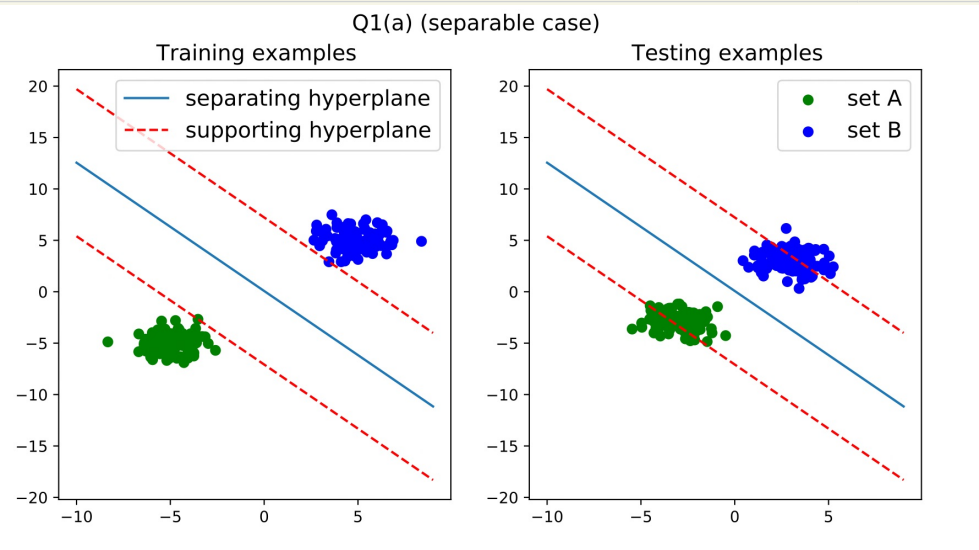
Q1 Consider the classification problem in Homework 2. Download the datasets `train_separable.mat` and `test_separable.mat` from the course website. Download `CVX` from `http://cvxr.com/cvx/` (or `http://www.cvxpy.org/en/latest/` if you use Python) and learn how to use it. Implement the following using `CVX`.

a) Apply the C-Hull formulation to train a classifer, i.e.,

$$\begin{aligned} &\text{minimize}_{u,v} \ \|Au - Bv\|_2^2 \\ &\text{subject to } 1^T u = 1, \ u \succeq 0 \\ &\qquad\qquad 1^T v = 1, \ v \succeq 0 \end{aligned}$$

Visualize the training data together with the classifier. Also visualize the testing data and the classifier in another figure, and report the classification error on the testing data using the true labels provided in `test_separable.mat` . (15%)

Following figure shows the data points and classifier (blue hyperplane)



Let $\hat{u}$ and $\hat{v}$ denote the solution returned by the solver (CVX), then we can find the hyperplane

$$w^T x = \gamma$$

by taking $w = \hat{u} - \hat{v}$ and $\gamma = \frac{\|\hat{u}\|^2 - \|\hat{v}\|^2}{2}$

The red hyperplanes are

$$w^T x = \gamma_u \quad \text{and}$$

$$w^T x = \gamma_v$$

where,

$$\gamma_u = w^T \hat{u}$$

$$\gamma_v = w^T \hat{v}$$

The classification error on the test data was 0%.

d) Repeat the above for `train_overlap.mat` and `test_overlap.mat` using the reduced C-Hull, i.e.,

$$\begin{aligned} \text{minimize}_{u,v} \ & \|Au - Bv\|_2^2 \\ \text{subject to } & \mathbf{1}^T u = 1, \ d\mathbf{1} \succeq u \succeq 0 \\ & \mathbf{1}^T v = 1, \ d\mathbf{1} \succeq v \succeq 0. \end{aligned}$$

Report the classification error on the testing data **using an appropriate** d. (15%)

(Please also submit your scripts. In case that your do not know what is C-Hull, check out the paper by Kristin P. Bennett, and Erin J. Bredensteiner, "**Duality and geometry in SVM classifiers**," ICML 2000. In addition, for background of classification, check out the slides of **Lecture 1**.)

To find an appropriate d for the reduced convex hull formulation, we use trial and error.

