Stochastic Optimization for Machine Learning

ICML 2010, Haifa, Israel

Tutorial by
Nati Srebro and Ambuj Tewari
Toyota Technological Institute at Chicago

Goals

- Introduce Stochastic Optimization setup, and its relationship to Statistical Learning and Online Learning
- Understand Stochastic Gradient Descent: formulation, analysis and use in machine learning
- Learn about extensions and generalizations to Gradient Descent and its analysis
- Become familiar with concepts and approaches Stochastic Optimization, and their Machine Learning counterparts

Main Goal: Machine Learning *is* Stochastic Optimization

Outline

- Gradient Descent and Stochastic Gradient Descent
  - Including sub-gradient descent
- The Stochastic Optimization setup and the two main approaches:
  - Statistical Average Approximation
  - Stochastic Approximation
- Machine Learning as Stochastic Optimization
  - Leading example: L_2 regularized linear prediction, as in SVMs
- Connection to Online Learning

(break)

- More careful look at Stochastic Gradient Descent
- Generalization to other norms: Mirror Descent
- Faster convergence under special assumptions

Prelude: Gradient Descent

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$$

Start at some $\mathbf{w}^{(0)}$

Iterate:

$$\mathbf{w}^{(k+1)} \leftarrow \qquad \mathbf{w}^{(k)} - \alpha^{(k)} \nabla F(\mathbf{w}^{(k)})$$

$\Pi_{\mathcal{W}}(\mathbf{w})$ = arg $\min_{v\,\in\,\mathcal{W}} ||\mathbf{v}\text{-}\mathbf{w}||_2$

Prelude: Gradient Descent

$$\min_{\mathbf{w} \in \mathcal{W}} F(\mathbf{w})$$

Start at some $\mathbf{w}^{(0)}$

Iterate:

$$\mathbf{w}^{(k+1)} \leftarrow \Pi_{\mathcal{W}} \left(\mathbf{w}^{(k)} - \alpha^{(k)} \nabla F(\mathbf{w}^{(k)}) \right)$$

$\Pi_{\mathcal{W}}(\mathbf{w}) = \arg\min_{\mathrm{v} \in \mathcal{W}} ||\mathbf{v}\text{-}\mathbf{w}||_2$

Gradient Descent: Analysis

- We will focus on convex, Lipschitz functions.
- *Lipschitz* functions:

$$|F(\mathbf{v})-F(\mathbf{u})| \leq G\cdot||\mathbf{u}-\mathbf{v}||_2$$

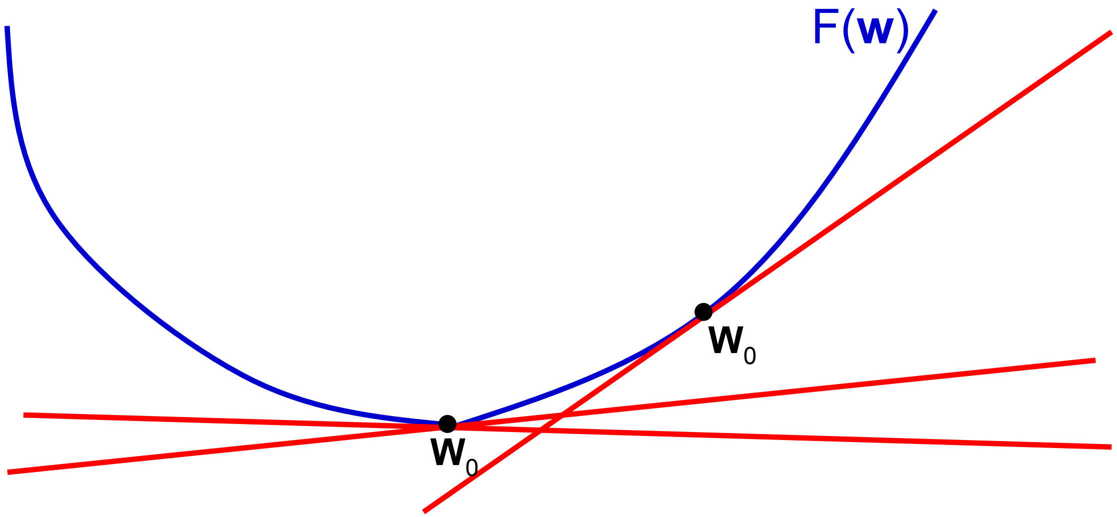
- If f is differentiable:

$$||F(\mathbf{w})||_2 \leq G$$

- What if f is not differentiable?

  Subgradient!

Subgradient of a Convex Function



- If F(·) is differentiable at $\mathbf{w}_0$, gradient gives linear lower bound on F(·):
  $\forall_{\mathbf{v}} \quad F(\mathbf{v}) \geq F(\mathbf{w}_0) + \langle \mathbf{v}\text{-}\mathbf{w}_0, \mathbf{g} \rangle \qquad\qquad \mathbf{g} = \nabla F(\mathbf{w}_0)$

- In general, *subgradient* is any **g** corresponding to a linear lower bound:
  $\forall_{\mathbf{v}} \quad F(\mathbf{v}) \geq F(\mathbf{w}_0) + \langle \mathbf{v}\text{-}\mathbf{w}_0, \mathbf{g} \rangle \quad \Leftrightarrow \quad \mathbf{g} \in \nabla F(\mathbf{w}_0)$

- G-Lipschitz: $||\mathbf{g}||_2 \leq G$ for *all* subgradients $\mathbf{g} \in \nabla F(\mathbf{w})$

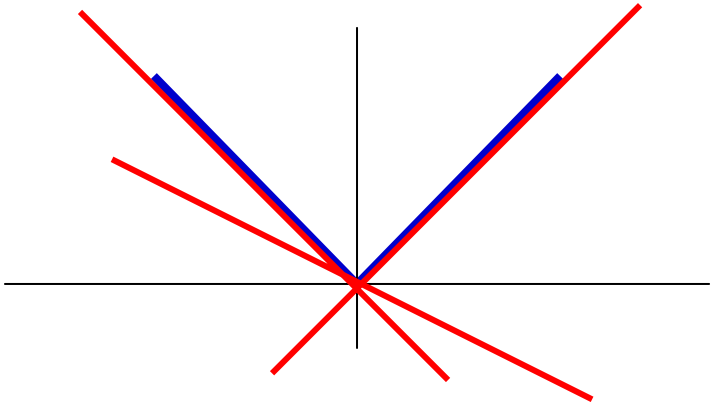
Subgradients: Examples

- $F(z) = |z|$

  $\nabla F(z) = \{-1\}$ $z<0$

  $\nabla F(0) = [-1,1]$

  $\nabla F(z) = \{1\}$ $z>0$

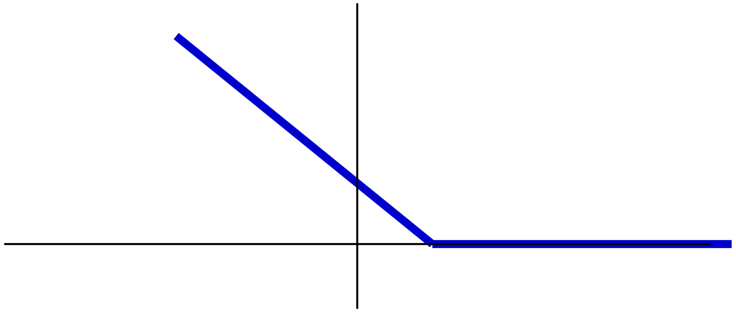


- $F(z) = [1-z]_+$

  $\nabla F(z) = \{-1\}$ $z<1$

  $\nabla F(1) = [-1,0]$

  $\nabla F(z) = \{0\}$ $z>1$



- $F(\mathbf{w}) = ||\mathbf{w}||_1$

  $\nabla F(\mathbf{w})[i] \ni \mathrm{sign}(\mathbf{w}[i])$

Prelude II: Sub-Gradient Descent

$$\min_{\mathbf{w} \in \mathcal{W}} F(\mathbf{w})$$

Start at some $\mathbf{w}^{(0)}$

Iterate:

Get subgradient $\mathbf{g}^{(k)} = \nabla F(\mathbf{w}^{(k)})$

$\mathbf{w}^{(k+1)} \leftarrow \Pi_{\mathcal{W}} (\mathbf{w}^{(k)} - \alpha^{(k)} \mathbf{g}^{(k)})$

$\Pi_{\mathcal{W}}(\mathbf{w}) = \arg\min_{v \in \mathcal{W}} ||\mathbf{v}-\mathbf{w}||_2$

$$\alpha^{(k)} = \frac{B/G}{\sqrt{k}}$$

Guarantee on sub-Optimality:

$||\nabla F(\mathbf{w})||_2 \leq G$

$||\mathbf{w}^*||_2 \leq B$

$$F(\mathbf{w}^{(k)}) - F(\mathbf{w}^*) \leq O\left(\frac{GB}{\sqrt{k}}\right) \Longrightarrow O\left(\frac{G^2B^2}{\epsilon^2}\right) \text{ iterations}$$

This is the best possible using only F(**w**) and ∇F(**w**)

(if the dimension is unbounded)

Stochastic Sub-Gradient Descent

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$$

Start at some $\mathbf{w}^{(0)}$

Iterate:

Get subgradient *estimate* $\mathbf{g}^{(k)}$, s.t. $\mathbb{E}[\mathbf{g}^{(k)}] \in \nabla F(\mathbf{w}^{(k)})$

