ETH zürich

Large scale learning with adaptive sample sizes

Hadi Daneshmand, Aurelien Lucchi, Thomas Hofmann

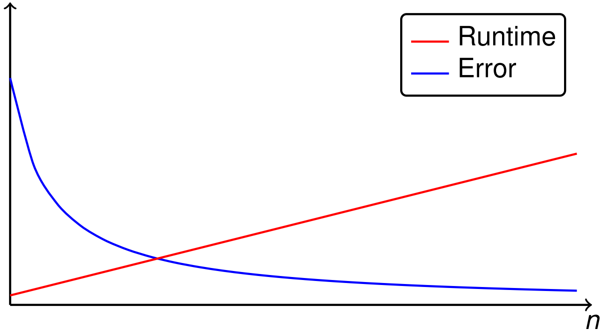
WSS 2017

Computer Science vs. Statistics

$$\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Error: $\mathbf{E}\left[\|\widehat{\mu} - \mathbf{E}\left[\widehat{\mu}\right]\|\right] < C/\sqrt{n}$

Runtime: $O(n)$



Time-Data Tradeoff

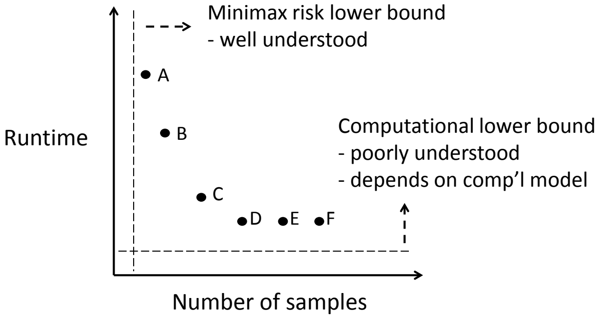


Figure: Time-Data Tradeoff[1]

[1] Chandrasekaran, V. & Jordan, M. I. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences* **110,** E1181–E1190 (2013).

Example: Denoising Problem

- Recovering unknown signal $\boldsymbol{x}^* \in \mathcal{S} \subset \mathbb{R}^d$ from noisy observations:

$$\boldsymbol{y}_i = \boldsymbol{x}^* + \sigma^2 \boldsymbol{z}_i, \boldsymbol{z}_i \sim \mathcal{N}(0, \mathbf{I}), i = 1, \ldots, n \tag{1}$$

  Let $\bar{y} = \sum_{i=1}^{n} y_i / n$.
- Optimization problem:

$$\arg\min_{\boldsymbol{x}} \frac{1}{2} \|\bar{y} - \boldsymbol{x}\|^2, \text{ s.t } \boldsymbol{x} \in \mathcal{S} \tag{2}$$

Denoising: Convex Relaxation

- Non-convex problem:

$$\arg\min_{\boldsymbol{x}} \frac{1}{2}\|\bar{y} - \boldsymbol{x}\|^2, \text{ s.t } \boldsymbol{x} \in \mathcal{S} \tag{3}$$

- Convex relaxed problem:

$$\boldsymbol{x}_n(\mathcal{C}) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\bar{y} - \boldsymbol{x}\|^2, \text{ s.t } \boldsymbol{x} \in \mathcal{C} \tag{4}$$

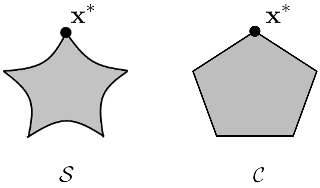


Figure: Convex Relaxation[2]

[2] Chandrasekaran, V. & Jordan, M. I. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences* **110,** E1181–E1190 (2013).

Denoising: Recovering Gaurantees

$$\mathbf{E}\left[\|\boldsymbol{x}_n(\mathcal{C}) - \boldsymbol{x}^*\|^2\right] \leq \frac{\sigma^2}{n} \underbrace{g(T(\boldsymbol{x}^*, \mathcal{C}))}_{\text{Monotonic in } \mathcal{C}} \tag{5}$$

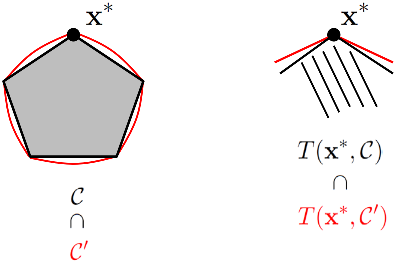


Figure: Recovery condition[3]

[3] Chandrasekaran, V. & Jordan, M. I. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences* **110,** E1181–E1190 (2013).

Convexity, Classification, and Risk Bounds

Peter L. BARTLETT, Michael I. JORDAN, and Jon D. MCAULIFFE

Many of the classification algorithms developed in the machine learning literature, including the support vector machine and boosting, can be viewed as minimum contrast methods that minimize a convex surrogate of the 0–1 loss function. The convexity makes these algorithms computationally efficient. The use of a surrogate, however, has statistical consequences that must be balanced against the computational virtues of convexity. To study these issues, we provide a general quantitative relationship between the risk as assessed using the 0–1 loss and the risk as assessed using any nonnegative surrogate loss function. We show that this relationship gives nontrivial upper bounds on excess risk under the weakest possible condition on the loss function—that it satisfies a pointwise form of Fisher consistency for classification. The relationship is based on a simple variational transformation of the loss function that is easy to compute in many applications. We also present a refined version of this result in the case of low noise, and show that in this case, strictly convex loss functions lead to faster rates of convergence of the risk than would be implied by standard uniform convergence arguments. Finally, we present applications of our results to the estimation of convergence rates in function classes that are scaled convex hulls of a finite-dimensional base class, with a variety of commonly used loss functions.

