Robust ML Training with Conditional Gradients

Sebastian Pokutta

Technische Universität Berlin
and
Zuse Institute Berlin

pokutta@math.tu-berlin.de
@spokutta

CO@Work 2020 Summer School
September, 2020

Berlin Mathematics Research Center
MATH+

Opportunities in Berlin

Shameless plug

Postdoc and PhD positions in **optimization/ML.**

At **Zuse Institute Berlin** and **TU Berlin**.

What is this talk about?

Introduction

Can we train, e.g., Neural Networks so that they are (more) robust to noise and adversarial attacks?

What is this talk about?

Introduction

Can we train, e.g., Neural Networks so that they are (more) robust to noise and adversarial attacks?

Outline

- A simple example
- The basic setup of supervised Machine Learning
- Stochastic Gradient Descent
- Stochastic Conditional Gradient Descent

(Hyperlinked) References are not exhaustive; check references contained therein.

Statements are simplified for the sake of exposition.

Supervised Machine Learning and ERM

A simple example

Consider the following simple learning problem, a.k.a. linear regression:

Given:

Set of points $X \doteq \{x_1, \ldots, x_k\} \subseteq \mathbb{R}^n$

Vector $y \doteq (y_1, \ldots, y_k) \in \mathbb{R}^k$

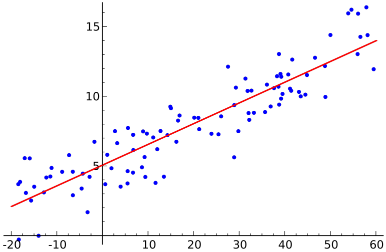
Find:

Linear function $\theta \in \mathbb{R}^n$, such that

$$x_i\theta \approx y_i \qquad \forall i \in [k],$$

or in matrix form:

$$X\theta \approx y.$$



[Wikipedia]

Supervised Machine Learning and ERM

A simple example

Consider the following simple learning problem, a.k.a. linear regression:

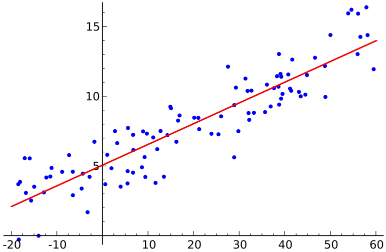
Given:
Set of points $X \doteq \{x_1, \ldots, x_k\} \subseteq \mathbb{R}^n$
Vector $y \doteq (y_1, \ldots, y_k) \in \mathbb{R}^k$

Find:
Linear function $\theta \in \mathbb{R}^n$, such that

$$x_i \theta \approx y_i \qquad \forall i \in [k],$$

or in matrix form:

$$X\theta \approx y.$$



[Wikipedia]

The search for the best θ can be naturally cast as an optimization problem:

$$\min_{\theta} \sum_{i \in [k]} |x_i \theta - y_i|^2 = \min_{\theta} \|X\theta - y\|_2^2 \qquad \text{(linReg)}$$

Supervised Machine Learning and ERM

Empirical Risk Minimization

More generally, interested in the Empirical Risk Minimization problem:

$$\min_{\theta} L(\theta) \doteq \min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \ell(f(x, \theta), y). \tag{ERM}$$

Supervised Machine Learning and ERM

Empirical Risk Minimization

More generally, interested in the Empirical Risk Minimization problem:

$$\min_{\theta} L(\theta) \doteq \min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y). \tag{ERM}$$

The ERM approximates the General Risk Minimization problem:

$$\min_{\theta} \widehat{L}(\theta) \doteq \min_{\theta} \mathbb{E}_{(x,y)\in\widehat{\mathcal{D}}}\, \ell(f(x,\theta),y). \tag{GRM}$$

Supervised Machine Learning and ERM

Empirical Risk Minimization

More generally, interested in the Empirical Risk Minimization problem:

$$\min_\theta L(\theta) \doteq \min_\theta \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y). \tag{ERM}$$

The ERM approximates the General Risk Minimization problem:

$$\min_\theta \widehat{L}(\theta) \doteq \min_\theta \mathbb{E}_{(x,y)\in\widehat{\mathcal{D}}}\, \ell(f(x,\theta),y). \tag{GRM}$$

Note: If $\mathcal{D}$ is chosen large enough, under relatively mild assumptions, a solution to (ERM) is a good approximation to a solution to (GRM):

$$\widehat{L}(\theta) \leq L(\theta) + \sqrt{\frac{\log|\Theta| + \log\frac{1}{\delta}}{|\mathcal{D}|}},$$

with probability $1-\delta$. This bound is typically very loose.