Mainly, we want to find a d that makes the reduced convex hulls separable, i.e. $\text{conv } A \cap \text{conv } B = \phi$

We can see that

$$\min_{x \in \text{conv}A,\, y \in \text{conv}B} \|x - y\|_2^2 \begin{cases} > 0 & \text{if } \text{conv}A \cap \text{conv}B = \phi \\ = 0 & \text{if } \text{conv}A \cap \text{conv}B \neq \phi \end{cases}$$

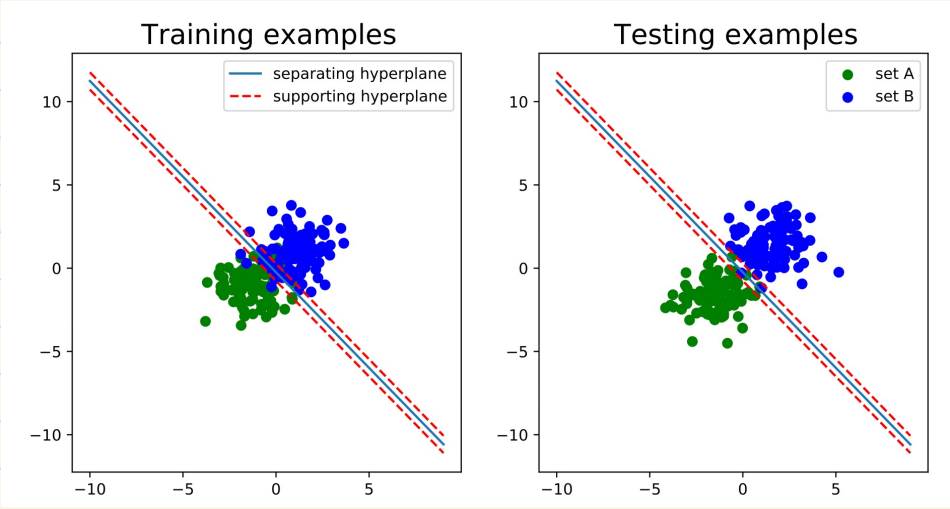
Therefore, one way to find an appropriate d is by searching for d such that

$$\| A\hat{u} - B\hat{v} \|_2^2 > \epsilon$$

By taking $\epsilon = 0.1$, and searching for d using grid method with gridsize $= 0.01$,

we get $d = 0.03$

And the classification error $= 2\%$.



Q2 Under the same setting of **Q1**, do the following:

(a) Implement C-Hull and Reduced C-Hull using projected gradient. Do the same visualization as in Q1. (15%)

(b) Repeat the above using Nesterov's acceleration. (15%)

(check out Beck, Amir, and Marc Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." SIAM journal on imaging sciences 2.1 (2009): 183-202.)

(a) Projected Gradient Method

Let $f(u,v) = \|Au - Bv\|_2^2$

For C-Hull, projected gradient method alternates between the following two steps

$$u^{(r,1)} \leftarrow u^{(r)} - \alpha_u \nabla_u f(u^{(r)}, v^{(r)})$$
$$v^{(r,1)} \leftarrow v^{(r)} - \alpha_v \nabla_v f(u^{(r)}, v^{(r)})$$

$$u^{(r+1)} \leftarrow \text{Proj}_{X_u}(u^{(r,1)})$$
$$v^{(r+1)} \leftarrow \text{Proj}_{X_v}(v^{(r,1)})$$

where $X_u = \{u \mid u \geq 0,\ 1^T u = 1\}$
$X_v = \{v \mid v \geq 0,\ 1^T v = 1\}$

we choose $\alpha_u = \frac{1}{\lambda_{max}(A^T A)}$

$\alpha_v = \frac{1}{\lambda_{max}(B^T B)}$

The gradients are obtained as follows:

$$\nabla_u f(u,v) = A^T A u - A^T B v$$

$$\nabla_v f(u,v) = B^T B v - B^T A u$$

For the reduced convex hull, we only need to change the projection set

$$X_u = \{ u \mid u^T 1 = 1,\ u \geq 0,\ u \leq d \}$$

$$X_v = \{ v \mid v^T 1 = 1,\ v \geq 0,\ v \leq d \}$$

The projection operator is defined as follows

$$\text{proj}_X(y) = \min_{x \in X} \frac{1}{2} \|x - y\|_2^2$$

We use CVX to obtain $\text{proj}_{X_u}(y)$ by solving the following

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & \frac{1}{2} \|u - y\|_2^2 \\ & u^T 1 = 1 \\ & u \geq 0 \\ & u \leq d \end{aligned}$$

