Appendix: Saddle Points and Accelerated Perceptron Algorithms

Adams Wei Yu[†] WEIYU@CS.CMU.EDU
Fatma Kılınç-Karzan[‡] FKILINC@ANDREW.CMU.EDU
Jaime G. Carbonell[†] JGC@CS.CMU.EDU
[†]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
[‡]Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA

This appendix provides the omitted proofs from the main text, detailed descriptions and derivations for mirror prox framework, and supplement for numerical experiments.

9. Proofs Omitted from the Main Text

Proof of Lemma 1.

Proof. (a) Definition of $\rho(A)$ and the fact $\min_{j=1,\dots,n} y^T A^j = \min_{x\in\Delta_n} y^T Ax$ implies $\rho(A) = \max_{\|y\|_2=1} \min_{x\in\Delta_n} y^T Ax$.

(b) Let $f(y) := \min_{x\in\Delta_n} y^T Ax$, and note that $f(y)$ is concave in y. Using (a), $\rho(A) = \max_{\|y\|_2=1} f(y)$, i.e., $\exists\, y^*$ s.t. $\|y^*\|_2 = 1$ and $\rho(A) = f(y^*) = \max_{\|y\|_2=1} f(y) \le \max_{\|y\|_2\le 1} f(y)$ where the last inequality follows from the obvious relaxation. Whenever $\rho(A) > 0$, we claim that this relaxation is tight. If not, then $\exists\ \bar{y}$ s.t. $\|\bar{y}\|_2 < 1$, and $f(\bar{y}) > f(y^*) = \rho(A) > 0 = f(0)$, and thus $\bar{y} \neq 0$. Now consider $\hat{y} := \frac{\bar{y}}{\|\bar{y}\|_2}$. Note that $\|\hat{y}\|_2 = 1$ and $f(\hat{y}) = \frac{1}{\|\bar{y}\|_2} f(\bar{y}) > f(\bar{y}) > f(y^*)$, where the first inequality is due to $\|\bar{y}\|_2 < 1$. But this contradicts to the optimality of y^*, and hence we conclude $\rho(A) = \max_{\|y\|_2\le 1} f(y) = \text{Opt}$. □

Proof of Theorem 2.

Proof. (a) $0 < \underline{\phi}(y_t) = \min_{x\in\Delta_n} y_t^T Ax \le y_t^T Ae^i = (A^T y_t)_i$ for all $i = 1, \dots, n$, and hence $A^T y_t > 0$.

(b) In this case, $\underline{\phi}(y_t) \le 0$ and $\epsilon_{\text{sad}}(z_t) = \overline{\phi}(x_t) - \underline{\phi}(y_t) \le \epsilon$ implying $\overline{\phi}(x_t) \le \underline{\phi}(y_t) + \epsilon \le \epsilon$. Moreover, $\overline{\phi}(x_t) = \max_{\|y\|_2\le 1} y^T Ax = \|Ax_t\|_2$, and hence we have $\|Ax_t\|_2 \le \epsilon$. □

Proof of Theorem 3.

Proof. From the definition of Opt, $\overline{\phi}(\cdot)$, and $\underline{\phi}(\cdot)$, for any $x \in X$ and $y \in Y$, we have $\underline{\phi}(y) \le \text{Opt} \le \overline{\phi}(x)$.

(a) If $\rho(A) > 0$, by Lemma 1, we have $\rho(A) = \text{Opt}$. Combining the above relations with Theorem 1, for every iteration $t \ge 1$, we obtain

$$\overline{\phi}(x_t) - \frac{\Omega\mathcal{L}}{t} \le \underline{\phi}(y_t) \le \rho(A) \le \overline{\phi}(x_t) \le \underline{\phi}(y_t) + \frac{\Omega\mathcal{L}}{t}.$$

Note that by rearranging terms in this inequality, we get

$$\rho(A) - \frac{\Omega\mathcal{L}}{t} \le \underline{\phi}(y_t) \le \rho(A),$$

By setting $N = \frac{\Omega\mathcal{L}}{\rho(A)} + 1$, we get $\underline{\phi}(y_N) \ge \rho(A) - \frac{\Omega\mathcal{L}}{N} > 0$. By Theorem 2, we know y_N is a feasible solution of LDFP.