KEY WORDS: Boosting; Convex optimization; Empirical process theory; Machine learning; Rademacher complexity; Support vector machine.

1. INTRODUCTION

Convexity has become an increasingly important theme in applied mathematics and engineering, having taken on a prominent role akin to that played by linearity for many decades. Building on the discovery of efficient algorithms for linear programs, researchers in convex optimization theory have developed computationally tractable methods for large classes of convex programs (Nesterov and Nemirovskii 1994). Many fields in which optimality principles form the core conceptual

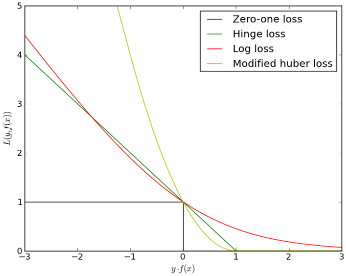
derstand these algorithms not only from a computational standpoint, but also in terms of their statistical properties. What are the statistical consequences of choosing models and estimation procedures so as to exploit the computational advantages of convexity?

In this article we study this question in the context of discriminant analysis, a topic referred to as *classification* in the machine learning field. We consider the setting in which a covariate vector $X \in \mathcal{X}$ is to be classified according to a binary response $Y \in \{-1, 1\}$. The goal is to choose a discriminant

Convex Relaxation of Classification Problem

- $\boldsymbol{x}$ is dependent random variable, $y \in \{\mp 1\}$ is dependent random variable
- Goal: minimizing classification error

$$\min_{f_{\boldsymbol{w}} \in \mathcal{F}} \left[\mathcal{L}(\boldsymbol{w}) := \mathbf{E}_{\boldsymbol{x}, y \sim \mathcal{P}} \left[\mathbf{I}(y f_{\boldsymbol{w}}(\boldsymbol{x}) < 0) \right] \right] \tag{6}$$



Classification Problem: Convex Relaxation Error

- Main Objective

$$\min_{f_{\boldsymbol{w}} \in \mathcal{F}} \left[\mathcal{L}(\boldsymbol{w}) := \mathbf{E}_{\boldsymbol{x}, y \sim \mathcal{P}} \left[\mathbf{I}(y f_{\boldsymbol{w}}(\boldsymbol{x}) < 0)\right]\right] \tag{7}$$

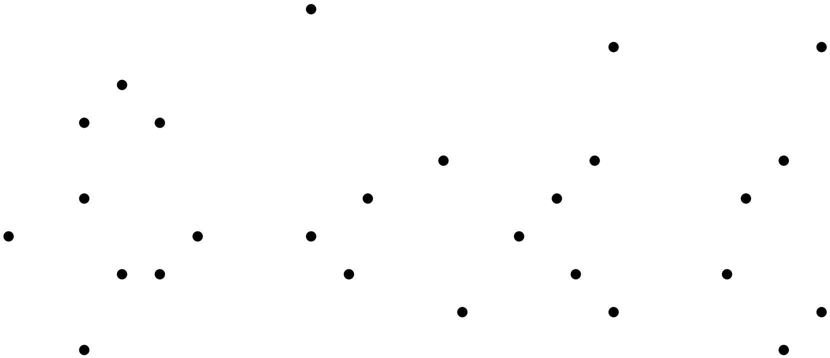
- Convex Relaxation:

$$\min_{f_{\boldsymbol{w}} \in \mathcal{F}} \left[\mathcal{R}(\boldsymbol{w}) := \mathbf{E}_{\boldsymbol{x}, y \sim \mathcal{P}} \left[\varphi(y f_{\boldsymbol{w}}(\boldsymbol{x}))\right]\right] \tag{8}$$

- Relaxation Error:

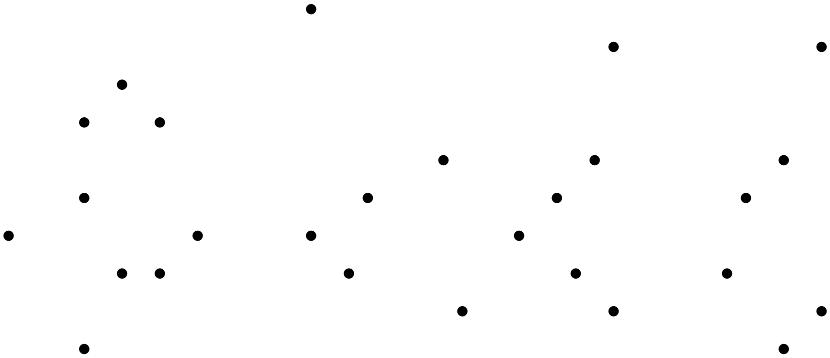
$$g(\mathcal{L}(\boldsymbol{w}) - \mathcal{L}^*) \leq \mathcal{R}(\boldsymbol{w}) - \mathcal{R}^* \tag{9}$$

