Tutorial: PART 1

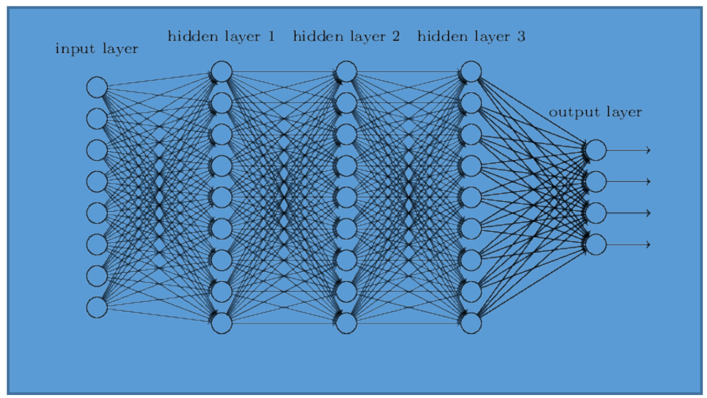
Optimization for machine learning

Elad Hazan
Princeton University

+ help from Sanjeev Arora, Yoram Singer

ML paradigm

Machine



Chair/car

Distribution over $\{a\} \in R^n$

label

$b = f_{parameters}(a)$

This tutorial - training the machine

- Efficiency
- generalization

Agenda

1. Learning as mathematical optimization
   - Stochastic optimization, ERM, online regret minimization
   - Offline/online/stochastic gradient descent
2. Regularization
   - AdaGrad and optimal regularization
3. Gradient Descent++
   - Frank-Wolfe, acceleration, variance reduction, second order methods, non-convex optimization

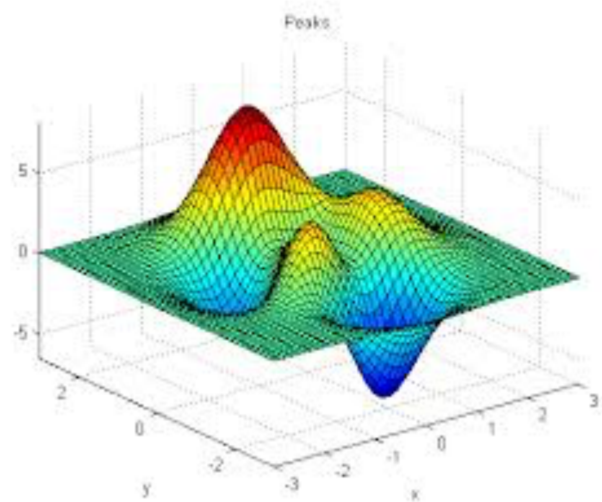
NOT touch upon:

- Parallelism/distributed computation (asynchronous optimization, HOGWILD etc.), Bayesian inference in graphical models, Markov-chain-monte-carlo, Partial information and bandit algorithms

Mathematical optimization

Input: function $f: K \mapsto R$, for $K \subseteq R^d$

Output: minimizer $x \in K$, such that $f(x) \leq f(y) \; \forall \; y \in K$

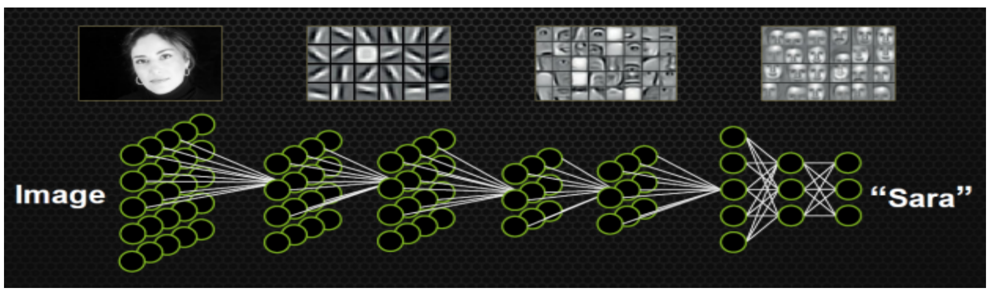


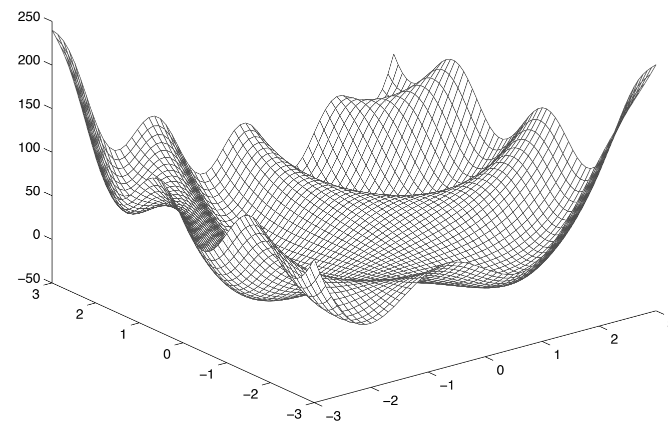
Accessing f? (values, differentials, …)

Generally NP-hard, given full access to function.