$\mathbf{w}^{(k+1)} \leftarrow \Pi_{\mathcal{W}}(\ \mathbf{w}^{(k)} - \alpha^{(k)}\mathbf{g}^{(k)}\ )$

Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathbf{w}^{(i)}$

$\alpha^{(k)} = \frac{B/G}{\sqrt{k}}$

Guarantee on sub-Optimality:

$||\mathbf{g}^{(k)}||_2 \le G$

$||\mathbf{w}^*||_2 \le B$

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \le O\left(\frac{GB}{\sqrt{k}}\right) \Longrightarrow O\left(\frac{G^2B^2}{\epsilon^2}\right) \text{ iterations}$$

Same guarantee as (best possible) full-gradient guarantee:

# of stochastic iterations = # of full gradient iterations

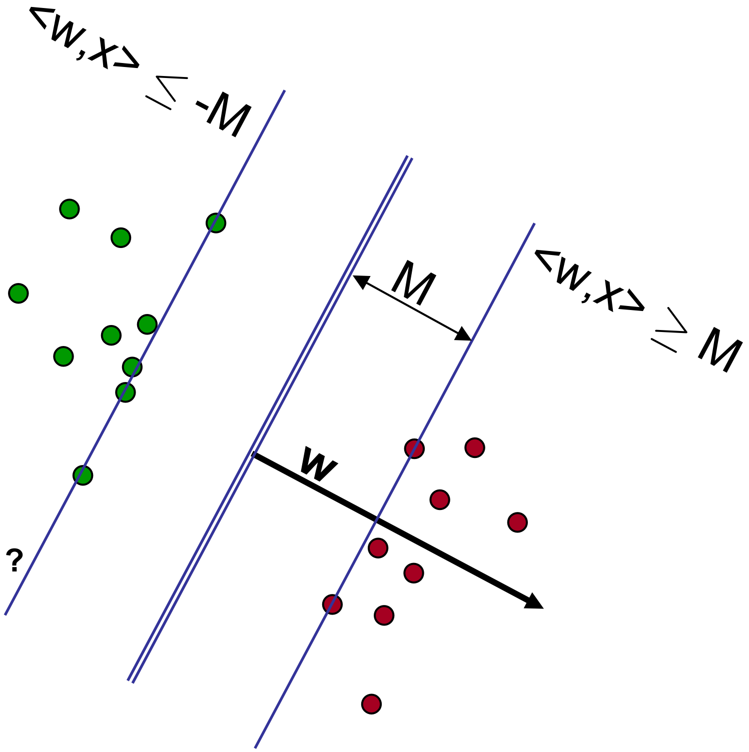
SGD for Machine Learning

$$\min_{\mathbf{w}} \widehat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \mathit{loss}(\mathbf{w} \text{ on } (\mathbf{x}_i, y_i))$$

Subgradient estimate: $\mathbf{g}^{(k)} = \nabla_{\mathbf{w}}\, \mathit{loss}(\, \mathbf{w}^{(k)} \text{ on } (\mathbf{x}_i, y_i)\,)$

Example: linear prediction with hinge loss (SVM)

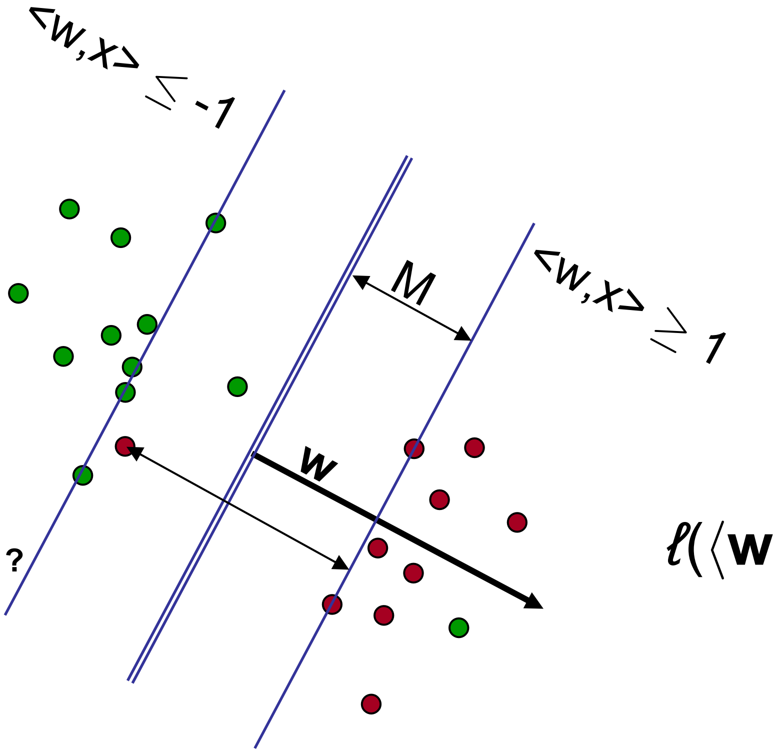
L_2-regularized Linear Classification aka Support Vector Machines



|w|=1

L_2-regularized Linear Classification aka Support Vector Machines

$$\min_{\|\mathbf{w}\|_2 \le B} \frac{1}{m}\sum_{i=1}^{m} \ell(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i) \qquad \equiv \qquad \min_{\mathbf{w}} \frac{1}{m}\sum_{i=1}^{m} \ell(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$



Margin: M = 1/|**w**|

$\ell(\langle \mathbf{w},\mathbf{x} \rangle, y) = [1 - y\langle \mathbf{w},\mathbf{x} \rangle]_+$

SGD for Machine Learning

$$\min_{\mathbf{w}} \hat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} loss(\mathbf{w} \text{ on } (\mathbf{x}_i, y_i))$$

Subgradient estimate: $\mathbf{g}^{(k)} = \nabla_{\mathbf{w}}\, loss(\, \mathbf{w}^{(k)} \text{ on } (\mathbf{x}_i, y_i)\,)$

Example: linear prediction with hinge loss (SVM)

$$\min_{\|\mathbf{w}\|_2 \leq B} \hat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i)$$

$\ell(\langle \mathbf{w},\mathbf{x} \rangle, y) = [1 - y\langle \mathbf{w},\mathbf{x} \rangle]_+$

$$\mathbf{g}^{(k)} = \ell'\left(\left\langle \mathbf{w}^{(k)}, \mathbf{x}_i \right\rangle, y_i\right)\mathbf{x}_i$$

$$= \begin{cases} -y_i \mathbf{x}_i & y_i\left\langle \mathbf{w}^{(k)}, \mathbf{x}_i \right\rangle < 1 \\ 0 & \text{otherwise} \end{cases}$$

$\|\mathbf{g}^{(k)}\|_2 \leq G = \sup \|\mathbf{x}_i\|_2$

```
Start at some $\mathbf{w}^{(0)}$
Iterate:  Draw $i \in 1..n$ at random
          If $y_i\langle \mathbf{w},\mathbf{x}_i \rangle < 1$,
                $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{(k)} y_i \mathbf{x}_i$
          If $\|\mathbf{w}\|_2 \geq B$,
                $\mathbf{w} \leftarrow B\,\mathbf{w} / \|\mathbf{w}\|_2$
          $\mathbf{w}_{sum}$ += $\mathbf{w}$
Output $\mathbf{w}_{sum}/k$
```

Stochastic vs Batch Gradient Descent

$$\min_{\mathbf{w}} \hat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} loss(\mathbf{w} \text{ on } (\mathbf{x}_i, y_i))$$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_1$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_2$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_3$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_4$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_5$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_{m\text{-}1}$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\mathbf{g}_m$

$\mathbf{g}_1 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_1, y_1))$

$\mathbf{g}_2 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_2, y_2))$

$\mathbf{g}_3 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_3, y_3))$

$\mathbf{g}_4 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_4, y_4))$

$\mathbf{g}_5 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_5, y_5))$

$\mathbf{g}_m = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_m, y_m))$

$\mathbf{x}_1, y_1$
$\mathbf{x}_2, y_2$
$\mathbf{x}_3, y_3$
$\mathbf{x}_4, y_4$
$\mathbf{x}_5, y_5$
$\vdots$
$\mathbf{x}_m, y_m$

$$\left.\begin{array}{l}
\mathbf{g}_1 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_1, y_1)) \\
\mathbf{g}_2 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_2, y_2)) \\
\mathbf{g}_3 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_3, y_3)) \\
\mathbf{g}_4 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_4, y_4)) \\
\mathbf{g}_5 = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_5, y_5)) \\
\\
\mathbf{g}_m = \nabla loss(\mathbf{w} \text{ on } (\mathbf{x}_m, y_m))
\end{array}\right\} \nabla \hat{L}(\mathbf{w}) = \tfrac{1}{m}\mathbf{g}_i$$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\sum \mathbf{g}_{\mathrm{i}}$

Stochastic vs Batch Gradient Descent

- Intuitive argument: if only taking simple gradient steps, better to be stochastic (will return to this later)

- Formal result:

$\|\mathbf{x}\|_2 \leq X$

- Stochastic Gradient Descent Runtime: $O\left(\frac{X^2B^2}{\epsilon^2}d\right)$

- Batch Gradient Descent Runtime: $O\left(\frac{X^2B^2}{\epsilon^2}md\right)$

if only using gradients, and only assuming Lipschitz, this is the optimal runtime.

- Compared with second order methods?
- For specific objectives? With stronger assumptions?