Empirical Risk Minimization



Goal: $\min_{\boldsymbol{w} \in \mathcal{F}} \mathcal{R}(\boldsymbol{w}) = \mathbf{E}_{\boldsymbol{x} \sim \mathcal{P}} f_{\boldsymbol{x}}(\boldsymbol{w})$

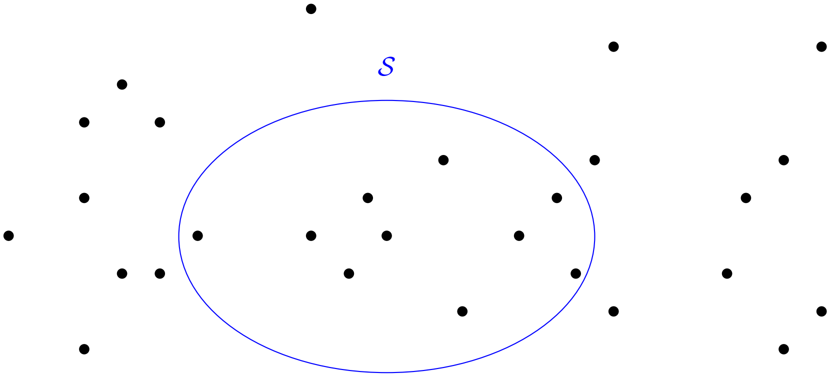
Empirical Risk Minimization



Goal: $\min_{\boldsymbol{w} \in \mathcal{F}} \mathcal{R}(\boldsymbol{w}) = \mathbf{E}_{\boldsymbol{x} \sim \mathcal{P}} f_{\boldsymbol{x}}(\boldsymbol{w})$

$\mathcal{P}$ is unknown!

Empirical Risk Minimization

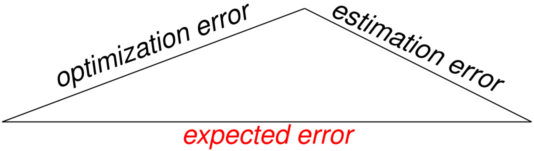


Goal: $\min_{\boldsymbol{w} \in \mathcal{F}} \mathcal{R}_{\mathcal{S}}(\boldsymbol{w})$,

$$\mathcal{R}_{\mathcal{S}}(\boldsymbol{w}) := \frac{1}{n} \sum_{\boldsymbol{x} \in \mathcal{S}} f_{\boldsymbol{x}}(\boldsymbol{w})$$

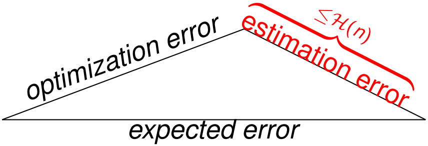
Optimization-Estimation Error Decomposition

- **Goal:** bound expected error $\mathcal{R}(\boldsymbol{w}) - \mathcal{R}^*$
- expected error $\leq$ optimization error + estimation error[4]



[4] Bousquet, O. & Bottou, L. *The tradeoffs of large scale learning.* in *NIPS* (2008).

Generalization Error



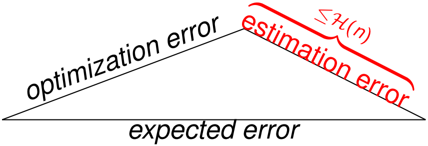
- Data **independent** uniform convergence bounds:

$$\mathcal{R}(\boldsymbol{w}) \leq \mathcal{R}_{\mathcal{S}}(\boldsymbol{w}) + \mathrm{VC}/\sqrt{n}, \forall \boldsymbol{w}$$

- Data **dependent** uniform convergence bounds:

$$\mathcal{R}(\boldsymbol{w}) \leq \mathcal{R}_{\mathcal{S}}(\boldsymbol{w}) + c_1 \sqrt{\mathrm{var}(\mathcal{R}(\boldsymbol{w}))/n} + c_2/n, \forall \boldsymbol{w}$$

Estimation Error



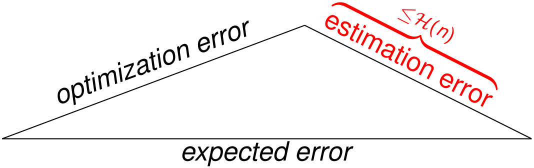
- Regular estimation error

$$\mathbf{E}_{\mathcal{S}}|\mathcal{R}^*_{\mathcal{S}} - \mathcal{R}^*| \leq \mathrm{VC}/\sqrt{n}$$

- Fast estimation error

$$\mathbf{E}_{\mathcal{S}}|\mathcal{R}^*_{\mathcal{S}} - \mathcal{R}^*| \leq \mathrm{RC}/n$$

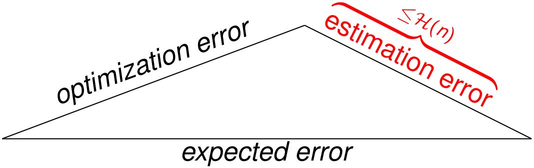
Estimation Error



- Estimation error can be bounded as:

