Computational and Statistical Learning Theory

TTIC 31120

Prof. Nati Srebro

Lecture 16:

Mirror Descent

Online to Batch, Stochastic Optimization

Stability and Learning

$\Psi(w)$ is α-strongly convex w.r.t. $\|w\|$

$\ell(w,z)$ is G-Lipschitz w.r.t $\|w\|$

$\hat{w}_\lambda = \min L_S(w) + \lambda\Psi(w)$ is $\beta(m) = \frac{G^2}{\lambda\alpha m}$ stable

$\frac{G^2}{\lambda\alpha m}$ regret on $\tilde{\ell}_\lambda(w,z) = \ell(w,z) + \lambda\Psi(w)$

$\tilde{L}_\lambda(\hat{w}_\lambda) \leq \inf_w \tilde{L}_\lambda(w) + \frac{G^2}{\lambda\alpha m}$

$\sqrt{\frac{B^2G^2}{\alpha m}}$ regret on $\ell(w,z)$ over $\mathcal{H} = \{\Psi(w) \leq B^2\}$ with FTRL

$L(\hat{w}_\lambda) \leq \inf_{\Psi(w)\leq B^2} L(w) + \sqrt{\frac{B^2G^2}{\alpha m}}$

$\sqrt{\frac{B^2G^2}{\alpha m}}$ regret on $\ell(w,z)$ over $\mathcal{H} = \{\Psi(w) \leq B^2\}$ with L-FTRL (dual averaging)

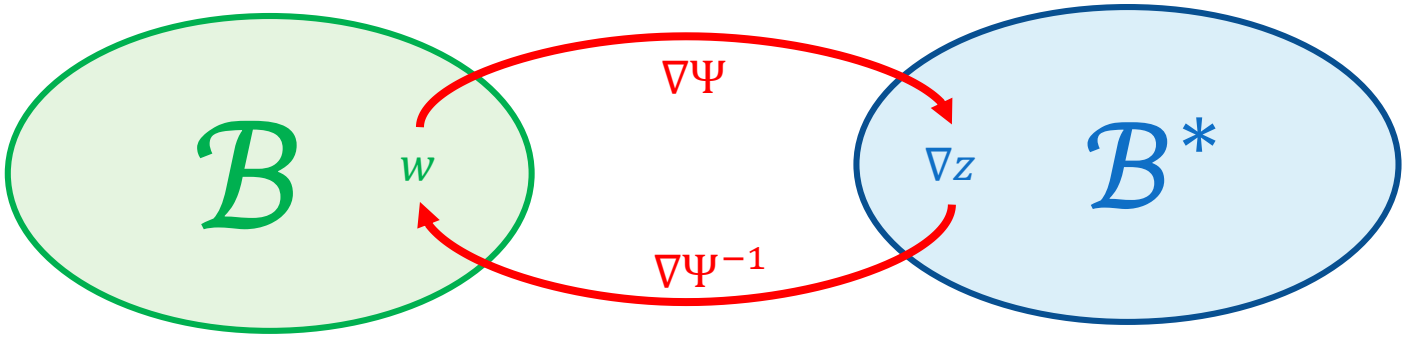
Linearized FTRL

$$w_{t+1} = \arg\min_{w \in \overline{\mathcal{H}}} \frac{1}{t} \sum_{i=1}^{t} \langle \nabla z_i(w_i), w \rangle + \lambda_t \Psi(w)$$

- If $\overline{\mathcal{H}} = \mathcal{B}$: $w_{t+1} = \nabla\Psi^{-1}\left(-\frac{1}{\lambda_t t}\sum_{i=1}^{t} \nabla z_i(w_i)\right)$

$$= \nabla\Psi^{-1}\left(\frac{\lambda_{t-1}(t-1)}{\lambda_t t}\nabla\Psi(w_t) - \frac{1}{\lambda_t t}\nabla z_i(w_i)\right)$$

- If $\overline{\mathcal{H}} \subset \mathcal{B}$: $w_{t+1} = \Pi_{\Psi}^{\overline{\mathcal{H}}}\left(\nabla\Psi^{-1}\left(-\frac{1}{\lambda_t t}\sum_{i=1}^{t} \nabla z_i(w_i)\right)\right)$