(b) If $\rho(A) < 0$, then $\underline{\phi}(y_t) < 0$. Let $N = \frac{\Omega\mathcal{L}}{\epsilon} + 1$. By Theorem 1, $\epsilon_{\text{sad}}(z_N) \le \frac{\Omega\mathcal{L}}{N} < \epsilon$. Then by Theorem 2, x_N is an ϵ-solution of LAP. □

10. Mirror Prox Setup for Feasibility Problems

Let $Z = X \times Y$ where $X,\ Y$ are two given domains with respective norms $\|\cdot\|_x$ and $\|\cdot\|_y$, their dual norms $\|\cdot\|_{(x,*)}$ and $\|\cdot\|_{(y,*)}$, and d.g.f.'s $\omega_x(\cdot)$ and $\omega_y(\cdot)$. Given two scalars $\alpha_x, \alpha_y > 0$, we build the setup for the Mirror Prox algorithm, i.e., $\|\cdot\|$ and d.g.f. $\omega(\cdot)$ for the domain Z, where $z = [x; y]$, as follows:

$$\|z\| = \sqrt{\alpha_x \|x\|_x^2 + \alpha_y \|y\|_y^2}$$

with the dual norm

$$\|\zeta\|_* = \sqrt{\frac{1}{\alpha_x}\|\zeta_x\|_{(x,*)}^2 + \frac{1}{\alpha_y}\|\zeta_y\|_{(y,*)}^2}$$

and corresponding d.g.f. given by

$$\omega(z) = \alpha_x \omega_x(x) + \alpha_y \omega_y(y).$$

With these choices, we arrive at $\Omega = \Omega_z \le \alpha_x\Omega_x + \alpha_y\Omega_y$ where $\Omega_x := \max_{x\in X} V_{x_\omega}(x) \le \max_{x\in X}\omega(x) - \min_{x\in X}\omega(x)$ and Ω_y is defined similarly. Moreover, our prox mapping $\text{Prox}_z(\xi)$ becomes decomposable, i.e., by letting $\xi = [\xi_x;\xi_y]$, we have

$$\begin{aligned}
&\text{Prox}_z^\omega(\xi)\\
&= \operatorname*{argmin}_{w\in Z}\{\langle\xi, w\rangle + V_z(w)\}\\
&= \operatorname*{argmin}_{w\in Z}\{\langle\xi, w\rangle + (\omega(w) - \omega(z) - \langle\omega'(z), w - z\rangle)\}\\
&= \operatorname*{argmin}_{(w_x;w_y)\in Z}\{\langle\xi_x, w_x\rangle + \langle\xi_y, w_y\rangle + \alpha_x\omega_x(w_x) + \alpha_y\omega_y(w_y)\\
&\qquad - \alpha_x\langle\omega_x'(x), w_x - x\rangle - \alpha_y\langle\omega_y'(y), w_y - y\rangle\}\\
&= \operatorname*{argmin}_{(w_x;w_y)\in Z}\{\langle\xi_x, w_x\rangle + \langle\xi_y, w_y\rangle + \alpha_x\omega_x(w_x) + \alpha_y\omega_y(w_y)\\
&\qquad - \alpha_x\langle\omega_x'(x), w_x\rangle - \alpha_y\langle\omega_y'(y), w_y\rangle\}\\
&= \left[\operatorname*{argmin}_{w_x\in X}\{\langle\xi_x - \alpha_x\omega_x'(x), w_x\rangle + \alpha_x\omega_x(w_x)\}\ ;\right.\\
&\qquad \left.\operatorname*{argmin}_{w_y\in Y}\{\langle\xi_y - \alpha_y\omega_y'(y), w_y\rangle + \alpha_y\omega_y(w_y)\}\right]\\
&= \left[\text{Prox}_x^{\omega_x}\left(\frac{\xi_x}{\alpha_x}\right);\ \text{Prox}_y^{\omega_y}\left(\frac{\xi_y}{\alpha_y}\right)\right].
\end{aligned}$$