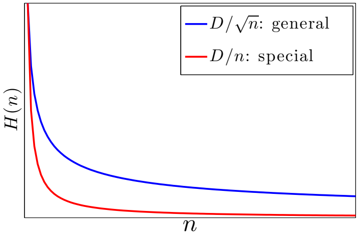
$$\mathbf{E}_{\mathcal{S}}\left[\mathcal{R}^*_{\mathcal{S}} - \mathcal{R}^*\right] \leq \mathcal{H}(n)$$

Estimation Error

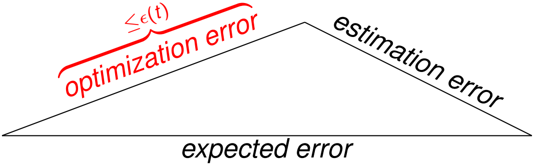


- Estimation error can be bounded as:

$$\mathbf{E}_{\mathcal{S}}\left[\mathcal{R}_{\mathcal{S}}^{*} - \mathcal{R}^{*}\right] \leq \mathcal{H}(n)$$

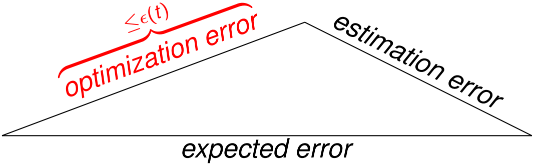


Optimization Error



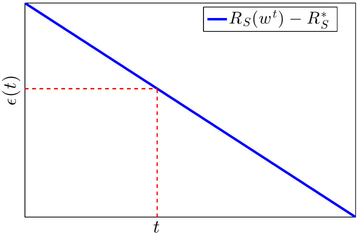
- Standard assumptions: μ-strongly convex and L-smoothness

Optimization Error



- Standard assumptions: μ-strongly convex and L-smoothness
  - Computational limits causes optimization error as:

$$\mathcal{R}_{\mathcal{S}}(\boldsymbol{w}^t) - \mathcal{R}^*_{\mathcal{S}} \leq \epsilon(t)$$



Nesterov's Accelerated Gradient Descent

- ODE model for gradient descent update:

$$\frac{\boldsymbol{x}_{t+1} - \boldsymbol{x}_t}{\eta} = -\nabla f(\boldsymbol{x}) \Rightarrow \dot{\boldsymbol{x}}(t) + \nabla f(\boldsymbol{x}(t)) = 0 \tag{10}$$

- Heavy ball ODE (Boris Polyak):

$$\ddot{\boldsymbol{x}}(t) + \dot{\boldsymbol{x}}(t) + \nabla f(\boldsymbol{x}(t)) = 0 \tag{11}$$

- Discretized heavy ball ODE obtains AGD updates (Yurii Nesterov)e.
- Improvement: $(1 - \mu/L)^t \Rightarrow (1 - \sqrt{\mu/L})^t$.

Stochastic Gradient Descent

- SGD uses a stochastic approximation of gradient:

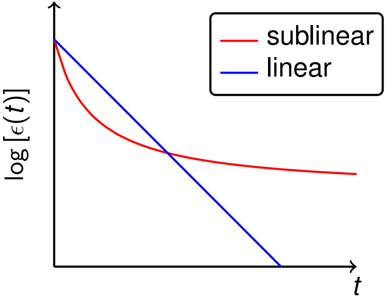
$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \nabla_r f(\boldsymbol{w}_t), \mathbf{E}_r \left[\nabla_r f(\boldsymbol{w}_t)\right] = \nabla f(\boldsymbol{w}_t) \tag{12}$$

- For small-scale learning, "stochastic optimization is wasteful"[5]
- SGD is popular for large-scale learning.

[5] Vladimir Vapnik

SGD vs. GD

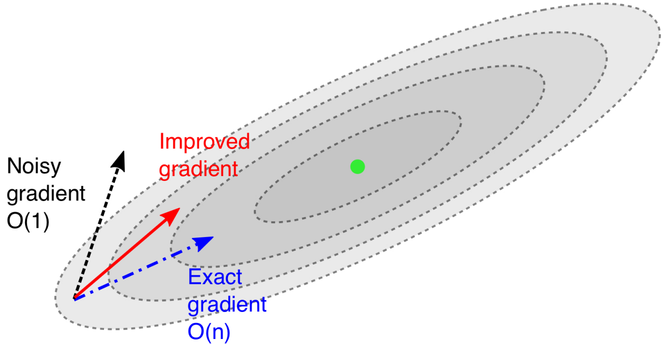
method	update	complexity	$\epsilon(t)$
GD	$\boldsymbol{w}_t - \sum_{i=1}^{n} \nabla f_i(\boldsymbol{w}_t)/(Ln)$	$n \times d$	$(1 - \mu/L)^t c_0$
SGD	$\boldsymbol{w}_t - \nabla f_r(\boldsymbol{w}_t)/(tn)$	d	c_0/t



Optimization with SVRG

- Variance reduced SGD: SAGA[6], SVRG, SAG, etc.
- These methods use a variance correction term as:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \left[\nabla f_{\mathbf{x}}(\mathbf{w}^t) - g_{\mathbf{x}} \right], \quad g_{\mathbf{x}} := \nabla f_{\mathbf{x}}(\mathbf{w}^{old}) - \tilde{\nabla} \mathcal{R}_{\mathcal{S}}$$



[6] Defazio, A., Bach, F. & Lacoste-Julien, S. *SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives.* in *NIPS* (2014).