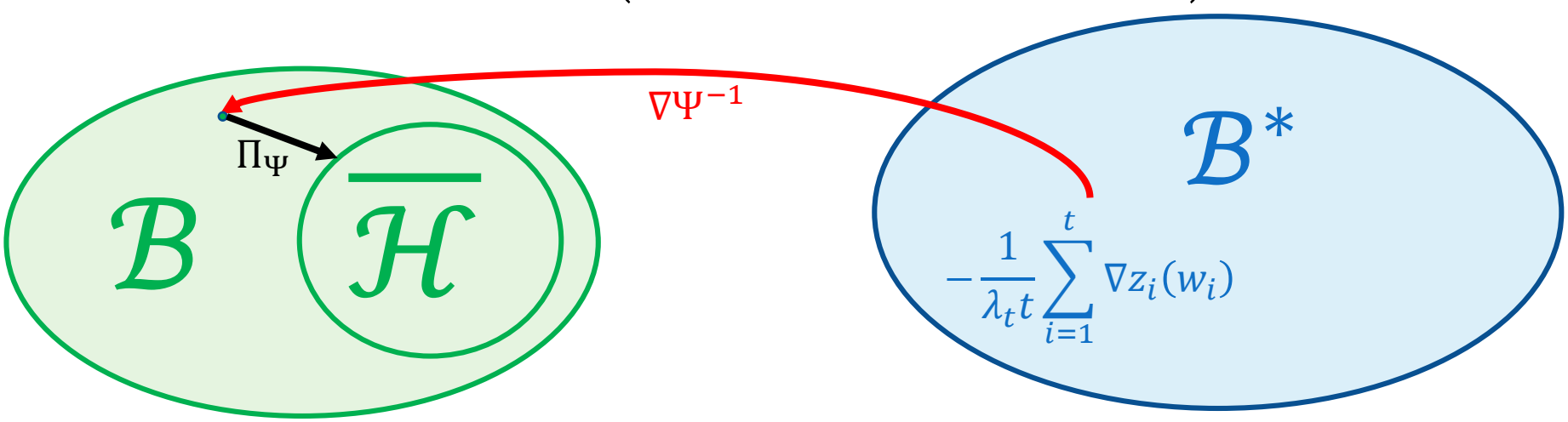
Linearized FTRL

$$w_{t+1} = \arg\min_{w \in \overline{\mathcal{H}}} \frac{1}{t} \sum_{i=1}^{t} \langle \nabla z_i(w_i), w \rangle + \lambda_t \Psi(w)$$

- If $\overline{\mathcal{H}} = \mathcal{B}$: $w_{t+1} = \nabla\Psi^{-1}\left(-\frac{1}{\lambda_t t}\sum_{i=1}^{t} \nabla z_i(w_i)\right)$

$$= \nabla\Psi^{-1}\left(\frac{\lambda_{t-1}(t-1)}{\lambda_t t}\nabla\Psi(w_t) - \frac{1}{\lambda_t t}\nabla z_i(w_i)\right)$$

- If $\overline{\mathcal{H}} \subset \mathcal{B}$: $w_{t+1} = \Pi_{\Psi}^{\overline{\mathcal{H}}}\left(\nabla\Psi^{-1}\left(-\frac{1}{\lambda_t t}\sum_{i=1}^{t} \nabla z_i(w_i)\right)\right)$



From FTRL to Mirror Descent

- FTRL easily implementable only for linear objectives (or for a linearization)---**otherwise need to store all the history**
- Linearized FTRL ***almost*** yields online gradient descent:

$$w_{t+1} = \frac{\lambda_{t-1}(t-1)}{\lambda_t t} w_t - \frac{1}{2\lambda_t t} \nabla z_t(w_t)$$

but only **if** $\overline{\mathcal{H}} = \mathcal{B}$

Non-Linearized Mirror Descent

Initialize: $w_t = \arg\min_{w \in \overline{\mathcal{H}}} \Psi(w)$

Iterate: $\boldsymbol{w_{t+1} = \arg\min_{w \in \overline{\mathcal{H}}} \ell(w, z_t) + t\lambda_t D_\Psi(w || w_t)}$

