Subspace Frank-Wolfe Optimization

Akanksha Makkar
160062

Mohammad Areeb
160403

Mentor- Prof. Ketan Rajawat

Optimization for Big Data (EE698U)

Contents

1 Abstract

We present a stochastic descent algorithm that applies to constrained optimization and is particularly efficient when the objective function is slow to evaluate and gradients are not easily obtained, as in some PDE-constrained optimization and machine learning problems. The basic algorithm projects the gradient onto a random subspace at each iteration, similar to coordinate descent but without restricting directional derivatives to be along the axes and apply Frank-Wolfe using that gradient direction. We provide proofs of convergence under convexity assumption and show favorable results when compared to subspace gradient descent on different constraints.

2 Motivation

In some cases of optimizing a strongly convex function on a constrained set χ, Frank-Wolfe works faster than corresponding Projected Gradient Descent, e.g. when Projection involves a "Bisection" step, etc. In these cases, if we are given $\mathbf{P}^\top \nabla f(\mathbf{x})$ instead of $\nabla f(\mathbf{x})$ as the oracle, we can use below provided Subspace version of Frank-Wolfe to reach the optimum.

In such cases, if $\mathbf{P}$ is a $d \times 2$ matrix, then the subspace is 2 dimensional, and we have to do Frank-Wolfe in 2-D, which can be quite easy.

2.1 Intuition

In Frank-Wolfe Algorithm, we have $\mathbf{y}_t = \arg\min_{\mathbf{y}\in\chi} \langle \nabla f(\mathbf{x}_t), \mathbf{y}\rangle$
Here, we are trying to do the same operation in a projected sub-space, i.e.

$$\mathbf{u}_t = \operatorname*{arg\,min}_{\mathbf{u}\in\mathbf{P}_t^\top\chi} \langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{u}\rangle$$

where $\mathbf{u}_t = \mathbf{P}_t^\top \mathbf{v}_t$ for some $\mathbf{v}_t \in \chi$

$$\begin{aligned}
\mathbf{u}_t &= \operatorname*{arg\,min}_{\mathbf{v}\in\mathbf{P}_t^\top\chi} \langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{u}\rangle \\
&= \mathbf{P}_t^\top \left(\operatorname*{arg\,min}_{\mathbf{v}\in\chi} \langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{P}_t^\top \mathbf{v}\rangle \right) \\
&= \mathbf{P}_t^\top \left(\operatorname*{arg\,min}_{\mathbf{v}\in\chi} \left(\nabla f(\mathbf{x}_t)^\top \mathbf{P}_t \mathbf{P}_t^\top \mathbf{v} \right) \right) \\
&= \mathbf{P}_t^\top \left(\operatorname*{arg\,min}_{\mathbf{v}\in\chi} \langle \mathbf{P}_t \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{v}\rangle \right)
\end{aligned}$$

$\Rightarrow \mathbf{v}_t = \arg\min_{\mathbf{v}\in\chi} \langle \mathbf{P}_t \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{v}\rangle$

$\therefore$ $\mathbf{v}_t$ is the direction that minimizes inner product with $\mathbf{P}_t \mathbf{P}_t^\top \nabla f(\mathbf{x}_t)$ and $\mathbb{E}\left[\mathbf{P}_t \mathbf{P}_t^\top \nabla f(\mathbf{x}_t)\right] = \nabla f(\mathbf{x}_t)$

Hence, $\mathbf{u}_t$ is the descent direction in the sub-space, and $\mathbf{y}_t$ is a good estimator of descent direction in original space.

This gives our update equation as $\mathbf{y}_t = \mathbf{P}_t \mathbf{u}_t$, and we use this $\mathbf{y}_t$ for the Frank-Wolfe update in original space i.e. $\mathbf{x}_{t+1} = (1-\alpha_t)\mathbf{x}_t + \alpha_t \mathbf{y}_t$

3 Introduction

We consider optimization problems of the form

$$\min_{\mathbf{x}\in\chi} f(\mathbf{x}) \quad \text{or} \quad \arg\min_{\mathbf{x}\in\chi} f(\mathbf{x}) \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ has a $\mathcal{L}$-Lipschitz derivative. We also consider additional restriction of convexity of f and compactness of set χ. In the most basic form discussed in Section 2.1, we project the gradient onto a random ℓ-dimensional subspace (χ_2) and descend along that subspace as in Frank-Wolfe. We use the following iteration scheme.

1. Find a vector v_t in χ_2 that minimizes its inner product with g_t (gradient in the subspace)
2. Project that vector into original d-dimensional space
3. Use the projected vector for Frank-Wolfe iteration in original d-dimensional space

3.1 Related Work

3.1.1 Subspace Descent Schemes

The simplest subspace algorithm uses the following scheme:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \mathbf{P}_k \mathbf{P}_k^\top \nabla f(\mathbf{x}_k)$$

where matrix $\mathbf{P}_k$ has properties defined in section 4.1

This algorithm has convergence properties consistent with gradient descent.

Here, we note that $\mathbb{E}[\mathbf{P}_k \mathbf{P}_k^\top \nabla f(\mathbf{x}_k)] = \nabla f(\mathbf{x}_k)$, hence the iterate uses an unbiased estimator of the gradient.

3.1.2 Sketched Gradient Descent

SEGA is a randomized first order optimization method which, in each iteration, updates the current estimate of the gradient through a sketch-and-project operation using the information provided by the latest sketch, and subsequently uses it to compute an unbiased estimate of the true gradient through a random relaxation procedure. the gradient.

$$\mathbf{h}_{k+1} = \arg\min \|\mathbf{h} - \mathbf{h}_k\|_2^2 \quad \text{subject to} \quad \mathbf{P}_k^\top \mathbf{h}_{k+1} = \mathbf{P}_k^\top \nabla f(\mathbf{x}_k)$$

3.1.3 Co-ordinate Descent Schemes

The simplest variant of subspace descent is a deterministic method that cycles over the co-ordinates.

