Online optimization

good ML training algorithms regret less

Ed Bueler

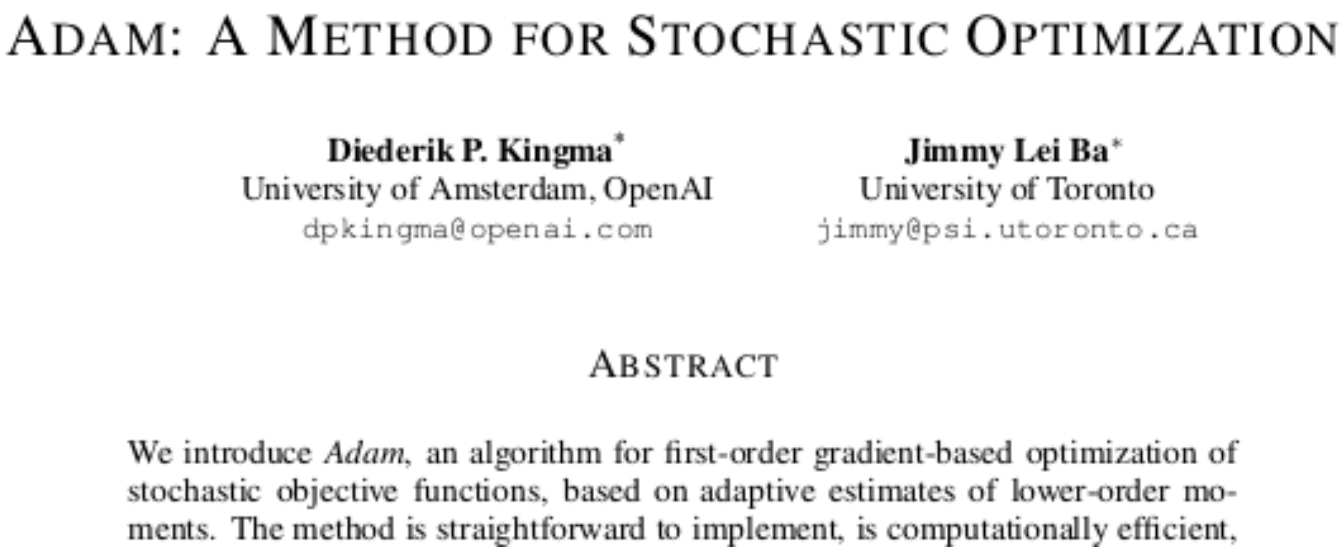
MATH 692 Mathematics for Machine Learning

17 March 2022

Outline

my motivations

- why is SGD so popular? is my algorithm better than yours?
- it was easy to stumble upon Kingma & Ba (2014):



ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION

Diederik P. Kingma*
University of Amsterdam, OpenAI
dpkingma@openai.com

Jimmy Lei Ba*
University of Toronto
jimmy@psi.utoronto.ca

ABSTRACT

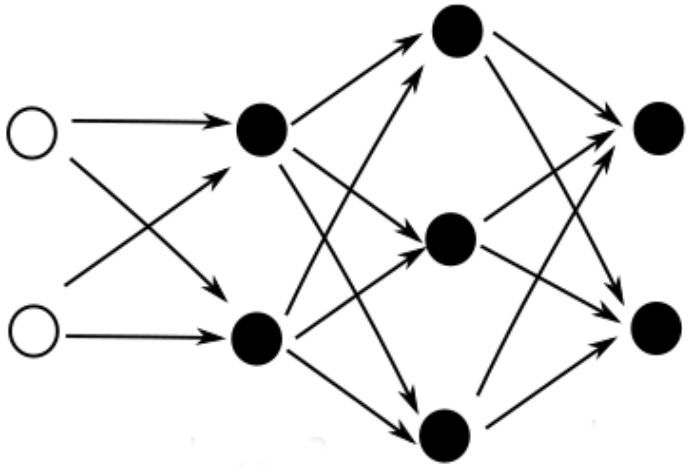
We introduce *Adam*, an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient,

  - 101000 citations; Google Scholar's top-cited paper in 2020
  - the Adam optimizer is the default for tensorflow, pytorch, ...