- Enjoys same regret guarantees as Follow-The-Regularized-Leader:
- For a convex G-Lipschitz problem w.r.t. norm $\|w\|$, if $\Psi(w) \geq 0$ is α-strongly convex w.r.t. $\|w\|$, for any $w \in \overline{\mathcal{H}}$,

$$\frac{1}{m}\sum_{t=1}^{m} \ell(w_t, z_t) \leq \frac{1}{\mathrm{m}}\sum_{t=1}^{m} \ell(w, z_t) + \frac{1}{m}\sum_{t=1}^{m} \lambda_t \Psi(w) + \frac{1}{m}\sum_{t=1}^{m} \frac{2G^2}{\alpha \lambda_t t}$$

- If also $\forall_{w \in \mathcal{H}} \Psi(w) \leq B^2$, using $\lambda_t = \sqrt{\frac{2G^2}{\alpha B^2 t}}$, for any $w \in \mathcal{H}$,

$$\frac{1}{m}\sum_{t=1}^{m} \ell(w_t, z_t) \leq \frac{1}{\mathrm{m}}\sum_{t=1}^{m} \ell(w, z_t) + \sqrt{\frac{32G^2B^2}{\alpha m}}$$

(Linearized) Mirror Descent

- Linearized Mirror Descent:

$$w_{t+1} = \arg\min_{w \in \overline{\mathcal{H}}} \langle \nabla z_t(w_t), w \rangle + t\lambda_t D_\Psi(w || w_t)$$

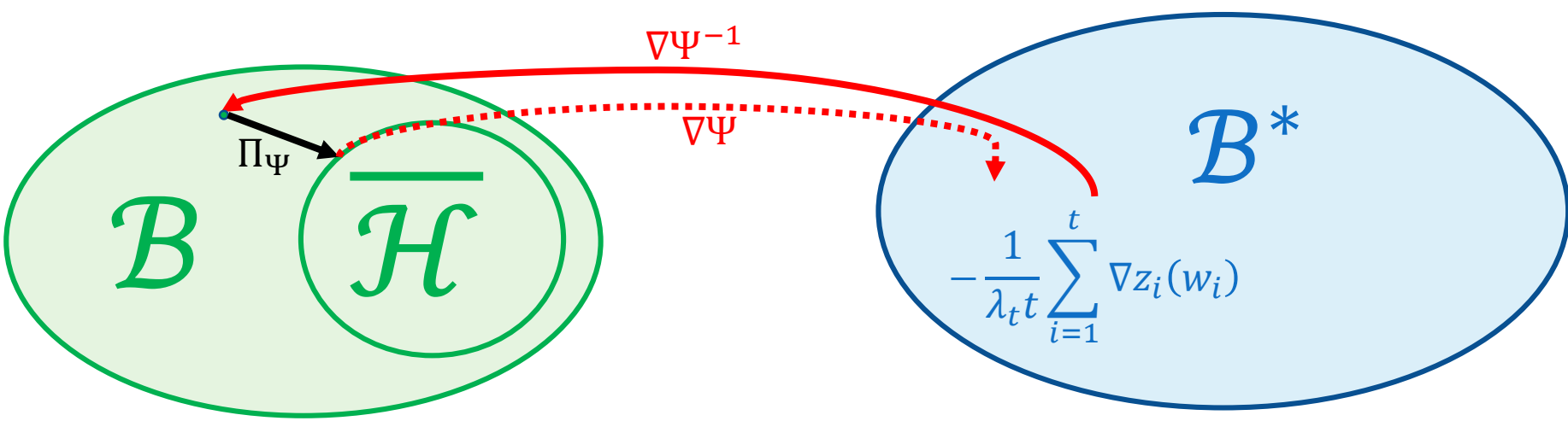
$$= \Pi_\Psi^{\overline{\mathcal{H}}}\left(\nabla\Psi^{-1}\left(\nabla\Psi(w_t) - \frac{1}{t\lambda_t}\nabla z_t(w_t) \right) \right)$$