Furthermore for bilinear saddle point problems, we have

$$\begin{aligned}
\|F(z) - F(z')\|_* &= \|(A^T(y-y'); A(x-x'))\|_*\\
&= \sqrt{\frac{1}{\alpha_x}\|A^T(y-y')\|_{(x,*)}^2 + \frac{1}{\alpha_y}\|A(x-x')\|_{(y,*)}^2}\\
&\le \mathcal{L}\|z - z'\|
\end{aligned}$$

with $\mathcal{L} := \sqrt{\frac{1}{\alpha_x}\mathcal{L}_{xy}^2 + \frac{1}{\alpha_y}\mathcal{L}_{yx}^2}$ where

$$\begin{aligned}
\mathcal{L}_{xy} &\ge \max_y\left\{\|A^Ty\|_{(x,*)} :\ \|y\|_y \le 1\right\} \quad \text{and}\\
\mathcal{L}_{yx} &\ge \max_x\left\{\|Ax\|_{(y,*)} :\ \|x\|_x \le 1\right\}.
\end{aligned}$$

Hence we arrive at

$$\Omega\mathcal{L} \le (\alpha_x\Omega_x + \alpha_y\Omega_y)\sqrt{\frac{1}{\alpha_x}\mathcal{L}_{xy}^2 + \frac{1}{\alpha_y}\mathcal{L}_{yx}^2}.$$

By minimizing this upper bound in terms of α_x, α_y, we get

$$\begin{aligned}
\alpha_x &= \frac{\mathcal{L}_{xy}}{\sqrt{\Omega_x}(\mathcal{L}_{xy}\sqrt{\Omega_x} + \mathcal{L}_{yx}\sqrt{\Omega_y})} \text{ and}\\
\alpha_y &= \frac{\mathcal{L}_{yx}}{\sqrt{\Omega_y}(\mathcal{L}_{xy}\sqrt{\Omega_x} + \mathcal{L}_{yx}\sqrt{\Omega_y})},
\end{aligned}$$

which leads to

$$\mathcal{L} = \mathcal{L}_{xy}\sqrt{\Omega_x} + \mathcal{L}_{yx}\sqrt{\Omega_y},$$

and

$$\Omega \le \alpha_x\Omega_x + \alpha_y\Omega_y \le 1.$$

Setup for Linear Feasibility Problems

In the particular case of LDFP, taking into account the setup $X\times Y = \Delta_n\times\mathcal{B}_m$, we select $\|\cdot\|_x = \|\cdot\|_1$ and $\|\cdot\|_y = \|\cdot\|_2$ with d.g.f.'s $\omega_x(x) = \text{Entr}(x) := \sum_{i=1}^n x_i\ln(x_i)$, i.e., the Entropy d.g.f., and $\omega_y(y) = \frac{1}{2}y^Ty$ Euclidean d.g.f. with

$$\text{Prox}_z^\omega(\xi) = \left[\text{Prox}_x^{\omega_x}\left(\frac{\xi_x}{\alpha_x}\right);\ \text{Prox}_y^{\omega_y}\left(\frac{\xi_y}{\alpha_y}\right)\right].$$

By considering the form of the d.g.f.s and the associated domains, we can utilize the closed form expressions for the corresponding optimal solutions in the above optimization problems for prox mapping computations and hence arrive at, for $i = 1,\dots,n$,

$$[\text{Prox}_x^{\omega_x}(\xi_x)]_i = \frac{x_i\exp\{-[\xi_x]_i\}}{\sum_{j=1}^n x_j\exp\{-[\xi_x]_j\}} \quad \text{and}$$

$$[\text{Prox}_y^{\omega_y}(\xi_y)]_i = \begin{cases} y_i - [\xi_y]_i, & \text{if } \|y - \xi_y\|_2 \le 1\\ \frac{y_i - [\xi_y]_i}{\|y-\xi_y\|_2}, & \text{otherwise}\end{cases}.$$