This method is popular because many problems have structure that makes a co-ordinate update very cheap. Choosing the co-ordinates in an appropriate manner can lead to results on par with gradient descent.

They have the following iterate:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha e_i e_i^\top \nabla f(\mathbf{x}_k)$$

where e_i are standard basis vectors of $\mathbb{R}^d$ and i is chosen randomly at each iterate.

4 Problem Formulation:

To minimize $f(\mathbf{x}), \mathbf{x} \in \mathbb{R}^d$ over the compact set χ, when given Oracle is $\mathbf{P}^\top \nabla f(\mathbf{x})$

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\chi} f(\mathbf{x})$$

4.1 Assumptions:

(A1) The function $f(\mathbf{x})$ is $\mathcal{L}$-Smooth.
(A2) The set χ is closed, convex.

Like subspace descent, given oracle is $\mathbf{P}^\top \nabla f(\mathbf{x})$.

$\mathbf{P} \in \mathbb{R}^{d\times l}$ for some $\ell \ll d$ is a random matrix which maps the gradient to an ℓ-dimensional subspace.

It has the following properties:-

(A3) $\mathbb{E}\left[\mathbf{P}\mathbf{P}^\top\right] = \mathbf{I}_d$
(A4) $\mathbf{P}^\top\mathbf{P} = \left(\frac{d}{l}\right)\mathbf{I}_l$

5 Proposed Algorithm:

5.1 Original Frank-Wolfe Algorithm in d-dimensional space

1. $\mathbf{y}_t = \arg\min_{\mathbf{y}\in\chi}\langle\nabla f(\mathbf{x}_t), \mathbf{y}\rangle$
2. $\mathbf{x}_{t+1} = (1-\alpha_t)\mathbf{x}_t + \alpha_t\mathbf{y}_t$

5.2 Subspace Frank-Wolfe

Now, we have a constrained subset $\chi_2 = \{\mathbf{v}|\mathbf{v} = \mathbf{P}^\top\mathbf{x}; \mathbf{x}\in\chi\}$

1. $\mathbf{u}_t = \arg\min_{\mathbf{u}\in\chi_2}\langle\mathbf{P}_t^\top\nabla f(\mathbf{x}_t), \mathbf{u}\rangle$
2. $\mathbf{y}_t = \mathbf{P}_t\mathbf{u}_t$
3. $\mathbf{x}_{t+1} = (1-\alpha_t)\mathbf{x}_t + \alpha_t\mathbf{y}_t$

Step 1:

Find a vector $\mathbf{u}_t \in \chi_2$ which minimizes value of inner product with $\mathbf{P}_t^\top\nabla f(\mathbf{x}_t)$.
This is analogous to finding a $\mathbf{y}_t$ from Step 1 of original Frank-Wolfe, which minimizes inner product with entire d-dimensional gradient $\nabla f(\mathbf{x})$, only in a ℓ-dimensional subspace.

Step 2:

Project the above found $\mathbf{u}_t$ into the original space to get $\mathbf{y}_t$

Step 3:

Perform the known Frank-Wolfe update in the original space using above found $\mathbf{y}_t$.

6 Proof of Convergence for subspace Frank-Wolfe:

Define the diameter of convex set χ as $\mathcal{D}$ i.e.

$$\|\mathbf{x}-\mathbf{y}\| \leq \mathcal{D}, \quad \forall \quad \mathbf{x},\mathbf{y} \in \chi$$

Also, Frank-Wolfe update gives $\mathbf{x}_{t+1} - \mathbf{x}_t = \eta_t(\mathbf{y}_t - \mathbf{x}_t)$

We begin by using the Q.U.B property to obtain

$$\begin{aligned}
f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) &\leq \langle \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}_t \rangle + \frac{L}{2}\left\|\mathbf{x}_{t+1} - \mathbf{x}_t\right\|^2 \\
&\leq \langle \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}_t \rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2 \\
&= \eta_t\langle \nabla f(\mathbf{x}_t), \mathbf{y}_t - \mathbf{x}_t \rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2 \\
&= \eta_t\langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{u}_t - \mathbf{x}_t \rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2 \quad \text{(same } \mathbf{u}_t \text{ as mentioned in Intuition)}
\end{aligned}$$

$$\boxed{\text{Here, } \mathbf{u}_t = \underset{\mathbf{u}\in\chi_2}{\arg\min}\langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{u}\rangle = \underset{\mathbf{u}\in\chi_2}{\arg\min}\langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{u}\rangle}$$

$$\therefore \langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{u}_t\rangle \leq \langle \mathbf{P}_t^\top \nabla f(\mathbf{x}_t), \mathbf{P}_t^\top\mathbf{x}^*\rangle \Rightarrow \langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{u}_t\rangle \leq \langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{P}_t^\top\mathbf{x}^*\rangle$$

$$\therefore f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \leq \eta_t\langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{P}_t^\top\mathbf{x}^* - \mathbf{x}_t\rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2$$