[e.g., Suriya Gunasekar' lecture notes] [The Elements of Statistical Learning, Hastie et al]

Supervised Machine Learning and ERM

Empirical Risk Minimization: Examples

1. Linear Regression
   $\ell(z_i, y_i) \doteq |z_i - y_i|^2$ and $z_i = f(\theta, x_i) \doteq x_i \theta$

Supervised Machine Learning and ERM

Empirical Risk Minimization: Examples

1. Linear Regression
   $\ell(z_i, y_i) \doteq |z_i - y_i|^2$ and $z_i = f(\theta, x_i) \doteq x_i \theta$

2. Classification / Logistic Regression over classes C
   $\ell(z_i, y_i) \doteq -\sum_{c \in [C]} y_{i,c} \log z_{i,c}$ and, e.g., $z_i = f(\theta, x_i) \doteq x_i \theta$ (or a neural network)

Supervised Machine Learning and ERM

Empirical Risk Minimization: Examples

1. Linear Regression
   $\ell(z_i, y_i) \doteq |z_i - y_i|^2$ and $z_i = f(\theta, x_i) \doteq x_i\theta$
2. Classification / Logistic Regression over classes C
   $\ell(z_i, y_i) \doteq -\sum_{c \in [C]} y_{i,c} \log z_{i,c}$ and, e.g., $z_i = f(\theta, x_i) \doteq x_i\theta$ (or a neural network)
3. Support Vector Machines
   $\ell(z_i, y_i) \doteq y_i \max(0, 1 - z_i) + (1 - y_i) \max(0, 1 + z_i)$ and $z_i = f(\theta, x_i) \doteq x_i\theta$

Supervised Machine Learning and ERM

Empirical Risk Minimization: Examples

1. Linear Regression
   $\ell(z_i, y_i) \doteq |z_i - y_i|^2$ and $z_i = f(\theta, x_i) \doteq x_i\theta$

2. Classification / Logistic Regression over classes C
   $\ell(z_i, y_i) \doteq -\sum_{c\in[C]} y_{i,c} \log z_{i,c}$ and, e.g., $z_i = f(\theta, x_i) \doteq x_i\theta$ (or a neural network)

3. Support Vector Machines
   $\ell(z_i, y_i) \doteq y_i \max(0, 1 - z_i) + (1 - y_i) \max(0, 1 + z_i)$ and $z_i = f(\theta, x_i) \doteq x_i\theta$

4. Neural Networks
   $\ell(z_i, y_i)$ some loss function and $z_i = f(\theta, x_i)$ neural network with weights θ

...and many more choices and combinations possible.

Optimizing the ERM Problem
Stochastic Gradient Descent

How to solve Problem (ERM)?

Optimizing the ERM Problem
Stochastic Gradient Descent

How to solve Problem (ERM)?

Simple idea: Gradient Descent

[see blog for background on conv opt]

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla L(\theta_t) \tag{GD}$$

Optimizing the ERM Problem

Stochastic Gradient Descent

How to solve Problem (ERM)?

Simple idea: Gradient Descent [see blog for background on conv opt]

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla L(\theta_t) \tag{GD}$$

Unfortunately, this might be too expensive if (ERM) has a lot of summands.

Optimizing the ERM Problem

Stochastic Gradient Descent

How to solve Problem (ERM)?

Simple idea: Gradient Descent [see blog for background on conv opt]

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla L(\theta_t) \tag{GD}$$

Unfortunately, this might be too expensive if (ERM) has a lot of summands.

However, reexamine:

$$\nabla L(\theta) = \nabla \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \ell(f(x,\theta), y) = \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \nabla \ell(f(x,\theta), y), \tag{ERMgrad}$$

Optimizing the ERM Problem
Stochastic Gradient Descent

How to solve Problem (ERM)?