- **Q.** how do they analyze Adam and show that it is good?

connecting to my earlier talk: training a neural net

- forward pass through an artificial neural net (ANN) with L layers:

$x^{\{i\}} \rightarrow$  $\rightarrow a^{[L]}$ (supposed to be $y^{\{i\}}$)

- output activations are a function of input and parameters:

$$a^{[L]} = a^{[L]}(x^{\{i\}}; p)$$

- parameters p collect weights and biases: $p = \{W^{[\ell]}, b^{[\ell]}\} \in \mathbb{R}^n$
- cost of one labeled pair $(x^{\{i\}}, y^{\{i\}})$:

$$C^{\{i\}}(p) = \frac{1}{2} \left\| y^{\{i\}} - a^{[L]}(x^{\{i\}}; p) \right\|_2^2$$

- **training goal:** find p so the costs $C^{\{i\}}(p)$ are small

notation (standardization and simplification)

- $\theta = p$ is vector of parameters
- $(x_i, y_i) = (x^{\{i\}}, y^{\{i\}})$ is a labeled pair
- N is total size of training set
- $c_i(\theta) = \frac{1}{2} \left\| y_i - a^{[L]}(x_i; \theta) \right\|_2^2 = C^{\{i\}}(p)$ is cost of one pair
  - the detailed form of $c_i(\theta)$ will not matter much

total cost versus online training

total cost over training set

original goal: minimize total (average) cost over fixed training set

$$\frac{1}{N}\sum_{i=1}^{N} c_i(\theta)$$

online training = sequence of cost objectives

assume infinite sequence of cost functionals:

$$c_i(\theta)$$

- the basic online training **method** is clear: train the neural net incrementally, a little for each c_i
- but what's the online training **goal**?

online training algorithms

- choose θ_1 as the initial parameters iterate
- an *online training algorithm* computes each new iterate θ_{i+1}
- θ_{i+1} is computed from cost functionals $\{c_1, c_2, \ldots, c_i\}$ and (parameter) iterates $\{\theta_1, \theta_2, \ldots, \theta_i\}$
- **key assumption:** an online training algorithm does not use *future* cost functionals in constructing θ_{i+1}:

$$\theta_{i+1} = F(c_1, \ldots, c_i, \theta_1, \ldots, \theta_i)$$

online training algorithms: examples

- *online gradient descent* (OGD) with learning rates $\eta_i > 0$:

$$\theta_{i+1} = \theta_i - \eta_i \nabla c_i(\theta_i)$$

  - probability is not needed, so jettison "stochastic": SGD $\to$ OGD
- Adam (*more later*)
- Adaline, Adadelta, Adagrad, RMSprop, mini-batching, dropout, ...
  - google search for buzzwords: `tensorflow optimizers`
- follow the leader (*more later*)
- (quasi-)Newton methods which use (approximate) 2nd derivatives

definition (Zinkevich, 2003; decision theory in 1980s?)

the *regret* of an online algorithm, at the kth training step, is

$$R_j = \sum_{i=1}^{j} c_i(\theta_i) - \min_{\theta} \sum_{i=1}^{j} c_i(\theta)$$

- regret R_j is difference between algorithm's result for the costs so far and the best-possible cost from a single parameter setting
  - best setting so far: $\theta_j^* = \operatorname{argmin}_{\theta} \sum_{i=1}^{j} c_i(\theta)$
- negative regret is possible!
- $R_j > 0$: the player regrets not choosing θ_j^*

regret in game theory

- treat the algorithm as a *player* and the online stream of cost functionals c_i as an *adversary*
- the player chooses θ_i **before** knowing c_i
- the adversarial sequence $\{c_i\}$ is totally-uncontrolled

online convex game (Abernathy et al 2008)

for $i = 1, \ldots, j$
player chooses θ_i
then adversary chooses c_i
player suffers regret

$$R_j = \sum_{i=1}^{j} c_i(\theta_i) - \min_{\theta} \sum_{i=1}^{j} c_i(\theta)$$

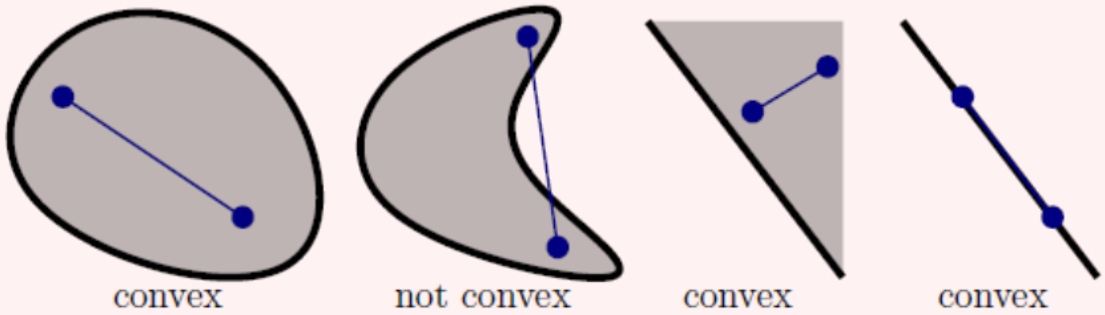
- previous slide says “convex”; we need the definition

