi^*_h}{dC_{ij}} + \sum_{h,\ell=1}^{n,n} C_{h\ell}S^*_{\epsilon,h\ell}\frac{d\zeta^*_\ell}{dC_{ij}}\right)\nabla_w C_{ij}, \quad (15)$$

where $\begin{bmatrix}\nabla_C\xi^* \\ \nabla_C\zeta^*\end{bmatrix} = \begin{bmatrix}-H^{-1}D \\ \mathbf{0}\end{bmatrix}$ with $-H^{-1}D \in \mathbb{R}^{(2n-1)\times n\times n}, \mathbf{0} \in \mathbb{R}^{1\times n\times n}$,

$$D_{\ell ij} = \frac{1}{\epsilon}\begin{cases}\delta_{\ell i}S^*_{\epsilon,ij}, & \ell = 1,\cdots,n;\\ \delta_{\ell j}S^*_{\epsilon,ij}, & \ell = n+1,\cdots,2n-1,\end{cases}$$

$$H^{-1} = -\epsilon\begin{bmatrix}(\text{diag}(\mu))^{-1} + (\text{diag}(\mu))^{-1}\bar{S}^*_\epsilon\mathcal{K}^{-1}\bar{S}^{*T}_\epsilon(\text{diag}(\mu))^{-1} & -(\text{diag}(\mu))^{-1}\bar{S}^*_\epsilon\mathcal{K}^{-1} \\ -\mathcal{K}^{-1}\bar{S}^{*T}_\epsilon(\text{diag}(\mu))^{-1} & \mathcal{K}^{-1}\end{bmatrix},$$

and $\mathcal{K} = \text{diag}(\bar{\nu}) - \bar{S}^{*T}_\epsilon(\text{diag}(\mu))^{-1}\bar{S}^*_\epsilon, \quad \bar{\nu} = \nu_{1:n-1}, \quad \bar{S}^*_\epsilon = S^*_{\epsilon,1:n,1:n-1}.$

Proof. This result is straightforward combining the Sinkhorn's scaling theorem and Theorem 3 in Xie et al. (2020). □

D ALGORITHM OF THE FORWARD PASS FOR ROBOT-ROBUST

For better numerical stability, in practice we add two more regularization terms,

$$S^*_{\text{r}}(w), \bar{\mu}^*, \bar{\nu}^* = \arg\min_{S\in\Pi(\bar{\mu},\bar{\nu}),\ \bar{\mu},\bar{\nu}\in\Delta_n}\langle C(w), S\rangle + \epsilon H(S) + \epsilon_1 h(\bar{\mu}) + \epsilon_2 h(\bar{\nu}), \quad (16)$$
$$\text{s.t. } \mathcal{F}(\bar{\mu}, \mu) \le \rho_1,\ \mathcal{F}(\bar{\nu}, \nu) \le \rho_2,$$

where $h(\bar{\mu}) = \sum_i \bar{\mu}_i \log\bar{\mu}_i$ is the entropy function for vectors. This can avoid the entries of $\bar{\mu}$ and $\bar{\nu}$ shrink to zeros when updated by gradient descent. We remark that since we have entropy term $H(S)$, the entries of S would not be exactly zeros. Furthermore, we have $\bar{\mu} = S\mathbf{1}$ and $\bar{\mu} = S\mathbf{1}$. Therefore, theoretically the entries of $\bar{\mu}$ and $\bar{\nu}$ will not be zeros. We only add the two more entropy terms for numerical consideration. The detailed algorithm is in Algorithm 1. Although the algorithm is not guaranteed to converge to a feasible solution, in practice it usually converges to a good solution (Wang et al., 2015).

Algorithm 1 Solving S_{r}^* for robust matching

Require: $C \in \mathbb{R}^{m\times n}, \mu, \nu, K, \epsilon, L, \eta$
$G_{ij} = e^{-\frac{C_{ij}}{\epsilon}}$
$\bar{\mu} = \mu, \bar{\nu} = \nu$
$b = \mathbf{1}_n$
for $l = 1, \cdots, L$ **do**
$a = \bar{\mu}/(Gb), b = \bar{\nu}/(G^T a)$
$\bar{\mu} = \bar{\mu} - \eta(e^{\frac{a}{\epsilon}} + \epsilon_1 * \log \bar{\mu}), \bar{\nu} = \bar{\nu} - \eta(e^{\frac{b}{\epsilon}} + \epsilon_2 * \log \bar{\nu})$
$\bar{\mu} = \max\{\bar{\mu}, 0\}, \bar{\nu} = \max\{\bar{\nu}, 0\}$
$\bar{\mu} = \bar{\mu}/(\bar{\mu}^\top \mathbf{1}), \bar{\nu} = \bar{\nu}/(\bar{\nu}^\top \mathbf{1})$
if $\|\bar{\mu} - \mu\|_2^2 > \rho_1$ **then**
$\bar{\mu} = \mu + \sqrt{\rho_1}\frac{\bar{\mu}-\mu}{\|\bar{\mu}-\mu\|_2}$
end if
if $\|\bar{\nu} - \nu\|_2^2 > \rho_2$ **then**
$\bar{\nu} = \nu + \sqrt{\rho_2}\frac{\bar{\nu}-\nu}{\|\bar{\nu}-\nu\|_2}$
end if
end for
$S = \mathrm{diag}(a) \odot G \odot \mathrm{diag}(b)$

E Algorithm of the Backward Pass for ROBOT-robust

Since the derivation is tedious, we first summarize the outline of the derivation, then provide the detailed derivation.