Simple idea: Gradient Descent [see blog for background on conv opt]

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla L(\theta_t) \tag{GD}$$

Unfortunately, this might be too expensive if (ERM) has a lot of summands.

However, reexamine:

$$\nabla L(\theta) = \nabla \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \nabla \ell(f(x,\theta),y), \tag{ERMgrad}$$

Thus if we sample $(x,y) \in \mathcal{D}$ uniformly at random, then

$$\nabla L(\theta) = \mathbb{E}_{(x,y)\in\mathcal{D}} \nabla \ell(f(x,\theta),y) \tag{gradEst}$$

Optimizing the ERM Problem

Stochastic Gradient Descent

This leads to Stochastic Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(f(x, \theta_t), y) \qquad \text{with } (x, y) \sim \mathcal{D}, \tag{SGD}$$

one of the most-used algorithm for ML training (together with its many variants).

Optimizing the ERM Problem

Stochastic Gradient Descent

This leads to Stochastic Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(f(x, \theta_t), y) \qquad \text{with } (x, y) \sim \mathcal{D}, \tag{SGD}$$

one of the most-used algorithm for ML training (together with its many variants).

Typical variants include

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.

Optimizing the ERM Problem

Stochastic Gradient Descent

This leads to Stochastic Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(f(x, \theta_t), y) \qquad \text{with } (x, y) \sim \mathcal{D}, \tag{SGD}$$

one of the most-used algorithm for ML training (together with its many variants).

Typical variants include

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.
- Learning rate schedules. To ensure convergence the learning rate η is dynamically managed.

Optimizing the ERM Problem

Stochastic Gradient Descent

This leads to Stochastic Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(f(x, \theta_t), y) \qquad \text{with } (x, y) \sim \mathcal{D}, \tag{SGD}$$

one of the most-used algorithm for ML training (together with its many variants).

Typical variants include

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.
- Learning rate schedules. To ensure convergence the learning rate η is dynamically managed.
- Adaptive Variants and Momentum. RMSProp, Adagrad, Adadelta, Adam, …

This leads to Stochastic Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla \ell(f(x, \theta_t), y) \qquad \text{with } (x, y) \sim \mathcal{D}, \tag{SGD}$$

one of the most-used algorithm for ML training (together with its many variants).

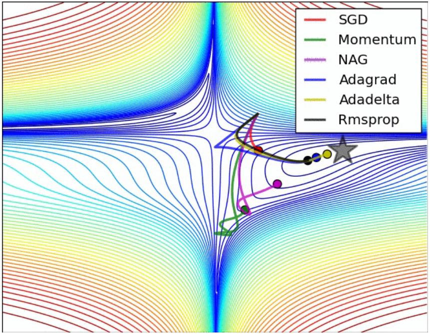
Typical variants include

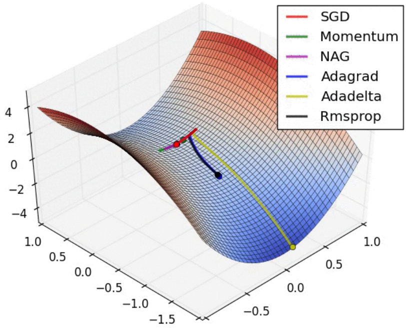
- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.
- Learning rate schedules. To ensure convergence the learning rate η is dynamically managed.
- Adaptive Variants and Momentum. RMSProp, Adagrad, Adadelta, Adam, ...
- Variance Reduction. Compute exact gradient once in a while as reference point, e.g., SVRG.

[for an overview of variants: blog of Sebastian Ruder]

A comparison between different variants

Stochastic Gradient Descent





[Graphics from blog of Sebastian Ruder; see also for animations]

(More) robust ERM training

Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y).$$

(More) robust ERM training

Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y).$$

In the standard formulation θ is unbounded and can get quite large.

(More) robust ERM training

Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \ell(f(x,\theta), y).$$

In the standard formulation θ is unbounded and can get quite large.