Stochastic Dual Coordinate Ascent

- Primal Ridge Regression:

$$\mathcal{Q}_p(\boldsymbol{w}) = \frac{1}{2n}\boldsymbol{w}^\top(\mathbf{X}^\top\mathbf{X} + \mu\mathbf{I})\boldsymbol{w} - \frac{1}{n}y^\top\mathbf{X}\boldsymbol{w}, \boldsymbol{w} \in \mathbb{R}^d \tag{13}$$

- Dual Ridge Regression:

$$\mathcal{Q}_d(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^\top\left(\mathbf{X}\mathbf{X}^\top/(\mu n) + \mathbf{I}\right)\boldsymbol{\alpha} - y^\top\boldsymbol{\alpha}/(n), \boldsymbol{\alpha} \in \mathbb{R}^n \tag{14}$$

- SDCA update:

$$\boldsymbol{\alpha}^+(r) = \max_{\boldsymbol{\alpha}_r} \mathcal{Q}_d(\boldsymbol{\alpha}), r \sim \text{uniform}\{1, \dots, n\} \tag{15}$$

SDCA vs SGD

- Primal interpretation of SDCA:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_r(t)\nabla f_r(\boldsymbol{w}_t) \tag{16}$$

- SGD udpates:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta(t)\nabla f_r(\boldsymbol{w}_t) \tag{17}$$

- SDCA modification improves sub-linear convergence of SGD to a linear convergence rate.

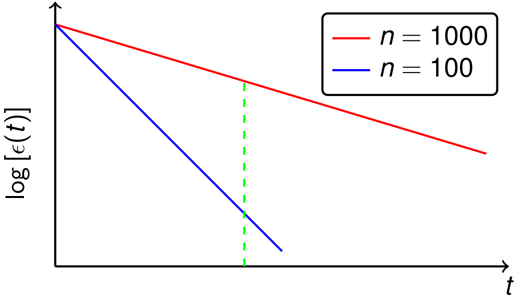
Challenge of Large Scale Learning

$$\epsilon(t) = \left(1 - \min\left\{\frac{\mu}{L}, \frac{1}{n}\right\}\right)^t C$$

In large scale setting, convergence rate is $\rho_n = 1 - 1/n$

$$\epsilon(t) = \left(1 - \min\left\{\frac{\mu}{L}, \frac{1}{n}\right\}\right)^t C$$

In large scale setting, convergence rate is $\rho_n = 1 - 1/n$



Why a Smaller Set?

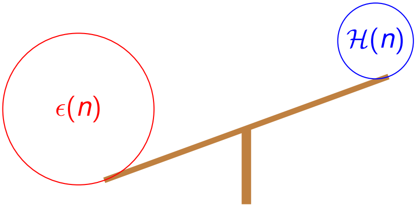
- Baseline: one "epoch" with full sample:

$$\epsilon(n) = \left(1 - \frac{1}{n}\right)^n C \simeq \frac{C}{e}$$

Why a Smaller Set?

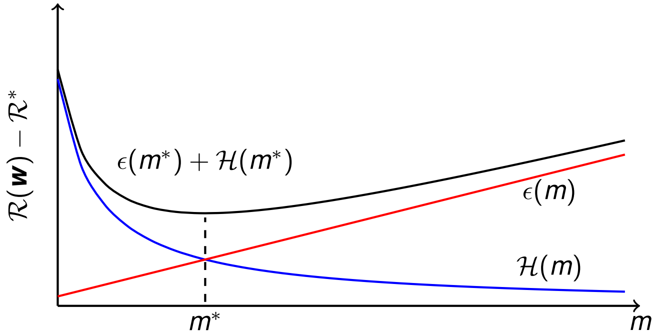
- Baseline: one "epoch" with full sample:

$$\epsilon(n) = \left(1 - \frac{1}{n}\right)^n C \simeq \frac{C}{e}$$



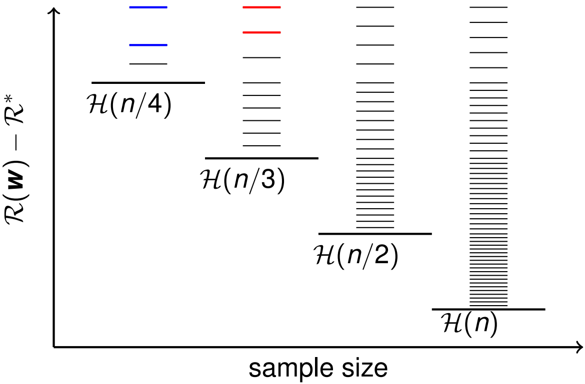
- expected error $\simeq \epsilon(n) \simeq \frac{C}{e}$