E.1 Summary

Given $\bar{\mu}^*, \bar{\nu}^*, S_{\mathrm{r}}^*(w)$, we compute the Jacobian matrix $dS_{\mathrm{r}}^*(w)/dw$ using implicit differentiation and differentiable programming techinques. Specifically, the Lagrangian function of Problem (16) is

$$\begin{aligned}\mathcal{L} =& \langle C, S\rangle + \epsilon H(S) + \epsilon_1 h(\bar{\mu}) + \epsilon_2 h(\bar{\nu}) - \xi^\top(\Gamma \mathbf{1}_m - \mu) - \zeta^\top(\Gamma^\top \mathbf{1}_n - \nu) \\ &+ \lambda_1(\bar{\mu}^\top \mathbf{1}_n - 1) + \lambda_2(\bar{\nu}^\top \mathbf{1}_m - 1) + \lambda_3(\|\bar{\mu} - \mu\|_2^2 - \rho_1) + \lambda_4(\|\bar{\nu} - \nu\|_2^2 - \rho_2).\end{aligned}$$

where ξ and ζ are dual variables. The KKT conditions (Stationarity condition) imply that the optimal solution $\Gamma^{*,\epsilon}$ can be formulated using the optimal dual variables ξ^* and ζ^* as,

$$S_{\mathrm{r}}^* = \mathrm{diag}(e^{\frac{\xi^*}{\epsilon}}) e^{-\frac{C}{\epsilon}} \mathrm{diag}(e^{\frac{\zeta^*}{\epsilon}}). \tag{17}$$

By the chain rule, we have

$$\frac{dS_{\mathrm{r}}^*}{dw} = \frac{dS_{\mathrm{r}}^*}{dC}\frac{dC}{dw} = \left(\frac{\partial S_{\mathrm{r}}^*}{\partial C} + \frac{\partial S_{\mathrm{r}}^*}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial S_{\mathrm{r}}^*}{\partial \zeta^*}\frac{d\zeta^*}{dC}\right)\frac{dC}{dw}.$$

Therefore, we can compute $dS_{\mathrm{r}}^*(w)/dw$ if we obtain $\frac{d\xi^*}{dC}$ and $\frac{d\zeta^*}{dC}$.

Substituting (17) into the Lagrangian function, at the optimal solutions we obtain

$$\mathcal{L} = \mathcal{L}(\xi^*, \zeta^*, \bar{\mu}^*, \bar{\nu}^*, \lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*; C).$$

Denote $r^* = [(\xi^*)^\top, (\zeta^*)^\top, (\bar{\mu})^\top, (\bar{\nu})^\top, \lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*]^\top$, and $\phi(r^*; C) = \partial\mathcal{L}(r^*; C)/\partial r^*$. At the optimal dual variable r^*, the KKT condition immediately yields $\phi(r^*; C) \equiv 0$. By the chain rule, we have

$$\frac{d\phi(r^*; C)}{dC} = \frac{\partial\phi(r^*; C)}{\partial C} + \frac{\partial\phi(r^*; C)}{\partial r^*}\frac{dr^*}{dC} = 0. \tag{18}$$

Rerranging terms, we obtain

$$\frac{dr^*}{dC} = -\left(\frac{\partial\phi(r^*; C)}{\partial r^*}\right)^{-1}\frac{\partial\phi(r^*; C)}{\partial C}. \tag{19}$$

Combining (17), (18), and (19), we can then obtain $dS_{\mathrm{r}}^*(w)/dw$.

E.2 DETAILS

Now we provide the detailed derivation for computing dS_{r}^*/dw.

Since S_{r}^* is the optimal solution of an optimization problem, we can follow the implicit function theorem to solve for the closed-form expression of the gradient. Specifically, we adopt $\mathcal{F}(\bar{\mu}, \nu) = \sum_i (\bar{\mu}_i - \mu_i)^2$, and rewrite the optimization problem as

$$\begin{aligned}
&\min_{\bar{\mu},\bar{\nu},S} \langle C, S\rangle + \epsilon \sum_{ij} S_{ij}(\log S_{ij} - 1) + \epsilon_1 \sum_i \bar{\mu}_i(\log \bar{\mu}_i - 1) + \epsilon_2 \sum_j \bar{\nu}_j(\log \bar{\nu}_j - 1),\\
\text{s.t.,}\ &\sum_j S_{ij} = \bar{\mu}_i, \quad \sum_i S_{ij} = \bar{\nu}_j,\\
&\sum_i \bar{\mu}_i = 1, \quad \sum_j \bar{\nu}_j = 1,\\
&\sum_i (\bar{\mu}_i - \mu_i)^2 \le \rho_1, \quad \sum_j (\bar{\nu}_j - \nu_j)^2 \le \rho_2.
\end{aligned}$$