Stochastic Optimization Setting

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$$

based on only stochastic information on F:

- Only access to unbiased estimates of $F(\mathbf{w})$ and $\nabla F(\mathbf{w})$
- No direct access to $F(\mathbf{w})$

- E.g. when distribution of *z* is unknown, and can only get sample $z^{(i)}$
  - $\mathbf{g}^{(k)} = \nabla_{\mathbf{w}} f(\mathbf{w}^{(k)}, z^{(k)})$ unbiased estimator of $\nabla F(\mathbf{w})$

- Traditional applications:
  - Optimization under uncertainty
    - Uncertainty about network performance
    - Uncertainty about client demands
    - Uncertainty about system behavior in control problems
  - Complex systems where its easier to sample then integrate over *z*
    - “monte carlo” optimization

Machine Learning *is* Stochastic Optimization

- Up to now: apply stochastic optimization to minimizing *empirical* error

- But learning a good predictor is itself a stochastic optimization problem:

  $$\min_h L(h) = \mathbb{E}_{x,y}[\mathrm{loss}(h(x),y)]$$

  without knowing true distribution of (x,y),
  given sample $(x_1,y_1),\ldots,(x_m,y_m)$

- Special case of stochastic optimization:
  - optimization variable is the predictor (hypothesis) *h*
  - stochastic objective is generalization error (risk)
  - stochasticity is over instances we would like to be able to predict

- Vapnik's "General Learning Setting" is generic stochastic optimization:

  $$\min_h F(h) = \mathbb{E}_z[f(h,z)]$$

  [Vapnik95]

  based on iid sample $z_1,\ldots,z_m$

General Learning: Examples

Minimize $F(h)=\mathbb{E}_z[f(h;z)]$ based on sample $z_1,z_2,\ldots,z_n$

- Supervised learning:

  $z = (x,y)$
  h specifies a predictor $h: \mathcal{X} \to \mathcal{Y}$
  $f(\ h\ ;\ (x,y)\ ) = \text{loss}(h(x),y)$

- Unsupervised learning, e.g. k-means clustering:

  $\mathbf{z} = \mathbf{x} \in \mathbb{R}^{\mathrm{d}}$
  $h = (\mu[1],\ldots,\mu[k]) \in \mathbb{R}^{d\times k}$ specifies k cluster centers
  $f(\ (\mu[1],\ldots,\mu[k])\ ;\ \mathbf{x}\ ) = \min_j ||\mu[i]-\mathbf{x}||^2$

- Density estimation:

  h specifies probability density $p_h(x)$
  $f(\ h\ ;\ x\ ) = -\log p_h(x)$

- Optimization in uncertain environment, e.g.:

  $\mathbf{z}$ = traffic delays on each road segment
  $\mathbf{h}$ = route chosen (indicator over road segments in route)
  $f(\ \mathbf{h}\ ;\ \mathbf{z}\ ) = \langle \mathbf{h},\mathbf{z} \rangle$ = total delay along route

Stochastic Convex Optimization

- We will focus mostly on stochastic ***convex*** optimization:

$$\min_{w\in\mathcal{W}} F(w) = \mathbb{E}[f(w,z)]$$

  - $\mathcal{W}$ is a convex subset of a normed vector space (e.g. $\mathbb{R}^d$)
  - $f(w,z)$, and so also $F(w)$, is convex in w.

- For supervised learning:

$$\min_{w\in\mathcal{W}} L(w) = \mathbb{E}[\mathrm{loss}(\langle w,\phi(x,y)\rangle,y)]$$

convex subset of normed vector space

convex loss

A non-linear predictor will not yield convex L(w) with any meaningful-for-prediction loss function (linear in some implicit feature space is OK).

Stochastic Convex Optimization in Machine Learning

$$\min_{w\in\mathcal{W}} L(w) = \mathbb{E}[\mathrm{loss}(\langle w,\phi(x,y)\rangle,y)]$$

convex subset of normed vector space

convex loss

- Can capture different:
  - convex loss functions
  - norms (regularizers)
  - explicit or implicit feature maps
- Including:
  - SVMs (L_2 norm with hinge loss)
  - Regularized Logistic Regression
  - CRFs, Structural SVMs (L_2 norm with structured convex loss functions)
  - LASSO (L_1 norm with squared loss)
  - Group LASSO (Group $L_{2,1}$ or $L_{\infty\Leftrightarrow\infty}$ norm)
  - Trace-Norm Regularization (as in MMMF, multi-task learning)
- Does NOT include, e.g.:
  - Non-convex loss (e.g. 0/1 loss)
  - Decision trees, decision lists
  - Fromulas (CNF, DNF, and variants)

These *are* instances of stochastic optimization, but not stochastic *convex* optimization

Stochastic Optimization vs Statistical Learning

- Focus on computational efficiency

- Generally assumes unlimited sampling
  - as in monte-carlo methods for complicated objectives

- Optimization variable generally a vector in a normed space
  - complexity control through norm

- Discussion mostly parametric

  BUT: - most convergence results are dimension-independent
  - methods and analysis applicable also to non-parametric problems

- Mostly convex objectives (or at least convex relaxations)

- Focus on sample size

- What can be done with a fixed number of samples?

- Abstract hypothesis classes
  - linear predictors, but also combinatorial hypothesis classes
  - generic measures of complexity such as VC-dim, fat shattering, Radamacher

- Parametric (finite-dim) and non-parametric classes

- Non-convex classes and loss functions
  - multi-layer networks
  - sparse and low-rank models
  - combinatorial classes

Two Approaches to Stochastic Optimization

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}) = \mathbb{E}[f(\mathbf{w},z)]$$

- Sample Average Approximation (SAA):

  [Kleywegt, Shapiro, Homem-de-Mello 2001], [Rubinstein Shapiro 1990], [Plambeck et al 1996]

  – Collect sample $z_1,\ldots,z_m$

  – Minimize $\hat{F}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} f(\mathbf{w}, z_i)$

  – In our terminology: Empirical Risk Minimization

  – Analysis typically based on Uniform Concentration

- Sample Approximation (SA):

  [Robins Monro 1951]

  – Update $\mathbf{w}^{(k)}$ based on weak estimator to $F(\mathbf{w}^{(k)})$, $\nabla F(\mathbf{w}^{(k)})$, etc

    - E.g., based on $\mathbf{g}^{(k)} = \nabla f(\mathbf{w},z^{(k)})$

  – Simplest method: stochastic gradient descent

  – Similar to online approach in learning (more on this later)

Stochastic Approximation for Machine Learning

$$\min_{\mathbf{w}} L(\mathbf{w}) = \mathbb{E}[\ell(\langle \mathbf{w},\mathbf{x}\rangle, y)]$$

$|\ell'| \leq 1$

$\|x\|_2 \leq X$

- Our previous approach was a mixed approach:
  - SAA: collect sample of size *m* and minimize empirical error (w/ norm constraint):

$$\min_{\|\mathbf{w}\|_2 \leq B} \hat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i)$$

  - Optimize this with SGD, i.e. applying SA to the *empirical objective*
    - At each SGD iteration, pick random (x,y) from empirical sample
  - SGD guarantee is on *empirical* suboptimality:

$$\hat{L}(\overline{\mathbf{w}}^{(k)}) \leq \hat{L}(\hat{\mathbf{w}}) + O\left(\sqrt{\frac{X^2 B^2}{k}}\right)$$

  - To get guarantee on $L(w^{(k)})$, need to combined with uniform concentration:

$$\sup_{\|\mathbf{w}\| \leq B} \left| \hat{L}(\mathbf{w}) - L(\mathbf{w}) \right| \leq O\left(\sqrt{\frac{X^2 B^2}{m}}\right)$$

- Pure SA approach:
  - Optimize L(w) directly
    - At each iteration, use an independent sample *from the source distribution*
  - Same SGD guarantee, but directly to the generalization error:

$$L(\overline{\mathbf{w}}^{(k)}) \leq L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2 \|\mathbf{w}^*\|_2^2}{k}}\right)$$

Stochastic Approximation (SGD) for Machine Learning

SGD on Empirical Objective (SA inside SAA):

$$\min_{\|\mathbf{w}\|_2 \leq B} \hat{L}(\mathbf{w})$$

```
Draw $(x_1,y_1),\ldots,(x_m,y_m) \sim \mathcal{D}$
Start at some $\mathbf{w}^{(0)}$
Iterate:
    Draw $i = i^{(k)} \sim$ Unif(1..m)
    $\mathbf{g}^{(k)} = \ell'(\langle \mathbf{w}^{(k)},\mathbf{x}_i \rangle, y_i)\, \mathbf{x}_i$
    $\mathbf{w}^{(k+1)} \leftarrow \Pi_B(\, \mathbf{w}^{(k)} - \alpha^{(k)} \mathbf{g}^{(k)}\, )$

Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{j=1}^{k} \mathbf{w}^{(k)}$
```

Direct SA Approach:

```
Start at some $\mathbf{w}^{(0)}$
Iterate:
    Draw $(\mathbf{x}^{(k)}, y^{(k)}) \sim \mathcal{D}$
    $\mathbf{g}^{(k)} = \ell'(\langle \mathbf{w}^{(k)},\mathbf{x}^{(k)} \rangle, y^{(k)})\, \mathbf{x}^{(k)}$
    $\mathbf{w}^{(k+1)} \leftarrow \mathbf{w}^{(k)} - \alpha^{(k)} \mathbf{g}^{(k)}$

Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{j=1}^{k} \mathbf{w}^{(k)}$
```

- SA requires fresh sample at every iteration, i.e. needs m $\geq$ k
- If m<k, similar, except for projection (and sampling with replacement)
- "SA inside SAA" allows #iterations k > sample size m

- Do we need k > m iterations?
- And also, recall earlier question: Is SAA with 2nd Order Optimization better then SGD ?