Moreover by selecting $z_\omega = \left[\frac{1}{n}\mathbf{1}; \mathbf{0}_m\right]$, where $\mathbf{1}$ stands for the vector in $\mathbb{R}^n$ with all coordinates equal to 1 and $\mathbf{0}_m$ denotes the zero vector in $\mathbb{R}^m$, we get

$$\Omega_x = \log(n), \text{ and } \Omega_y = \frac{1}{2} \tag{9}$$

Also, $\mathcal{L} = \sqrt{\frac{1}{\alpha_x}\mathcal{L}_{xy}^2 + \frac{1}{\alpha_y}\mathcal{L}_{yx}^2}$ with $\mathcal{L}_{xy} \ge \max_y\{\|A^Ty\|_\infty : \|y\|_2 \le 1\}$ and $\mathcal{L}_{yx} \ge \max_x\{\|Ax\|_2 : \|x\|_1 \le 1\}$. In fact in this current setup, one can pick

$$\mathcal{L}_{xy} = \mathcal{L}_{yx} = \max_j\|A_j\|_2 = 1$$

due to our normalization. Hence by considering $\Omega \le \alpha_x\Omega_x + \alpha_y\Omega_y$, we arrive at

$$\Omega\mathcal{L} \le (\alpha_x\Omega_x + \alpha_y\Omega_y)\sqrt{\frac{1}{\alpha_x}\mathcal{L}_{xy}^2 + \frac{1}{\alpha_y}\mathcal{L}_{yx}^2}.$$

The minimization of this upper bound in α_x, α_y, as discussed above, leads to the corresponding parameters associated with Mirror Prox algorithm with $\Omega \le 1$ and $\mathcal{L} = \mathcal{L}_{xy}\sqrt{\Omega_x} + \mathcal{L}_{yx}\sqrt{\Omega_y}$. Therefore, in the case of linear feasibility problems LDFP and LAP, we have

$$\mathcal{L} = \mathcal{L}_{xy}\sqrt{\Omega_x} + \mathcal{L}_{yx}\sqrt{\Omega_y} = \sqrt{\log(n)} + \sqrt{1/2}$$

and hence

$$\Omega\mathcal{L} \le \sqrt{\log(n)} + \sqrt{1/2}.$$

Setup for Kernelized and Conic Feasibility Problems

The kernelized case is the same as the linear case. For the conic case, as mentioned before, $\Omega\mathcal{L}$ determines the convergence rate of the MPFP framework. As mentioned in the previous section, the domains involved X, Y determine both $\Omega\mathcal{L}$ and the prox mappings. For conic feasibility problems, in all cases Y domain is $\mathcal{B}_m$, always resulting in $\Omega_y = \frac{1}{2}$, yet Ω_x is case dependent, for $\mathcal{K} = \mathbb{R}^n_+$ and $\mathcal{K} = \mathbb{S}^n_+$, we have $\Omega_x = O(\log(n))$ and for $\mathcal{K} = \mathbb{L}^n$, we have $\Omega_x = \frac{1}{2}$, which leads to the resulting rate of convergences which is no worse than $O(\frac{\sqrt{\log(n)}}{\epsilon})$ for Mirror Prox algorithm.

11. Supplement for Numerical Experiments

In this section, we supplement our real data results and present our synthetic data generation procedures and the additional experimental results on the synthetic data, which could not have been incorporated to the main text because of space limitation. Recall that, in the following tables, 'TO' means 'Timeout' and 'Fail' means the algorithm fails due to running into numerical issues or falling into dead loop.