definition

- a set $K \subset \mathbb{R}^n$ is *convex* if for all $x, y \in K$ and $0 \leq t \leq 1$,

$$t\,x + (1-t)y \in K$$

  - line segment from x to y is inside K



convex functions

- convex sets *and* functions

definition

- a function $f : K \to \mathbb{R}$ is *convex* if for all $x, y \in K$ and $0 \leq t \leq 1$,

$$t\,f(x) + (1-t)f(y) \geq f(t\,x + (1-t)y)$$

- convex = concave *up*
- for $f : K \to \mathbb{R}$ to be convex, note K must be convex

definition

given a convex set $K \subset \mathbb{R}^n$ and a convex function $f : K \to \mathbb{R}$,

$$\min_{\theta \in K} f(\theta)$$

is a *convex optimization problem*

- K is often described by equality and inequality constraints
- note $K = \mathbb{R}^n$ is convex (unconstrained optimization)
- SVM is convex optimization
- actual ML training is often not convex (e.g. ANN)
- *I don't know the answer yet*:
  for an ANN, when is $c_i(\theta) = \frac{1}{2} \left\| y_i - a^{[L]}(x_i; \theta) \right\|_2^2$ a convex function on $K = \mathbb{R}^n$?

convex optimization: a standard algorithm

- to solve: $\min_{\theta \in K} f(\theta)$
- assume $K \subset \mathbb{R}^n$ is closed and convex
  - then each $y \in \mathbb{R}^n$ has a unique *projection onto* K, the closest point in K to y:

$$\Pi_K(y) = \text{argmin}_{x \in K} \|y - x\|_2$$

- assume $f : K \to \mathbb{R}$ is differentiable and convex
- assume learning rates (*step sizes*) $\eta_i > 0$

projected gradient descent

choose θ_1
for $i = 1, 2, \ldots$
$$\theta_{i+1} = \Pi_K\big(\theta_i - \eta_i \nabla f(\theta_i)\big)$$

online convex optimization

- convex optimization is *so* 20th century . . . let's go online!

definition

online convex optimization (programming):

- fix a convex set $K \subset \mathbb{R}^n$
- assume a sequence of convex functions $c_i : K \to \mathbb{R}$
- an algorithm for generating $\theta_{i+1} \in K$ from previous c_i, θ_i

the online goal is small regret

- the goal of an online algorithm is **slowly-growing regret**

$$R_j = \underbrace{\sum_{i=1}^{j} c_i(\theta_i)}_{\text{online alg. result}} - \underbrace{\min_{\theta} \sum_{i=1}^{j} c_i(\theta)}_{\text{offline result}}$$

Zinkevich (2003), regarding a regret analysis: *We make **no distributional assumptions** about the convex cost functions. . . . We cannot hope to choose a point θ_i that minimizes c_i, because c_i can be anything. Instead we try to **minimize regret**. . . . If the sequence of cost functions $\{c_i\}$ is relatively stationary, then an online algorithm can learn what the cost functions will look like in the future. If the sequence of cost functions varies drastically, then the offline algorithm will not be able to take advantage of this because it selects a single θ.*

regret bounds

- the online optimization framework evaluates an ML training algorithm via a *regret bound*:
  - how fast does R_j grow?
- regret bounds are informative about algorithms
  - what properties of the objectives c_i determine the bound?
  - what algorithmic settings determine the bound?
- we need an example!

logarithmic OGD regret bound: $R_j = O(\log j)$

- online gradient descent (OGD):

$$\theta_{i+1} = \Pi_K\big(\theta_i - \eta_i \nabla c_i(\theta_i)\big)$$

- assume well-behaved costs c_i: smooth, strictly-convex

theorem (Hazan et al., 2007)

Suppose the c_i are uniformly strictly convex: $\nabla^2 c_i \succ H > 0$. Compute iterates $\theta_1, \ldots, \theta_j$ from OGD. Let $G = \max_{i=1,\ldots,j} \|\nabla c_i(\theta_i)\|$.