$$\begin{aligned}
\therefore \mathbb{E}\left[f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t)\right] &\leq \eta_t\mathbb{E}\left[\langle \nabla f(\mathbf{x}_t), \mathbf{P}_t\mathbf{P}_t^\top\mathbf{x}^* - \mathbf{x}_t\rangle\right] + \frac{L}{2}\eta_t^2\mathcal{D}^2 \\
&= \eta_t\langle \nabla f(\mathbf{x}_t), \mathbb{E}\left[\mathbf{P}_t\mathbf{P}_t^\top\right]\mathbf{x}^* - \mathbf{x}_t\rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2 \\
&= \eta_t\langle \nabla f(\mathbf{x}_t), \mathbf{x}^* - \mathbf{x}_t\rangle + \frac{L}{2}\eta_t^2\mathcal{D}^2 \\
&\leq \eta_t(f(\mathbf{x}^*) - f(\mathbf{x}_t)) + \frac{L}{2}\eta_t^2\mathcal{D}^2 \quad \text{(using convexity of } f(\mathbf{x})\text{)}
\end{aligned}$$

$$\therefore \mathbb{E}\left[f(\mathbf{x}_{t+1}) - f(\mathbf{x}^*)\right] \leq (1-\eta_t)\left[f(\mathbf{x}_t) - f(\mathbf{x}^*)\right] + \frac{L}{2}\eta_t^2\mathcal{D}^2$$

Define $\delta_t = \mathbb{E}\left[f(\mathbf{x}_t) - f(\mathbf{x}^*)\right]$

$\therefore$ We have the recursive relation $\delta_{t+1} \leq (1-\eta_t)\delta_t + \frac{L}{2}\eta_t^2\mathcal{D}^2$

Rest of the proof is same as for original Frank-Wolfe algorithm. Using Induction, we prove :

$$\delta_t = \mathbb{E}\left[f(\mathbf{x}_t) - f(\mathbf{x}^*)\right] \leq \frac{2L\mathcal{D}^2}{t+2}$$

$$\therefore \delta_T = \mathbb{E}\left[f(\mathbf{x}_T) - f(\mathbf{x}^*)\right] \leq \frac{2L\mathcal{D}^2}{T+2}$$

Hence, the algorithm converges in $\mathcal{T} = \mathcal{O}\left(\frac{1}{\epsilon}\right)$ iterations.

7 Experimental Results

7.1 Haar Matrix ($\mathbf{P}_t$) Construction:

We use the pseudo-code provided in following algorithm to draw from a scaled Haar-distributed Matrix

---

Algorithm 1: Generate a scaled, Haar distributed matrix as in 3.1.1

Input: ℓ, d
Output: P matrix satisfying *A3* and *A4* of 4.1
Initialize X $\in \mathbb{R}^{d\times l}$
Set $\mathrm{X}_{i,j} \sim \mathcal{N}(0,1)$
Calculate thin QR decomposition of X = QR

Let $\Lambda = \begin{bmatrix} \frac{R_{1,1}}{|R_{1,1}|} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \frac{R_{\ell,\ell}}{|R_{\ell,\ell}|} \end{bmatrix}$

P = $\sqrt{\frac{d}{l}}\mathbf{Q\Lambda}$

7.2 Comparison of Subspace Frank-Wolfe and Subspace Gradient Descent for box constraints

We compared Convergence Rates of Subspace Gradient Descent and Subspace Frank-Wolfe in cases of very high dimensional data.

We tried to optimize the function $f(\mathbf{x}) = \frac{1}{2}\left\|\mathbf{x}\right\|^2$ over the set $\chi = \text{Box}(l_1, l_2)$, $l_1, l_2 \in \mathbb{R}^d$, and compare the convergence of Sub-space Gradient descent and Sub-space Frank-Wolfe Descent. We solved for $\chi = \text{Box}[-3, 3]$, $\epsilon = 10^{-3}$, d=10000, l=10.

7.2.1 Implementation

The Frank-Wolfe version is applied as follows:

Algorithm 2: Subspace Frank-Wolfe with box constraints

---

while $\|\mathbf{x}_{t+1} - \mathbf{x}_t\| \geq \epsilon$
do
1.generate scaled Haar distributed matrix $\mathbf{P}_t$ using algorithm given in Paper 7.1
2.get $\mathbf{P}_t^\top \nabla f(\mathbf{x}_t)$
3.for each $i \in [d]$
if *sign*$\left[\mathbf{P}_t^\top \nabla f(\mathbf{x}_t)\right]_i = 1$ **then**
$\mathbf{u}_t^i = l_1$
else
$\mathbf{u}_t^i = l_2$
end
3.$\mathbf{y}_t = \mathbf{P}_t \mathbf{u}_t$
4.$\mathbf{x}_{t+1} = (1 - \eta_t)\mathbf{x}_t + \eta_t \mathbf{y}_t$
end

7.2.2 Result

Even for constraints involving simple projections like this one, Subspace Frank-Wolfe performs equivalent (maybe even better for some cases) to Subspace Gradient Descent.

7.3 Comparison of Subspace Frank-Wolfe and Subspace Gradient Descent for L-1 norm ball constraint

We compared Convergence Rates of Subspace Gradient Descent and Subspace Frank-Wolfe in cases of very high dimensional data.

We tried to optimize the function $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$ over the set $\chi = \{\mathbf{x} \quad | \quad \|\mathbf{x}\|_1 \leq \alpha, \mathbf{x} \in \mathbb{R}^d\}$, and compare the convergence of Sub-space Gradient descent and Sub-space Frank-Wolfe Descent. We solved for $\alpha = 1$, $\epsilon = 10^{-5}$, d=10000, l=10.