Learning = optimization over data
(a.k.a. Empirical Risk Minimization)

Fitting the parameters of the model ("training") = optimization problem:





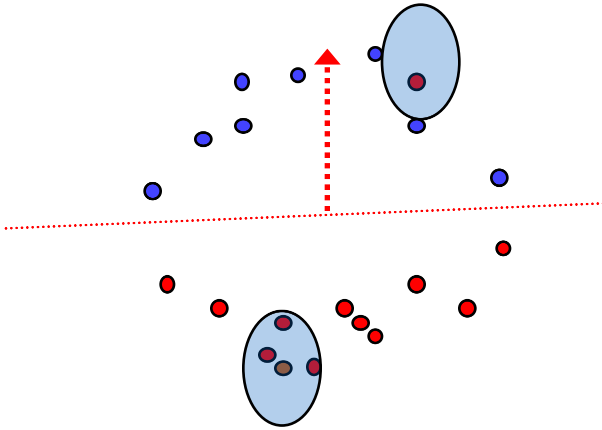
$$\arg\min_{x \in R^d} \frac{1}{m} \sum_{i=1\ to\ m} \ell_i(x, a_i, b_i) + R(x)$$

m = # of examples (a,b) = (features, labels)

d = dimension

Example: linear classification

Given a sample $S = \{(a_1, b_1), \ldots, (a_m, b_m)\}$,
find hyperplane (through the origin w.l.o.g)
such that:



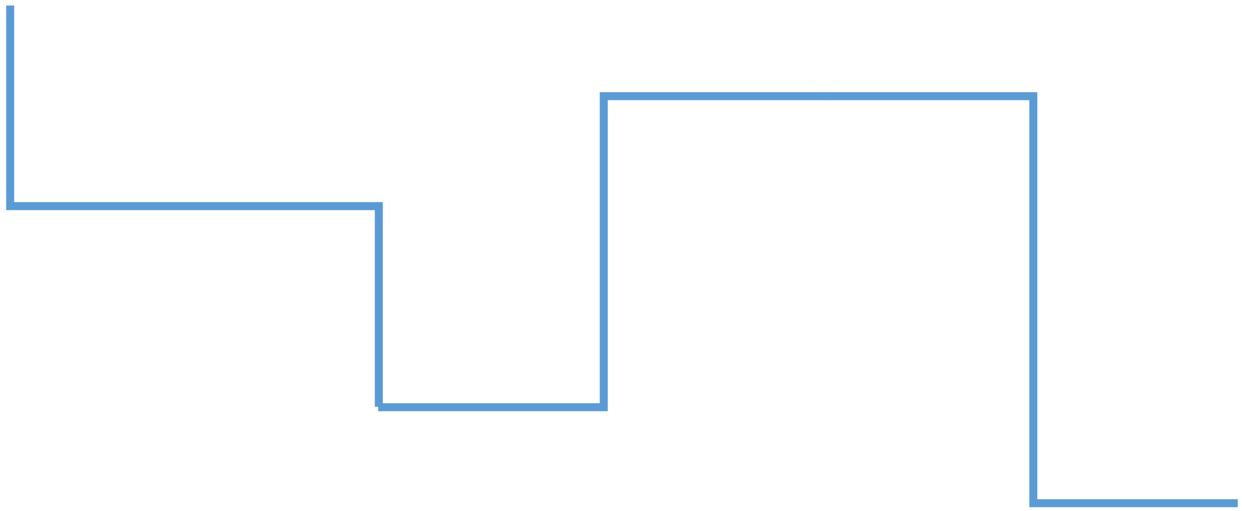
$x = \arg\min_{|x| \le 1}$ # of mistakes =

$$\arg\min_{|x| \le 1} \; |\{i \;\; s.t.\; sign\,(x^T a_i) \neq b_i\}|$$

$$\arg\min_{|\mathrm{x}| \le 1} \frac{1}{m} \sum_i \ell(x, a_i, b_i) \quad \text{for} \;\; \ell(x, a_i, b_i) = \begin{cases} 1 & x^\top a \neq b \\ 0 & x^\top a = b \end{cases}$$

NP hard!

Sum of signs → global optimization NP-hard!
but locally verifiable…



Local property that ensures global optimality?

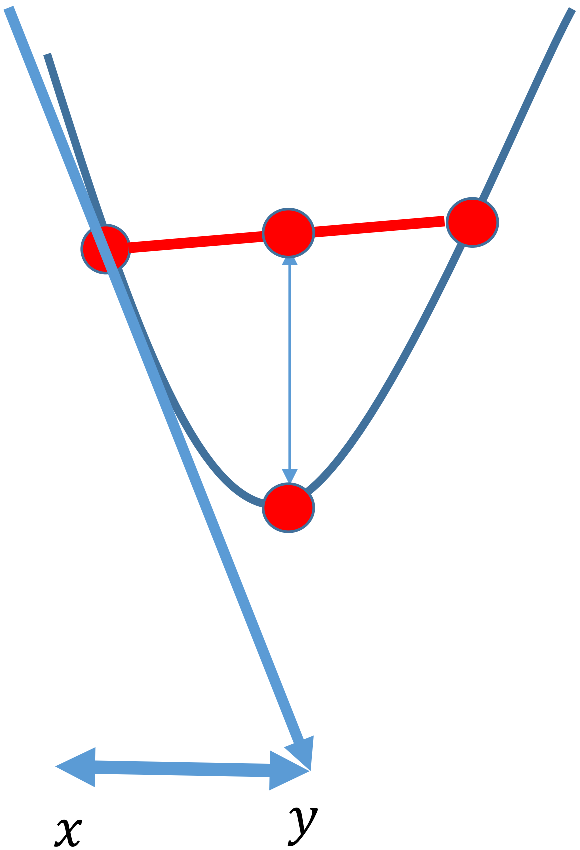
Convexity

A function $f: R^d \mapsto R$ is convex if and only if:

$$f\left(\frac{1}{2}x + \frac{1}{2}y\right) \leq \frac{1}{2}f(x) + \frac{1}{2}f(y)$$

- Informally: smiley ☺
- Alternative definition:

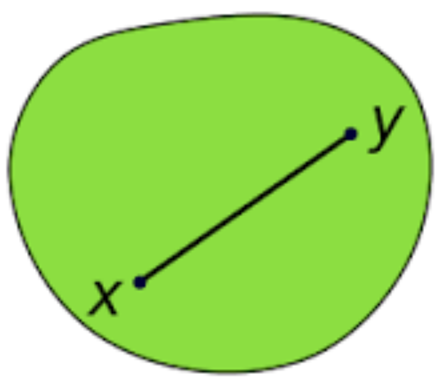
$$\mathrm{f(y)} \geq \mathrm{f(x)} + \nabla f(x)^\top (y - x)$$