The Language of the above problem is

$$\begin{aligned}
&\mathcal{L}(C, S, \bar{\mu}, \bar{\nu}, \xi, \zeta, \lambda_1, \lambda_2, \lambda_3, \lambda_4)\\
&= \langle C, S\rangle + \epsilon \sum_{ij} S_{ij}(\log S_{ij} - 1) + \epsilon_1 \sum_i \bar{\mu}_i(\log \bar{\mu}_i - 1) + \epsilon_2 \sum_j \bar{\nu}_j(\log \bar{\nu}_j - 1)\\
&- \xi^\top (S\mathbf{1}_m - \bar{\mu}) - \zeta^\top (S^\top \mathbf{1}_n - \bar{\nu})\\
&+ \lambda_1(\sum_i \bar{\mu}_i - 1) + \lambda_2(\sum_j \bar{\nu}_j - 1) + \lambda_3(\sum_i (\bar{\mu}_i - \mu_i)^2 - \rho_1) + \lambda_4(\sum_j (\bar{\nu}_j - \nu_j)^2 - \rho_2).
\end{aligned}$$

Easy to see that the Slater's condition holds. Denote

$$\mathcal{L}^* = \mathcal{L}(C, S_{\mathrm{r}}^*, \bar{\mu}^*, \bar{\nu}^*, \xi^*, \zeta^*, \lambda_1^*, \lambda_2^*, \lambda_3^*, \lambda_4^*).$$

Following the KKT conditions,

$$\frac{d\mathcal{L}^*}{dS_{\mathrm{r},ij}^*} = C_{ij} + \epsilon \log S_{\mathrm{r},ij}^* - \xi_i^* - \zeta_j^* = 0.$$

Therefore, $S_{\mathrm{r},ij}^* = e^{\frac{\xi_i^* + \zeta_j^* - C_{ij}}{\epsilon}}$. Then we have

$$\frac{dS_{\mathrm{r}}^*}{dw} = (\frac{\partial S_{\mathrm{r}}^*}{\partial C} + \frac{\partial S_{\mathrm{r}}^*}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial S_{\mathrm{r}}^*}{\partial \zeta^*}\frac{d\zeta^*}{dC})\frac{dC}{dw}.$$

So all we need to do is to compute $\frac{d\xi^*}{dC}$ and $\frac{d\zeta^*}{dC}$. Denote $F_{ij} = e^{\frac{\xi_i + \zeta_j - C_{ij}}{\epsilon}}$. Denote

$$\begin{aligned}
\phi &= \frac{d\mathcal{L}}{d\xi} = \bar{\mu} - F\mathbf{1}_m,\\
\psi &= \frac{d\mathcal{L}}{d\zeta} = \bar{\nu} - F^\top \mathbf{1}_n,\\
p &= \frac{d\mathcal{L}}{d\bar{\mu}} = \xi + \lambda_1 \mathbf{1}_n + 2\lambda_3(\bar{\mu} - \mu) + \epsilon_1 \log \bar{\mu},\\
q &= \frac{d\mathcal{L}}{d\bar{\nu}} = \zeta + \lambda_2 \mathbf{1}_m + 2\lambda_4(\bar{\nu} - \nu) + \epsilon_2 \log \bar{\nu},\\
\chi_1 &= \frac{d\mathcal{L}}{d\lambda_1} = \bar{\mu}^\top \mathbf{1}_n - 1,\\
\chi_2 &= \frac{d\mathcal{L}}{d\lambda_2} = \bar{\nu}^\top \mathbf{1}_m - 1,\\
\chi_3 &= \lambda_3(\|\bar{\mu} - \mu\|_2^2 - \rho_1),\\
\chi_4 &= \lambda_4(\|\bar{\nu} - \nu\|_2^2 - \rho_2).
\end{aligned}$$

Denote $\chi = [\chi_1, \chi_2, \chi_3, \chi_4]$, and $\lambda = [\lambda_1, \lambda_2, \lambda_3, \lambda_4]$. Following the KKT conditions, we have