If $\eta_i = \dfrac{1}{i\,H}$ then $\boxed{R_j \leq \dfrac{G^2}{2H}(1 + \log j).}$

sketch of Hazan proof

- detailed proof in extra slides at end

proof sketch:

- Taylor expand $c_i(\theta^*)$ from basepoint θ_i, to 2nd order.
- Use Hessian lower bound to control $c_i(\theta_i) - c_i(\theta^*)$ in terms of $\nabla c_i(\theta_i)^\top(\theta_i - \theta^*)$ and $\|\theta^* - \theta_i\|^2$.
- OGD bounds $\nabla c_i(\theta_i)^\top(\theta_i - \theta^*)$ by Pythagorean-ish argument.
- Use $\eta_i = \frac{1}{iH}$ and telescoping to get $2R_j \leq \frac{G^2}{H}\sum_{i=1}^{j}\frac{1}{i}$.
- Integral test gives $2R_j \leq \frac{G^2}{H}(1 + \log j)$. □

square root OGD regret bound: $R_j = O(\sqrt{j}\,)$

- $\nabla^2 c_i \succ H > 0$ (strictly convex) is too strong
  - also, $\eta_i = \frac{1}{iH}$ depends on H, typically unknown
- we now allow $\nabla^2 c_i = 0$, so we need to assume K is bounded
  - $c_i(\theta)$ could be linear in θ

theorem (Zinkevich, 2003)

Suppose $K \subset \mathbb{R}^n$ is closed, convex, and bounded with diameter d_K. Suppose each $c_i : K \to \mathbb{R}$ is differentiable and convex. Compute iterates $\theta_1, \ldots, \theta_j$ from OGD. Let $G = \max_{i=1,\ldots,j} \|\nabla c_i(\theta_i)\|$.

If $\eta_i = \dfrac{1}{\sqrt{i}}$ then $\boxed{R_j \leq \left(\dfrac{d_K^2}{2} + G^2\right)\sqrt{j} - \dfrac{G^2}{2}.}$

sketch of Zinkevich proof

- detailed proof in extra slides at end

proof sketch:

- Replace c_i by linear functions with same gradients: $\tilde{c}_i(\theta) = g_i^\top \theta$ where $g_i = \nabla c_i(\theta_i)$.
- Convexity of c_i shows $\tilde{R}_j = \sum_{i=1}^{j} g_i^\top (\theta_i - \theta^*) \geq R_j$. (Regret increases.)
- OGD bounds $g_i^\top (\theta_i - \theta^*)$ by Pythagorean-ish argument.
- Use telescoping to get $2\tilde{R}_j \leq \frac{d_K^2}{\eta_j} + G^2 \sum_{i=1}^{j} \eta_i$
- Choose $\eta_i = i^{-p}$ and use integral test.
- $p = 0.5$ gives the slowest-growing bound. □

OGD average regret goes to zero

corollary

under *either hypothesis*, the average regret of OGD goes to zero:

$$\lim_{j\to\infty} \frac{R_j}{j} = 0,$$

equivalently, average cost converges to its optimal value:

$$\lim_{j\to\infty} \frac{1}{j}\sum_{i=1}^{j} c_i(\theta_i) = \lim_{j\to\infty} \min_{\theta} \frac{1}{j}\sum_{i=1}^{j} c_i(\theta)$$

- explains why OGD=SGD is a *good* ML training algorithm?
- both proofs used a learning rate schedule with $\eta_j \to 0$
- neither regret bound proof actually required θ_j^* to be optimal!
- apparently, *some* convexity is needed?

other algorithms: follow the leader

- an example showing $O(\sqrt{j})$ regret bound is nontrivial!
- "follow the leader" is obvious and old (1957?)

follow the leader (FTL)

choose θ_1
for $i = 1, 2, \ldots$

$$\theta_{i+1} = \operatorname{argmin} \sum_{\ell=1}^{i} c_\ell(\theta)$$

- FTL has no sub-linear regret bound: $\frac{R_j}{j} \not\to 0$
  - concrete example in notes by M. Razaviyayn[1]; $c_i(\theta)$ are linear and Zinkevich argument for OGD applies
- lesson: don't try too hard to optimize the c_i you already know!