- Contrast with Linearized FLTR (dual averaging):

$$w_{t+1} = \Pi_\Psi^{\overline{\mathcal{H}}}\left(\nabla\Psi^{-1}\left(-\frac{1}{\lambda_t t}\sum_{i=1}^{t} \nabla z_i(w_i) \right) \right)$$

Only if $\overline{\mathcal{H}} = \mathcal{B}$

$$= \nabla\Psi^{-1}\left(\frac{\lambda_{t-1}(t-1)}{\lambda_t t}\nabla\Psi(w_t) - \frac{1}{\lambda_t t}\nabla z_i(w_i) \right)$$



Online Gradient Descent

- With $\Psi(w) = \|w\|^2$, Online Mirror Descent becomes Online Projected Gradient Descent:

$$w_{t+1} = \arg\min_{w \in \overline{\mathcal{H}}} \langle \nabla z_t(w_t), w \rangle + t\lambda_t \|w - w_t\|^2$$

$$= \Pi^{\overline{\mathcal{H}}}\left(w_t - \eta_t \nabla z_t(w_t)\right)$$

$$\Pi^{\overline{\mathcal{H}}}(w) = \arg\min_{w' \in \overline{\mathcal{H}}} \|w - w'\|$$

$$\eta_t = \frac{2}{t\lambda_t}$$

Advantages of Mirror Descent over FTRL / Dual Averaging

- Allows direct derivation of Online Gradient Descent with arbitrary stepsizes and without scaling of iterates
- Allows incremental algorithm (keeping only w_t, and not entire history) *even without linearization*
- Can also partially linearize, or use any other convex lower bound on the objective

- E.g., for: $\ell(w,(x,y)) = loss(\langle w,x\rangle;y) + \|w\|_1$ can use:
  - $w_{t+1} = \arg\min_{w\in\overline{\mathcal{H}}} loss(\langle w,x\rangle;y) + \|w\|_1 + t\lambda_t D_\Psi(w||w_t)$
  - $w_{t+1} = \arg\min_{w\in\overline{\mathcal{H}}} \langle \nabla loss(w_t), w\rangle + \|w\|_1 + t\lambda_t D_\Psi(w||w_t)$

Summary

- For a convex G-Lipschitz problem w.r.t. norm $\|w\|$, if we have a α-strongly convex regularizer $0 \leq \Psi(w)$ s.t. $\forall_{w \in \mathcal{H}} \Psi(w) \leq B^2$, we can get regret $O\left(\sqrt{\frac{B^2 G^2}{\alpha m}}\right)$ using:
  - FTRL
  - Linearized FLTR (dual averaging)
  - Non-Linearized MD
  - Linearized MD (e.g. Projected GD)
  - Partially linearized MD

- Under same condition, using stability arguments, RERM agnostically PAC learns with sample complexity $O\left(\frac{B^2 G^2}{\alpha \epsilon^2}\right)$ in the statistical setting
- And… its also possible to show matching bounds on the Radamacher complexity of norm-constrained **linear** predictors is such a strongly convex regularizer exists

Online vs Statistical Learning

- Problem specified by:
  - Domain and objective: $\ell: \overline{\mathcal{H}} \times \mathcal{Z} \to \mathbb{R}$
  - Hypothesis class to compete with: $\mathcal{H} \subseteq \overline{\mathcal{H}}$
- Online Regret of learning rule $A: \mathcal{Z}^* \to \overline{\mathcal{H}}$

$$\forall_{z_1, z_2, \ldots, z_m} \frac{1}{m} \sum_{t=1}^{m} \ell(A(z_1 .. z_{t-1}), z_t) \leq \inf_{h \in \mathcal{H}} \frac{1}{m} \sum_{t=1}^{m} \ell(h, z_t) + Reg(m)$$

- Statistical (PAC) "Regret" / excess error:

$$\forall_{\mathcal{D}(\mathcal{Z})} \forall^{\delta}_{S \sim \mathcal{D}^m} L_{\mathcal{D}}\big(A(S)\big) \leq \inf_{h \in \mathcal{H}} \frac{1}{m} \sum_{t=1}^{m} L_{\mathcal{D}}(h) + \epsilon(m, \delta)$$

Low Statistical Regret $\not\Rightarrow$ Low Online Regret

(e.g. linear classification)

- But does low online regret imply low statistical regret?