Problem. Large θ for, e.g., Neural Networks lead to large Lipschitz constants. Trained network becomes sensitive to input noise and perturbations. [Tsuzuku, Sato, Sugiyama, 2018]

(More) robust ERM training

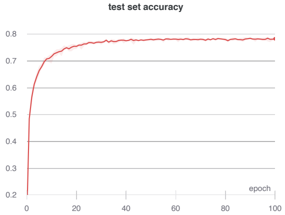
Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y).$$

In the standard formulation θ is unbounded and can get quite large.

Problem. Large θ for, e.g., Neural Networks lead to large Lipschitz constants. Trained network becomes sensitive to input noise and perturbations. [Tsuzuku, Sato, Sugiyama, 2018]



Performance for Neural Network trained on MNIST.

(More) robust ERM training

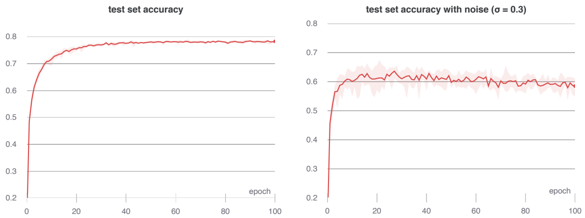
Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y).$$

In the standard formulation θ is unbounded and can get quite large.

Problem. Large θ for, e.g., Neural Networks lead to large Lipschitz constants. Trained network becomes sensitive to input noise and perturbations. [Tsuzuku, Sato, Sugiyama, 2018]



Performance for Neural Network trained on MNIST.

(More) robust ERM training

Stochastic Conditional Gradients

Recall Problem (ERM):

$$\min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x,\theta),y).$$

In the standard formulation θ is unbounded and can get quite large.

Problem. Large θ for, e.g., Neural Networks lead to large Lipschitz constants. Trained network becomes sensitive to input noise and perturbations. [Tsuzuku, Sato, Sugiyama, 2018]

Performance for Neural Network trained on MNIST.

(More) robust ERM training

Stochastic Conditional Gradients

(Partial) Solution. Constrained ERM training:

$$\min_{\theta \in P} \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \ell(f(x, \theta), y), \tag{cERM}$$

where P is a compact convex set.

(More) robust ERM training

Stochastic Conditional Gradients

(Partial) Solution. Constrained ERM training:

$$\min_{\theta \in P} \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} \ell(f(x,\theta), y), \tag{cERM}$$

where P is a compact convex set.

Rationelle. Find “better conditioned” local minima θ.

The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

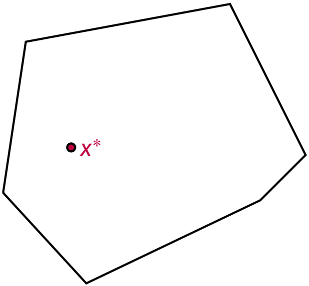
Stochastic Conditional Gradients

[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

$x_0 \in \mathcal{V}$
for $t = 0$ **to** $T-1$ **do**
    $v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
    $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
end for

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

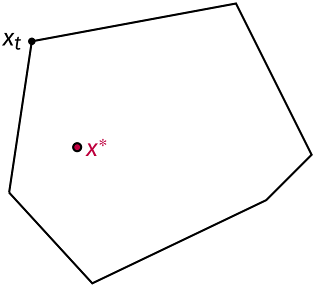
Stochastic Conditional Gradients

[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

$$f(x) = \|x - x^*\|_2^2$$

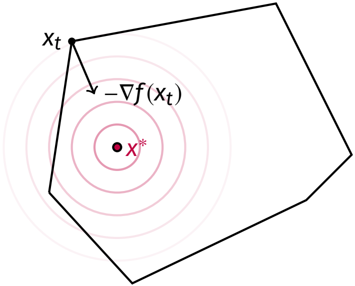


[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

$$f(x) = \|x - x^*\|_2^2$$



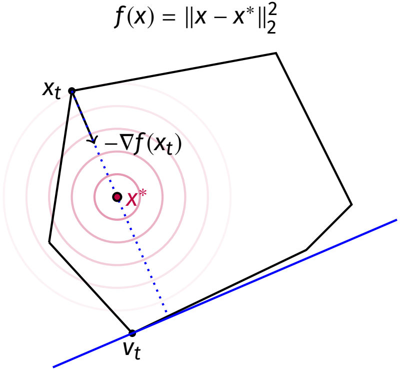
The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