And for $\text{Proj}_{X_v}(y)$ by solving the following

$$\begin{aligned} \underset{v}{\text{minimize}} \quad & \frac{1}{2} \|v - y\|_2^2 \\ & v^T 1 = 1 \\ & v \geq 0 \\ & v \leq d \end{aligned}$$

Following is the pseudocode for projected gradient descent

$v^{(0)} \leftarrow 0$

$u^{(0)} \leftarrow 0$

$r \leftarrow 0$

while $r < \text{max\_iteration}$ and $\|u^{(r)} - u^{(r-1)}\|_2^2 + \|v^{(r)} - v^{(r-1)}\|_2^2 > \varepsilon$ (when $r > 0$)

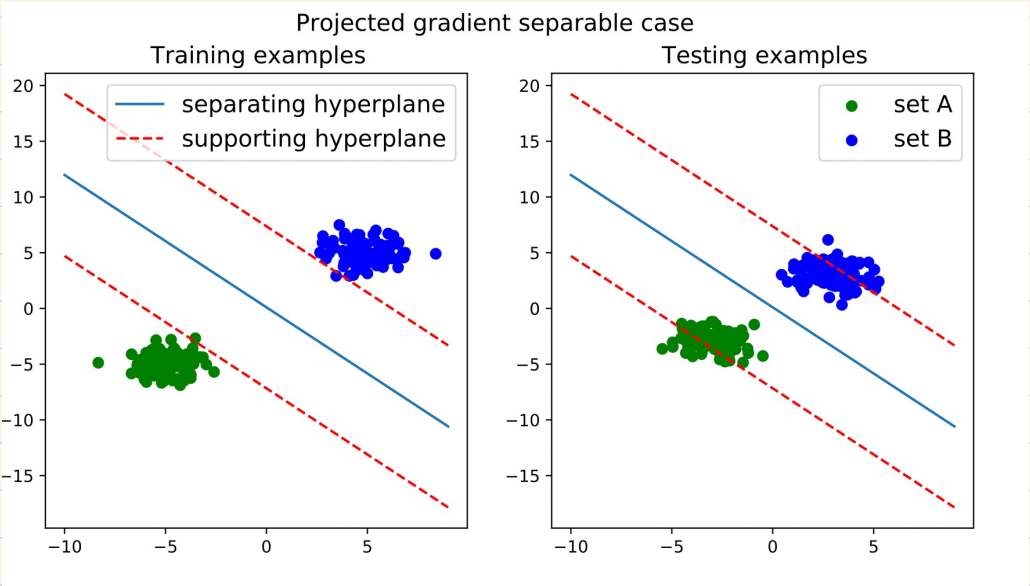
$r \leftarrow r + 1$

$u^{(r,1)} \leftarrow u^{(r)} - \alpha_u \nabla_u f(u, v)$

$v^{(r,1)} \leftarrow v^{(r)} - \alpha_v \nabla_v f(u, v)$

$u^{(r+1)} \leftarrow \text{Proj}_{X_u}(u^{(r,1)})$

$v^{(r+1)} \leftarrow \text{Proj}_{X_v}(v^{(r,1)})$



Above figure shows the result of using projected gradient descent in the separable case. we set Max_Iteration = 200 and $\varepsilon = 10^{-5}$. The figure shows that the separating and supporting hyperplanes are accurately estimated and is visually similar to the result from CVX. The classification error is 0 %.

