Coin Betting for Backprop without Learning Rates and More

Francesco Orabona

Stony Brook University
Stony Brook, NY USA

August 24, 2017

Motivation

The Dream: Truly Automatic Machine Learning

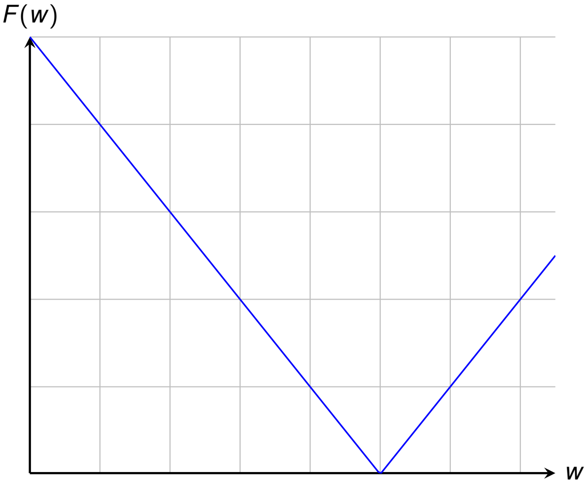
- No hyperparameters to tune
- No humans in the loop
- Some guarantees

Outline

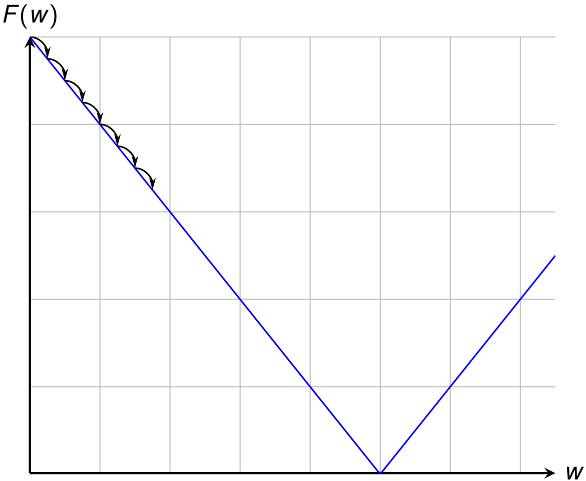
Outline

Convex Optimization

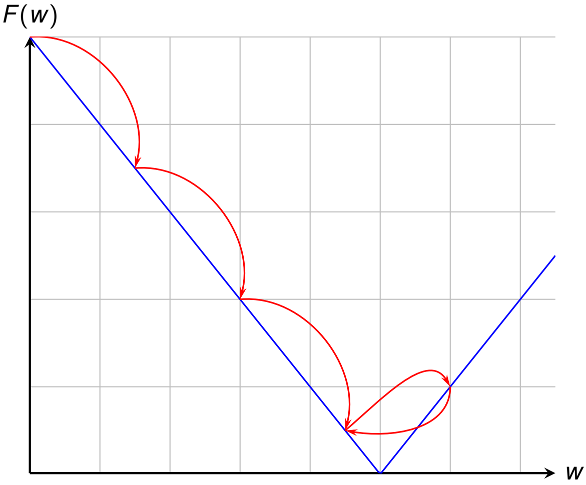
How Subgradient Descent Work?



How Subgradient Descent Work?



How Subgradient Descent Work?



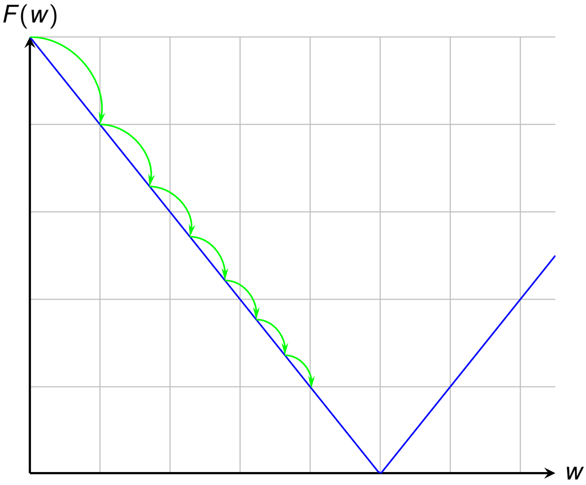
What About "Adaptive" Algorithms?

- In the stochastic setting is even more challenging: the function at each round is always the same only in expectation

What About “Adaptive” Algorithms?

- In the stochastic setting is even more challenging: the function at each round is always the same only in expectation
- What about AdaGrad?

AdaGrad



What the Theory Says?