Stochastic Approximation (Stochastic Gradient Descent) for Machine Learning

SGD on Empirical Objective (SA inside SAA):

$$\min_{\|\mathbf{w}\|_2 \le B} \hat{L}(\mathbf{w})$$

```
Draw $(x_1,y_1),\ldots,(x_m,y_m) \sim \mathcal{D}$
Start at some $\mathbf{w}^{(0)}$
Iterate:
    Draw $i = i^{(k)} \sim$ Unif(1..m)
    $\mathbf{g}^{(k)} = \ell'(\langle \mathbf{w}^{(k)}, \mathbf{x}_i \rangle, y_i)\ \mathbf{x}_i$
    $\mathbf{w}^{(k+1)} \leftarrow \Pi_B(\ \mathbf{w}^{(k)} - \alpha^{(k)}\, \mathbf{g}^{(k)}\ )$

Output    $\overline{\mathbf{w}}^{(k)} = \frac{1}{k} \sum_{j=1}^{k} \mathbf{w}^{(k)}$
```

$$\hat{L}(\overline{\mathbf{w}}^{(k)}) \le \hat{L}(\hat{\mathbf{w}}) + O\left(\sqrt{\frac{X^2B^2}{k}}\right)$$

$$\sup_{\|\mathbf{w}\| \le B} \left| \hat{L}(\mathbf{w}) - L(\mathbf{w}) \right| \le O\left(\sqrt{\frac{X^2B^2}{m}}\right)$$

$$\Downarrow$$

$$L(\overline{\mathbf{w}}^{(k)}) \le L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2B^2}{k}}\right) + O\left(\sqrt{\frac{X^2B^2}{m}}\right)$$

Direct SA Approach:

```
Start at some $\mathbf{w}^{(0)}$
Iterate:
    Draw $(\mathbf{x}^{(k)}, y^{(k)}) \sim \mathcal{D}$
    $\mathbf{g}^{(k)} = \ell'(\langle \mathbf{w}^{(k)}, \mathbf{x}^{(k)} \rangle, y^{(k)})\ \mathbf{x}^{(k)}$
    $\mathbf{w}^{(k+1)} \leftarrow \mathbf{w}^{(k)} - \alpha^{(k)}\, \mathbf{g}^{(k)}$

Output  $\overline{\mathbf{w}}^{(k)} = \frac{1}{k} \sum_{j=1}^{k} \mathbf{w}^{(k)}$
```

$$\alpha^{(k)} = \frac{B/X}{\sqrt{k}}$$

$$L(\overline{\mathbf{w}}^{(k)}) \le L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2B^2}{k}}\right)$$

$\|w^*\| \le B$

SA vs SAA for L$_2$ Regularized Learning

$$L(\mathbf{w}) = \mathbb{E}[\ell(\langle \mathbf{w},\mathbf{x}\rangle, y)]$$

$|\ell'| \leq 1$ $||x||_2 \leq X$

- SA (Single-Pass Stochastic Gradient Descent)
  - fresh sample ($\mathbf{x}^{(k)}$,$y^{(k)}$) at each iterations
  - i.e. single pass over the data
  - After k iterations: $L(\overline{\mathbf{w}}^{(k)}) \leq L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2\|\mathbf{w}^*\|_2^2}{k}}\right)$

  $\Rightarrow$ to get L(w) $\leq$ L(w*) + ϵ:

  $$\text{sample size } m = O\left(\frac{X^2\|\mathbf{w}^*\|_2^2}{\epsilon^2}\right) \qquad \text{runtime} = O(md) = O\left(\frac{X^2\|\mathbf{w}^*\|_2^2}{\epsilon^2}d\right)$$

- SAA (Empirical Risk Minimization)
  - Sample size to gurantee L(w) $\leq$ L(w*) + ϵ: $m = \Omega\left(\frac{X^2\|\mathbf{w}^*\|_2^2}{\epsilon^2}\right)$

    $\Rightarrow$ using any method: $\text{runtime} \geq \Omega\left(\frac{X^2\|\mathbf{w}^*\|_2^2}{\epsilon^2}d\right)$
  - And with a sample of size m, whatever we do, can't guarantee generalization error better then: $L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2\|\mathbf{w}^*\|_2^2}{m}}\right)$

SA vs SAA for L_2 Regularized Learning

$$L(\mathbf{w}) = \mathbb{E}[\ell(\langle \mathbf{w},\mathbf{x}\rangle, y)]$$

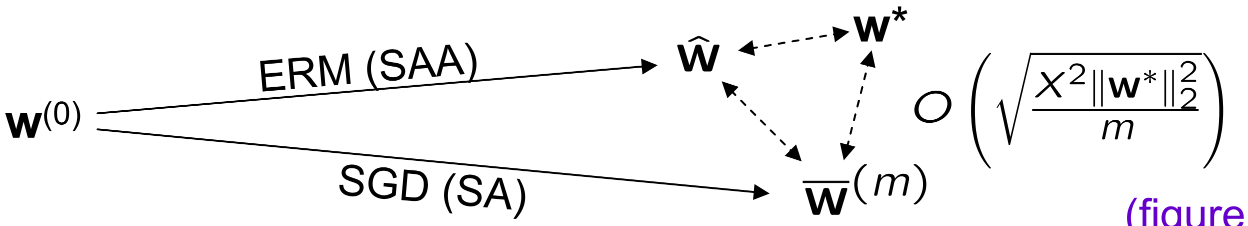
$|\ell'| \leq 1$ $\qquad$ $\|x\|_2 \leq X$

$$\hat{\mathbf{w}} = \arg\min_{\|\mathbf{w}\|\leq B} \hat{L}(\mathbf{w}) \qquad \overline{\mathbf{w}}^{(m)} = \text{output of one-pass SGD on } m \text{ samples}$$

- Summary:
  - Can obtain familiar SVM generalization guarantee directly from [Nemirovski Yudin 78]: with *m* samples, and after *k*=*m* iterations:

$$L(\overline{\mathbf{w}}^{(m)}) \leq L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2\|\mathbf{w}^*\|_2^2}{m}}\right)$$

  - Even with limited sample size, can't beat SA (single-pass SGD): guarantees best-possible generalization error with optimal runtime*



$\mathbf{w}^{(0)}$
ERM (SAA)
SGD (SA)
$\hat{\mathbf{w}}$
$\mathbf{w}^*$
$\overline{\mathbf{w}}^{(m)}$
$O\left(\sqrt{\frac{X^2\|\mathbf{w}^*\|_2^2}{m}}\right)$

(figure adapted from Leon Bottou)

* Up to constant factors

* Without further assumptions (tightness is "worst-case" over source distribution)

Those pesky constant factors…

$\hat{\mathbf{w}} = \arg\min_{\|\mathbf{w}\| \le B} \hat{L}(\mathbf{w})$ $\overline{\mathbf{w}}^{(m)} =$ output of one-pass SGD on *m* samples

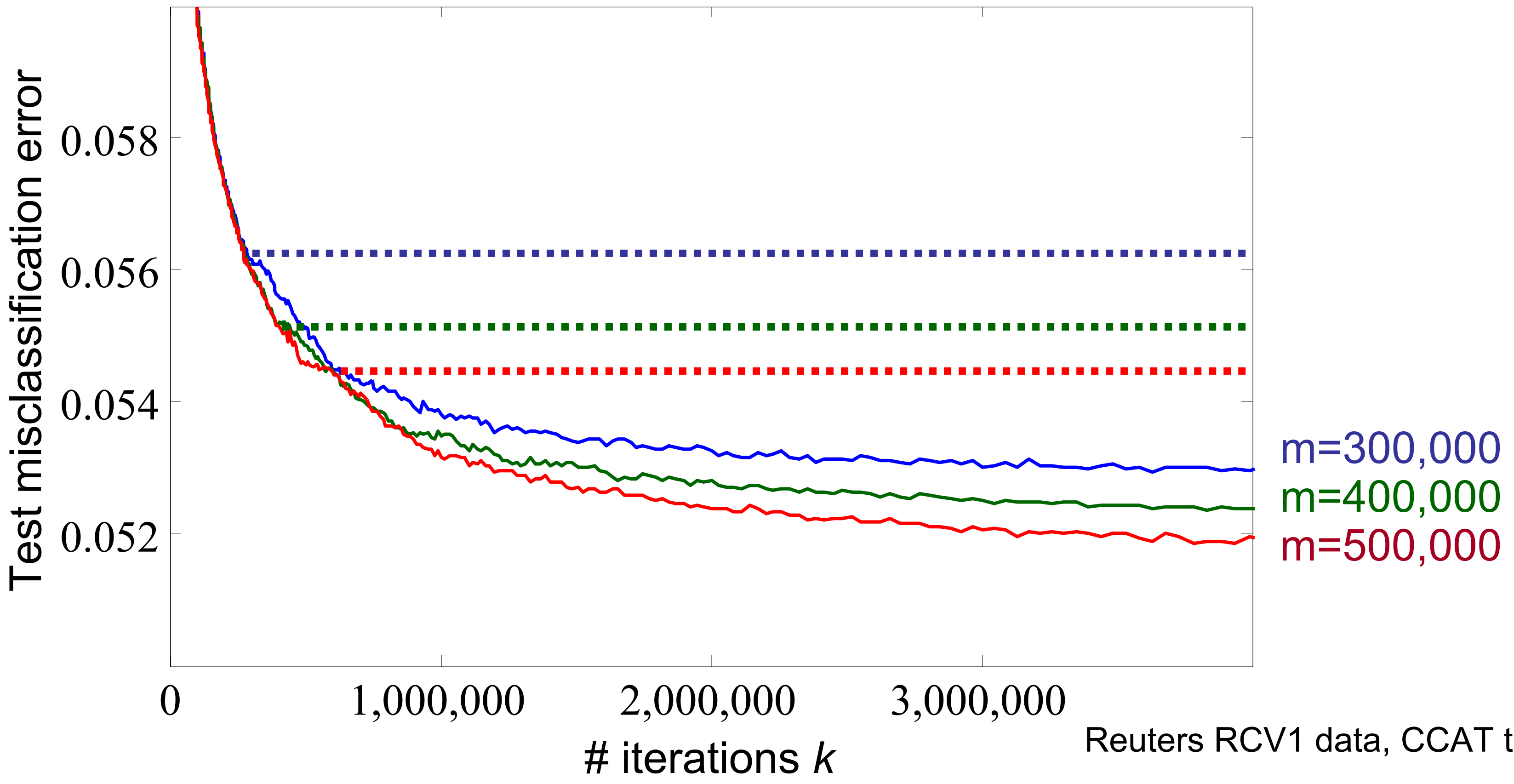
- The constant factor in the theoretical guarantees we can show for SA is actually a bit better then in the ERM guarantee (two vs four)