7.3.1 Implementation

The Frank-Wolfe version is applied as follows:

1. Generate scaled Haar distributed matrix $\mathbf{P}_t$ at each iteration using the algorithm mentioned in Section 7.1
2. Get $\mathbf{g}_t = \mathbf{P}_t^\top \nabla f(\mathbf{x}_t)$
3. Find the coordinate i $\in$ [d] for which $|[\mathbf{g}_t]_i|$ is maximum
4. Get unit vector $\mathbf{u}_t = -sign\{[\mathbf{g}_t]_i\}\mathbf{e}_i$, where $\mathbf{e}_i$ is a standard basis vector of $\mathbb{R}^l$
5. Get $\mathbf{y}_t = \mathbf{P}_t\mathbf{u}_t$
6. $\mathbf{x}_{t+1} = (1-\eta_t)\mathbf{x}_t + \eta_t\mathbf{y}_t$
7. Repeat till convergence

Algorithm 3: Subspace Frank-Wolfe with L-1 norm constraint

while $\|\mathbf{x}_{t+1} - \mathbf{x}_t\| \geq \epsilon$
do
1. generate scaled Haar distributed matrix $\mathbf{P}_t$ using algorithm given in Paper 7.1
2. get $\mathbf{P}_t^\top \nabla f(\mathbf{x}_t)$
3. $\mathrm{i}_t = \arg\max |[\mathbf{P}_t^\top \nabla f(\mathbf{x}_t)]_i|$
4. $\mathrm{u}_t = -sign\{[\mathbf{P}_t^\top \nabla f(\mathbf{x}_t)]_{i_t}\}\mathbf{e}_{i_t}$
5. $\mathbf{y}_t = \mathbf{P}_t\mathbf{u}_t$
6. $\mathbf{x}_{t+1} = (1-\eta_t)\mathbf{x}_t + \eta_t\mathbf{y}_t$

end

7.3.2 Result

In this case, the Subspace Gradient Descent Algorithm involves projection onto L-1 norm ball, which involves a Bisection Step, rendering it slow as compared to Subspace Frank-Wolfe which has a simple projection free step for the update.

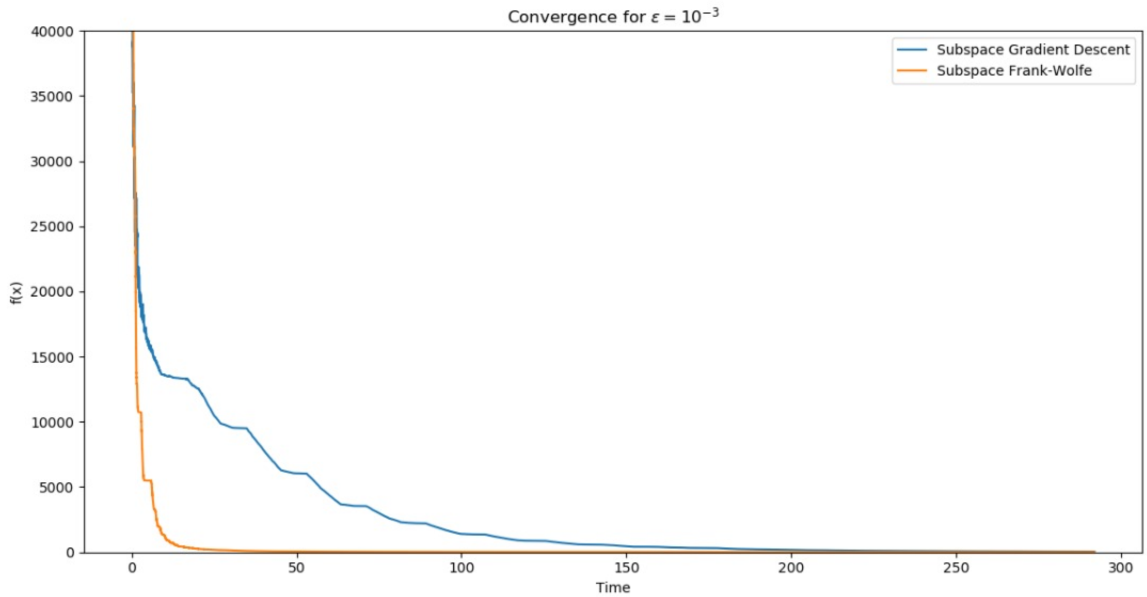

Figure 1: Subspace Gradient Descent v/s Subspace Frank-Wolfe with Box Constraints(Linear Scale)