$$\phi = 0, \psi = 0, p = 0, q = 0, \chi = 0,$$

at the optimal solutions. Therefore, for the optimal solutions we have

$$\begin{aligned}
\frac{d\phi}{dC} &= \frac{\partial \phi}{\partial C} + \frac{\partial \phi}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial \phi}{\partial \zeta^*}\frac{d\zeta^*}{dC} + \frac{\partial \phi}{\partial \bar{\mu}^*}\frac{d\bar{\mu}^*}{dC} + \frac{\partial \phi}{\partial \bar{\nu}^*}\frac{d\bar{\nu}^*}{dC} + \frac{\partial \phi}{\partial \lambda^*}\frac{d\lambda^*}{dC} = 0,\\
\frac{d\psi}{dC} &= \frac{\partial \psi}{\partial C} + \frac{\partial \psi}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial \psi}{\partial \zeta^*}\frac{d\zeta^*}{dC} + \frac{\partial \psi}{\partial \bar{\mu}^*}\frac{d\bar{\mu}^*}{dC} + \frac{\partial \psi}{\partial \bar{\nu}^*}\frac{d\bar{\nu}^*}{dC} + \frac{\partial \psi}{\partial \lambda^*}\frac{d\lambda^*}{dC} = 0,\\
\frac{dp}{dC} &= \frac{\partial p}{\partial C} + \frac{\partial p}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial p}{\partial \zeta^*}\frac{d\zeta^*}{dC} + \frac{\partial p}{\partial \bar{\mu}^*}\frac{d\bar{\mu}^*}{dC} + \frac{\partial p}{\partial \bar{\nu}^*}\frac{d\bar{\nu}^*}{dC} + \frac{\partial p}{\partial \lambda^*}\frac{d\lambda^*}{dC} = 0,\\
\frac{dq}{dC} &= \frac{\partial q}{\partial C} + \frac{\partial q}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial q}{\partial \zeta^*}\frac{d\zeta^*}{dC} + \frac{\partial q}{\partial \bar{\mu}^*}\frac{d\bar{\mu}^*}{dC} + \frac{\partial q}{\partial \bar{\nu}^*}\frac{d\bar{\nu}^*}{dC} + \frac{\partial q}{\partial \lambda^*}\frac{d\lambda^*}{dC} = 0\\
\frac{d\chi}{dC} &= \frac{\partial \chi}{\partial C} + \frac{\partial \chi}{\partial \xi^*}\frac{d\xi^*}{dC} + \frac{\partial \chi}{\partial \zeta^*}\frac{d\zeta^*}{dC} + \frac{\partial \chi}{\partial \bar{\mu}^*}\frac{d\bar{\mu}^*}{dC} + \frac{\partial \chi}{\partial \bar{\nu}^*}\frac{d\bar{\nu}^*}{dC} + \frac{\partial \chi}{\partial \lambda^*}\frac{d\lambda^*}{dC} = 0.
\end{aligned}$$

Therefore, we have

$$\begin{bmatrix}
\frac{d\xi^*}{dC}\\ \frac{d\zeta^*}{dC}\\ \frac{d\bar{\mu}^*}{dC}\\ \frac{d\bar{\nu}^*}{dC}\\ \frac{d\lambda^*}{dC}
\end{bmatrix} = -\begin{bmatrix}
\frac{\partial \phi}{\partial \xi^*} & \frac{\partial \phi}{\partial \zeta^*} & \frac{\partial \phi}{\partial \bar{\mu}^*} & \frac{\partial \phi}{\partial \bar{\nu}^*} & \frac{\partial \phi}{\partial \lambda^*}\\
\frac{\partial \psi}{\partial \xi^*} & \frac{\partial \psi}{\partial \zeta^*} & \frac{\partial \psi}{\partial \bar{\mu}^*} & \frac{\partial \psi}{\partial \bar{\nu}^*} & \frac{\partial \psi}{\partial \lambda^*}\\
\frac{\partial p}{\partial \xi^*} & \frac{\partial p}{\partial \zeta^*} & \frac{\partial p}{\partial \bar{\mu}^*} & \frac{\partial p}{\partial \bar{\nu}^*} & \frac{\partial p}{\partial \lambda^*}\\
\frac{\partial q}{\partial \xi^*} & \frac{\partial q}{\partial \zeta^*} & \frac{\partial q}{\partial \bar{\mu}^*} & \frac{\partial q}{\partial \bar{\nu}^*} & \frac{\partial q}{\partial \lambda^*}\\
\frac{\partial \chi}{\partial \xi^*} & \frac{\partial \chi}{\partial \zeta^*} & \frac{\partial \chi}{\partial \bar{\mu}^*} & \frac{\partial \chi}{\partial \bar{\nu}^*} & \frac{\partial \chi}{\partial \lambda^*}
\end{bmatrix}^{-1}\begin{bmatrix}
\frac{\partial \phi}{\partial C}\\ \frac{\partial \psi}{\partial C}\\ \frac{\partial p}{\partial C}\\ \frac{\partial q}{\partial C}\\ \frac{\partial \chi}{\partial C}
\end{bmatrix}.$$

After some derivation, we have

$$\begin{bmatrix}
\frac{d\xi^*}{dC}\\ \frac{d\zeta^*}{dC}\\ \frac{d\bar{\mu}^*}{dC}\\ \frac{d\bar{\nu}^*}{dC}\\ \frac{d\lambda_1^*}{dC}\\ \frac{d\lambda_2^*}{dC}\\ \frac{d\lambda_3^*}{dC}\\ \frac{d\lambda_4^*}{dC}
\end{bmatrix} = -\begin{bmatrix}
-\frac{1}{\epsilon}\mathrm{diag}(\bar{\mu}) & -\frac{1}{\epsilon}S_{\mathrm{r}}^* & \boldsymbol{I}_n & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
-\frac{1}{\epsilon}(S_{\mathrm{r}}^*)^\top & -\frac{1}{\epsilon}\mathrm{diag}(\bar{\nu}) & \mathbf{0} & \boldsymbol{I}_m & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\boldsymbol{I}_n & \mathbf{0} & 2\lambda_3\boldsymbol{I}_n + \mathrm{diag}(\frac{\epsilon_1}{\bar{\mu}}) & \mathbf{0} & \mathbf{1}_n & \mathbf{0} & 2(\bar{\mu}-\mu) & \mathbf{0}\\
\mathbf{0} & \boldsymbol{I}_m & \mathbf{0} & 2\lambda_4\boldsymbol{I}_m + \mathrm{diag}(\frac{\epsilon_2}{\bar{\nu}}) & \mathbf{0} & \mathbf{1}_m & \mathbf{0} & 2(\bar{\nu}-\nu)\\
\mathbf{0} & \mathbf{0} & \mathbf{1}_n^\top & \mathbf{0} & 0 & 0 & 0 & 0\\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}_m^\top & 0 & 0 & 0 & 0\\
\mathbf{0} & \mathbf{0} & 2\lambda_3(\bar{\mu}-\mu)^\top & \mathbf{0} & 0 & 0 & \|\bar{\mu}-\mu\|_2^2-\rho_1 & 0\\
\mathbf{0} & \mathbf{0} & \mathbf{0} & 2\lambda_4(\bar{\nu}-\nu)^\top & 0 & 0 & 0 & \|\bar{\nu}-\nu\|_2^2-\rho_2
\end{bmatrix}^{-1}\begin{bmatrix}
\frac{\partial \phi}{\partial C}\\ \frac{\partial \psi}{\partial C}\\ \mathbf{0}\\ \mathbf{0}\\ 0\\ 0\\ 0\\ 0
\end{bmatrix},$$