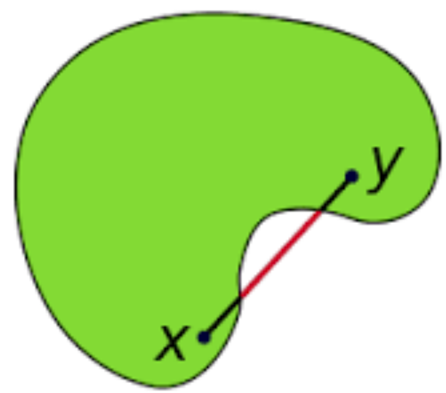


Convex sets

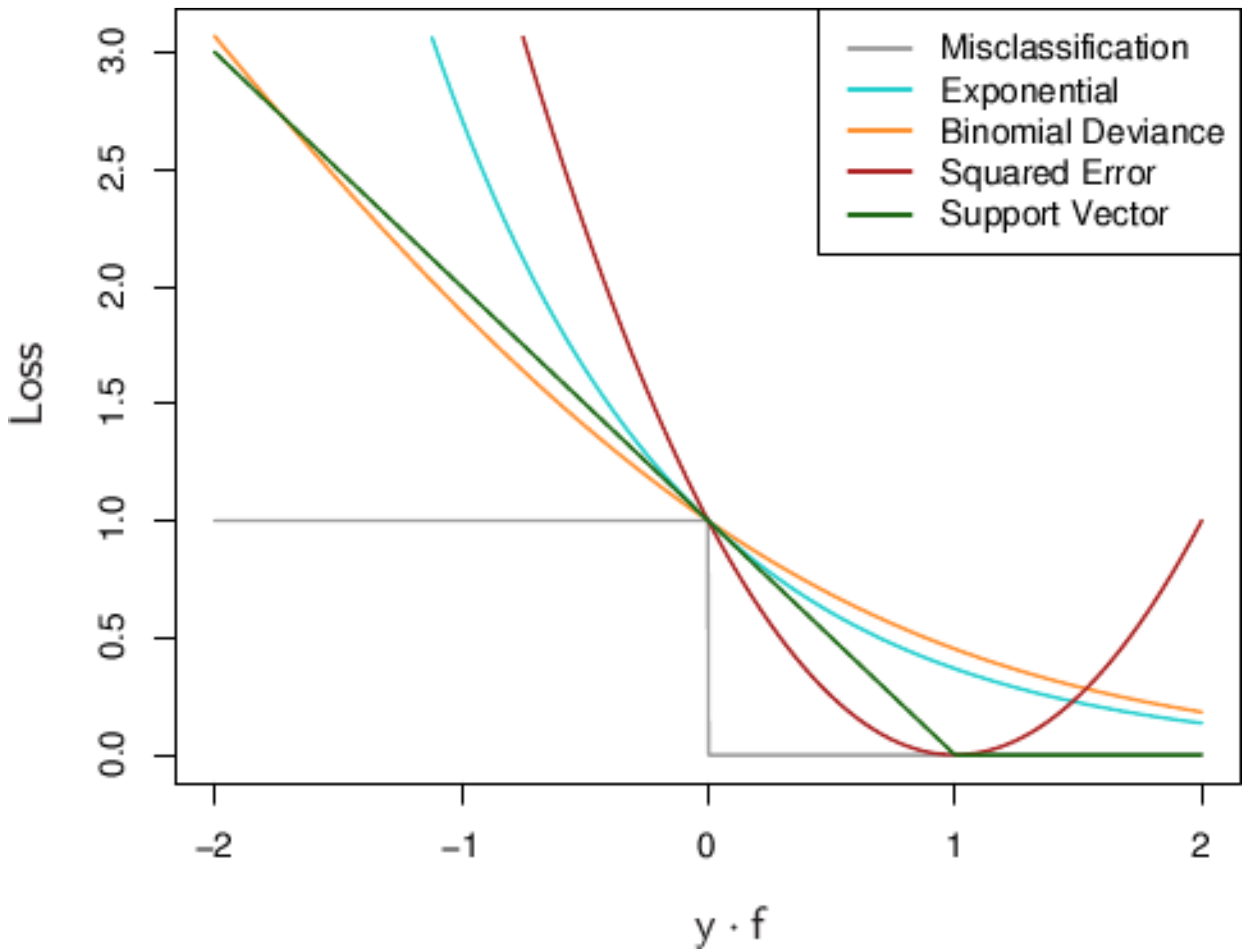
Set K is convex if and only if:

$x, y \in K \ \Rightarrow (½x + ½y) \in K$



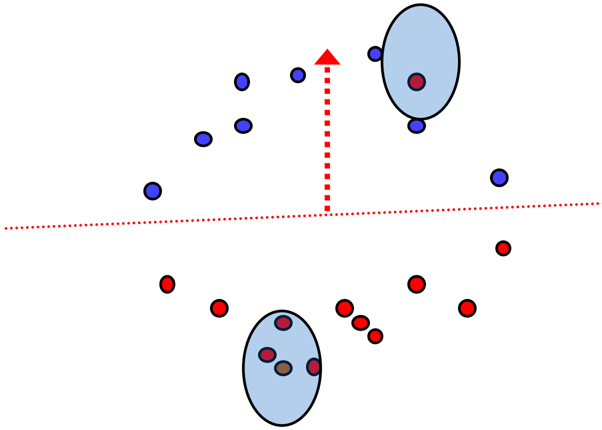


Loss functions $\ell(x, a_i, b_i) = \ell(x^T a_i \cdot b_i)$



Convex relaxations for linear (&kernel) classification

$$x = \arg\min_{|x| \leq 1} |\{i \ \ s.t.\ sign\,(x^T a_i) \neq b_i\}|$$



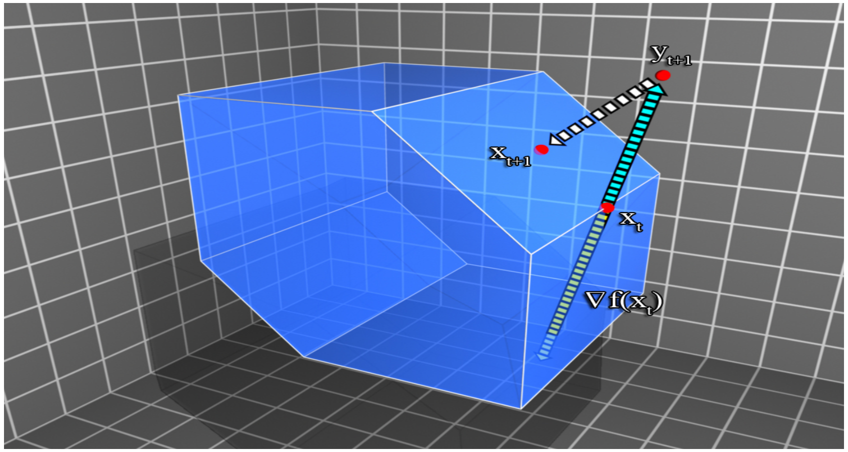
1. Ridge / linear regression $\ell(x^\top a_i, y_i) = (x^\top a_i - b_i)^2$
2. SVM $\ell(x^\top a_i, y_i) = \max\{0, 1 - b_i \ x^\top a_i\}$
3. Logistic regression $\ell(x^\top a_i, y_i) = \log(1 + e^{-b_i \cdot x^\top a_i})$

We have: cast learning as mathematical optimization, argued convexity is algorithmically important

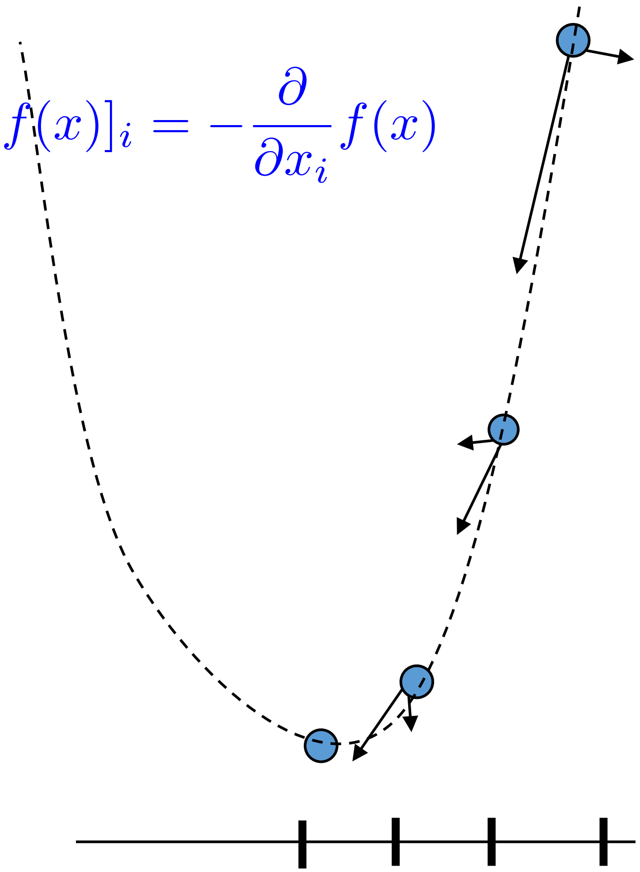
Next ➔ algorithms!

Gradient descent, constrained set

$$y_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$$
$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$



$$-[\nabla f(x)]_i = -\frac{\partial}{\partial x_i} f(x)$$



Convergence of gradient descent

$$y_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$$
$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$

Theorem: for step size $\eta = \frac{D}{G\sqrt{T}}$

$$f\left(\frac{1}{T}\sum_t x_t\right) \leq \min_{x^* \in K} f(x^*) + \frac{DG}{\sqrt{T}}$$

Where:

- G = upper bound on norm of gradients

$$|\nabla f(x_t)| \leq G$$

- D = diameter of constraint set

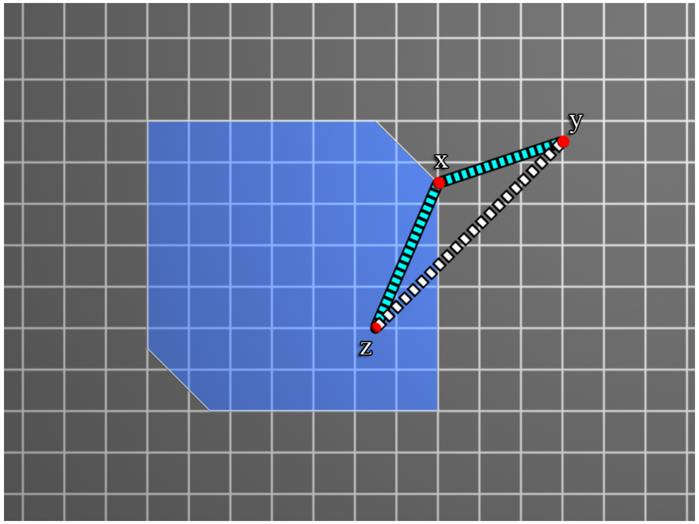
$$\forall x, y \in K \quad . \quad |x - y| \leq D$$

$$y_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$$
$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$

Proof:

1. Observation 1:
$$|x^* - y_{t+1}|^2 = |x^* - x_t|^2 - 2\eta \nabla f(x_t)(x_t - x^*) + \eta^2 |\nabla f(x_t)|^2$$
2. Observation 2:
$$|x^* - x_{t+1}|^2 \leq |x^* - y_{t+1}|^2$$

This is the Pythagorean theorem:



$$y_{t+1} \leftarrow x_t - \eta \nabla f(x_t)$$
$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$

Proof:

1. Observation 1:
$$|x^* - y_{t+1}|^2 = |x^* - x_t|^2 - 2\eta \nabla f(x_t)(x_t - x^*) + \eta^2 |\nabla f(x_t)|^2$$
2. Observation 2:
$$|x^* - x_{t+1}|^2 \leq |x^* - y_{t+1}|^2$$