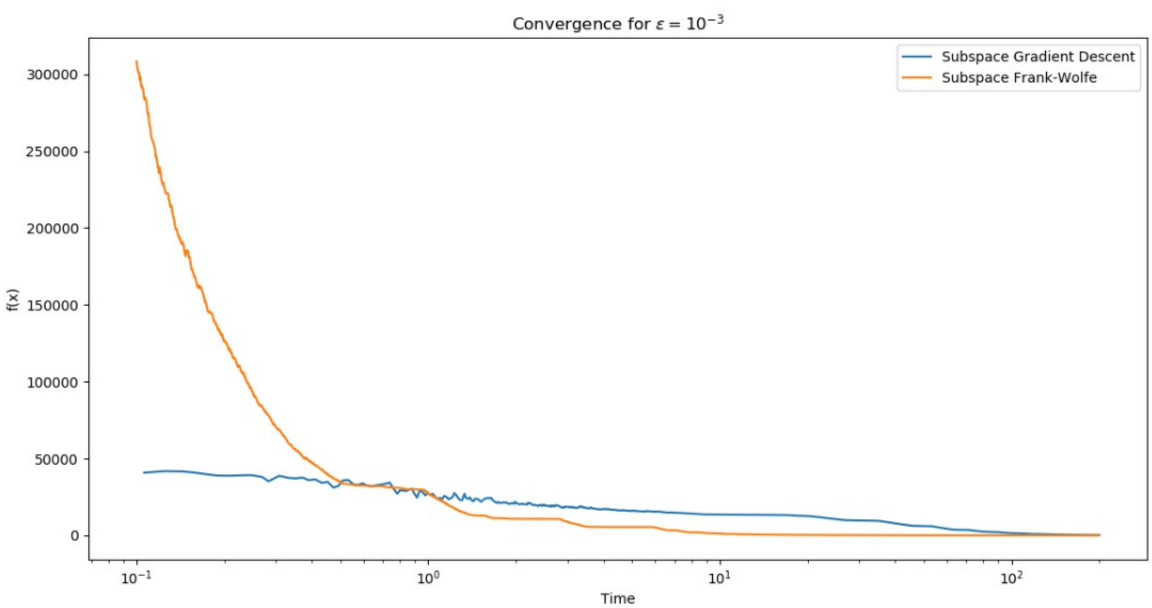

Figure 2: Subspace Gradient Descent v/s Subspace Frank-Wolfe with Box Constraints(Semilog Scale)

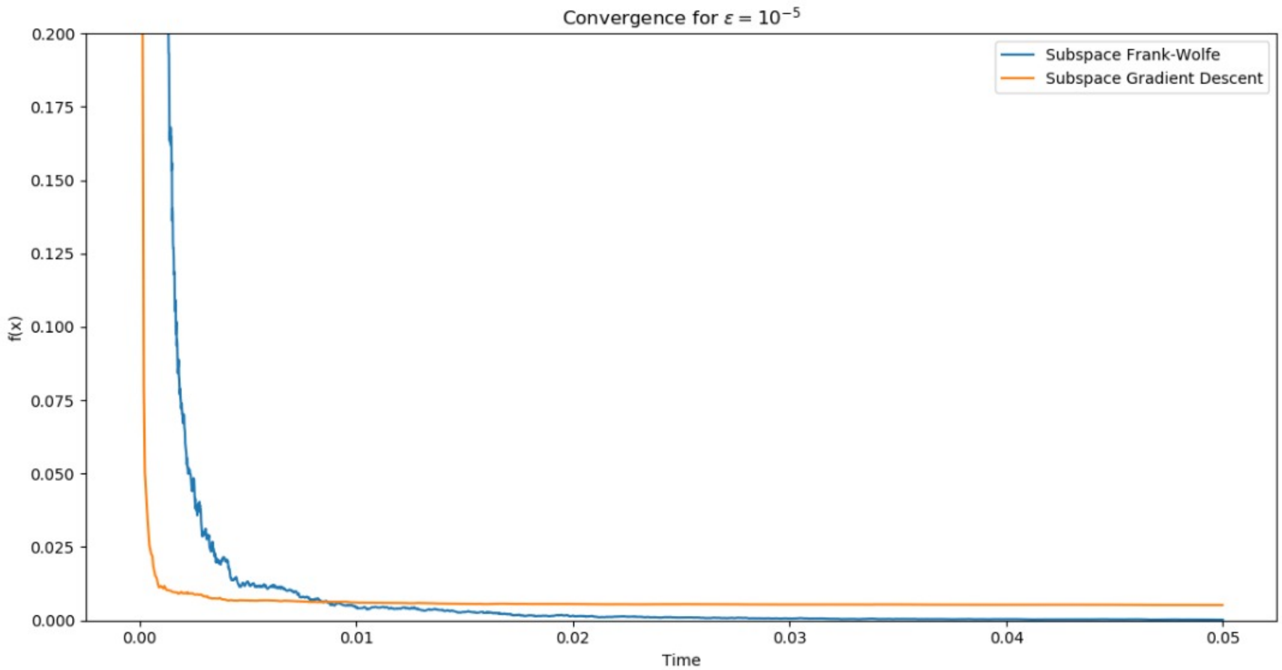

Figure 3: Subspace Gradient Descent v/s Subspace Frank-Wolfe with L-1 Norm Ball Constraints(Linear Scale)