- Its tight, in the worst case, up to a factor of eight.

- But in practice, the ERM does seem to be better...

- Said differently: with a fixed-size sample, after a single SGD pass over the data, we still don't obtain the same generalization performance as the ERM.

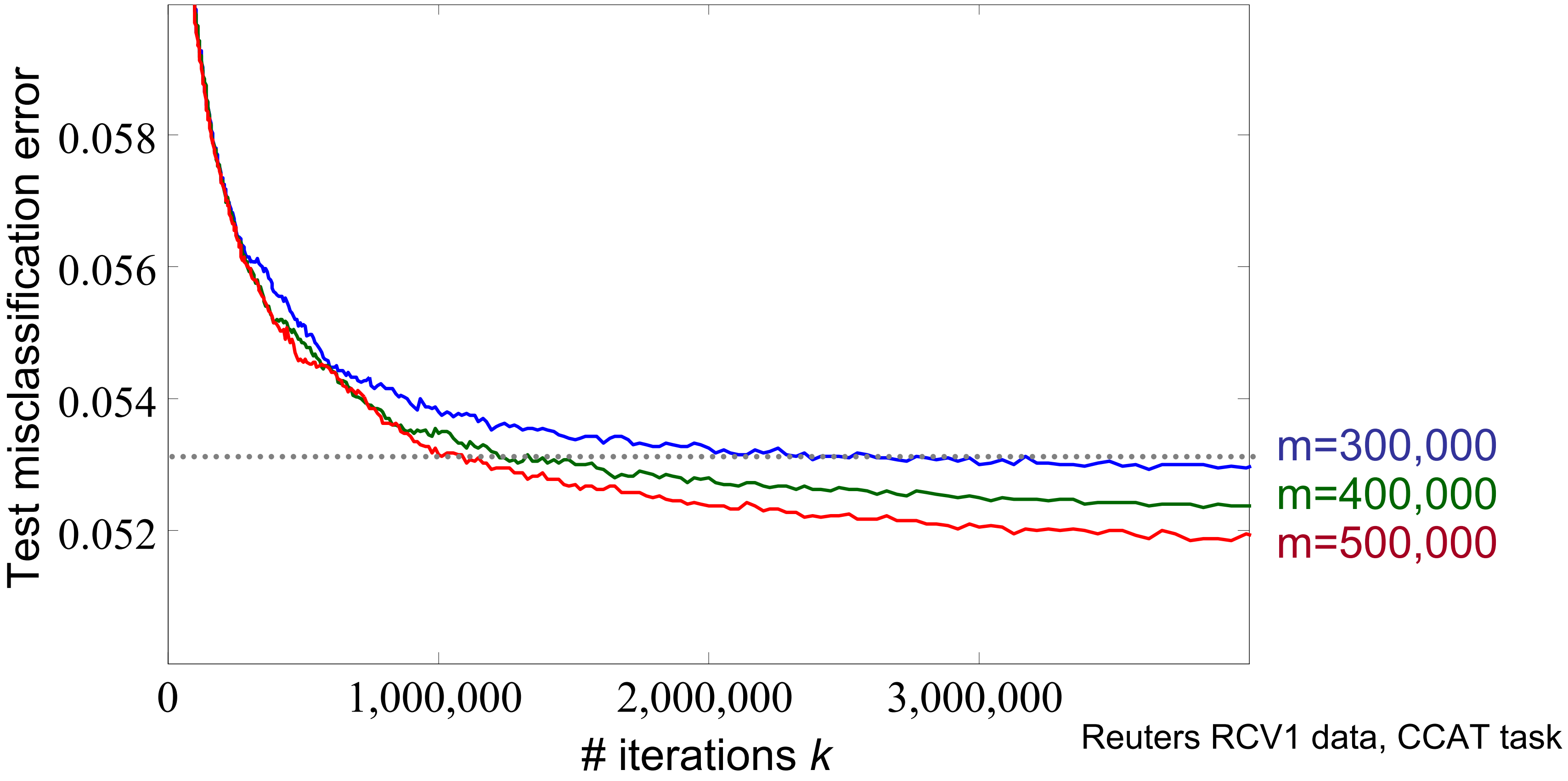
Mixed approach: SGD on Empirical Error



Reuters RCV1 data, CCAT task

[Shalev-Shwartz, Srebro 2008]

Mixed approach: SGD on Empirical Error



Reuters RCV1 data, CCAT task

- The mixed approach (reusing examples) can make sense
- Still: fresh samples are better
  - ⇒ With a larger training set, can reduce generalization error faster
  - ⇒ *Larger* training set means *less* runtime to get target generalization error

[Shalev-Shwartz, Srebro 2008]

Outline

- Gradient Descent and Stochastic Gradient Descent
  - Including sub-gradient descent
- The Stochastic Optimization setup and the two main approaches:
  - Statistical Average Approximation
  - Stochastic Approximation
- Machine Learning as Stochastic Optimization
  - Leading example: L_2 regularized linear prediction, as in SVMs
- Connection to Online Learning

(break)

- More careful look at Stochastic Gradient Descent
- Generalization to other norms: Mirror Descent
- Faster convergence under special assumptions

Online Optimization (and Learning)

- Online optimization setup:
  - As in stochastic optimization fixed and known f(w,z) and domain $\mathcal{W}$
  - $z^{(1)}$,$z^{(2)}$,… presented sequentially by "adversary"
  - "Learner" responds with $w^{(1)}$,$w^{(2)}$,…

$$\begin{tikzcd}
\text{Adversary:} & & z^{(1)} \arrow[dr] & & z^{(2)} \arrow[dr] & & z^{(3)} \arrow[dr] & \\
\text{Learner:} & w^{(1)} \arrow[ur] & & w^{(2)} \arrow[ur] & & w^{(3)} \arrow[ur] & & \cdots\cdot
\end{tikzcd}$$

  - Learner's goal: minimize regret versus best single response in hindsight.

$$\frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{(j)}, z^{(j)}) - \inf_{\mathbf{w}^{*}\in\mathcal{W}} \frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{*}, z^{(j)})$$

  - E.g., investment return:

    $\mathbf{w}[i]$ = investment in holding *i*

    $\mathbf{z}[i]$= return on holding *i*

    f($\mathbf{w}$,$\mathbf{z}$) = -$\langle\mathbf{w}$,z$\rangle$

  - Learning: f($\mathbf{w}$,(x,y)) = loss($h_{\mathbf{w}}(x)$ on y)

Online Optimization (and Learning)

- Online optimization setup:
  - As in stochastic optimization fixed and known f(w,z) and domain $\mathcal{W}$
  - $z^{(1)}, z^{(2)}, \ldots$ presented sequentially by "adversary"
  - "Learner" responds with $w^{(1)}, w^{(2)}, \ldots$

$$\begin{tikzcd}
\text{Adversary:} & & z^{(1)} \arrow[dr] & & z^{(2)} \arrow[dr] & & z^{(3)} \arrow[dr] & \\
\text{Learner:} & w^{(1)} \arrow[ur] & & w^{(2)} \arrow[ur] & & w^{(3)} \arrow[ur] & & \ldots.
\end{tikzcd}$$

  - Learner's goal: minimize regret versus best single response in hindsight.