Thus:

$$|x^* - x_{t+1}|^2 \leq |x^* - x_t|^2 - 2\eta \nabla f(x_t)(x_t - x^*) + \eta^2 G^2$$

And hence:

$$f\left(\frac{1}{T}\sum_t x_t\right) - f(x^*) \leq \frac{1}{T}\sum_t [f(x_t) - f(x^*)] \leq \frac{1}{T}\sum_t \nabla f(x_t)(x_t - x^*)$$

$$\leq \frac{1}{T}\sum_t \frac{1}{2\eta}(|x^* - x_{t+1}|^2 - |x^* - x_t|^2) + \frac{\eta}{2} G^2$$

$$\leq \frac{1}{T \cdot 2\eta} D^2 + \frac{\eta}{2} G^2 \leq \frac{DG}{\sqrt{T}}$$

Recap

Theorem: for step size $\eta = \frac{D}{G\sqrt{T}}$

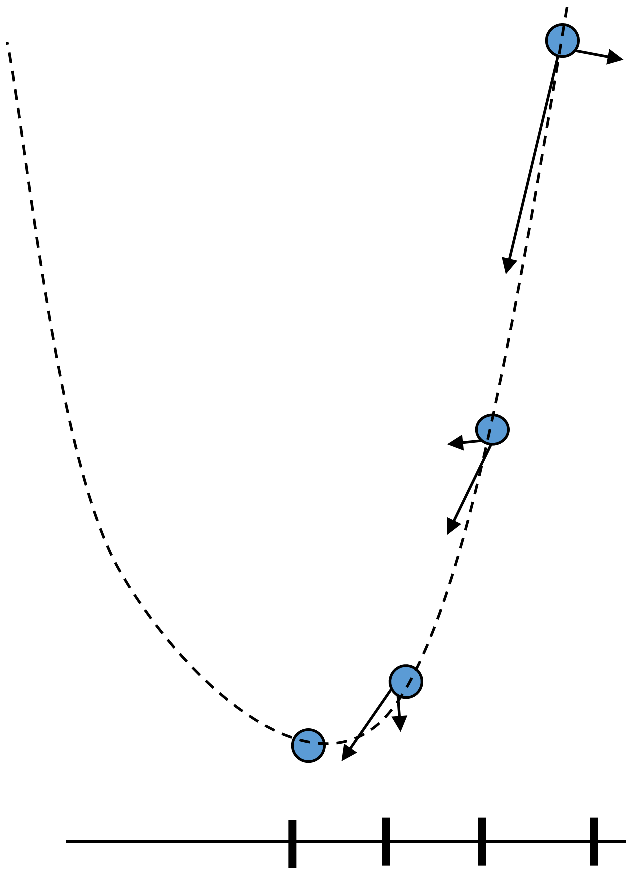
$$f\left(\frac{1}{T}\sum_t x_t\right) \leq \min_{x^* \in K} f(x^*) + \frac{DG}{\sqrt{T}}$$

Thus, to get ϵ-approximate solution, apply $O\left(\frac{1}{\epsilon^2}\right)$ gradient iterations.

Gradient Descent - caveat

For ERM problems

$$\arg\min_{x \in R^d} \frac{1}{m} \sum_{i=1 \; to \; m} \ell_i(x, a_i, b_i) + R(x)$$



1. Gradient depends on all data
2. What about generalization?

Next few slides:

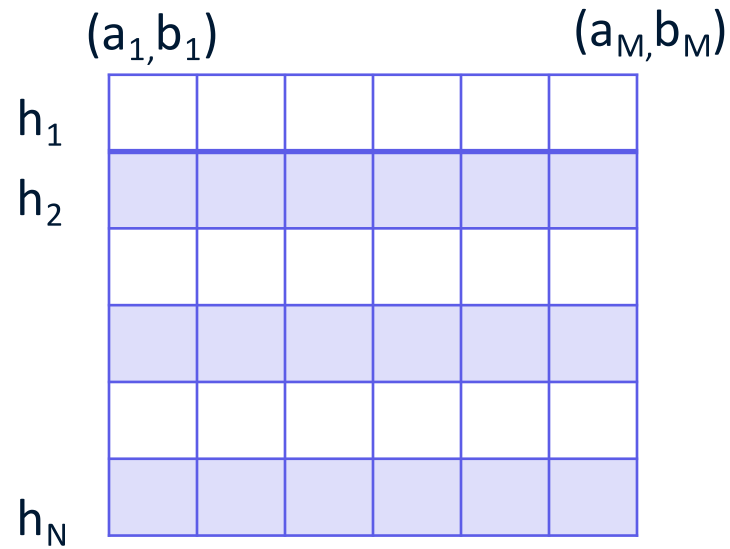
Simultaneous optimization and generalization

➔ Faster optimization! (single example per iteration)

Statistical (PAC) learning

Nature: i.i.d from distribution D over

$A \times B = \{(a, b)\}$



learner:

Hypothesis h

Loss, e.g. $\ell(h,(a,b)) = (h(a) - b)^2$

$$err(h) = \mathbb{E}_{a,b\sim D}[\ell(h,(a,b))]$$

Hypothesis class H: X -> Y is learnable if $\forall \epsilon, \delta > 0$ exists algorithm s.t. after seeing m examples, for $m = poly(\delta, \epsilon, dimension(H))$

finds h s.t. w.p. 1- δ:

$$\mathrm{err}(h) \leq \min_{h^* \in \mathcal{H}} \mathrm{err}(h^*) + \epsilon$$

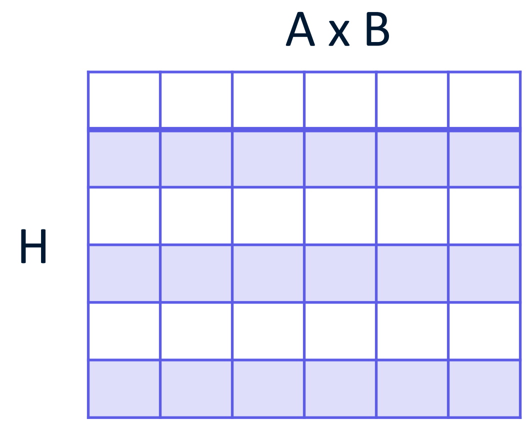
More powerful setting:
Online Learning in Games