and

$$\begin{aligned}
\frac{\partial \phi_h}{\partial C_{ij}} &= \frac{1}{\epsilon}\delta_{hi}S_{ij}, \forall h = 1,\cdots,n, \quad i = 1,\cdots,n, \quad j = 1,\cdots,m\\
\frac{\partial \psi_\ell}{\partial C_{ij}} &= \frac{1}{\epsilon}\delta_{\ell j}S_{ij}, \forall \ell = 1,\cdots,m-1, \quad i = 1,\cdots,n, \quad j = 1,\cdots,m.
\end{aligned}$$

To efficiently solve for the inverse in the above equations, we denote

$$A = \begin{bmatrix}
-\frac{1}{\epsilon}\mathrm{diag}(\bar{\mu}) & -\frac{1}{\epsilon}S_{\mathrm{r}}^* & \boldsymbol{I}_n & \mathbf{0}\\
-\frac{1}{\epsilon}(S_{\mathrm{r}}^*)^\top & -\frac{1}{\epsilon}\mathrm{diag}(\bar{\nu}) & \mathbf{0} & \boldsymbol{I}_m\\
\boldsymbol{I}_n & \mathbf{0} & 2\lambda_3\boldsymbol{I}_n + \mathrm{diag}(\frac{\epsilon_1}{\bar{\mu}}) & \mathbf{0}\\
\mathbf{0} & \boldsymbol{I}_m & \mathbf{0} & 2\lambda_4\boldsymbol{I}_m + \mathrm{diag}(\frac{\epsilon_2}{\bar{\nu}})
\end{bmatrix},$$

$$B_1 = \begin{bmatrix}
\mathbf{1}_n & \mathbf{0} & 2(\bar{\mu}-\mu) & \mathbf{0}\\
\mathbf{0} & \mathbf{1}_m & \mathbf{0} & 2(\bar{\nu}-\nu)
\end{bmatrix},$$

$$C_1 = \begin{bmatrix}
\mathbf{1}_n^\top & \mathbf{0}\\
\mathbf{0} & \mathbf{1}_m^\top\\
2\lambda_3(\bar{\mu}-\mu)^\top & \mathbf{0}\\
\mathbf{0} & 2\lambda_4(\bar{\nu}-\nu)^\top
\end{bmatrix},$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \|\bar{\mu} - \mu\|_2^2 - \rho_1 & 0 \\ 0 & 0 & 0 & \|\bar{\nu} - \nu\|_2^2 - \rho_2 \end{bmatrix}.$$

We first A^{-1} using the rules for inverting a block matrix,

$$A^{-1} = \begin{bmatrix} K & -KL \\ -LK & L + LKL \end{bmatrix} =: \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$$

where

$$L = \begin{bmatrix} 2\lambda_3 \boldsymbol{I}_n + \mathrm{diag}(\frac{\epsilon_1}{\bar{\mu}}) & \mathbf{0} \\ \mathbf{0} & 2\lambda_4 \boldsymbol{I}_m + \mathrm{diag}(\frac{\epsilon_1}{\bar{\nu}}) \end{bmatrix}^{-1}, \quad K = \left(\frac{1}{\epsilon} \begin{bmatrix} \mathrm{diag}(\bar{\mu}) & S_{\mathrm{r}}^* \\ (S_{\mathrm{r}}^*)^\top & \mathrm{diag}(\bar{\nu}) \end{bmatrix} + L \right)^{-1}.$$