- Online Gradient Descent **[Zinkevich 03]**:

```
Start at some $\mathbf{w}^{(0)}$
Iterate:   Predict $\mathbf{w}^{(k)}$, receive $z^{(k)}$, pay $f(\mathbf{w}^{(k)}, z^{(k)})$
           $\mathbf{w}^{(k+1)} \leftarrow \Pi_{\mathcal{W}}(\mathbf{w}^{(k)} - \alpha^{(k)} \nabla f(\mathbf{w}^{(k)}, z^{(k)}))$
```

$\alpha^{(k)} = \frac{B/G}{\sqrt{k}}$

$\|\nabla f(\mathbf{w},z)\|_2 \leq G$

$$\frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{(j)}, z^{(j)}) - \frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{*}, z^{(j)}) \leq O\left(\frac{GB}{\sqrt{k}}\right)$$

$\|w^*\| \leq B$

Online Optimization vs Stochastic Approximation

- In both Online Setting and Stochastic Approximation
  - Receive samples sequentially
  - Update **w** after each sample

- But, in Online Setting:
  - Objective is empirical regret, i.e. behavior on observed instances
  - $z^{(k)}$ chosen adversarialy (no distribution involved)
- As opposed on Stochastic Approximation:
  - Objective is $\mathbb{E}_z[f(\mathbf{w},z)]$, i.e. behavior on “future” samples
  - i.i.d. samples $z^{(k)}$

- Stochastic Approximation is a computational approach, Online Learning is an analysis setup
  - E.g. “Follow the leader” is an online algorithm that solves an ERM problem at each iteration. It is still fully in the online setting, and is sensible to analyze as such

Online To Stochastic

- Any online algorithm with regret guarantee:

$$\frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{(j)}, z^{(j)}) - \frac{1}{k}\sum_{j=1}^{k} f(\mathbf{w}^{*}, z^{(j)}) \leq R(k)$$

can be converted to a Stochastic Approximation algorithm, by outputting the average of the iterates **[Cesa-Bianchi et al 04]**:

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^{*}) \leq R(k) \qquad \overline{\mathbf{w}}^{(k)} = \tfrac{1}{k}\textstyle\sum_{i=1}^{k} \mathbf{w}^{(i)}$$

(in fact, even in high confidence rather then in expectation)

Onlined Gradient Descent
[Zinkevich 03]

online2stochastic
[Cesa-Binachi et al 04]

Stochastic Gradient Descent
[Nemirovski Yudin 78]

Break

Outline

- Gradient Descent and Stochastic Gradient Descent
  - Including sub-gradient descent
- The Stochastic Optimization setup and the two main approaches:
  - Statistical Average Approximation
  - Stochastic Approximation
- Machine Learning as Stochastic Optimization
  - Leading example: L_2 regularized linear prediction, as in SVMs
- Connection to Online Learning

(break)

- More careful look at Stochastic Gradient Descent
- Generalization to other norms: Mirror Descent
- Faster convergence under special assumptions

Stochastic Gradient Descent

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w})$$

```
Start at $\mathbf{w}^{(0)}=0$
Iterate:
    Get subgradient *estimate* $\mathbf{g}^{(k)}$
    $\mathbf{w}^{(k+1)} \leftarrow \Pi_{\mathcal{W}}(\ \mathbf{w}^{(k)} - \alpha^{(k)}\mathbf{g}^{(k)}\ )$
Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{j=1}^{k}\mathbf{w}^{(j)}$
```

$\Pi_{\mathcal{W}}(\mathbf{w})$ = arg $\min_{\mathbf{v}\in\mathcal{W}} ||\mathbf{v}\text{-}\mathbf{w}||_2$

Assumptions for analysis:

- F(**w**) is convex in **w**
- Independent and unbiased (sub)-gradient estimates: $\mathbb{E}[\mathbf{g}^{(k)}] \in \nabla F(\mathbf{w}^{(k)})$
- $\mathbb{E}[||\mathbf{g}^{(k)}||_2^{\ 2}] \leq G^2$
  - Equivalently: $\sup_{\mathbf{w}}||\nabla F(\mathbf{w})||^2$ + Var$[\mathbf{g}^{(k)}] \leq G^2$
  - Slightly weaker then $||\mathbf{g}^{(k)}||_2 \leq G$
- We do *not* need $\mathcal{W}$ to be bounded (could be $\mathbb{R}^d$)
  - But stepsize and convergence gurantee will depend on $||\mathbf{w}^*||_2$

Stochastic Gradient Descent: Stepsizes and Convergence

- Main inequality:

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)}) - F(\mathbf{w}^*)\right] \le \frac{\left\|\mathbf{w}^* - \mathbf{w}^{(0)}\right\|^2 + \sum_{j=0}^{k-1}(\alpha^{(k)})^2\mathbb{E}\left[\left\|\mathbf{g}^{(k)}\right\|^2\right]}{2\sum_{j=0}^{k-1}\alpha^{(k)}}$$

(same as for Gradient Descent analysis, but in expectation)

- With any $\alpha^{(k)} \to 0$ and $\sum_{j=1..k}\alpha^{(j)} \to \infty$: $\lim \mathbb{E}\left[F(\overline{\mathbf{w}}^{(K)})\right] \le F(\mathbf{w}^*)$
- Fixed stepsizes:

$$\alpha^{(j)} = \frac{\|\mathbf{w}^*\|_2}{G\sqrt{K}} \quad \Longrightarrow \quad \mathbb{E}\left[F(\overline{\mathbf{w}}^{(K)})\right] \le F(\mathbf{w}^*) + \frac{G\,\|\mathbf{w}^*\|_2}{\sqrt{K}}$$

$$\alpha^{(j)} = \frac{\epsilon}{G^2} \quad \Longrightarrow \quad \mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \le \epsilon \quad \text{with} \quad k = \frac{G^2\|\mathbf{w}^*\|_2^2}{\epsilon^2}$$

- Decaying stepsizes:

$$\alpha^{(k)} = \frac{\|\mathbf{w}^*\|_2}{G\sqrt{k}} \quad \Longrightarrow \quad \mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] \le F(\mathbf{w}^*) + 4\frac{G\,\|\mathbf{w}^*\|_2}{\sqrt{k}}$$

- If we don't know *G*,||**w***||, getting the stepsize wrong is not too bad:

$$\alpha^{(k)} = \beta\frac{\|\mathbf{w}^*\|_2}{G\sqrt{k}} \quad \Longrightarrow \quad \mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] \le F(\mathbf{w}^*) + 4\frac{G\,\|\mathbf{w}^*\|_2}{\sqrt{k}}\max\left(\beta,\tfrac{1}{\beta}\right)$$

Stochastic Gradient Descent: Comments

$$\alpha^{(k)} = \Theta\left(\frac{1}{\sqrt{k}}\right) \qquad \mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] \leq F(\mathbf{w}^*) + O\left(\frac{G\,\|\mathbf{w}^*\|_2}{\sqrt{k}}\right)$$

- Fairly robust to stepsize

- Projections:
  - If minimizing L(**w**) stochastically, using fresh samples at each iteration, can take $\mathcal{W}=\mathbb{R}^d$, and no need to project
  - In mixed SA/SAA approach (SGD on $\hat{L}$(**w**), reusing sample): must take $\mathcal{W}$={**w** | ||**w**||$\leq$ B} to ensure generalization

- Sampling with/without replacement:
  - In mixed SA/SAA approach, when reusing sample, theory only valid when sampling iid *with* replacements.
  - In practice: better to take random permutation of data (ie sample *without* replacement). When permutation is exhusted (finished a pass over the data), take another random permutation, etc. Warning: No theory for this!
  - See Leon Bottou’s webpage.

Stochastic Gradient Descent: Comments

- Averaging:
  - As presented, SGD outputs the *average* over the iterates $\mathbf{w}^{(k)}$
  - Instead of taking the average, same gurantee holds for random iterate:
    - When done, pick $j \in 1..K$ at random, output $\mathbf{w}^{(j)}$
    - Equivalently, use a random number of iterations (pick number of iterations at random between 1..K: on average you are fine).
  - Not aware of guarantee of $\mathbf{w}^{(k)}$ for non-random, predetermined k
    - E.g., it could be that for some problem, taking exactly 7328 SGD iterations would be bad, even in expectation over the sample, but taking a random number of iterations between 1 and 7328 would be fine.
    - My guess: we are missing some theory here…
  - In practice: averaging reduces variance.

- High Confidence Guarantee:

With probability at least 1-δ over the samples:

$$F(\overline{\mathbf{w}}^{(k)}) \leq F(\mathbf{w}^*) + O\left(\frac{G\,\|\mathbf{w}^*\|_2 + \log\frac{1}{\delta}}{\sqrt{k}}\right)$$

e.g. using an online to stochastic conversion [Cesa Bianchi et al 2004]

- Only for average! (not for random iterate)

Other Regularizers

- Discussion so far focused on $||w||_2$, and L_2 regularization
- In particular, SGD sample complexity depends on $||w||_2$, and so matches the sample complexity of L_2 regularized learning.
- What about other regularizers?
  - E.g. L_1, group norms, matrix norms

- Option 1: SAA approach, minimizing:

$$\min_{\|\mathbf{w}\|_{\text{reg}} \le B} \hat{L}(\mathbf{w}) \qquad \text{or} \qquad \min_{\mathbf{w}} \hat{L}(\mathbf{w}) + \lambda \|\mathbf{w}\|_{\text{reg}}$$