Optimization-Estimation Tradeoff



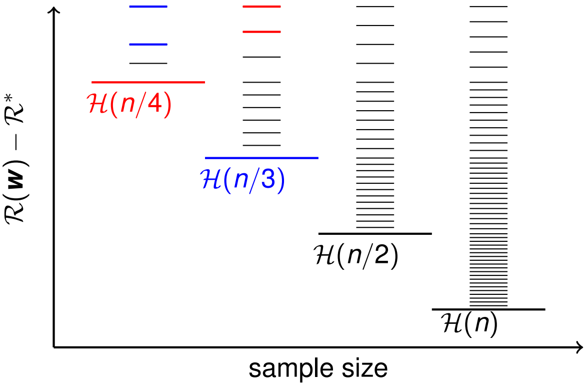
Optimization with Dynamic Sample Size

- Smaller set:
  - faster convergence

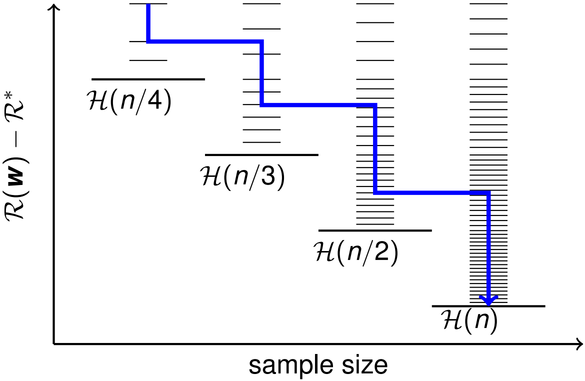


Optimization with Dynamic Sample Size

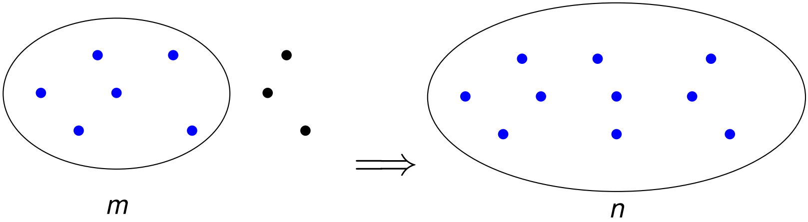
- Smaller set:
  - faster convergence
  - larger estimation error



Optimization with Dynamic Sample Size

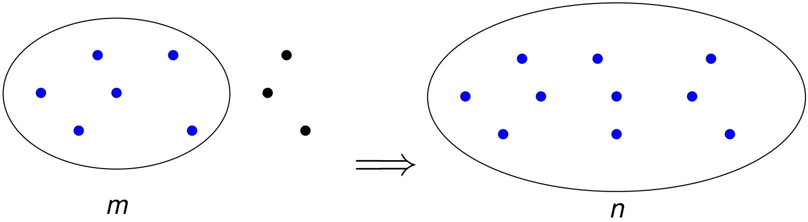


Switching Sample Size



$$\mathcal{R}_m(\boldsymbol{w}) - \mathcal{R}_m^* \leq \epsilon$$

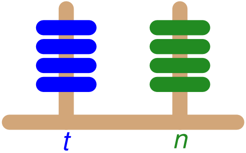
Switching Sample Size



$$\mathcal{R}_m(\boldsymbol{w}) - \mathcal{R}_m^* \leq \epsilon \qquad\qquad \mathcal{R}_n(\boldsymbol{w}) - \mathcal{R}_n^* \leq \epsilon + \boxed{\frac{n-m}{n}\mathcal{H}(m)}$$

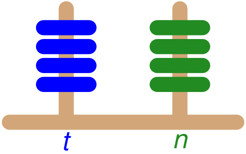
Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$



Finding Optimal Strategy

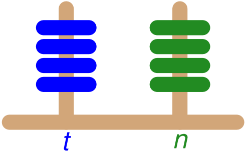
$\mathbf{U}(t, n) = ?$



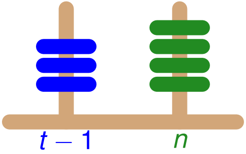
Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$

Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$

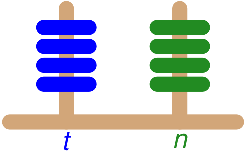


Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$

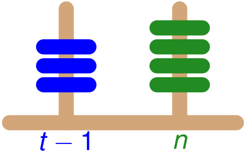


Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$



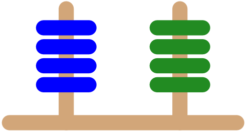
Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$



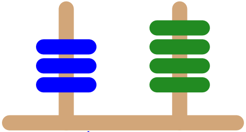
$$\alpha = \boxed{\left(1 - \frac{1}{n}\right)} \times \mathbf{U}(t-1, n)$$

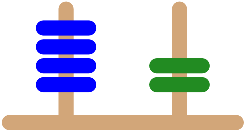
Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$



Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$

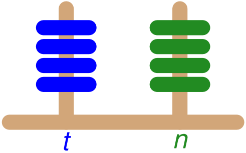




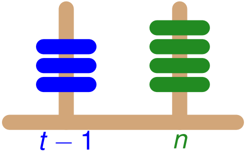
$$\alpha = \boxed{\left(1 - \frac{1}{n}\right)} \times \mathbf{U}(t-1, n)$$

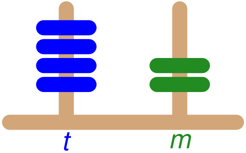
Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$



Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$



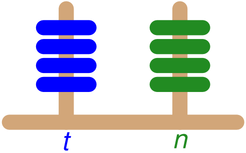


$$\alpha = \boxed{\left(1 - \frac{1}{n}\right)} \times \mathbf{U}(t-1, n)$$

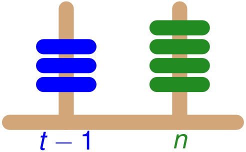
$$\beta = \min_{m<n} \left[\mathbf{U}(t, m) + \boxed{\frac{n-m}{n} \mathcal{H}(m)} \right]$$

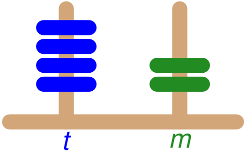
Finding Optimal Strategy

$\mathbf{U}(t, n) = ?$



Given $\mathbf{U}(k, m)$
$k < t$ or $m < n$





$$\alpha = \boxed{\left(1 - \frac{1}{n}\right)} \times \mathbf{U}(t-1, n)$$

$$\beta = \min_{m<n} \left[\mathbf{U}(t, m) + \boxed{\frac{n-m}{n} \mathcal{H}(m)} \right]$$

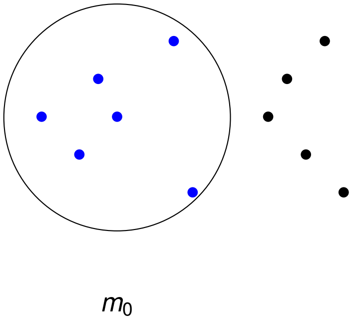
$$U(t, n) = min(\alpha, \beta)$$

Which Strategy Minimizes $\mathbf{U}(n, n)$?

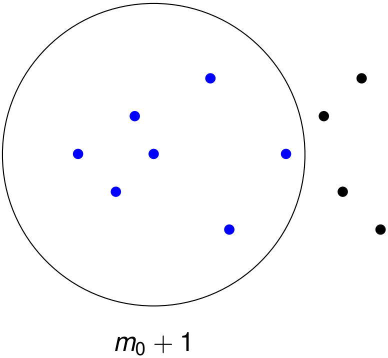
- We are interested in the case when $t = n$, i.e. $\mathbf{U}(n, n)$.
- LINEAR strategy minimizes $\mathbf{U}(n, n)$
  - Start with sample size $M_0 = L/\mu$ with $T_0 = 2M_0$.
  - Then *linearly* increase the size of set, i.e.

$$M(t) = \left\lceil \frac{t}{2} \right\rceil$$

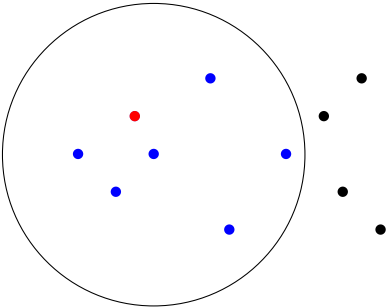
LINEAR Strategy



LINEAR Strategy

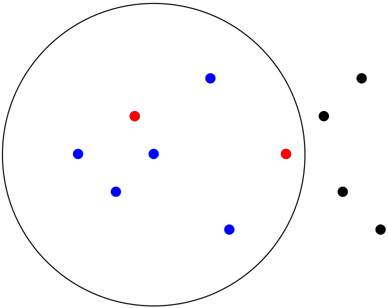


LINEAR Strategy



$m_0 + 1$

LINEAR Strategy



$m_0 + 1$

Theoretical Improvement

- DYNASAGA: SAGA with LINEAR sample size strategy

METHOD	OPTIMIZATION ERROR $\epsilon(n)$
DYNASAGA	$O(\mathcal{H}(n))$
SAGA	const.

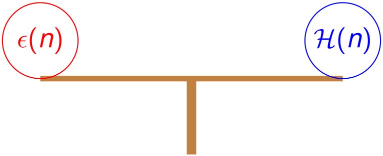
- Why $\epsilon(n) \simeq \mathcal{H}(n)$?

Theoretical Improvement

- DYNASAGA: SAGA with LINEAR sample size strategy

METHOD	OPTIMIZATION ERROR $\epsilon(n)$
DYNASAGA	$O(\mathcal{H}(n))$
SAGA	const.

- Why $\epsilon(n) \simeq \mathcal{H}(n)$?