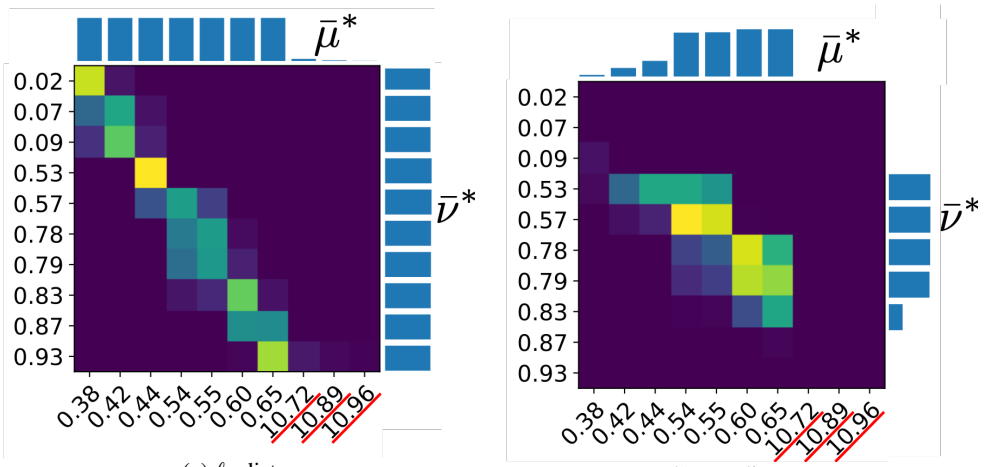
Then using the rules of inverting a block matrix again, we have

$$\begin{bmatrix} \frac{d\xi^*}{dC} \\ \frac{d\zeta^*}{dC} \end{bmatrix} = (A_1 + A_2 B_1 (D - C_1 A_4 B_1)^{-1} C_1 A_3) \begin{bmatrix} \frac{\partial \phi}{\partial C} \\ \frac{\partial \psi}{\partial C} \end{bmatrix}.$$

Therefore, the bottleneck of computation is the inverting step in computing K. Note L is a diagonal matrix, we can further lower the computation cost by applying the rules for inverting a block matrix again. The value of λ_3 and λ_4 can be estimated from the fact $p = 0, q = 0$. We detail the algorithm in Algorithm 2.

Algorithm 2 Computing the gradient for w

Require: $C \in \mathbb{R}^{m \times n}, \mu, \nu, \epsilon, \frac{dC}{dw}$
Run forward pass to get $S = S_{\mathrm{r}}^*, \bar{\mu}, \bar{\nu}, \xi, \zeta$
$x_1 = \sum_{i=1}^{\lceil n/2 \rceil} (\bar{\mu}_i - \mu_i), x_2 = \sum_{i=\lceil n/2 \rceil}^{n} (\bar{\mu}_i - \mu_i), b_1 = -\sum_{i=1}^{\lceil n/2 \rceil} \xi_i, b_2 = -\sum_{i=\lceil n/2 \rceil}^{n} \xi_i$
$[\lambda_1, \lambda_3]^\top = [[\lceil n/2 \rceil, x_1; n - \lceil n/2 \rceil, x_2]^{-1} [b1, b2]]^\top$
$x_1 = \sum_{j=1}^{\lceil m/2 \rceil} (\bar{\nu}_j - \nu_j), x_2 = \sum_{j=\lceil m/2 \rceil}^{m} (\bar{\nu}_j - \nu_j), b_1 = -\sum_{j=1}^{\lceil m/2 \rceil} \zeta_j, b_2 = -\sum_{j=\lceil m/2 \rceil}^{m} \zeta_j$
$[\lambda_2, \lambda_4]^\top = [[\lceil m/2 \rceil, x_1; m - \lceil m/2 \rceil, x_2]^{-1} [b1, b2]]^\top$
$\bar{\mu} = \bar{\mu} + \epsilon(2\lambda_3 \mathbf{1}_n + \frac{\epsilon_1}{\bar{\mu}})^{-1}, \bar{\nu} = \bar{\nu} + \epsilon(2\lambda_4 \mathbf{1}_m + \frac{\epsilon_2}{\bar{\nu}})^{-1}$
$\bar{\nu}' = \bar{\nu}[:-1], S' = S[:, :-1]$
$\mathcal{K} \leftarrow \mathrm{diag}(\bar{\nu}') - (S')^T (\mathrm{diag}(\bar{\mu}))^{-1} S'$
$H_1 \leftarrow (\mathrm{diag}(\bar{\mu}))^{-1} + (\mathrm{diag}(\bar{\mu}))^{-1} S' \mathcal{K}^{-1} (S')^\top (\mathrm{diag}(\bar{\mu}))^{-1}$
$H_2 \leftarrow -(\mathrm{diag}(\bar{\mu}))^{-1} S' \mathcal{K}^{-1}$
$H_3 \leftarrow (H_2)^\top$
$H_4 \leftarrow \mathcal{K}^{-1}$
Pad H_2 to be $[n, m]$ with value 0
Pad H_3 to be $[m, n]$ with value 0
Pad H_4 to be $[m, m]$ with value 0
$L = \mathrm{diag}([\epsilon(2\lambda_3 \mathbf{1}_n + \frac{\epsilon_1}{\bar{\mu}})^{-1}, \epsilon(2\lambda_4 \mathbf{1}_m + \frac{\epsilon_2}{\bar{\nu}})^{-1}])$
$A1 = [H_1, H_2; H_3, H_4]$
$A_2 = -A_1 \cdot L$
$A_3 = A_2^\top$
$A_4 = L + L \cdot A_1 \cdot L$
$E = A_1 + A_2 \cdot B1(D - C \cdot A_4 \cdot B)^{-1} C \cdot A_3$, where $B1, C_1, D$ defined above
$[J_1, J_2; J_3, J_4] = E$, where $J_1 \in \mathbb{R}^{n \times n}, J_2 \in \mathbb{R}^{n \times m}, J_3 \in \mathbb{R}^{m \times n}, J_4 \in \mathbb{R}^{m \times m}$
$[\frac{d\xi^*}{dC}]_{nij} \leftarrow [J_1]_{ni} S_{ij} + [J_2]_{nj} S_{ij}$
$[\frac{d\zeta^*}{dC}]_{mij} \leftarrow [J_3]_{mi} S_{ij} + [J_4]_{mj} S_{ij}$
Pad $\frac{d\zeta^*}{dC}$ to be $[m, n, m]$ with value 0
$[\frac{d\mathcal{L}}{dC}]_{ij} \leftarrow \frac{1}{\epsilon}(-C_{ij} S_{ij} + \sum_{n,m} C_{nm} S_{nm} [\frac{da^*}{dC}]_{nij} + \sum_{n,m} C_{nm} S_{nm} [\frac{db^*}{dC}]_{mij}) + S_{ij}$
return $\frac{d\mathcal{L}}{dC} \frac{dC}{dw}$