Online ⇒ Batch ?

- If $A: \mathcal{Z}^* \to \overline{\mathcal{H}}$ has low online regret $Reg(m)$, can we use $A(S)$ for statistical learning and guarantee low excess error?

- Example: $\forall_z \ell(h_{good}, z) = 0, \ell(h_{bad}, z) = 1$

$$A(z_1, \dots, z_t) = \begin{cases} h_{good}, & t \neq 1000 \\ h_{bad}, & t = 1000 \end{cases}$$

$Reg(1000) = 0.001$, but for $m = 1000$, $L(A(S)) = \inf_h L(h) + 1$

- Solution: return an average of the all online iterates.

Online ⇒ Batch

- For a convex learning problem (i.e. $\overline{\mathcal{H}}$ is convex and $\ell(h,z)$ is convex in h):

$$\overline{A}(z_1,\dots,z_m) = \frac{1}{m}\sum_{t=1}^{m} A(z_1,\dots,z_{t-1})$$

$\overline{A}(S)$- Batch Conversion of online rule A

Input: training set $S = \{z_1, z_2, \dots, z_m\}$

1. Run A on $z_1,\dots,z_m$ to obtain $h_1, h_2, \dots, h_{m+1}$
2. Return $\overline{h}_m = \frac{1}{m}\sum_{t=1}^{m} h_t$

- Theorem: $\mathbb{E}_{S\sim\mathcal{D}^m}\left[L_{\mathcal{D}}\left(\overline{A}(S)\right)\right] \le \inf_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + Reg(m)$

Proof: $\mathbb{E}\left[L\left(\overline{A}(S)\right)\right] = \mathbb{E}\left[L\left(\frac{1}{m}\sum_{t=1}^{m} h_t\right)\right] \le \mathbb{E}\left[\frac{1}{m}\sum_{t=1}^{m} L(h_t)\right] = \mathbb{E}\left[\frac{1}{m}\sum_{t=1}^{m} \ell(h_t, z_t)\right]$

$$\le \mathbb{E}\left[\frac{1}{m}\sum_{t=1}^{m} \ell(h, z_t) + Reg(m)\right] = L(h) + Reg(m)$$

- High probability?

If $|\ell(h,z,)| \le a$, $\quad \forall^{\delta}_{S\sim\mathcal{D}^m} L_{\mathcal{D}}\left(\overline{A}(S)\right) \le \inf_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + Reg(m) + 3a\sqrt{\frac{\log^1/_\delta}{m}}$

Proof sketch: apply Azuma's inequality (generalization of Hoeffding)

Non-Convex: Online ⇒ Batch

- If learning problem is non-convex?
  - E.g. supervised learning problem with $0/1$ loss.
- Return randomized predictor:

$$\overline{A}(S) = h_t \text{ with probability } \frac{1}{m}$$

- Guarantee is now on $\tilde{L}_{\mathcal{D}}\left(\overline{A}(S)\right) \stackrel{\text{def}}{=} \mathbb{E}_{rand}\left[L_{\mathcal{D}}\left(\overline{A}(S)\right)\right]$
- Can be viewed as convexifying the problem:

$$\widetilde{\mathcal{H}} = conv\left(\overline{\mathcal{H}}\right) \approx \left\{P\left(\overline{\mathcal{H}}\right) \,\middle|\, P \text{ is a dist over } \overline{\mathcal{H}}\right\}$$

$$\tilde{\ell}\left(\sum_i \alpha_i h_i, z\right) = \sum_i \alpha_i \, \ell(h_i, z)$$

Or: $\tilde{\ell}(P, z) = \mathbb{E}_{h \sim P}[\ell(h, z)]$

Implications of Online-to-Batch

- Anything online learnable is also statistically learnable, with same regret/excess error
  - Explains why we got same expressions, eg for finite hypothesis classes, large margin linear learning, ℓ_p-regularized problems
  - Converse not true!
  - Important: online learnable $\not\Rightarrow$ ULLN nor learnability with ERM.
    - E.g.: generalized convex Lipschitz problems like mean estimation with missing data
    - Learnable with online-to-batch, but not ERM
    - Also learnable with RERM (established using stability)
- More computationally efficient learning methods
  - E.g. Perceptron(+online-to-batch) vs SVM
- Useful even as optimization approach!