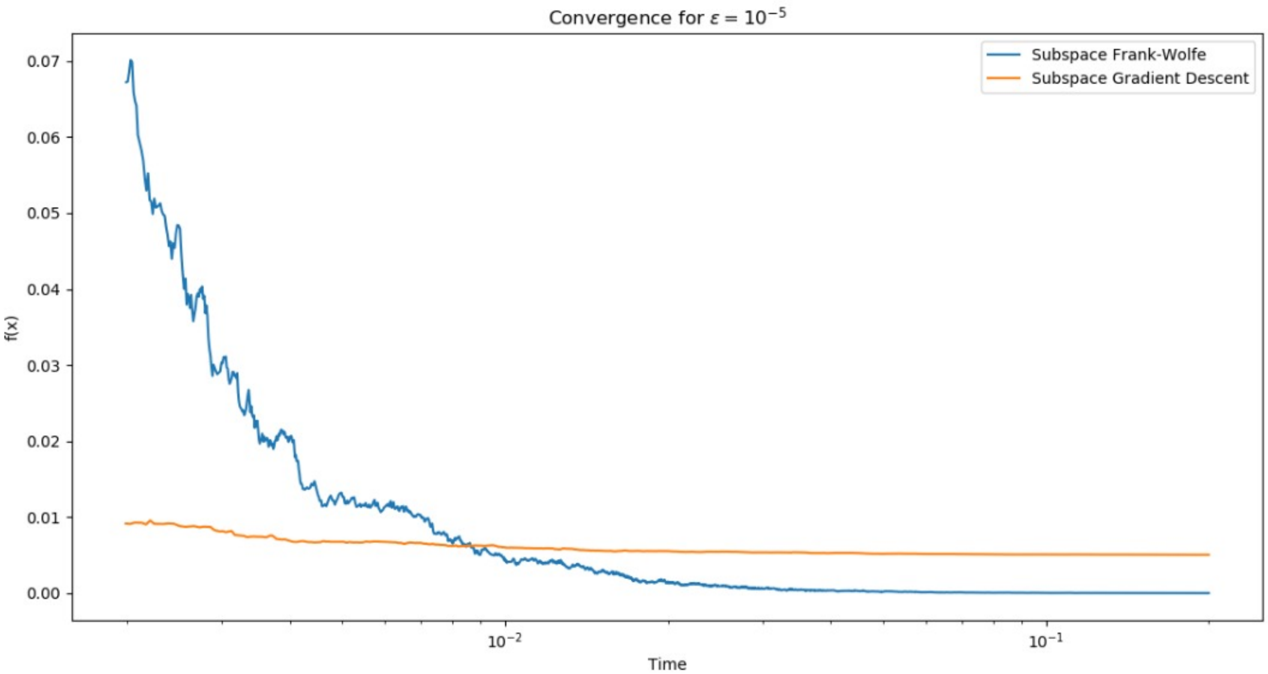

Figure 4: Subspace Gradient Descent v/s Subspace Frank-Wolfe with L-1 Norm Ball Constraints(Semilog Scale)

7.4 Comparison of Subspace Frank-Wolfe and SEGA for L-1 norm ball constraint

We compared Convergence Rates of Subspace Franjk-Wolfe and SEGA (Sketched Gradient) for optimization under L-1 norm ball constraint.

We tried to optimize the function $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$ over the set $\chi = \{\mathbf{x} \quad | \quad \|\mathbf{x}\|_1 \leq \alpha, \mathbf{x} \in \mathbb{R}^d\}$, We solved for $\alpha = 1$, $\epsilon = 10^{-4}$, $d = 100$, $l = 3$

7.4.1 Implementation

Subspace Frank-Wolfe updates are same as defined in Algorithm 3

SEGA tries to find $\mathbf{h}_{t+1} = \arg\min\|\mathbf{h} - \mathbf{h}_t\|_2^2$ subject to $\mathbf{P}^\top \mathbf{h}_{t+1} = \mathbf{P}^\top \nabla f(\mathbf{x})$
SEGA uses the following closed form update:

Algorithm 4: SEGA with L-1 norm constraint

Initialize x_0, h_0
while $\|\mathbf{x}_{t+1} - \mathbf{x}_t\| \geq \epsilon$
do

1. $\mathrm{x}_{t+1} = \mathbb{P}_\chi\{\mathbf{x}_t - \alpha \mathbf{h}_t\}$
2. $\mathrm{h}_{t+1} = (\mathbf{I} - \mathbb{Z}_t)\mathbf{h_t} + \mathbb{Z}_t \nabla f(\mathbf{x}_t)$, where $\mathbb{Z}_t = \left(\frac{l}{d}\right) \mathbf{P}_t \mathbf{P}_t^\top$

end

7.4.2 Result

SEGA and Subspace Frank-Wolfe have similar convergence rate.

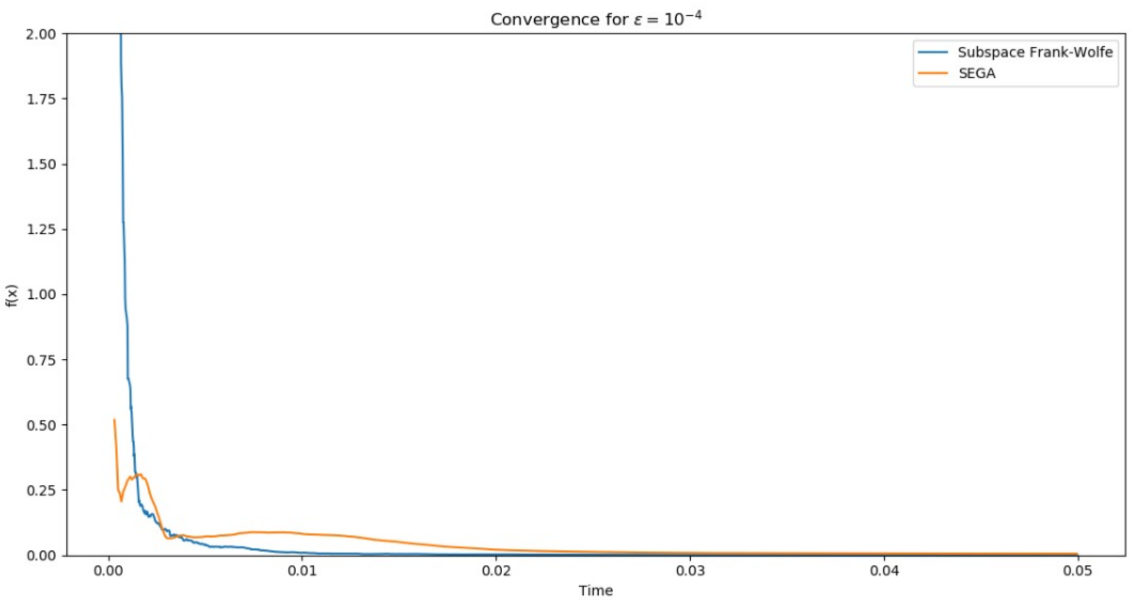

Figure 5: Subspace Frank-Wolfe v/s SEGA for L-1 norm Ball Constraint(Linear Scale)