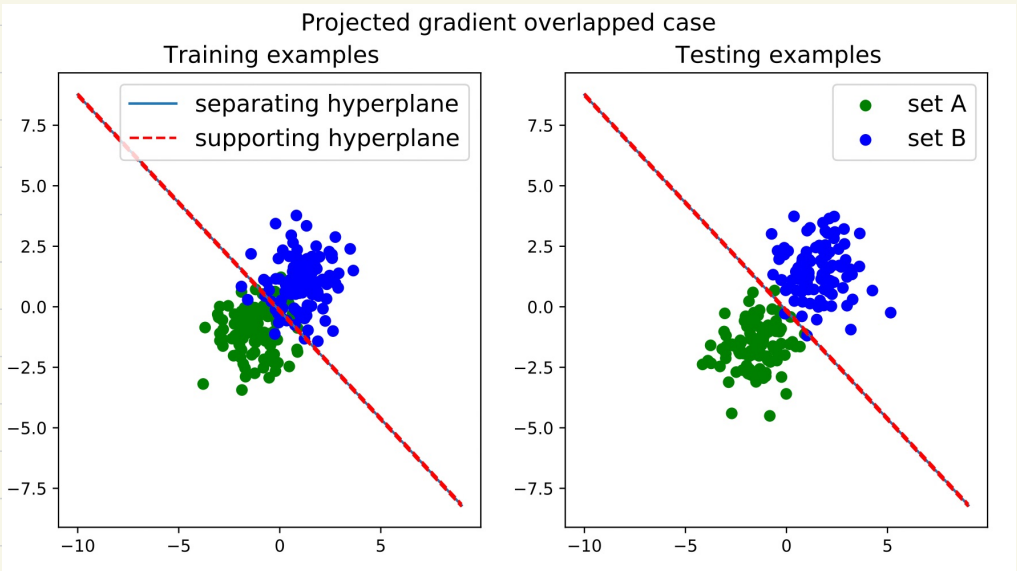


figure above shows the plot for the hyperplane found by projected gradient descent method for the case where the two clusters overlap. The classification error obtained was 0.15% when $\delta = 0.02$.

Following is the pseudocode for Nesterov's projected accelerated gradient

$v^{(0)} \leftarrow 0$; $y_v^{(0)} \leftarrow 0$

$u^{(0)} \leftarrow 0$; $y_u^{(0)} \leftarrow 0$

$r \leftarrow 1$

$a^{(0)} \leftarrow 0$

while $r <$ max_iteration

$$a^{(r+1)} \leftarrow \frac{1}{2}\left(1+\sqrt{4(a^{(r)})^2+1}\right)$$

$$t^{(r)} \leftarrow \left(a^{(r)}-1\right)/a^{(r+1)}$$

$$y_u^{(r+1)} \leftarrow (1+t^{(r)})\,u^{(r)} - t^{(r)}\,u^{(r-1)}$$

$$u^{(r,1)} \leftarrow y_u^{(r+1)} - \alpha_u \nabla_u f\left(y_u^{(r+1)}, v^{(r)}\right)$$

$$y_v^{(r+1)} \leftarrow (1+t^{(r)})\,v^{(r)} - t^{(r)}\,v^{(r-1)}$$

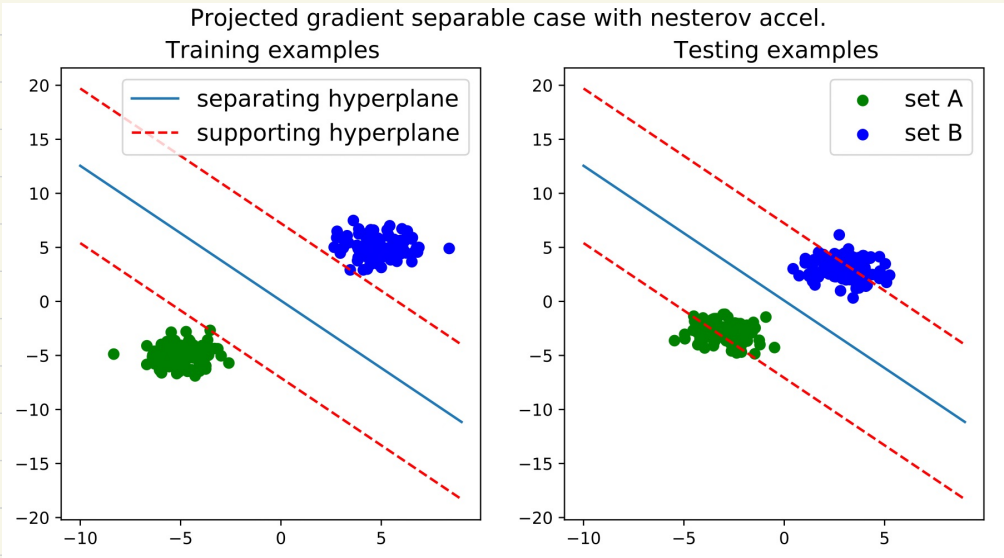
$$v^{(r,1)} \leftarrow y_v^{(r+1)} - \alpha_v \nabla_v f\left(u^{(r)}, y_v^{(r+1)}\right)$$

$$u^{(r+1)} \leftarrow \text{Proj}_{X_u}\left(u^{(r,1)}\right)$$

$$v^{(r+1)} \leftarrow \text{Proj}_{X_v}\left(v^{(r,1)}\right)$$

end while

Following figure shows the result of the case when Nesterov's acceleration is used in projected gradient descent for the separable case. Since Nesterov method does not guarantee decrease of objective in each iteration, we only use the max_iteration as the stopping criterion. Following figure shows that the end result obtained is similar to previous methods.