Stochastic Conditional Gradients

[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

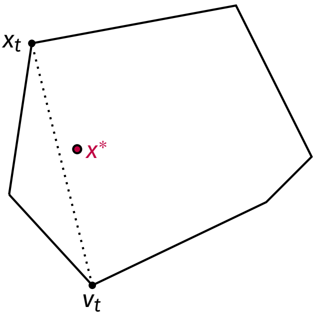


[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

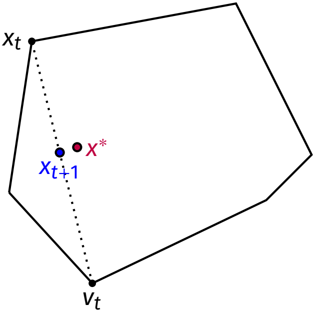
Stochastic Conditional Gradients

[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

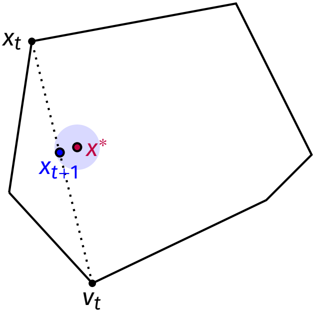
Stochastic Conditional Gradients

[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

Stochastic Conditional Gradients

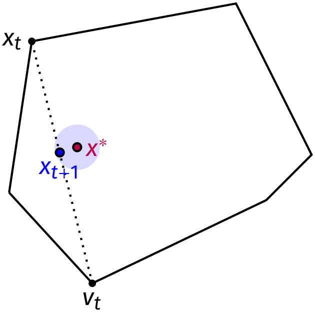
[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T-1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

- FW minimizes f over $\mathrm{conv}(\mathcal{V})$ by sequentially picking up vertices

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

Stochastic Conditional Gradients

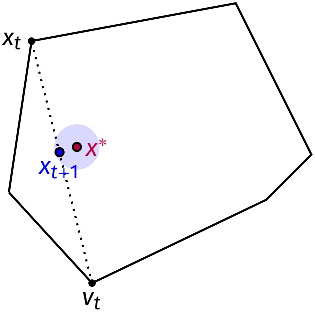
[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

- FW minimizes f over $\mathrm{conv}(\mathcal{V})$ by sequentially picking up vertices
- The final iterate x_T has cardinality at most $T + 1$

$$f(x) = \|x - x^*\|_2^2$$



The Frank-Wolfe Algorithm a.k.a. Conditional Gradients

Stochastic Conditional Gradients

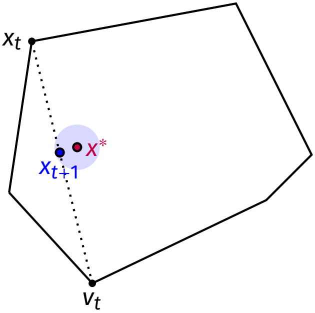
[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

$x_0 \in \mathcal{V}$
for $t = 0$ **to** $T - 1$ **do**
  $v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
  $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
end for

- FW minimizes f over $\text{conv}(\mathcal{V})$ by sequentially picking up vertices
- The final iterate x_T has cardinality at most $T + 1$
- Very easy implementation

$$f(x) = \|x - x^*\|_2^2$$

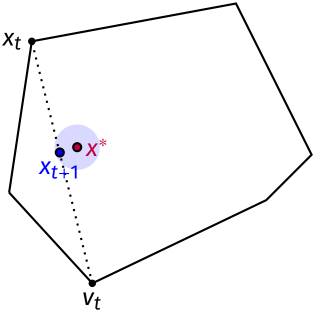


[Frank, Wolfe, 1956] [Levitin, Polyak, 1966]

Algorithm Frank-Wolfe Algorithm (FW)

$x_0 \in \mathcal{V}$
for $t = 0$ **to** $T - 1$ **do**
  $v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
  $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
end for

$$f(x) = \|x - x^*\|_2^2$$



- FW minimizes f over $\mathrm{conv}(\mathcal{V})$ by sequentially picking up vertices
- The final iterate x_T has cardinality at most $T + 1$
- Very easy implementation
- Algorithm is robust and depends on few parameters