(a) ℓ_2 distance (b) KL-divergence

Figure 9: *Illustration with different choice of $\mathcal{F}$.*

F Different forms of marginal relaxation

In this paper we adopt $\mathcal{F}$ to be the Euclidean distance. This is because this choice provides an OT plan that fits our intuition – the data points with significantly larger transportation cost should not be considered. Figure 9 shows an illustration. Here, the input distributions are the empirical distributions of the scalars on the left and the bottom. Notice that there are three support points in μ that are far away from others, i.e., $10.72, 10.89, 10.96$. In Figure 9 (a), the optimal solution Γ_{r}^* automatically ignores them, matching only the rest of the scalars. One alternative choice of $\mathcal{F}$ is the Kullback–Leibler (KL) divergence (Chizat et al., 2018b), whose resulted formulation possesses an efficient algorithm for the forward pass, and the differentiability for the backward pass. We do not adopt it because the OT plan generated by this choice does not fit out intuition: As shown in Figure 9 (b), the OT plan tends to ignore the points that are away from the mean, even with a very small ρ_1 and ρ_2. For both figures, we adopt $\epsilon = 10^{-5}$.

G More on Experiments

G.1 Unlabeled Sensing

We now provide more training details for experiments in Section 4.1. Here, AM and ROBOT is trained with batch size 500 and learning rate 10^{-4} for $2,000$ iterations. For the Sinkhorn algorithm in ROBOT we set $\epsilon = 10^{-4}$. We run RS for 2×10^5 iterations with inlier threshold as 10^{-2}. Other settings for the hyper-parameters in the baselines follows the default settings of their corresponding papers.

G.2 Nonlinear Regression

For the nonlinear regression experiment in Section 4.2, ROBOT and ROBOT-robust is trained with learning rate 10^{-4} for 80 iterations. For $n = 100, 200, 500, 1000, 2000$, we set batch size $10, 30, 50, 100, 300$, respectively.We set $\epsilon = 10^{-4}$ for the Sinkhorn algorithm in ROBOT. For Oracle and LS, we perform ordinary regression model and ensure convergence, i.e., learning rate 5×10^{-2} for 100 iterations.

G.3 Flow Cytometry

We provide more details for the Flow Cytometry experiment in Section 4.3. In the FC seting, ROBOT is trained with batch size 1260 and learning rate 10^{-4} for 80 iterations. In the GFC seting, ROBOT is trained with batch size 1260 and learning rate 6×10^{-4} for 60 iterations. We set $\epsilon = 10^{-4}$ for the Sinkhorn algorithm in ROBOT. Other settings for the hyper-parameters in the baselines follows the default settings of their corresponding papers. EM is initialized by AM.

G.4 MULTI-OBJECT TRACKING

For the MOT experiments in Section 4.4, the reported results of `MOT17` (train) and `MOT17` (dev) is trained on `MOT17` (train), and the reported results of `MOT20` (train) and `MOT20` (dev) is trained on `MOT20` (train). Each model is trained for 1 epoch. We adopt Adam optimizer with learning rate$= 10^{-5}$, $\epsilon = 10^{-4}$, and $\eta = 10^{-3}$. To track the birth and death of the tracks, we adapt the inference code of Xu et al. (2019b).

G.5 COMBINATION WITH RS

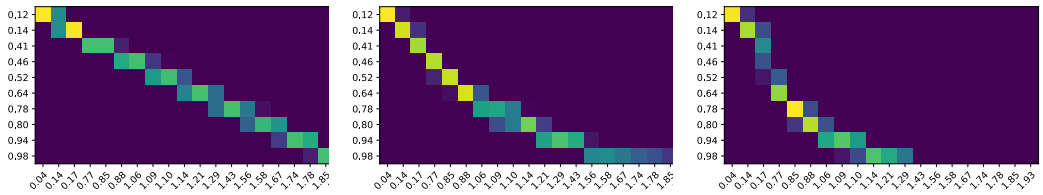
As suggested in Figure 2, although RS cannot perform well itself, retraining the output of RS using our algorithms increases the performance by a large margin. To show that combining RS and ROBOT can achieve better results than RS alone, we compare the following two cases: i). Subsample 2×10^5 times using RS; ii). Subsample 10^5 times using RS followed by ROBOT for 50 training steps. The result is shown in Table 2. For a larger permutation proportion, RS alone cannot perform as well as RS+ROBOT combination. Here, we have 10 runs for each proportion. We adopt `SNR`$= 100$, $d = 5$ for data, and $\epsilon = 10^{-4}$, learning rate 10^{-4} for ROBOT training.