However, the objective decreases more rapidly than projected gradient descent, as will be observed in solution to question 4.

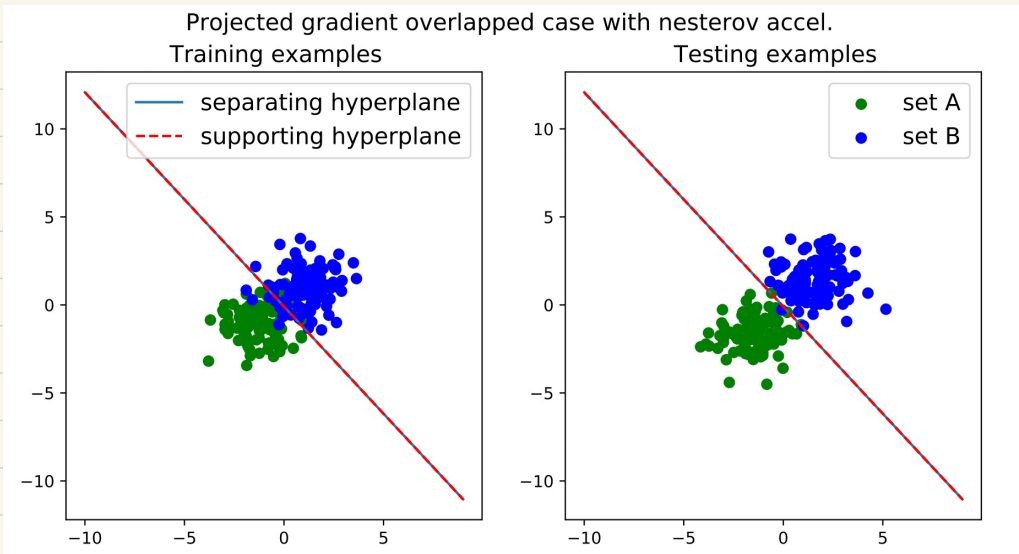


Figure above shows the result of using Nesterov's acceleration in

projected gradient descent for the overlapping case. We set $d = 0.03$. Here we obtain a classification error of 1.5%.

Q3 Under the same setting of **Q1**, implement C-Hull and Reduced C-Hull using ADMM. Do the same visualization as in Q1. (30%)

Let $x = \begin{bmatrix} u \\ v \end{bmatrix}$

Then $\|Au - Bv\|_2^2$ can be written as $\|\begin{bmatrix} A & -B \end{bmatrix} x\|_2^2$

And the constraint $1^T u = 1$, $1^T v = 1$ can be written as $\begin{bmatrix} 1^T & 0 \\ 0 & 1^T \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$$= \begin{bmatrix} 1^T & 0 \\ 0 & 1^T \end{bmatrix} x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Let $P = [A \quad -B]$

Then with variable splitting and after lifting the inequality constraint to the objective, we can rewrite C-Hull as

$$\underset{x_1, x_2}{\text{minimize}} \ \|Px_1\|_2^2 + I_+(x_2)$$

$$\text{subject to} \quad x_1 = x_2$$

$$\begin{bmatrix} 1^T & 0 \\ 0 & 1^T \end{bmatrix} x_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Further let

$$C = \begin{bmatrix} I & 0 \\ 0 & I \\ 1^T & 0 \\ 0 & 1^T \end{bmatrix}$$

$$D = \begin{bmatrix} I \\ 0 \end{bmatrix} \qquad e = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

of appropriate dimension such that the constraints can be re-written as

$$C x_1 = D x_2 + e$$

Finally C-Hull becomes

$$\underset{x_1, x_2}{\text{minimize}} \; \|P x_1\|_2^2 + I_+(x_2) \qquad (P1)$$

$$\text{subject to} \quad C x_1 - D x_2 - e = 0$$

The augmented Lagrangian of (P1) is

$$L(x_1, x_2, y) = \|P x_1\|_2^2 + I_+(x_2) + \frac{\rho}{2} \| C x_1 - D x_2 - e + y \|_2^2$$