Iteratively, for $t = 1, 2, \ldots, T$

Player: $h_t \in H$

Adversary: $(a_t, b_t) \in A$

Loss $\ell(h_t, (a_t, b_t))$



Goal: minimize (average, expected) regret:

$$\frac{1}{T}\left[\sum_t \ell(h_t, (a_t, b_t) - \min_{h^* \in \mathcal{H}} \sum_t \ell(h^*, (a_t, b_t))\right] \underset{T \to \infty}{\longrightarrow} 0$$

Vanishing regret → generalization in PAC setting! (online2batch)

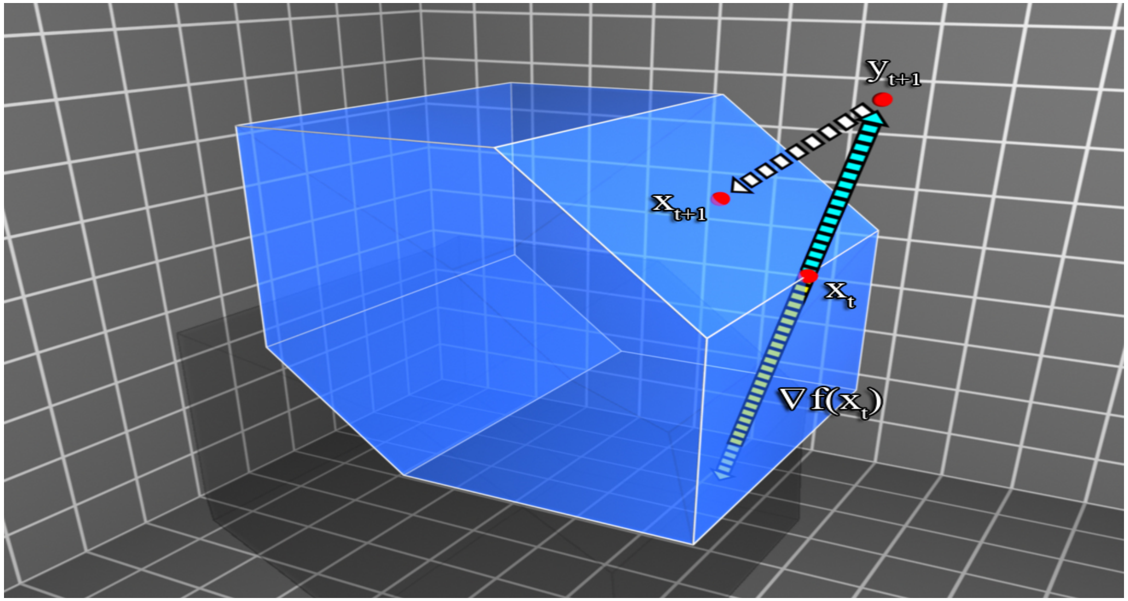
From this point onwards: $f_t(x) = \ell(x, a_t, b_t)$ = loss for one example

Can we minimize regret efficiently?

Online gradient descent [Zinkevich '05]

$$y_{t+1} = x_t - \eta \nabla f_t(x_t)$$

$$x_{t+1} = \arg\min_{x \in K} \|y_{t+1} - x_t\|$$



Theorem: Regret = $\sum_t f_t(x_t) - \sum_t f_t(x^*) = O(\sqrt{T})$

Analysis

$\nabla_t := \nabla f_t(x_t)$

Observation 1:

$$\|y_{t+1} - x^*\|^2 = \|x_t - x^*\|^2 - 2\eta \nabla_t (x^* - x_t) + \eta^2 \|\nabla_t\|^2$$

Observation 2: (Pythagoras)

$$\|x_{t+1} - x^*\| \leq \|y_{t+1} - x^*\|$$

Thus: $$\|x_{t+1} - x^*\|^2 \leq \|x_t - x^*\|^2 - 2\eta \nabla_t (x^* - x_t) + \eta^2 \|\nabla_t\|^2$$

Convexity: $$\sum_t [f_t(x_t) - f_t(x^*)] \leq \sum_t \nabla_t (x_t - x^*)$$

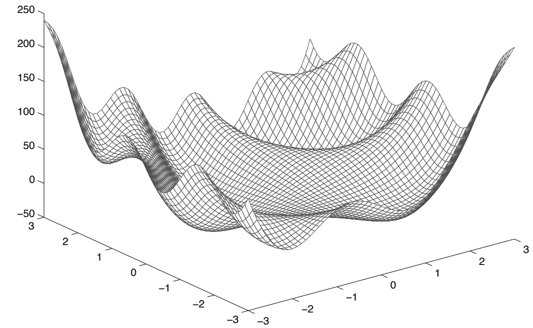
Lower bound

$$\text{Regret} = \Omega(\sqrt{T})$$

- 2 loss functions, T iterations:
  - $K = [-1,1],\ f_1(x) = x\ ,\ f_2(x) = -x$
  - Second expert loss = first * -1
- Expected loss = 0 (any algorithm)
- Regret = (compared to either -1 or 1)

$$E[|\#1's - \#(-1)'s|] = \Omega(\sqrt{T})$$

Stochastic gradient descent



Learning problem $\arg\min_{x \in R^d} F(x) = \ E_{(a_i,b_i)}[\ell_i(x, a_i, b_i)]$

random example: $f_t(x) = \ell_i(x, a_i, b_i)$

1. We have proved: (for any sequence of ∇_t)

$$\frac{1}{T}\sum_t \nabla_t^\top x_t \leq \min_{x^* \in K} \frac{1}{T}\sum_t \nabla_t^\top x^* + \frac{DG}{\sqrt{T}}$$