[1] https://cpb-us-e1.wpmucdn.com/sites.usc.edu/dist/3/137/files/2017/02/lec24-2bywoz5.pdf

other algorithms: quasi-Newton

- Hazan et al. (2007) also propose a *online Newton step* (ONS) algorithm
- actually a quasi-Newton method because it constructs an approximation to the Hessian on the fly
  - compare L-BFGS etc.

theorem (Hazan et al. 2007)

ONS has $R_j = O(\log j)$ if the c_i are α-exp-concave

- α-exp-concave $\supset$ strictly-convex
- Hazan (2007) shows ONS is *follow the approximate leader*

Adam algorithm

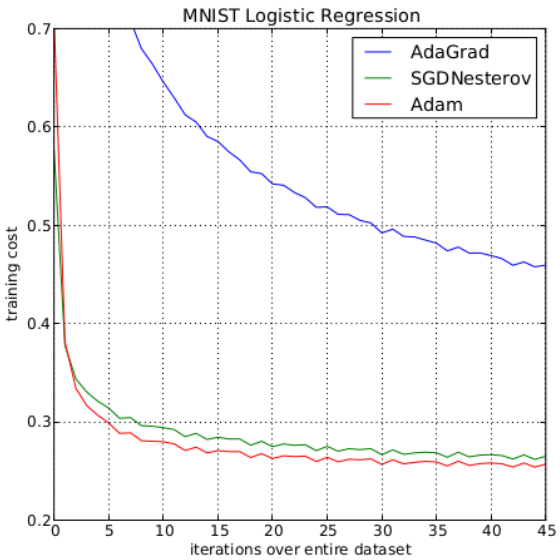
Adam (Kingma & Ba, 2014)

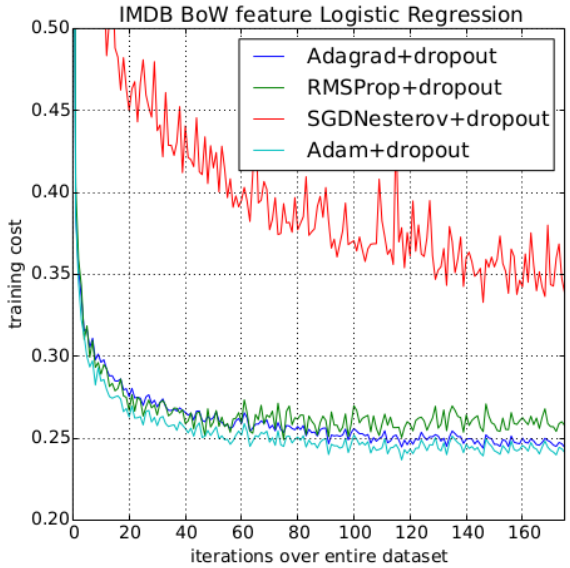
defaults: $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$
given: θ_1
$m_0 \leftarrow 0$, $v_0 \leftarrow 0$
for $i = 1, 2, \ldots$

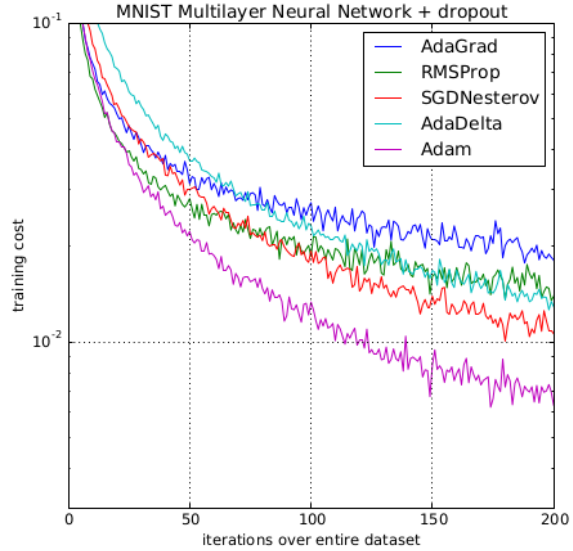
$$g_i = \nabla c_i(\theta_i)$$
$$m_i = \beta_1 m_{i-1} + (1-\beta_1) g_i \qquad \textit{(1st moment vector)}$$
$$v_i = \beta_2 v_{i-1} + (1-\beta_2) g_i^2 \qquad \textit{(2nd moment vector)}$$
$$\alpha_i = \alpha (1-\beta_2^i)^{1/2} / (1-\beta_1^i) \qquad \textit{(bias-corrected step)}$$
$$\theta_{i+1} = \theta_i - \alpha_i m_i / (\sqrt{v_i} + \epsilon)$$

- first-order optimization like OGD, but estimates 2nd moment
- no learning rate schedule (η_i) needed!
- θ updates are invariant to scaling of c_i
- performs "step-size annealing"

Adam results







- 99% of Kingma & Ba (2014) readers see above results graphs and are happy?