The ADMM updates are

$$x_1^{(r+1)} = \underset{x_1}{\text{argmin}} \; \|P x_1\|_2^2 + \frac{\rho}{2} \| C x_1 - D x_2^{(r)} - e + y^{(r)} \|_2^2$$

$$= \underset{x_1}{\text{argmin}} \; x_1^T \left(P^T P + \frac{\rho}{2} C^T C \right) x_1 - x_1^T \frac{\rho C^T}{2} \left(y^{(r)} - D x_2^{(r)} - e \right)$$

$$\Rightarrow \; 2 \left(P^T P + \frac{\rho}{2} C^T C \right) x_1 = \frac{\rho C^T}{2} \left(y^{(r)} - D x_2^{(r)} - e \right)$$

$$x_2^{(r+1)} = \underset{x_2}{\operatorname{argmin}} \; I_+(x_2) + \frac{\rho}{2} \| C x_1^{(r+1)} - D x_2 - e + y^{(r)} \|_2^2$$

Let $y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$.

Then $$x_2^{(r+1)} = \underset{x_2}{\operatorname{argmin}} \; I_+(x_2) + \frac{\rho}{2} \left\| \begin{bmatrix} I & 0 \\ 0 & I \\ 1^T & 0 \\ 0 & 1^T \end{bmatrix} x_1^{(r+1)} - \begin{bmatrix} I \\ 0 \end{bmatrix} x_2 - \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} y_1^{(r)} \\ y_2^{(r)} \end{bmatrix} \right\|_2^2$$

$$= \underset{x_2}{\operatorname{argmin}} \; I_+(x_2) + \frac{\rho}{2} \| x_1^{(r+1)} - x_2 + y_1^{(r)} \|_2^2$$

$$= \operatorname{Proj}_{\mathbb{R}_+^n} \left(x_1^{(r+1)} + y_1^{(r)} \right)$$

$$y^{(r+1)} = y^{(r)} + \left(C x_1^{(r+1)} - D x_2^{(r+1)} - e \right)$$

The pseudocode for the algorithm is given below:

$r \leftarrow 0$; $q^{(0)} \leftarrow \infty$; $s^{(0)} \leftarrow 0$

while $r <$ max_iteration and $\left(q^{(r)} > \epsilon_q \text{ or } s^{(r)} > \epsilon_s \right)$

$$x_1^{(r+1)} \leftarrow 2 \left(P^T P + \frac{\rho}{2} C^T C \right)^{-1} \frac{\rho C^T}{2} \left(y^{(r)} - D x_2^{(r)} - e \right)$$

$$x_2^{(r+1)} \leftarrow \operatorname{Proj}_{\mathbb{R}_+^n} \left(x_1^{(r+1)} + y_1^{(r)} \right)$$