Synthetic Data for LDFP Instances. As for the LDFP instance generation, we follow the scheme suggested by (Soheili & Peña, 2012) to construct the matrix A: we generate random vectors $\bar{y} \in \mathbb{R}^m$, $v \in \mathbb{R}^n$ where $\bar{y}$ is drawn from a uniform distribution on the unit sphere $\{y \in \mathbb{R}^m : \|y\|_2 = 1\}$ and each v_i is i.i.d. from Uniform$[0, 1]$. We set $\bar{A} = B + \bar{y}(\kappa v^T - \bar{y}^T B)$ where $\kappa \geq 0$ is a parameter and $B \in \mathbb{R}^{m\times n}$ is a randomly generated matrix, with each entry independently drawn from the standard normal distribution. Note that $\bar{A}^T\bar{y} = \kappa v$; hence, if $\kappa > 0$, LDFP is feasible with $\bar{y}$ as a solution. Further, κ serves as a proxy for the condition number $\rho(\bar{A})$, i.e., a larger κ corresponds to a larger $\rho(\bar{A})$ in (3). Since normalization does not change feasibility status of LDFP or $\rho(A)$, our A matrix is obtained by normalizing $\bar{A}$ such that all columns have their Euclidean norms equal to 1. Nevertheless, we observed that normalization drastically alters the computational behavior of two algorithms, SPCT and ISPVN. In particular, we observed that in all of the normalized instances, ISPVN algorithm failed by falling into a deadloop. Therefore we compared our algorithms against SPCT, PCT, and VN for normalized instances. On the other hand, we also performed a number of tests with unnormalized data, e.g., with the matrix $\bar{A}$ described above, and observed that in these cases, SPCT algorithm failed while we were able to run the ISPVN algorithm. We discuss the corresponding comparisons below.

Iterations of the LDFP. We present results on the number of iterations for algorithms tested in the LDFP experiments (normalized instances with A) in Table 3. This complements the corresponding runtime results, i.e., Table 2 in the LDFP section of the main text. From Table 3, we observe that the MPLFP outperforms competing algorithms in terms of number of iterations, under any (m, n) combination. This behavior also supports the superior performance of MPLFP in terms of its solution time given in Table 3.

We also present the results on the unnormalized data matrix $\bar{A}$, in Tables 3 and 4, which also support the superior performance of MPLFP.

	With column-normalized matrix, A			
(m, n)	MPLFP	SPCT	PCT	VN
$(10^2, 5\times10^3)$	**204.6**	605.3	92413.4	91450.1
$(10^3, 5\times10^3)$	**146.0**	338.9	15009.3	14864.0
$(10^2, 5\times10^4)$	**1692.2**	5144.2	TO	TO
$(10^3, 5\times10^4)$	**730.8**	2409.9	TO	TO
$(10^2, 5\times10^5)$	**11580**	Fail	TO	TO
$(10^3, 5\times10^5)$	**6970**	Fail	TO	TO
	With unnormalized matrix, A			
(m, n)	MPLFP	ISPVN	PCT	VN
$(10^2, 5\times10^3)$	**203.2**	1601.1	90649	90841
$(10^3, 5\times10^3)$	**30.9**	68.0	14869	14909
$(10^2, 5\times10^4)$	**2210.4**	Fail	TO	TO
$(10^3, 5\times10^4)$	**283.4**	2525.8	TO	TO
$(10^2, 5\times10^5)$	**18149**	Fail	TO	TO
$(10^3, 5\times10^5)$	**3415.4**	Fail	TO	TO

Table 3. Number of iterations of all methods for LDFP ($\kappa = 1$). Iteration limit is 10^6 and runtime limit is 5×10^4s.

	With unnormalized matrix, $\bar{A}$			
(m, n)	MPLFP	ISPVN	PCT	VN
$(10^2, 5\times10^3)$	**1.2**	19.1	98.3	110.8
$(10^3, 5\times10^3)$	**1.4**	4.9	171.3	181.2
$(10^2, 5\times10^4)$	**90.4**	Fail	TO	TO
$(10^3, 5\times10^4)$	**136.7**	1963.6	TO	TO
$(10^2, 5\times10^5)$	**43748**	Fail	TO	TO
$(10^3, 5\times10^5)$	**22753**	Fail	TO	TO

Table 4. Runtime (second) of all methods for LDFP ($\kappa = 1$). Iteration limit is 10^6 and runtime limit is 5×10^4s.