Adam's regret bound

Kingma & Ba (2014): *We analyze the convergence of Adam using the online learning framework proposed in (Zinkevich, 2003). . . . Since the nature of the [cost function] sequence is unknown in advance, we evaluate the algorithm using the regret . . .*

theorem

Compute iterates $\theta_1, \ldots, \theta_j$ from Adam. Then $\boxed{R_j = O(\sqrt{j}).}$

- $O(\sqrt{j})$ is the same as Zinkevich's bound, but Adam's constants are better, especially for $\theta \in \mathbb{R}^d$ with large d
- Reddi et al. (2019):
  - construct example where $\frac{R_j}{j} \not\to 0$ for Adam!
  - propose AMSGrad algorithm
  - actually prove $R_j = O(\sqrt{j})$ for AMSGrad . . . it seems

Adam's regret bound

Kingma & Ba (2014): *We analyze the convergence of Adam using the online learning framework proposed in (Zinkevich, 2003). . . . Since the nature of the [cost function] sequence is unknown in advance, we evaluate the algorithm using the regret . . .*

we regret to inform you . . .

Compute iterates $\theta_1, \ldots, \theta_j$ from Adam. Then $\boxed{R_j \neq O(\sqrt{j}).}$

- $O(\sqrt{j})$ is the same as Zinkevich's bound, but Adam's constants are better, especially for $\theta \in \mathbb{R}^d$ with large d
- Reddi et al. (2019):
  - construct example where $\frac{R_j}{j} \not\to 0$ for Adam!
  - propose AMSGrad algorithm
  - actually prove $R_j = O(\sqrt{j})$ for AMSGrad . . . it seems

alternative framework: empirical risk

- the *Deep Learning* book (Chapter 8), for example, tells you that ML optimization is special because the data is stochastic

definition

the *risk* of an ANN parameter setting is the expected cost over an assumed training data distribution

  - typically assumes each (x_i, y_i) is an independent sample

- but you don't tend to know the distribution, so ...

definition

the *empirical risk* of an ANN parameter setting is the average cost over the training data set

  - same as *maximum likelihood estimation* in many cases

the 5 frameworks for ML training optimization

- it seems ML optimization is portrayed in 5 different ways:
  1. **risk** = find parameters which minimize expected cost over assumed distribution for training data
  2. **empirical risk** = find parameters which minimize sample mean over training data
  3. **maximum likelihood estimation** = find parameters which maximize assumed distribution (e.g. exp of negative cost)
  4. naive **optimization** = find parameters which minimize average cost of training data
  5. **online regret** = find an algorithm $\theta_{i+1} = f(c_1, \ldots, c_i)$ which has slow-growing regret regardless of training data distribution

- typically, 1 is only notional
  - the training data distribution is unknown
- 2,3,4 are the same goal once you remove probabilistic edifice
- only 5 aligns with the *algorithm designer's* concerns
  - 5 is *meta*-optimization

the 5 frameworks: formulas

1. **risk**:

$$\min_{\theta} \mathbb{E}[c(\theta)] = \min_{\theta} \int_{\{\text{all } (x,y)\}} c(\theta; x, y)\, dp_{\text{data}}$$

2. **empirical risk** (*Deep Learning* book)
3. = **maximum likelihood estimation**
4. = **optimization** (Higham & Higham, 2019):

$$\min_{\theta} \frac{1}{j} \sum_{i=1}^{j} c_i(\theta)$$

5. **online regret**:

$$\min_{\text{alg. for } \theta_{i+1}} \left[\sum_{i=1}^{j} c_i(\theta_i) - \min_{\theta} \sum_{i=1}^{j} c_i(\theta) \right]$$

summary: why online regret bounds?