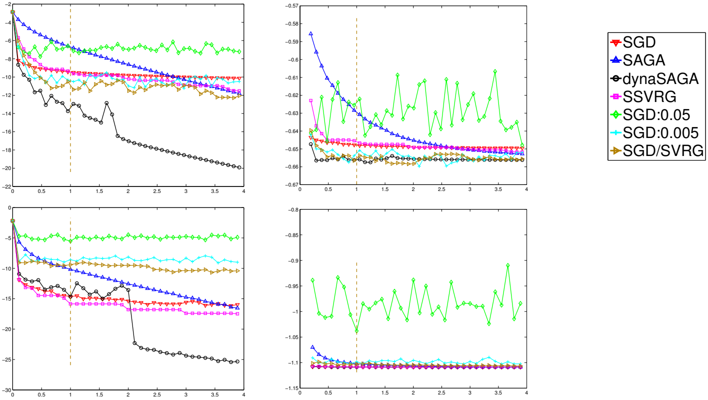
expected error $\simeq \max\{\epsilon(n), \mathcal{H}(n)\}$

Experiments

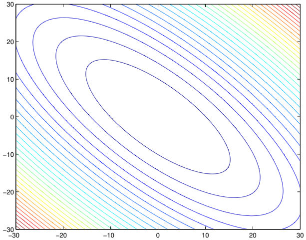
optimization error | test error

COVTYPE
$n = 580K$
$d = 54$

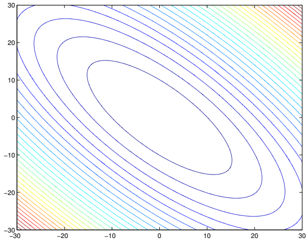
SUSY
$n = 5M$
$d = 18$



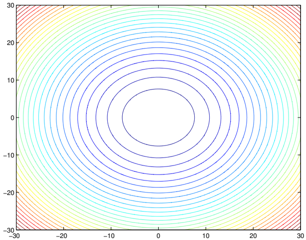
Ill-conditioned objective



Ill-conditioned objective



$\Rightarrow$



Newton's Method

$$\mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}) = g_{\mathcal{S}}(\boldsymbol{w}) + \frac{\gamma}{2}\|\boldsymbol{w}\|^2, \;\; g_{\mathcal{S}}(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell_{\boldsymbol{x}_i}(\boldsymbol{w})$$

- Newton's method involves the curvature of the objective in optimization

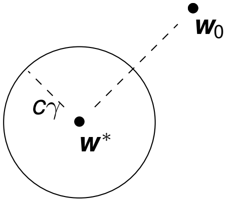
$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta H_{\mathcal{S},\gamma}^{-1}(\boldsymbol{w}^t)\left[\nabla \mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}^t)\right]$$

$$H_{\mathcal{S},\gamma}(\boldsymbol{w}) := \frac{1}{n}\sum_{i=1}^{n} \nabla^2 \ell_{\boldsymbol{x}_i}(\boldsymbol{w}) + \gamma \mathbf{I}$$

- Time complexity per iteration: $O(nd^2 + d^3)$
  - nd^2 for computing the Hessian matrix H and gradient $\nabla\mathcal{R}$
  - d^3 for inverting H

Global Convergence

- Sub-linear rate
- $\mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}^{t+1}) \leq \mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}^{t}) - \beta$



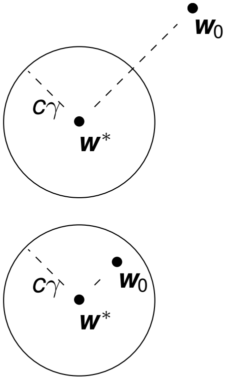
Convergence of Newton's Method

Global Convergence

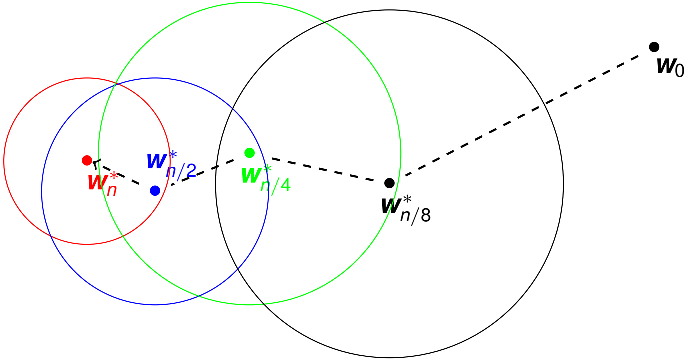
- Sub-linear rate
- $\mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}^{t+1}) \leq \mathcal{R}_{\mathcal{S},\gamma}(\boldsymbol{w}^{t}) - \beta$

Local Convergence

- Super-linear
- $\lambda(\boldsymbol{w}^{t+1}) \leq \left(\frac{\lambda(\boldsymbol{w}^t)}{1-\lambda(\boldsymbol{w}^t)}\right)^2$
- $\lambda(\boldsymbol{w}) :=$ $\langle H^{-1}(\boldsymbol{w})\nabla\mathcal{R}(\boldsymbol{w}), \nabla\mathcal{R}(\boldsymbol{w})\rangle^{1/2}$



Adaptive Newton



Summary

Large-scale classification:

- Convex relaxation of classification
- Using stochastic optimization to relax computational complexity
- Adaptive sample size to balance computational-statistical trade-off.

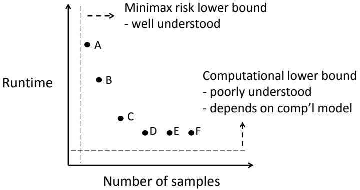
Off the convex path

Figure: Off the convex path [7]

[7] http://www.offconvex.org/

Open questions

- Which convex relaxation is better for large-scale learning?
- How we can achieve the computational lower-bound for classification?



- Classification without convex-relaxation?

Thank you for you attention!

Question mark credit: Mania Orand

Latex template credit: Lilyana Vankova