Iterations as κ ($\rho(A)$) changes. As a complementary of experiment on the effect of $\rho(A)$ (on normalized instances of A), we include Figure 3 here to show the corresponding number of iterations needed for different algorithms under different $\rho(A)$. Again, MPLFP is faster than the nearest competitor by at least one order of magnitude.

We also test the effect of $\rho(\bar{A})$ on the unnormalized instances with $\bar{A}$ matrix. The corresponding runtime and number of iteration figures are given in Figures 4 and 5.

Synthetic Data for LAP Instances. We generated difficult LAP instances as follows: Given an integer $r \geq 3$, and

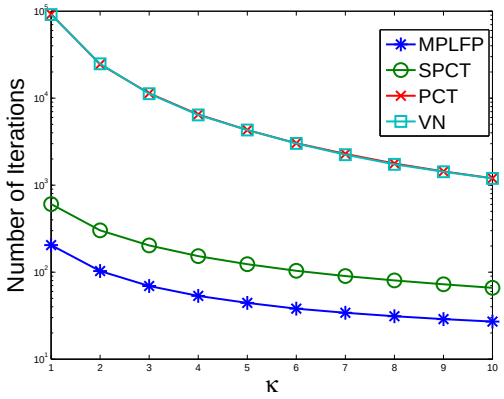

Figure 3. κ ($\rho(A)$) vs iteration with normalized matrix, A.

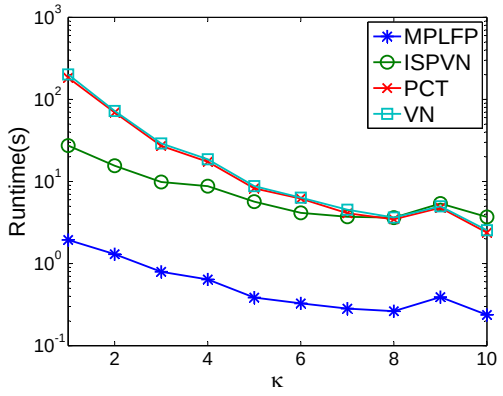

Figure 4. κ ($\rho(\bar{A})$) vs runtime with unnormalized matrix, $\bar{A}$.

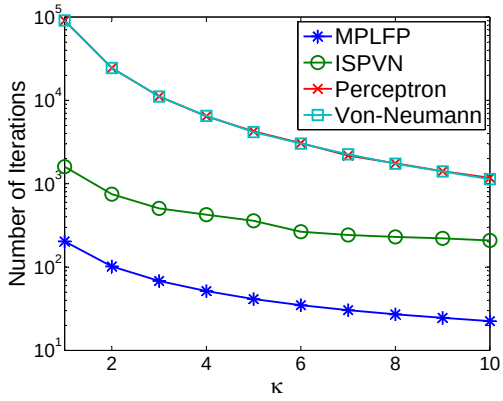

Figure 5. κ ($\rho(\bar{A})$) vs iteration with unnormalized matrix, $\bar{A}$.

number $\theta > 1$, we let $n = 2^r$ and generate a square $n \times n$ matrix for A. We control the difficulty of the instance with θ. We first generate $q = \lfloor \frac{(n-1)}{2} \rfloor$ points given by $S = \left\{ \cos(\pi \frac{k}{(q-1)}) : 0 \leq k \leq q-1 \right\} := \{s_0, \ldots, s_{q-1}\}$ and then translate and scale these points to obtain $p_i = \frac{1}{\theta} + (s_i - \min_j s_j) \frac{1-\frac{1}{\theta}}{\max_j s_j - \min_j s_j}$ for $i = 0, \ldots, q-1$, and the set $P = \{p_0, \ldots, p_{q-1}\}$. With this construction we have $\min\{p : p \in P\} = \frac{1}{\theta}$ and $\max\{p : p \in P\} = 1$. We define $(n-1) \times (n-1)$ diagonal matrix B with $2q$ diagonal entries $\{\pm\sqrt{p} : p \in P\}$, and set the remaining $n - 1 - 2q$ diagonal entries to 1. Then we let H be a Hadamard matrix of size $n \times n$ which is normalized to have all column norms equal to 1. We then generate a random $\bar{x}$ such that $\bar{x} \in \Delta_n = \{x \in \mathbb{R}^n : \sum_i x_i = 1, \, x_i \geq 0 \; \forall i\}$. Then we construct $\bar{A}$ matrix of the following form