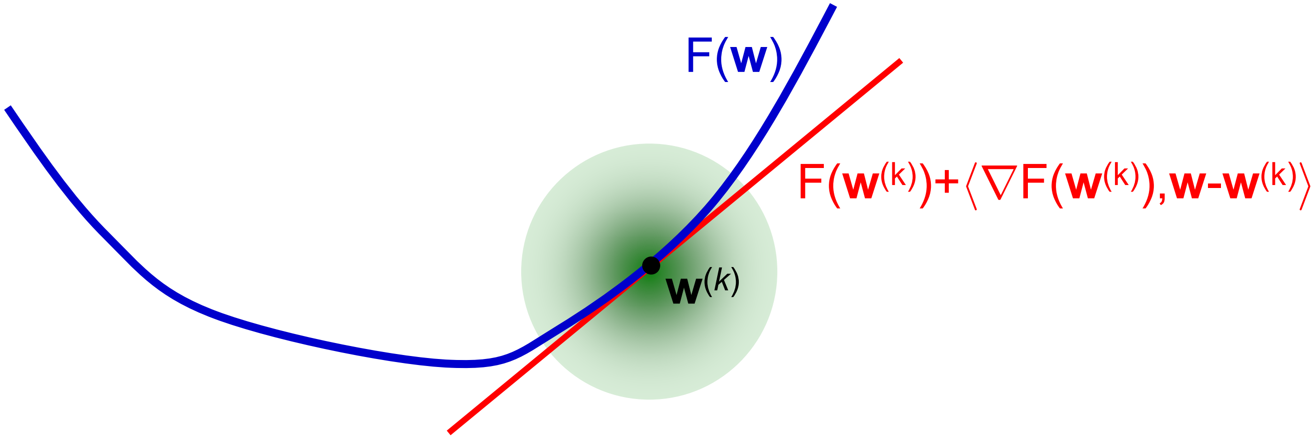
perhaps using SGD (runtime might depend on L_2, but sample complexity on $||w||_{reg}$)

- Option 2: SA approach geared towards other norms…

SGD as a Proximal Method

- Another motivation for (stochastic) gradient descent:

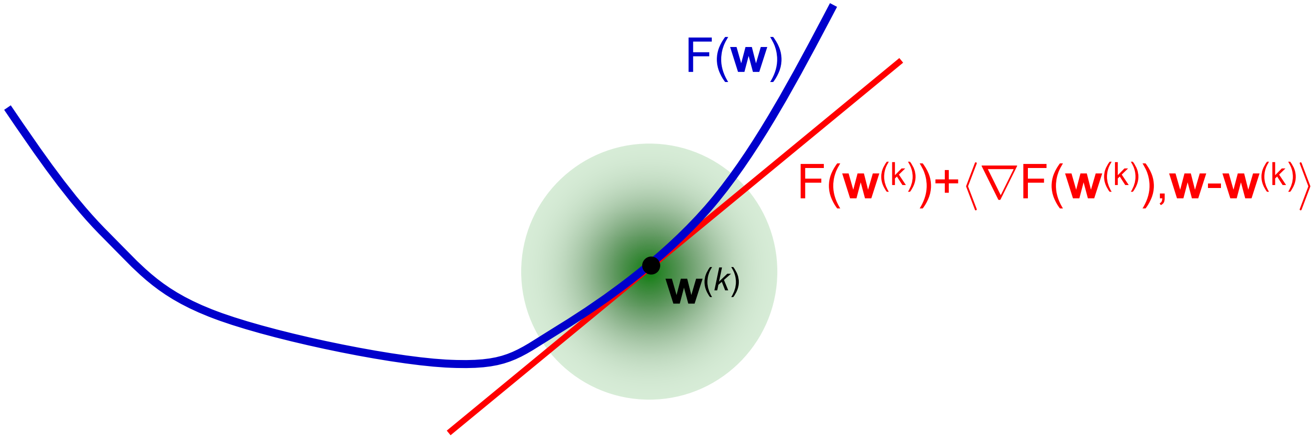
$$\mathbf{w}^{(k+1)} \leftarrow \arg\min_{\mathbf{w}} \underbrace{F(\mathbf{w}^{(k)}) + \left\langle \mathbf{g}^{(k)}, \mathbf{w} - \mathbf{w}^{(k)} \right\rangle}_{\text{1st order model of F}(\mathbf{w})\text{ around }\mathbf{w}^{(k)}\text{, based on }\mathbf{g}^{(k)}} + \underbrace{\frac{1}{2\alpha}\left\|\mathbf{w} - \mathbf{w}^{(k)}\right\|_2}_{\text{only valid near w}^{(k)}\text{, so don't go too far}}$$



SGD as a Proximal Method

- Another motivation for (stochastic) gradient descent:

$$\begin{aligned}
\mathbf{w}^{(k+1)} &\leftarrow \arg\min_{\mathbf{w}} F(\mathbf{w}^{(k)}) + \left\langle \mathbf{g}^{(k)}, \mathbf{w} - \mathbf{w}^{(k)} \right\rangle \ + \ \frac{1}{2\alpha}\left\| \mathbf{w} - \mathbf{w}^{(k)} \right\|_2 \\
&= \arg\min_{\mathbf{w}} \alpha \left\langle \mathbf{g}^{(k)}, \mathbf{w} \right\rangle \ + \ \frac{1}{2}\left\| \mathbf{w} - \mathbf{w}^{(k)} \right\|_2 \\
&= \mathbf{w}^{(k)} - \alpha \mathbf{g}^{(k)}
\end{aligned}$$



SGD as a Proximal Method

- Another motivation for (stochastic) gradient descent:

$$
\begin{aligned}
\mathbf{w}^{(k+1)} &\leftarrow \arg\min_{\mathbf{w}} F(\mathbf{w}^{(k)}) + \left\langle \mathbf{g}^{(k)}, \mathbf{w} - \mathbf{w}^{(k)} \right\rangle \; + \; \frac{1}{2\alpha} \left\| \mathbf{w} - \mathbf{w}^{(k)} \right\|_2 \\
&= \arg\min_{\mathbf{w}} \alpha \left\langle \mathbf{g}^{(k)}, \mathbf{w} \right\rangle \; + \; \frac{1}{2} \left\| \mathbf{w} - \mathbf{w}^{(k)} \right\|_2 \\
&= \mathbf{w}^{(k)} - \alpha \mathbf{g}^{(k)}
\end{aligned}
$$

Start at $\mathbf{w}^{(0)}=0$

Iterate: Get subgradient *estimate* $\mathbf{g}^{(k)}$

$\mathbf{w}^{(k+1)} \leftarrow \arg\min_{\mathbf{w}\in\mathcal{W}} \alpha^{(k)}\langle \mathbf{g}^{(k)},\mathbf{w}^{(k)}\rangle + \underbrace{\tfrac{1}{2}\|\mathbf{w}-\mathbf{w}^{(k)}\|_2}_{\text{replace with other norm?}}$

Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{j=1}^{k} \mathbf{w}^{(j)}$

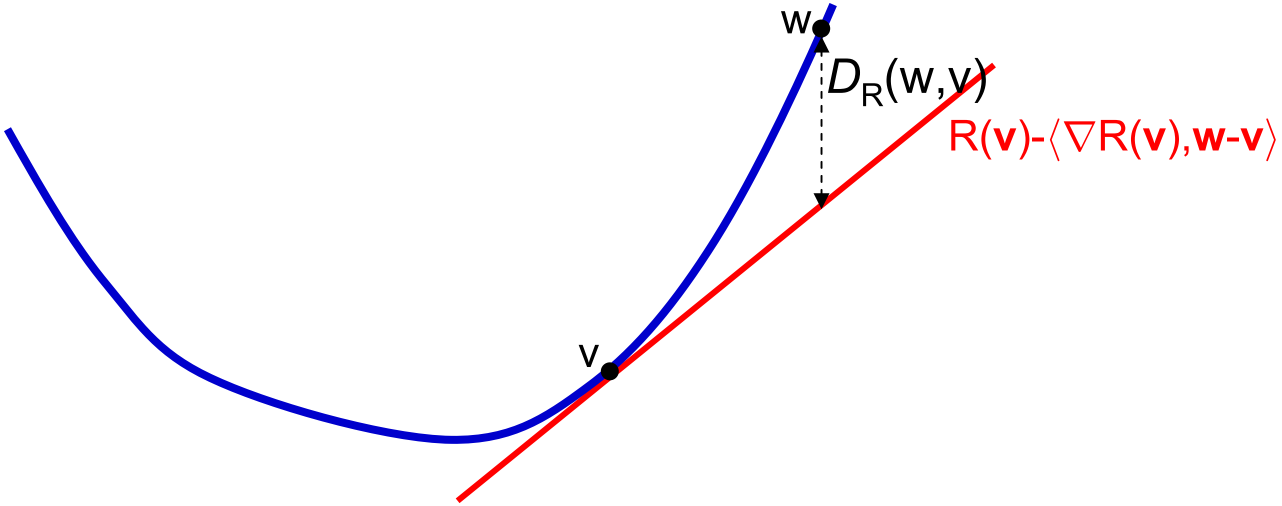
Bregman Divergences

- For a differentiable, convex R define the Bergman Divergence:

$$D_R(\mathbf{w},\mathbf{v}) = R(\mathbf{w})-R(\mathbf{v})-\langle \nabla R(\mathbf{v}),\mathbf{w}-\mathbf{v}\rangle$$

- We will need R that is non-negative and τ-strongly convex w.r.t. our norm of interest ||w||, i.e. s.t.:

$$D_R(\mathbf{w},\mathbf{v}) \geq \tau/2\ ||\mathbf{w}-\mathbf{v}||^2$$


w
$D_R(w,v)$
$R(\mathbf{v})-\langle \nabla R(\mathbf{v}),\mathbf{w}-\mathbf{v}\rangle$
v

Bregman Divergences

- For a differentiable, convex R define the Bergman Divergence:

$$D_R(\mathbf{w},\mathbf{v}) = R(\mathbf{w})-R(\mathbf{v})-\langle\nabla R(\mathbf{v}),\mathbf{w}-\mathbf{v}\rangle$$