$$y^{(r+1)} \leftarrow y^{(r)} + C x_1^{(r+1)} - D x_2^{(r+1)} - e$$

$$q^{(r+1)} \leftarrow \| C x_1^{(r+1)} - D x_2^{(r+1)} - e \|_2$$

$s^{(r+1)} \leftarrow \| PC^T D(x_2^{(r+1)} - z_2^{(r)}) \|_2$

$r \leftarrow r+1$

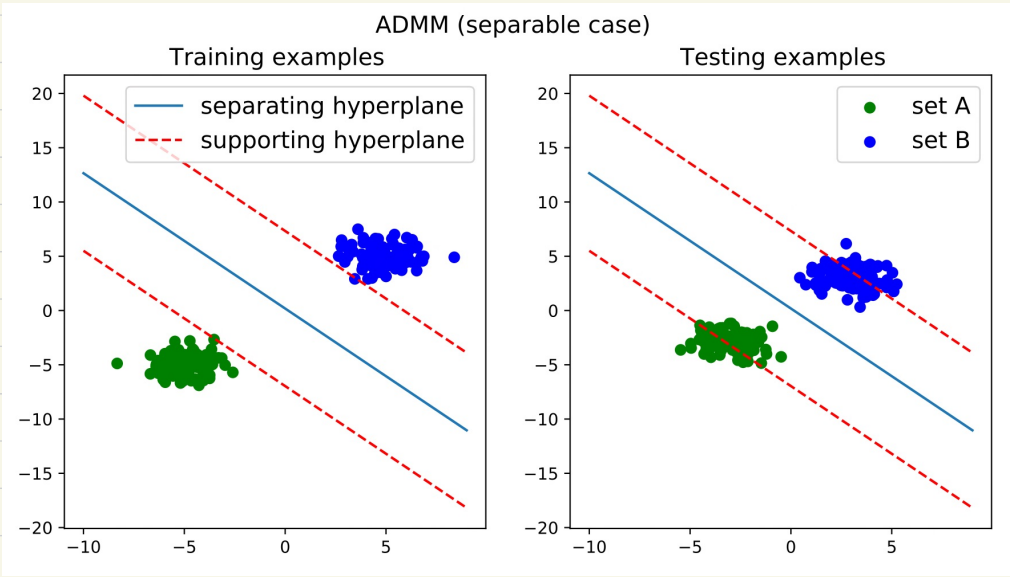
end while

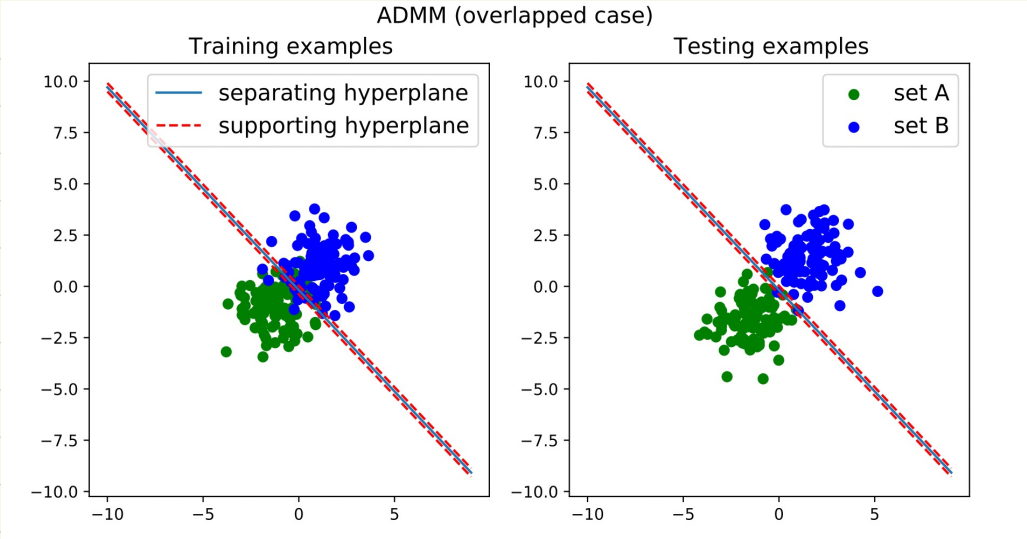
In the above algorithm $q^{(r)}$ and $s^{(r)}$ can be viewed as the residual for primal feasibility and dual feasibility, respectively. Therefore the stopping criterion is composed of maximum iteration and the primal and dual feasibility.

For Reduced C-Hull we can follow the same formulation as above except for the projection operator which can be replaced by projection onto the box constraint, i.e.,

$$\text{Proj}_{[0\ d]}\left(x_i^{(r+1)} + y_i^{(r)}\right)$$

Note that this projection can be obtain by taking entry wise projection of the operant on the segment $[0\ d]$.





Above figures show the result of ADMM for separable and overlapped cases respectively. We use $\epsilon_q = 0.001$, $\epsilon_s = 5$ and $\rho = 1000$. We use Max_Iterations = 1000. We obtain a classification error of 0%. for the separable case. For the overlapped case, with $d = 0.02$, we get a classification error of 1.5%.

Q4. Plot an "iteration v.s. objective value" figure for the training process. Compare all the algorithms that you implemented in this figure. In addition, plot a "time v.s. objective value" figure using all the algorithms. (10%)