- regret bounds are rigorous properties of ML-training-suitable, i.e. online, minimization algorithms
  - the ML community has adopted this way of analyzing algorithms
- quantitative regret bounds expose performance differences
  - between algorithms (e.g. FTL vs OGD vs Adam vs AMSGrad)
  - between cost-function classes (convex vs strictly-convex vs α-exp)
- regret bounds reveal how to set learning rates
  - compare η_i in Hazan, Zinkevich, Adam, AMSGrad proofs
- analysis of regret avoids probability
  - avoid distributional assumptions about the cost functions
  - avoid risk, empirical risk, MLE, Bayes, . . . when you really don't know probabilities for training data anyway
- analyzing regret is very 21st century!

online optimization references

- J. D. Abernathy, P. Bartlett, A. Rakhlin, and A. Tewari (2008). *Optimal strategies and minimax lower bounds for online convex games*, UC Berkeley Tech. Rep. UCB/EECS-2008-19
  - regret in game context
- E. Hazan, A. Agarwal, & S. Kale (2007). *Logarithmic regret algorithms for online convex optimization.* Machine Learning, 69(2), 169-192
  - $O(\log j)$ regret bounds for positive definite Hessians, and for Newton algorithms
- D. P. Kingma & J. Ba (2014). *Adam: A method for stochastic optimization*, preprint arXiv:1412.6980.
  - claims $O(\sqrt{j})$ regret bound
- S. J. Reddi, S. Kale, & S. Kumar (2019). *On the convergence of Adam and beyond*, preprint arXiv:1904.09237.
  - debunks Kingma & Ba (2014) regret bound proof
  - Adam $\overset{\text{replace with}}{\rightarrow}$ AMSGrad
- M. Zinkevich (2003). *Online convex programming and generalized infinitesimal gradient ascent*, Proceedings of the 20th International Conference on Machine Learning, 928-936
  - introduced regret
  - $O(\sqrt{j})$ regret bound of OGD

additional references

- L. Bottou, *Online algorithms and stochastic approximations.* In D. Saad, ed., *Online Learning and Neural Networks*, Cambridge University Press, 1998
  - first sentence:
    Almost all of the early work on Learning Systems focused on online algorithms (Hebb, 1949; Rosenblatt, 1957; Widrow and Hoff, 1960; Amari, 1967; ...)
  - regret is not mentioned
- I. Goodfellow, Y. Bengio, & A. Courville, *Deep Learning.* MIT Press, 2016
  - Chapter 8 addresses empirical risk
  - online optimization and regret is not mentioned

proof: Let $g_i = \nabla c_i(\theta_i)$ and $\theta^* = \theta^*_j$. Taylor and strict-convexity give

$$\begin{aligned} c_i(\theta^*) &= c_i(\theta_i) + g_i^\top(\theta^* - \theta_i) + \frac{1}{2}(\theta^* - \theta_i)^\top \nabla^2 c_i(\theta^* - \theta_i) \\ &\geq c_i(\theta_i) + g_i^\top(\theta^* - \theta_i) + \frac{H}{2}\|\theta^* - \theta_i\|^2 \end{aligned}$$

so $\quad 2\big(c_i(\theta_i) - c_i(\theta^*)\big) \leq 2g_i^\top(\theta_i - \theta^*) - H\|\theta^* - \theta_i\|^2 \quad$ for each i.

On the other hand, OGD, a projection property, and convexity give

$$\begin{aligned} \|\theta_{i+1} - \theta^*\|^2 = \|\Pi_K(\theta_i - \eta_i g_i) - \theta^*\|^2 &\leq \|\theta_i - \eta_i g_i - \theta^*\|^2 \\ &\leq \|\theta_i - \theta^*\|^2 - 2\eta_i g_i^\top(\theta_i - \theta^*) + \eta_i^2\|g_i\|^2 \end{aligned}$$

so $\quad 2g_i^\top(\theta_i - \theta^*) \leq \frac{1}{\eta_i}\left(\|\theta_i - \theta^*\|^2 - \|\theta_{i+1} - \theta^*\|^2\right) + \eta_i G^2 \quad$ for each i.