$$\bar{A} = H^T \begin{bmatrix} c & f^T \\ f & B \end{bmatrix} H,$$

where we select c, f to ensure that $\bar{A}\bar{x} = 0$. In particular, this leads to a square system of linear equations in c, f as a result of the special structure of B and H. Our A matrix is then obtained by normalizing $\bar{A}$ to have all Euclidean norms of the columns equal to 1. Note that by normalizing $\bar{x}$, we will also obtain $x \in \Delta_n$ such that $Ax = 0$. In all of our instances for LAP, we work with the normalized matrix A. In all of the generated instances, ISPVN algorithm failed, therefore the comparison is done against only VN algorithm.

Performance on LAP. We first test the performance of different methods to find an ϵ-solution for LAP by setting $\theta = 5$ and $\epsilon = 10^{-3}$. We report the performance comparison of MPLFP and VN in Tables 5 and 6. As suggested by our theoretical convergence rates, MPLFP algorithm is orders of magnitude faster than VN algorithm on the LAP instances.

	ϵ-solution of LAP ($\theta = 5$)	
(m, n)	MPLFP	VN
(1024, 1024)	**2.5**	21.6
(2048, 2048)	**10.6**	125.8
(4096, 4096)	**69.2**	948.6
(8192, 8192)	**284.5**	5465.9
(16384, 16384)	**1014.0**	25426

Table 5. Runtime (second) of all methods for LAP. Iteration limit is 10^6 and runtime limit is 5×10^4s.

The Effect of θ. We also test the effect of θ by varying θ from the set $5, 10, 100, 1000, 10000$ on instances of size $n = m = 2048$. The results on runtime and iteration numbers are provided in Tables 7 and 8.

Real Data: CIFAR-10 Image Classification. We complement the performance of the algorithms on the test data

	ϵ-solution of LAP ($\theta = 5$)	
(m, n)	MPLFP	VN
(1024, 1024)	**265.2**	8763.6
(2048, 2048)	**283.4**	12352.4
(4096, 4096)	**291.6**	16353.8
(8192, 8192)	**297.1**	22838.1
(16384, 16384)	**293.5**	31894.3

Table 6. Number of iteration of all methods for LAP. Iteration limit is 10^6 and runtime limit is 5×10^4s.

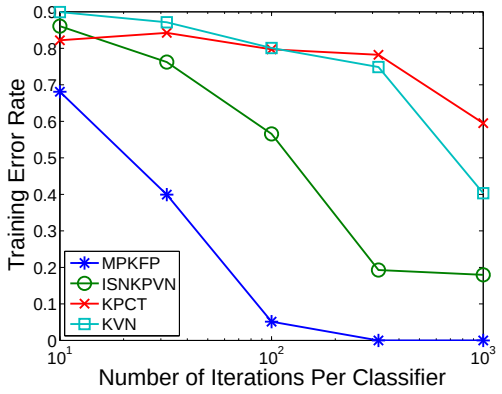

Figure 6. Iteration v.s. training error.

θ	5	10	10^2	10^3	10^4
MPLFP	10.4	10.7	14.2	13.7	13.5
VN	123.0	121.3	149.9	148.4	143.7

Table 7. Runtime (second) of MPLFP for LAP as θ varies.

θ	5	10	10^2	10^3	10^4
MPLFP	284.1	285.2	296.3	288.2	286.9
VN	12341	12149	11946	11800	11712

Table 8. Number of iterations of MPLFP for LAP as θ varies.

of CIFAR-10 Image Classification data set, e.g., Figure 2, with the corresponding comparison of their training errors on the training data set, e.g., Figure 6. Figure 6 indicates that within roughly 300 iterations, MPKFP achieves a training error of almost 0, and therefore supports the exceptional performance of MPKFP that we have observed in Figure 2.