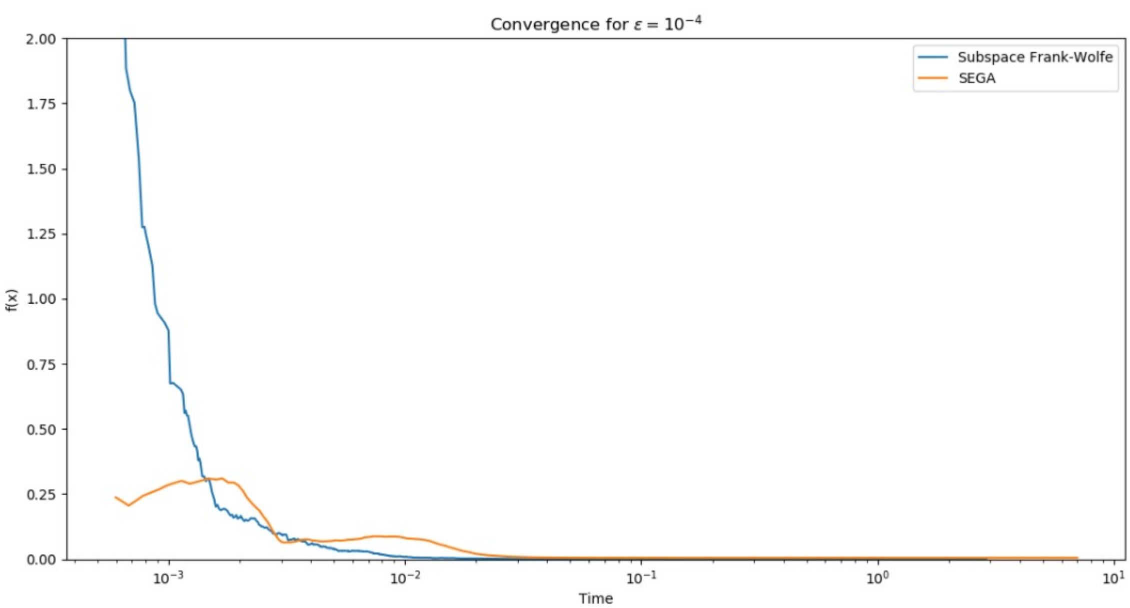

Figure 6: Subspace Frank-Wolfe v/s SEGA for L-1 norm Ball Constraint(Semilog Scale)

7.5 Comparison of Subspace Frank-Wolfe and Co-ordinate Descent for L-1 norm ball constraint

We compared Convergence Rates of Subspace Frank-Wolfe and Co-ordinate Descent for optimization under L-1 norm ball constraint.

We tried to optimize the function $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$ over the set $\chi = \{\mathbf{x} \quad | \quad \|\mathbf{x}\|_1 \leq \alpha, \mathbf{x} \in \mathbb{R}^d\}$,
We solved for $\alpha = 10$, $\epsilon = 10^{-4}$, d=100, l=1 (to compare with co-ordinate descent)

7.5.1 Implementation

Subspace Frank-Wolfe updates are same as defined in Algorithm 3

Co-ordinate Descent chooses a random $i \in [d]$ and maximizes along that co-ordinate using the following algorithm

Algorithm 5: Co-ordinate Descent with L-1 norm constraint

Initialize $y_0 = \mathbf{x}_0$
while $\|\mathbf{x}_{t+1} - \mathbf{x}_t\| \geq \epsilon$
do
1. $y_{t+1}^i = \mathbf{x}_t^i - \eta_t[\nabla f(\mathbf{x}_t)]_i$
2. $x_{t+1} = \mathbb{P}_\chi\{\mathbf{y}_{t+1}\}$

end

7.5.2 Result

Subspace Frank-Wolfe performs poorly as compared to Co-ordinate Descent.