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

```
x_0 ∈ 𝒱
for t = 0 to T − 1 do
    v_t ← arg min_{v∈𝒱} ⟨∇f(x_t), v⟩
    x_{t+1} ← x_t + γ_t(v_t − x_t)
end for
```

$x_0 \in \mathcal{V}$

for $t = 0$ **to** $T - 1$ **do**

  $v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$

  $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

end for

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

$x_0 \in \mathcal{V}$
for $t = 0$ **to** $T-1$ **do**
  $v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \nabla f(x_t), v \rangle$
  $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
end for

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \tilde{\nabla} f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \tilde{\nabla} f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

Similarly, many variants available

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \tilde{\nabla} f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

Similarly, many variants available

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.
- Learning rate schedules. To ensure convergence the learning rate η is dynamically managed.

Does it work?

Stochastic Conditional Gradients

As before choose an unbiased gradient estimator $\tilde{\nabla} f(x_t)$ with $\mathbb{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$.

Algorithm Stochastic Frank-Wolfe Algorithm (SFW)

1: $x_0 \in \mathcal{V}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle \tilde{\nabla} f(x_t), v \rangle$
4: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

Similarly, many variants available

- Batch versions. Rather than just taking one stochastic gradient, sample and average a mini batch. This also reduces variance of the gradient estimator.
- Learning rate schedules. To ensure convergence the learning rate η is dynamically managed.
- Variance Reduction. Compute exact gradient once in a while as reference point, e.g., SVRF, SVRCGS, ...

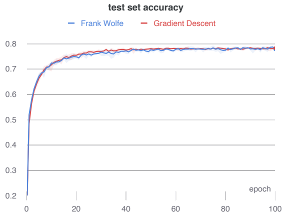
Does it work?

Stochastic Conditional Gradients

Same setup as before. SGD and SFW as solvers.

Does it work?
Stochastic Conditional Gradients

Same setup as before. SGD and SFW as solvers.

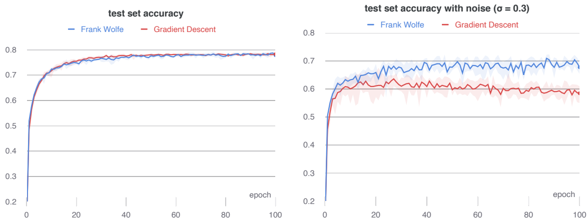


Performance for Neural Network trained on MNIST.

Does it work?

Stochastic Conditional Gradients

Same setup as before. SGD and SFW as solvers.

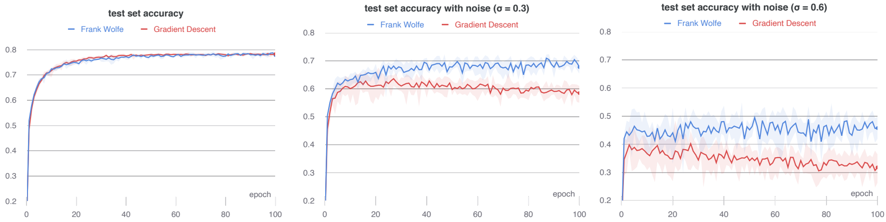


Performance for Neural Network trained on MNIST.

Does it work?

Stochastic Conditional Gradients

Same setup as before. SGD and SFW as solvers.

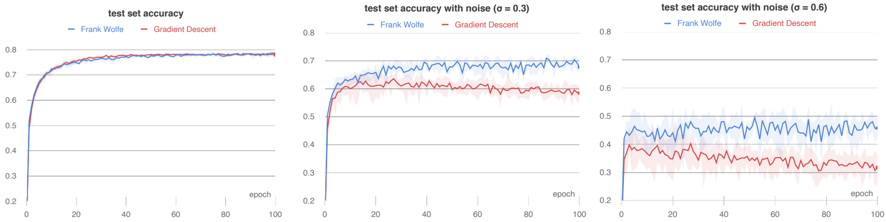


Performance for Neural Network trained on MNIST.

Does it work?

Stochastic Conditional Gradients

Same setup as before. SGD and SFW as solvers.



Performance for Neural Network trained on MNIST.

More details and experiments in the exercise...

Thank you!