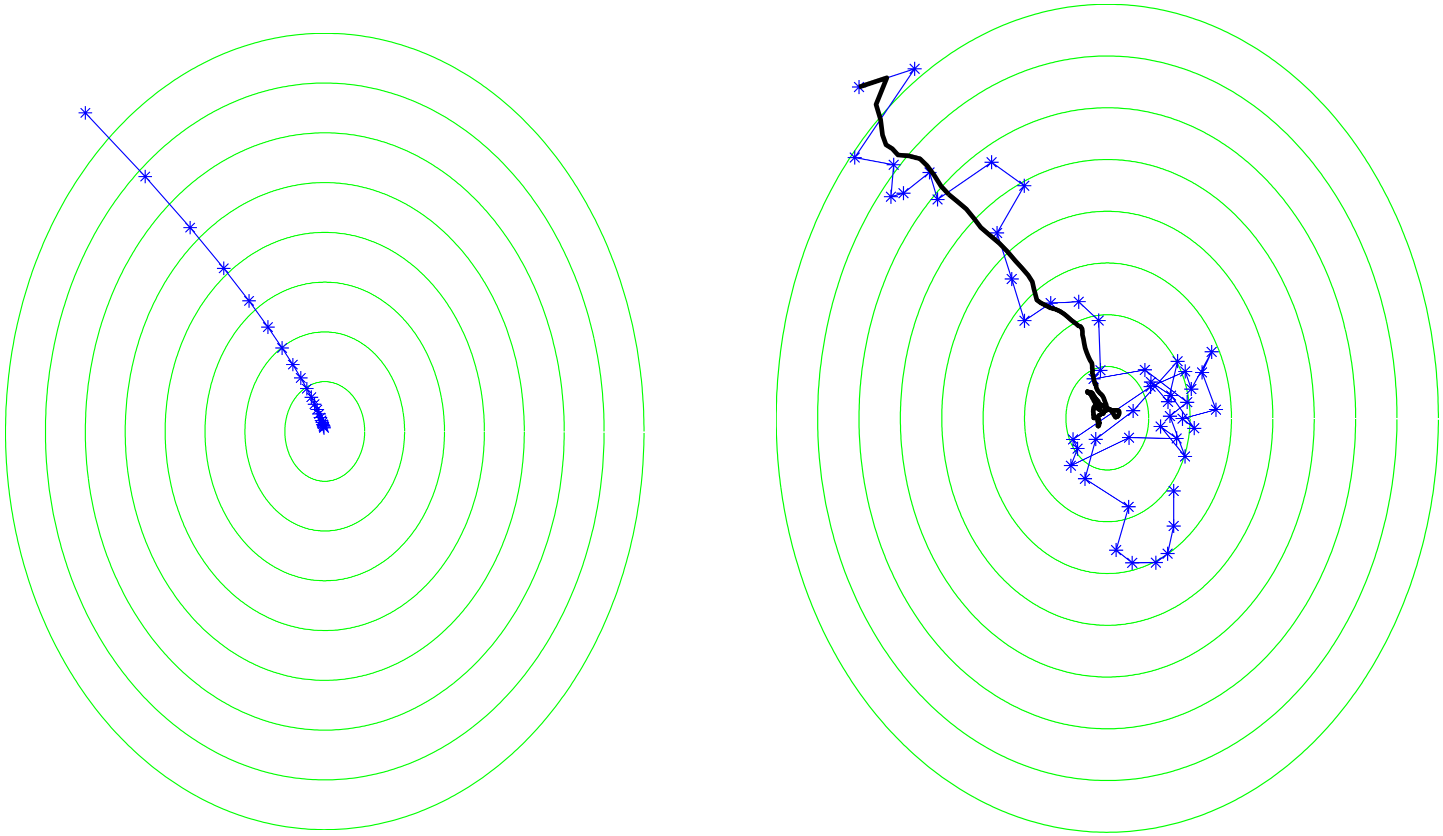
2. Taking (conditional) expectation:

$$E\left[F\left(\frac{1}{T}\sum_t x_t\right) - \min_{x^* \in K} F(x^*)\right] \leq E\left(\frac{1}{T}\sum_t \nabla_t^\top (x_t - x^*)]\right) \leq \frac{DG}{\sqrt{T}}$$

One example per step, same convergence as GD, & gives direct generalization! (formally needs martingales)

$O\left(\frac{d}{\epsilon^2}\right)$ vs. $O\left(\frac{md}{\epsilon^2}\right)$ total running time for ϵ generalization error.

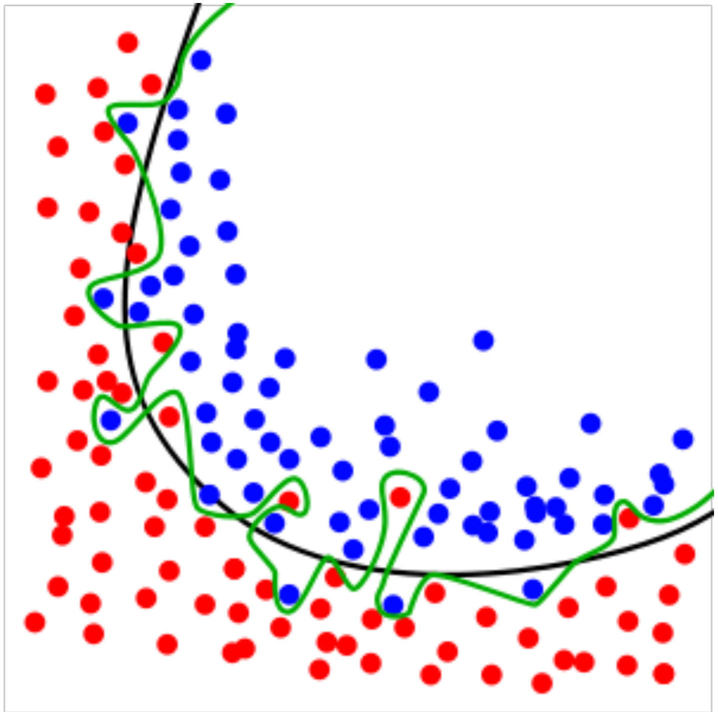
Stochastic vs. full gradient descent



Regularization & Gradient Descent++

Why "regularize"?