- We will need R that is non-negative and τ-strongly convex w.r.t. our norm of interest $||w||$, i.e. s.t.:

$$D_R(\mathbf{w},\mathbf{v}) \geq \tau/2\ ||\mathbf{w}-\mathbf{v}||^2$$

- Examples:
  - $R(w)=½||w||_2^2$ is 1-strongly convex w.r.t. $||w||_2$, $D_R(w,v)=½||w-v||_2^2$
  - $R(w)=½||w||_p^2$ is (p-1)-strongly convex w.r.t $||w||_p$, for p>1
  - $R(w)=\log(d)-\sum_i w[i]\cdot\log(w[i])$ is 1-strongly convex w.r.t. $||w||_1$ on $\{w \in \mathbb{R}^d_+ \mid ||w||<1 \}$

Stochastic Mirror Descent

Start at $\mathbf{w}^{(0)}$=arg min$_{\mathbf{w}}$ R($\mathbf{w}$)
Iterate: Get subgradient *estimate* $\mathbf{g}^{(k)}$
$\mathbf{w}^{(k+1)} \leftarrow$ arg min$_{\mathbf{w}\in\mathcal{W}}$ $\alpha^{(k)}\langle\mathbf{g}^{(k)},\mathbf{w}^{(k)}\rangle$+ $D_{\mathrm{R}}(\mathbf{w},\mathbf{w}^{(k)})$
Output $\overline{\mathbf{w}}^{(k)} = \frac{1}{k}\sum_{j=1}^{k} \mathbf{w}^{(j)}$

Bergman divergence of R, where R(w) is τ-strongly convex w.r.t. ||w||

Similar guarantee to stochastic gradient descent:

$||\mathbf{g}^{(k)}||_* \leq$ G

R($\mathbf{w}^*$) $\leq$ B^2

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \leq O\left(\sqrt{\frac{G^2B^2}{\tau k}}\right)$$

Stochastic Mirror Descent for Linear Prediction

$$\min_{\mathbf{w}} L(\mathbf{w}) = \mathbb{E}[\ell(\langle \mathbf{w},\mathbf{x}\rangle, y)]$$

$|\ell'| \leq 1$

- Using $R(w)=\frac{1}{2}||w||_p^2$ which is (p-1)-strongly convex w.r.t $||w||_p$

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \leq O\left(\sqrt{\frac{(\sup \|\mathbf{x}\|_q^2)\,\|\mathbf{w}^*\|_p^2}{(p-1)k}}\right)$$

- For $||w||_1$, either:
  - Use $R(w)=\frac{1}{2}||w||_p^2$ with p=log(d)
  - Or, use $R(w)=B^2\log(d)-B\sum_i x[i]\cdot\log(x[i])$, which is 1-strongly convex on $\{ x \mid x[i] \geq 0, ||x||_1 \leq B \}$

  Either way:

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \leq O\left(\sqrt{\frac{(\sup \|\mathbf{x}\|_\infty^2)\,\|\mathbf{w}^*\|_1^2 \log d}{k}}\right)$$

- Similarly also for other norms. E.g. for the $L_{2,1}$ “group Lasso”, can use $R(\mathbf{W}) = ||\mathbf{W}||_{2,p}^2$, with p=log(d).

Intermediate Summary

- SGD as proximal method with an L_2 “regularizer”
- Mirror Descent as a generalization to other regularizers
  - Versatile: Just plug in appropriate strongly convex R for your application (*L_p norms, Group norms, Schatten p-norms*)
  - Powerful: Typically gives bounds matching ERM
  - Optimal: In a “worst case” sense, i.e. without making further assumptions.
- Proximal point view also allows other extensions, such as partial linearization (e.g. FOBOS **[Duchi Singer 09]**)

Moving beyond $1/\sqrt{k}$ rates

- Need to assume stronger properties of the objective function F(**w**)

- We will discuss:
  - Strongly Convexity
  - Smoothness

Strongly Convex Objectives

- Focusing on the Euclidean norm $||w||_2$, F(**w**) is λ-strongly convex iff its Hessian is bounded from below:

$$\lambda_{\min}(\nabla^2 F(\mathbf{w})) \geq \lambda$$

- If F(**w**) is λ-strongly convex, then SGD converges much faster:

$||\mathbf{g}^{(k)}||_2 \leq G$

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \leq O\left(\frac{G \log k}{\lambda k}\right)$$

with $\alpha^{(k)} = \frac{1}{\lambda k}$

or, with more sophisticated randomly changing $\alpha^{(k)}$: **[Hazan Kale 10]**

$$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] - F(\mathbf{w}^*) \leq O\left(\frac{G}{\lambda k}\right)$$

- As with (weakly) convex case, this is the optimal rate, even when using full gradients, and for any method using only (full or stochastic) gradient information.

Strongly Convex Objectives in Machine Learning

- When do we encounter strongly convex objectives in Machine Learning?

$$\min_{\mathbf{w}} L(\mathbf{w}) = \mathbb{E}[\ell(\langle \mathbf{w},\mathbf{x}\rangle, y)]$$

- Only when:
  - loss $\ell(t,y)$ is strongly convex, e.g. squared loss $\ell(t,y)=(t-y)^2$
  - ***and*** data is well conditioned, i.e. $\lambda_{\min}(\mathrm{Var}[x]) \geq \lambda$
- Matches ERM gurantees

- More relevant for SGD on *regularized* empirical loss in mixed SAA/SA approach (next slide)

Regularized Empirical Risk Minimization

$$\min_{\mathbf{w}} F(\mathbf{w}) = \hat{L}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2 = \frac{1}{m}\sum_{i=1}^{m}\left(\ell(\langle \mathbf{w}, \mathbf{x}_i\rangle, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|^2\right)$$

- Regardless of (convex) loss function or data distribution, F(**w**) is always λ-strongly convex
- **PEGASOS** **[Shalev-Shwartz et al 07]**: Optimize this regularized empirical risk using SGD, with gradient estimates:

  $$\mathbf{g}^{(k)} = \ell'(\langle \mathbf{w}^{(k)}, \mathbf{x}_{\mathrm{i}}\rangle, y_i) y_i \mathbf{x}_i + \lambda \mathbf{w}^{(\mathbf{k})}$$

  yielding:

  $$\mathbb{E}\left[F(\overline{\mathbf{w}}^{(k)})\right] \leq F(\hat{\mathbf{w}}_\lambda) + O\left(\frac{X}{\lambda k}\right)$$

  But need to take $\lambda = \epsilon/||w^*||^2$, and so this still yields

  $$\mathbb{E}\left[L(\overline{\mathbf{w}}^{(k)})\right] \leq L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2\|\mathbf{w}^*\|_2^2}{k}}\right)$$

  and there ins't really a win here

Smooth Objectives

- A function F(**w**) is H-smooth if

  $$||\nabla F(\mathbf{w}) - \nabla F(\mathbf{v})||_2 \leq H \cdot ||\mathbf{w}-\mathbf{v}||$$

  i.e. *upper* bound on Hessian, or roughly speaking, 2nd derivative is bounded.

- Full (non-stochastic) Gradient Descent converges as:

  $$F(\mathbf{w}^{(k)}) \leq F(\mathbf{w}^*) + O\left(\frac{H\,\|\mathbf{w}^*\|_2^2}{k}\right)$$

- And, "accelerated" first order methods (using only gradient information and simple computations, but steps or not exactly in gradient direction):

  $$F(\mathbf{w}^{(k)}) \leq F(\mathbf{w}^*) + O\left(\frac{H\,\|\mathbf{w}^*\|_2^2}{k^2}\right)$$

SGD with a Smooth Objective

- Stochastic Gradient Descent:

$$F(\overline{\mathbf{w}}^{(k)}) \leq F(\mathbf{w}^*) + O\left(\frac{H\|\mathbf{w}^*\|_2^2}{k} + \sqrt{\frac{\sigma^2\|\mathbf{w}\|_2^2}{k}}\right)$$

- "Accelerated" stochastic-gradient method **[Lan 09]**:

$$F(\overline{\mathbf{w}}^{(k)}) \leq F(\mathbf{w}^*) + O\left(\frac{H\|\mathbf{w}^*\|_2^2}{k^2} + \sqrt{\frac{\sigma^2\|\mathbf{w}\|_2^2}{k}}\right)$$

- Where E[$\mathbf{g}^{(k)}$]$\in\nabla$F($\mathbf{x}^{(k)}$), and Var[$\mathbf{g}^{(k)}$]$\leq\sigma^2$
- In the typical machine learning situation, when stochastic estimates of the gradient are based on single samples, we can only guarantee Var[$\mathbf{g}^{(k)}$] $\leq$ X^2, and we do not get any improvement.

Beyond SGD / Mirror Descent

- Stochastic Coordinate Descent
  - Stochastic in method, not in setup (deterministic objective, in our case the empirical error)
  - L_1 regularization
    - update one random feature at a time
  - Dual of SVM objective [Hsieh et al 2008]
    - update a random dual variable at a time
    - similar in many ways to SGD on primal, which also updates one “multiplier” at a time
    - guarantee on dual sub optimality, but not on primal suboptimality

- Stochastic 2nd Order Methods
  - Leon Bottou’s web page and tutorials
  - Using stochastic Hessian information [Bryd Chin Neveitt Nocedal 2010]

Summary

Machine Learning *is* Stochastic Optimization

- Stochastic Approximation ($\approx$ Online) approaches in a sense “optimal” for machine learning problems.

- Classic Stochastic Approximation results, using Stochastic Mirror Descent and Stochastic Gradient Descent, match familiar Statistical Learning gurantees.

- Mixed approach often beneficial in practice: Optimize ERM using Stochastic Approximation