Thus since $1/\eta_i = iH$,

$$
\begin{aligned}
2R_j &= \sum_{i=1}^{j} 2\big(c_i(\theta_i) - c_i(\theta^*)\big) \le \sum_{i=1}^{j} 2g_i^\top(\theta_i - \theta^*) - H\sum_{i=1}^{j}\|\theta^* - \theta_i\|^2 \\
&\le \sum_{i=1}^{j} \frac{1}{\eta_i}\left(\|\theta_i - \theta^*\|^2 - \|\theta_{i+1} - \theta^*\|^2\right) + G^2\sum_{i=1}^{j}\eta_i - H\sum_{i=1}^{j}\|\theta^* - \theta_i\|^2 \\
&= \sum_{i=1}^{j} iH\left(\|\theta_i - \theta^*\|^2 - \|\theta_{i+1} - \theta^*\|^2\right) + \frac{G^2}{H}\sum_{i=1}^{j}\frac{1}{i} - H\sum_{i=1}^{j}\|\theta^* - \theta_i\|^2 \\
&= \sum_{i=1}^{j}(i-1)H\|\theta_i - \theta^*\|^2 - \sum_{i=1}^{j} iH\|\theta_{i+1} - \theta^*\|^2 + \frac{G^2}{H}\sum_{i=1}^{j}\frac{1}{i} \\
&= -jH\|\theta_{j+1} - \theta^*\|^2 + \frac{G^2}{H}\sum_{i=1}^{j}\frac{1}{i} \le 0 + \frac{G^2}{H}(1 + \log j)\,. \qquad \square
\end{aligned}
$$

extra: Zinkevich proof

- the per-iterate regret increases if we replace c_i by a linear function

Lemma: Suppose $c_i(\theta)$ is convex. For $\theta_i, \theta^* \in K$, if $g_i = \nabla c_i(\theta_i)$ then $c_i(\theta_i) - c_i(\theta^*) \leq g_i^\top(\theta_i - \theta^*)$.

proof: By convexity, $c_i(\theta^*) \geq c_i(\theta_i) + g_i^\top(\theta^* - \theta_i)$. □

- now we can prove the theorem

proof: As in Hazan, by OGD and projection,

$$\begin{aligned}\|\theta_{i+1} - \theta^*\|^2 = \|\Pi_K(\theta_i - \eta_i g_i) - \theta^*\|^2 &\leq \|\theta_i - \eta_i g_i - \theta^*\|^2 \\ &\leq \|\theta_i - \theta^*\|^2 - 2\eta_i g_i^\top(\theta_i - \theta^*) + \eta_i^2\|g_i\|^2\end{aligned}$$

so $\quad 2g_i^\top(\theta_i - \theta^*) \leq \dfrac{1}{\eta_i}\left(\|\theta_i - \theta^*\|^2 - \|\theta_{i+1} - \theta^*\|^2\right) + \eta_i G^2 \quad$ for each i.

Thus, assuming η_i is decreasing and nonnegative,

$$
\begin{aligned}
2R_j &= \sum_{i=1}^{j} 2\left(c_i(\theta_i) - c_i(\theta^*)\right) \leq \sum_{i=1}^{j} 2g_i^\top(\theta_i - \theta^*) \\
&\leq \sum_{i=1}^{j} \frac{1}{\eta_i}\left(\|\theta_i - \theta^*\|^2 - \|\theta_{i+1} - \theta^*\|^2\right) + G^2 \sum_{i=1}^{j} \eta_i \\
&\leq \frac{1}{\eta_1}\|\theta_1 - \theta^*\|^2 + \sum_{i=2}^{j}\left(\frac{1}{\eta_i} - \frac{1}{\eta_{i-1}}\right)\|\theta_i - \theta^*\|^2 \\
&\qquad - \frac{1}{\eta_{j+1}}\|\theta_{j+1} - \theta^*\|^2 + G^2 \sum_{i=1}^{j} \eta_i \\
&\leq d_K^2\left[\frac{1}{\eta_1} + \sum_{i=2}^{j}\left(\frac{1}{\eta_i} - \frac{1}{\eta_{i-1}}\right)\right] - 0 + G^2 \sum_{i=1}^{j} \eta_i
\end{aligned}
$$

By telescoping,

$$2R_j \le \frac{d_K^2}{\eta_j} + G^2 \sum_{i=1}^{j} \eta_i$$

Substitute $\eta_i = i^{-p}$ and use integral test:

$$2R_j \le d_K^2 j^p + G^2 \sum_{i=1}^{j} i^{-p} \le d_K^2 j^p + G^2 \left(1 + \int_1^j x^{-p}\, dx\right)$$

$$\le d_K^2 j^p + G^2 \left(1 + \frac{j^{1-p} - 1}{1-p}\right)$$

Using $p = 0.5$ balances powers for smallest growth rate:

$$2R_j \le d_K^2 \sqrt{j} + G^2 \left(1 + 2(\sqrt{j} - 1)\right)$$

Rearrange to $\quad R_j \le \left(\frac{d_K^2}{2} + G^2\right)\sqrt{j} - \frac{G^2}{2}.$ □