- Statistical learning theory / Occam's razor:
  # of examples needed to learn hypothesis class ~ it's "dimension"
  - VC dimension
  - Fat-shattering dimension
  - Rademacher width
  - Margin/norm of linear/kernel classifier



- PAC theory: Regularization <-> reduce complexity
- Regret minimization: Regularization <-> stability

Minimize regret: best-in-hindsight

$$\text{Regret} = \sum_t f_t(x_t) - \min_{x^* \in K} \sum_t f_t(x^*)$$

- Most natural:

$$x_t = \arg\min_{x \in K} \sum_{i=1}^{t-1} f_i(x)$$

- Provably works [Kalai-Vempala'05]:

$$x_t' = \arg\min_{x \in K} \sum_{i=1}^{t} f_i(x) = x_{t+1}$$

- So if $x_t \approx x_{t+1}$, we get a regret bound
- But instability $|x_t - x_{t+1}|$ can be large!

Fixing FTL: Follow-The-Regularized-Leader (FTRL)

- Linearize: replace f_t by a linear function, $\nabla f_t(x_t)^T x$
- Add **regularization:**

$$x_t = \arg\min_{x \in K} \sum_{i=1 \ldots t-1} \nabla_t^{\top} x + \frac{1}{\eta} R(x)$$

- *R(x)* is a strongly convex function, ensures stability:

$$\nabla_t^{\top} (x_t - x_{t+1}) = O(\eta)$$

FTRL vs. gradient descent

- $R(x) = \frac{1}{2} \| x \|^2$

$$x_t = \arg\min_{x \in K} \sum_{i=1}^{t-1} \nabla f_i(x_i)^\top x + \frac{1}{\eta} R(x)$$
$$= \prod_K \left(-\eta \sum_{i=1}^{t-1} \nabla f_i(x_i) \right)$$

- *Essentially* OGD: starting with $y_1 = 0$, for $t = 1, 2, \ldots$

$$x_t = \prod_K (y_t)$$
$$y_{t+1} = y_t - \eta \nabla f_t(x_t)$$

FTRL vs. Multiplicative Weights

- Experts setting: $K = \Delta_n$ distributions over experts
- $f_t(x) = c_t^T x$, where c_t is the vector of losses
- $R(x) = \sum_i x_i \log x_i$: negative entropy

$$x_t = \arg\min_{x \in K} \sum_{i=1}^{t-1} \nabla f_i(x_i)^\top x + \frac{1}{\eta} R(x)$$

$$= \exp\left(-\eta \sum_{i=1}^{t-1} c_i\right) / Z_t$$

Entrywise exponential

Normalization constant

- Gives the Multiplicative Weights method!

FTRL ⇔ Online Mirror Descent

$$x_t = \arg\min_{x \in K} \sum_{i=1}^{t-1} \nabla f_i(x_i)^\top x + \frac{1}{\eta} R(x)$$

Bregman Projection:

$$\prod_K^R(y) = \arg\min_{x \in K} B_R(x \| y)$$

$$B_R(x \| y) := R(x) - R(y) - \nabla R(y)^\top (x - y)$$

$$x_t = \prod_K^R(y_t)$$

$$y_{t+1} = (\nabla R)^{-1}(\nabla R(y_t) - \eta \nabla f_t(x_t))$$

Adaptive Regularization: AdaGrad

- Consider generalized linear model, prediction is function of $a^T x$

$$\nabla f_t(x) = \ell(a_t, b_t, x) a_t$$

- OGD update: $x_{t+1} = x_t - \eta \nabla_t = x_t - \eta \ell(a_t, b_t, x) a_t$
- features treated equally in updating parameter vector

- In typical text classification tasks, feature vectors a_t are very sparse, Slow learning!

- Adaptive regularization: per-feature learning rates

Optimal regularization

- The general RFTL form

$$x_t = \arg\min_{x \in K} \sum_{i=1\ldots t-1} f_i(x) \; + \frac{1}{\eta} R(x)$$

- Which regularizer to pick?
- AdaGrad: treat this as a learning problem!
  Family of regularizations:

$$R(x) = |x|_A^2 \quad s.t. \;\; A \succcurlyeq 0 \;, Trace(A) = d$$

- Objective in matrix world: best regret in hindsight!

AdaGrad (diagonal form)

- Set $x_1 \in K$ arbitrarily
- For $t = 1, 2, \ldots$,
  1. use x_t obtain f_t
  2. compute x_{t+1} as follows:

$$G_t = \mathrm{diag}(\textstyle\sum_{i=1}^{t} \nabla f_i(x_i) \nabla f_i(x_i)^\top)$$

$$y_{t+1} = x_t - \eta G_t^{-1/2} \nabla f_t(x_t)$$

$$x_{t+1} = \arg\min_{x \in K} (y_{t+1} - x)^\top G_t (y_{t+1} - x)$$

- Regret bound: [Duchi, Hazan, Singer '10]
  $O\left(\sum_i \sqrt{\sum_t \nabla_{t,i}^2}\right)$, can be $\sqrt{d}$ better than SGD
- Infrequently occurring, or small-scale, features have small influence on regret (and therefore, convergence to optimal parameter)

Agenda

1. Learning as mathematical optimization
   - Stochastic optimization, ERM, online regret minimization
   - Offline/stochastic/online gradient descent
2. Regularization
   - AdaGrad and optimal regularization
3. Gradient Descent++
   - Frank-Wolfe, acceleration, variance reduction, second order methods, non-convex optimization