Stochastic (Sub)-Gradient Descent

Online Gradient Descent
[Zinkevich 03]

$\xrightarrow[\text{[Cesa-Binachi et al 02]}]{\text{online2stochastic}}$ Stochastic Gradient Descent
[Nemirovski Yudin 78]

Online Mirror Descent
[Shalev-Shwatz Singer 07]

$\longrightarrow$ Stochastic Mirror Descent
[Nemirovski Yudin 78]

Optimize $\boldsymbol{F(w) = \mathbb{E}_{z\sim\mathcal{D}}[f(w,z)]}$ s.t. $\boldsymbol{w \in \mathcal{W}}$

1. Initialize $w_1 = 0 \in \mathcal{W}$
2. At iteration $t = 1,2,3,\ldots$
   1. Sample $z_t \sim \mathcal{D}$
   2. $w_{t+1} = \Pi^{\mathcal{W}}\left(w_t - \eta_t \underbrace{\nabla f(w_t, z_t)}_{\boldsymbol{g_t}}\right)$
3. Return $\overline{w}_m = \frac{1}{m}\sum_{t=1}^{m} w_t$

If $\|\nabla f(w,z)\|_2 \le G$ then with appropriate step size:

$$\mathbb{E}[F(\overline{w}_m)] \le \inf_{w\in\mathcal{W},\|w\|_2\le B} F(w) + O\left(\sqrt{\frac{B^2G^2}{m}}\right)$$

Similarly, also Stochastic Mirror Descent

Stochastic Optimization

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

based only on stochastic information on F

- Only unbiased estimates of $F(w), \nabla F(w)$
- No direct access to F

E.g., fixed $f(w, z)$ but $\mathcal{D}$ unknown

- Optimize $F(w)$ based on iid sample $z_1, z_2, \dots, z_m \sim \mathcal{D}$
- $g = \nabla f(w, z_t)$ is unbiased estimate of $\nabla F(w)$

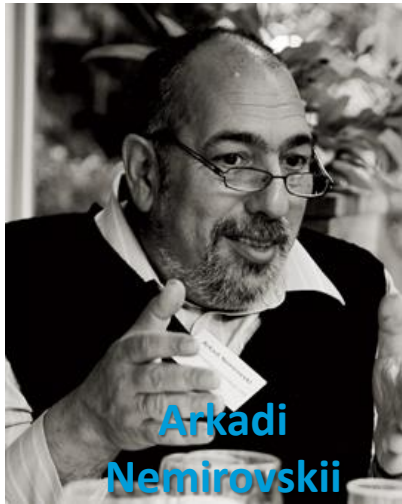
- Traditional applications
  - Optimization under uncertainty
    - Uncertainty about network performance
    - Uncertainty about client demands
    - Uncertainty about system behavior in control problems
  - Complex systems where its easier to sample then integrate over z

Machine Learning *is* Stochastic Optimization

$$\min_h L(h) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(h,z)] = \mathbb{E}_{x,y\sim\mathcal{D}}[loss(h(x),y)]$$

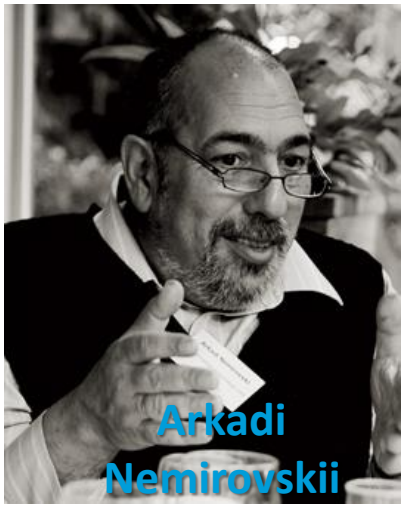
- Optimization variable: predictor h
- Objective: generalization error $L(h)$
- Stochasticity over $z = (x, y)$