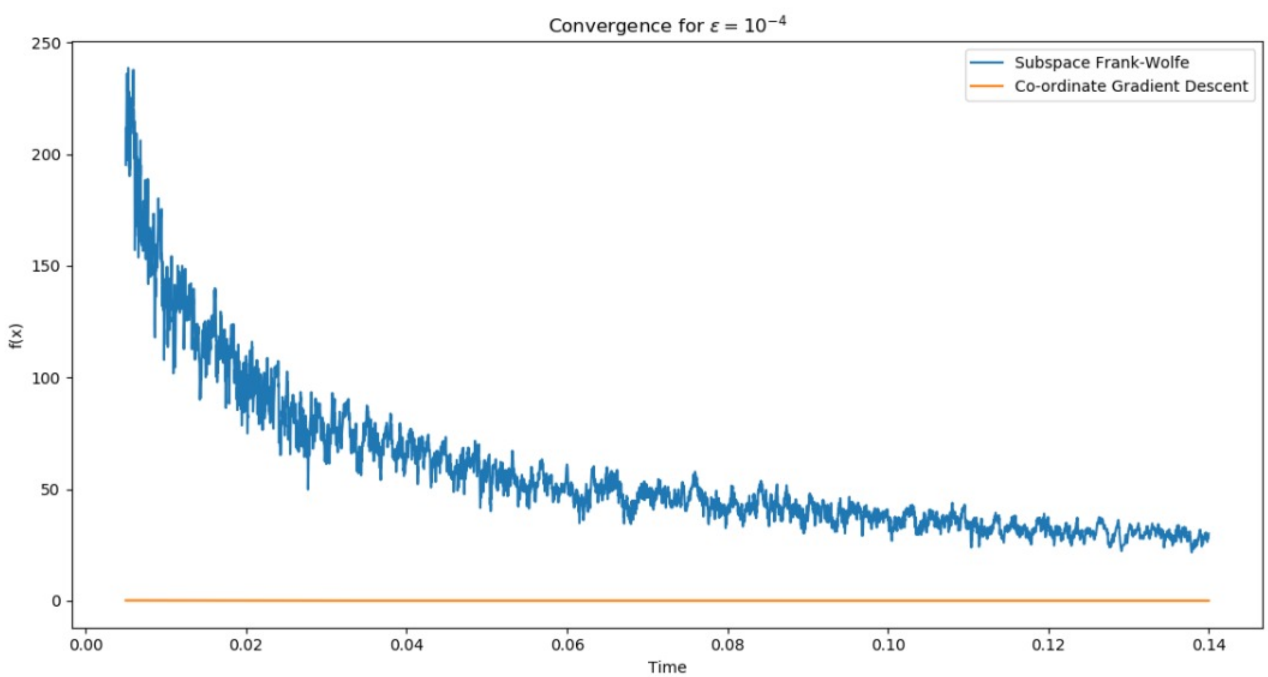

Figure 7: Subspace Frank-Wolfe v/s Co-ordinate Descent with L-1 Norm Ball Constraint (linear scale)

8 Conclusion

We have proposed a randomized optimization algorithm that offers advantage in terms of time when given oracle is $\mathbf{P}^\top \nabla f(\mathbf{x})$. This algorithm has same iteration complexity as Subspace Gradient Descent.

Table 1: Iteration Complexities of Used Algorithms

Algorithm	Iteration Complexity
Sub-space Frank-Wolfe	$\mathcal{O}\left(\frac{1}{\epsilon}\right)$
Sub-space Gradient Descent	$\mathcal{O}\left(\frac{1}{\epsilon}\right)$
Sketched Gradient(SEGA)	$\mathcal{O}\left(\log\frac{1}{\epsilon}\right)$
Co-ordinate Gradient Descent	$\mathcal{O}\left(\log\frac{1}{\epsilon}\right)$

However, proposed algorithm works faster in terms of time as compared to Subspace Gradient Descent and SEGA, but is slower than Co-ordinate Gradient Descent.

Following plot shows the comparison of convergence in time of the above used algorithms on semilog scale.

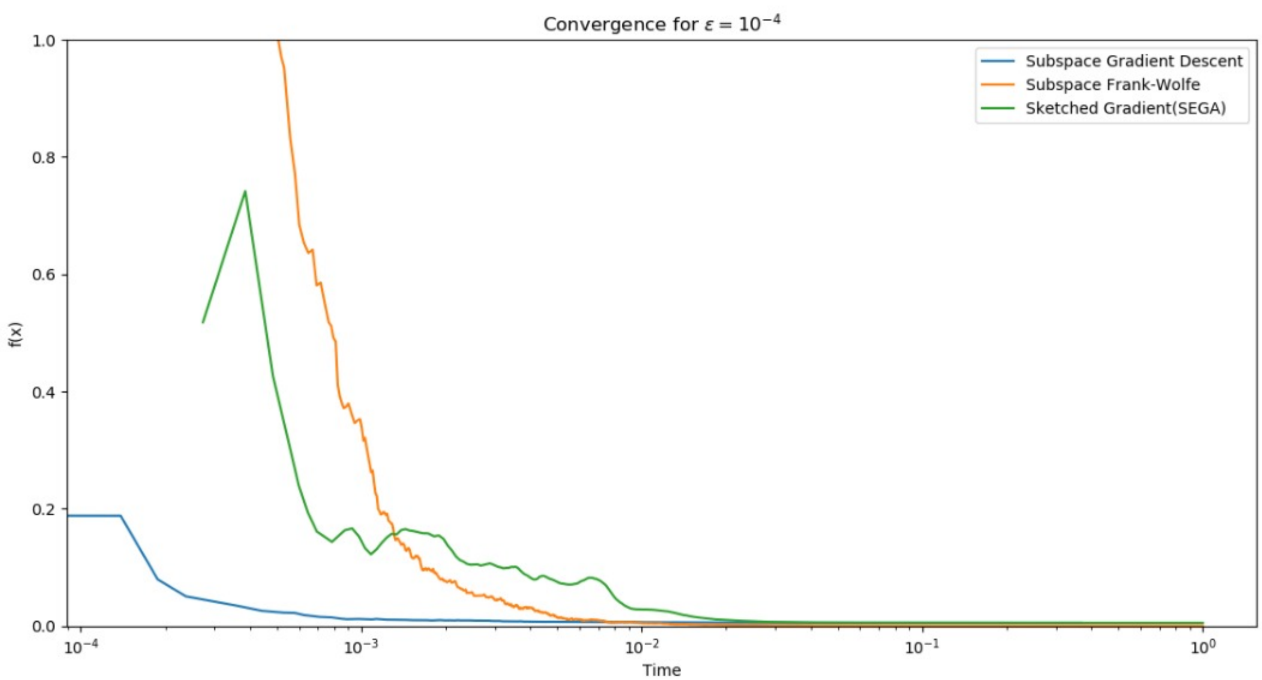

Figure 8: Comparison of all used algorithms

References

[1] David Kozak, Stephen Becker, Alizera Doostan, Luis Tenorio. Stochatic Subspace Descent

[2] Filip Hanzely, Konstantin Mishchenko, Peter Richtarik. SEGA: Variance Reduction via Gradient Sketching*

[3] Lijun Ding, Madeleine Udell. Frank-Wolfe Style Algorithms for Large Scale Optimization