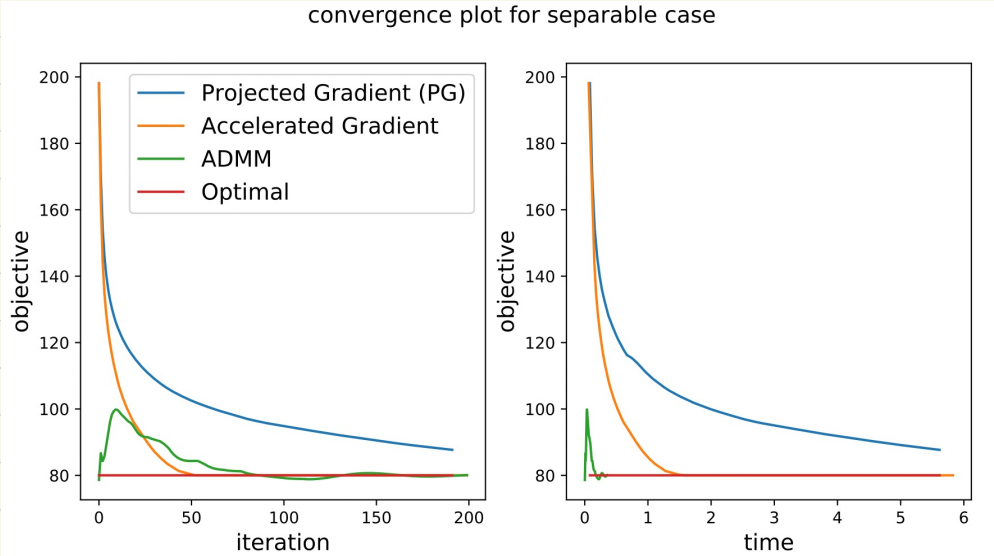
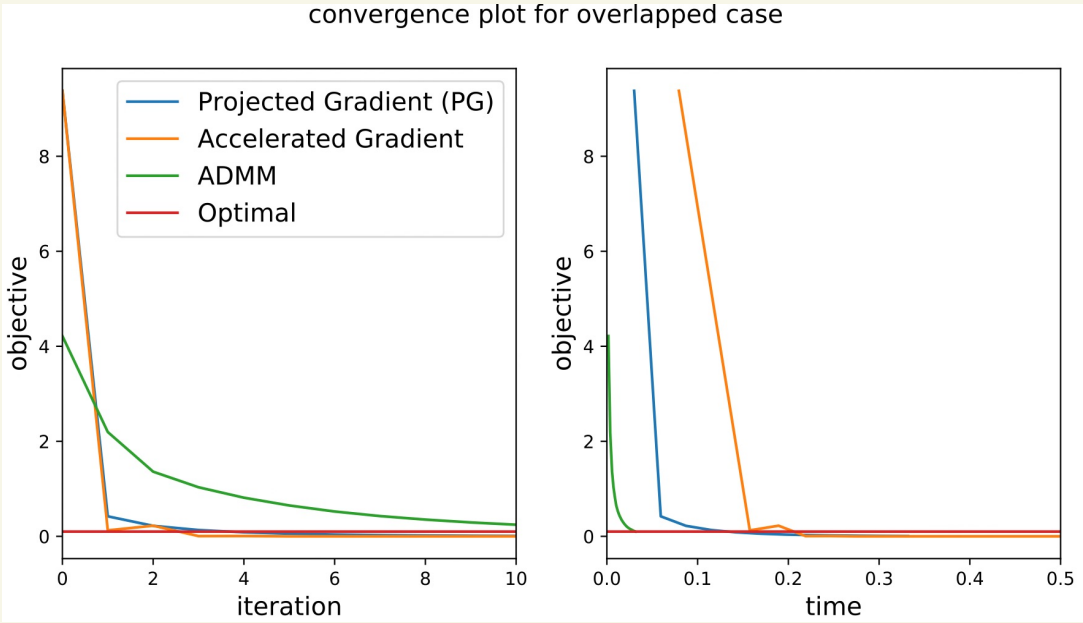


Figure above shows the convergence of different algorithms with respect to time and iteration for the separable case. We can see that the objective value of accelerated projected gradient reaches the optimal the quickest in terms of number of iterations. It also aligns with the convergence rates derived in class, i.e. $O\left(\frac{1}{\gamma^2}\right)$ for Nesterov's method and $O\left(\frac{1}{\gamma}\right)$ for vanilla projected gradient descent. Also ADMM is not monotonic because not all iterates are feasible. However, when we look at the time consumption, we see that ADMM is the fastest to approach the optimal.



Above figure shows the objective vs. iteration and time attained by all algorithms for the overlapped case. Since the solution is obtained in a couple of iterations by Projected Gradient and accelerated projected gradient, comparing the convergence rate in this case does not look meaningful. Nonetheless, in terms of iterations, Accelerated projected gradient is the quickest to converge to optimal. In terms of time, ADMM reaches the optimal the quickest.