"General Learning" $\equiv$ Stochastic 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
- Mostly convex objectives



- 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
- Also non-convex classes and loss functions

Two Approaches to Stochastic Optimization / Learning

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

- Empirical Risk Minimization (ERM) / Sample Average Approximation (SAA):
  - Collect sample $z_1, \ldots, z_m$
  - Minimize $F_S(w) = \frac{1}{m} \sum_i f(w, z_i)$
  - Analysis typically based on Uniform Convergence

- Stochastic Approximation (SA): [Robins Monro 1951]
  - Update w_t based on z_t
    - E.g., based on $g_t = \nabla f(w, z_t)$
  - E.g.: stochastic gradient descent
  - Online-to-batch conversion of online algorithm…

SGD as an Optimization Algorithm

Goal $\min_{\|w\|_2 \le B} F(w)$

- If $F(w)$ is G-Lipschitz (and all gradient estimates are bounded by G), after $T = O\left(\frac{B^2G^2}{\epsilon^2}\right)$ iterations of SGD, we have $F(\overline{w}_T) \le \inf_{\|w\|_2 \le B} F(w) + \epsilon$

- Claim: any optimization algorithm that only accesses $F(w)$ locally, i.e. only accesses $F(w), \nabla F(w)$,etc, and ensures $F(w_T) \le \inf_{\|w\|_2 \le B} F(w) + \epsilon$ after T accesses for any G-Lipschitz objective, must make at least $T = \Omega\left(\frac{B^2G^2}{\epsilon^2}\right)$ accesses

- SGD required only $T = O\left(\frac{B^2G^2}{\epsilon^2}\right)$ accesses to unbiased *stochastic* gradient estimates, i.e. g s.t. $\mathbb{E}[g] \in \partial F(w)$

- E.g., consider $F(w) = L_S(w) = \frac{1}{m}\sum_{i=1}^{m} loss^{hinge}(\langle w, \phi(x_i)\rangle; y_i)$

SGD for SVM

$$\min L_S(w) \;\; s.t. \;\; \|w\|_2 \leq B$$

Use $g_t = \nabla_w loss^{hinge}\left(\langle w_t, \phi_{i_t}(x)\rangle; y_{i_t}\right)$ for random i_t

Initialize $w^{(0)} = 0$

At iteration t:

- Pick $i \in 1 \ldots m$ at random
- If $y_i \langle w^{(t)}, \phi(x_i)\rangle < 1$,
  $w^{(t+1)} \leftarrow w^{(t)} + \eta_t y_i \phi(x_i)$
  else: $w^{(t+1)} \leftarrow w^{(t)}$
- If $\left\|w^{(t+1)}\right\|_2 > B$, then $w^{(t+1)} \leftarrow B \frac{w^{(t+1)}}{\left\|w^{(t+1)}\right\|_2}$

Return $\overline{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

$$\|g_t\|_2 \leq G \Rightarrow L_S\left(\overline{w}^{(T)}\right) \leq L_S(\widehat{w}) + \sqrt{\frac{B^2 G^2}{T}}$$

(in expectation over randomness in algorithm)

Stochastic vs Batch

$$\min L_S(w) \;\; s.t. \;\; \|w\|_2 \leq B$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\nabla L_S(w) = \frac{1}{m} \sum g_i$$

$\mathbf{w} \leftarrow \mathbf{w}\text{-}\sum \mathbf{g}_i$

Stochastic vs Batch

- Intuitive argument: if only taking simple gradient steps, better to be stochastic

- To get $L_S(w) \leq L_S(\widehat{w}) + \epsilon_{opt}$:

	#iter	cost/iter	runtime
Batch GD	B^2G^2/ϵ_{opt}^2	md	$\boldsymbol{m}d\frac{B^2G^2}{\epsilon_{opt}^2}$
SGD	B^2G^2/ϵ_{opt}^2	d	$d\frac{B^2G^2}{\epsilon_{opt}^2}$

- What about $L(w)$, which is what we really care about?
- How small should ϵ_{opt} be?
- Comparison to methods with a $\log 1/\epsilon$ dependence that use the structure of $L_S(w)$ (not only local access)?