Table 2: *Pairwise comparisons between RS alone and the combination of RS and ROBOT. The relative error ratio is the ratio of the relative errors of RS alone and RS+ROBOT combination. Ratios larger than* 1 *suggest that RS performs worse than RS+ROBOT combination.*

Proportion	25%	50%	75%
Rel. error ratio	1.04 ± 0.20	1.29 ± 0.32	1.27 ± 0.34

G.6 THE EFFECT OF ρ_1 AND ρ_2

We visualize S_{r}^* computed from the robust optimal transport problem in Figure 10. The two input distributions are Unif$(0, 2)$ and Unif$(0, 1)$. We can see that with large enough ρ_1 and ρ_2, Unif$(0, 1)$ would be aligned with the first half of Unif$(0, 2)$.



(a) $\rho_1 = 0, \rho_2 = 0$ (b) $\rho_1 = 0.1, \rho_2 = 0.1$ (c) $\rho_1 = 0.2, \rho_2 = 0.2$

Figure 10: Computed S^* for robust optimal transport problem.

G.7 COMPARISON OF RESIDUALS IN LINEAR REGRESSION

Settings. We generate n data points $\{(y_i, [x_i, z_i])\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ and $z_i \in \mathbb{R}^e$. We first generate $x_i \sim \mathcal{N}(\mathbf{0}_d, \boldsymbol{I}_d)$, $z_i \sim \mathcal{N}(\mathbf{0}_e, \boldsymbol{I}_e)$, $w \sim \mathcal{N}(\mathbf{0}_{d+e}, \boldsymbol{I}_{d+e})$, and $\varepsilon_i \sim \mathcal{N}(0, \rho_{\mathrm{noise}}^2)$. Then we compute $y_i = f([x_i, z_i]; w) + \varepsilon_i$. Next, we randomly permute the order of $\{z_i\}$ so that we lose the data correspondence. Here, $\mathcal{D}_1 = \{(x_i, y_i)\}$ and $\mathcal{D}_2 = \{z_j\}$ mimic two parts of data collected from two separate platforms.

We adopt a linear model $f(x; w) = x^\top w$. To evaluate model performance, we use error$= \sum_i (\widehat{y}_i - y_i)^2 / \sum_i (y_i - \bar{y})^2$, where $\widehat{y}_i$ is the predicted label, and $\bar{y}$ is the mean of $\{y_i\}$.

Baselines. We use Oracle, LS, Stochastic-EM as the baselines. Notice that without a proper initialization, Stochastic-EM performs well in partially permuted cases, but not in fully shuffled cases.

For better visualization, we only include this baseline in one experiment. Furthermore, we adopt two new baselines: Sliced-GW (Vayer et al., 2019) and Sinkhorn-GW (Xu et al., 2019a), which can be used to align distributions and points sets.

Results. We visualize the fitting error of regression models in Figure 11. We can see that ROBOT outperforms all the baselines except Oracle. Also, our model can beat the Oracle model when the dimension is low or when the noise is large.

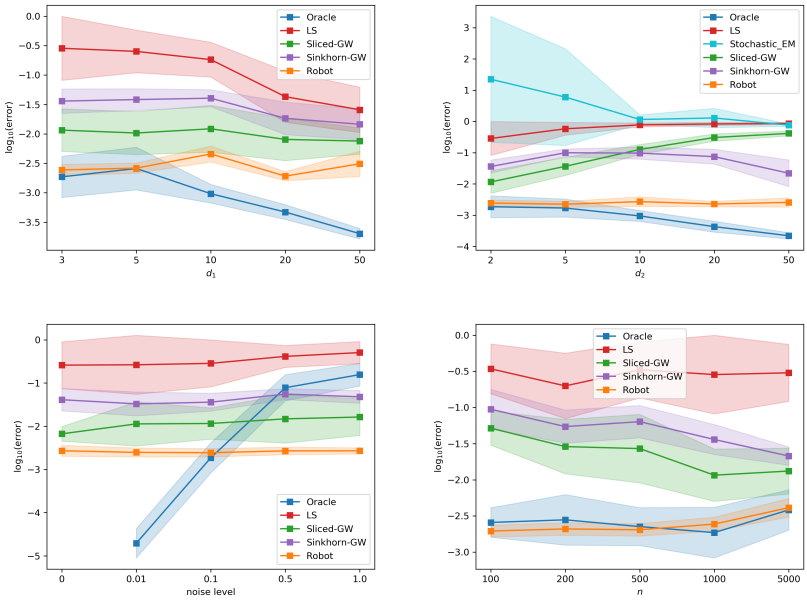


Figure 11: Linear regression. We use $n = 1000$, $d = 2$, $e = 3$, $\rho^2_{\text{noise}} = 0.1$ as defaults.