- Only strategy known: use a decreasing step size, $\mathcal{O}\left(\frac{\eta}{\sqrt{t}}\right)$
- Convergence after n iterations is $\mathcal{O}\left(\frac{1}{\sqrt{n}}\left(\frac{\|\boldsymbol{w}^*\|^2}{\eta}+\eta\right)\right)$
- $\boldsymbol{w}^*$ is the best solution

What the Theory Says?

- Only strategy known: use a decreasing step size, $\mathcal{O}\left(\frac{\eta}{\sqrt{t}}\right)$
- Convergence after n iterations is $\mathcal{O}\left(\frac{1}{\sqrt{n}}\left(\frac{\|\boldsymbol{w}^*\|^2}{\eta} + \eta\right)\right)$
- $\boldsymbol{w}^*$ is the best solution
- The optimal bound $\|\boldsymbol{w}^*\|\frac{1}{\sqrt{n}}$ can be obtained tuning $\eta = \|\boldsymbol{w}^*\|$

What the Theory Says?

- Only strategy known: use a decreasing step size, $\mathcal{O}\left(\frac{\eta}{\sqrt{t}}\right)$
- Convergence after n iterations is $\mathcal{O}\left(\frac{1}{\sqrt{n}}\left(\frac{\|\boldsymbol{w}^*\|^2}{\eta}+\eta\right)\right)$
- $\boldsymbol{w}^*$ is the best solution
- The optimal bound $\|\boldsymbol{w}^*\|\frac{1}{\sqrt{n}}$ can be obtained tuning $\eta = \|\boldsymbol{w}^*\|$ ...but you don't know $\boldsymbol{w}^*$...

What the Theory Says?

- Only strategy known: use a decreasing step size, $\mathcal{O}\left(\frac{\eta}{\sqrt{t}}\right)$
- Convergence after n iterations is $\mathcal{O}\left(\frac{1}{\sqrt{n}}\left(\frac{\|\boldsymbol{w}^*\|^2}{\eta}+\eta\right)\right)$
- $\boldsymbol{w}^*$ is the best solution
- The optimal bound $\|\boldsymbol{w}^*\|\frac{1}{\sqrt{n}}$ can be obtained tuning $\eta = \|\boldsymbol{w}^*\|$ ...but you don't know $\boldsymbol{w}^*$...

Why we cannot have an optimization algorithm that self-tunes its learning rate?

SGD without Learning Rates

- [McMahan&Streeter, NIPS'12][McMahan&Abernethy, NIPS'13], suboptimal bounds, 1D
- [Orabona, NIPS'13], suboptimal bound, Hilbert spaces
- [McMahan&Orabona, COLT'14], optimal bound, Hilbert space
- [Orabona, NIPS'14], data-dependent bound, analysis in RKHS, algorithm in VW
- [Cutkosky&Boahen, NIPS'16, COLT'17] unbounded gradients
- [Orabona&Pal, NIPS'16], coin-betting view
- [Orabona&Tommasi, ArXiv'17], data-dependent coin-betting for deep learning
- [Kotlowski, ALT'17], scale-free bound

Also, Learning with Experts algorithms: NormalHedge [Chaudhuri et al. NIPS'09], AdaNormalHedge [Luo&Schapire, NIPS'14, COLT'15], Squint [Koolen&Erven, COLT'15], etc.

SGD without Learning Rates

- [McMahan&Streeter, NIPS'12][McMahan&Abernethy, NIPS'13], suboptimal bounds, 1D
- [Orabona, NIPS'13], suboptimal bound, Hilbert spaces
- [McMahan&Orabona, COLT'14], optimal bound, Hilbert space
- [Orabona, NIPS'14], data-dependent bound, analysis in RKHS, algorithm in VW
- [Cutkosky&Boahen, NIPS'16, COLT'17] unbounded gradients
- [Orabona&Pal, NIPS'16], coin-betting view
- [Orabona&Tommasi, ArXiv'17], data-dependent coin-betting for deep learning
- [Kotlowski, ALT'17], scale-free bound

Also, Learning with Experts algorithms: NormalHedge [Chaudhuri et al. NIPS'09], AdaNormalHedge [Luo&Schapire, NIPS'14, COLT'15], Squint [Koolen&Erven, COLT'15], etc.

Outline

Betting on a Coin

- Start with \$1
- Bet w_t money on head ($w_t > 0$) or tails ($w_t < 0$)
  - Cannot borrow money
- Win or lose depending on the outcome of the coin $g_t \in \{-1, 1\}$
- $Wealth_t = Wealth_{t-1} + w_t g_t$

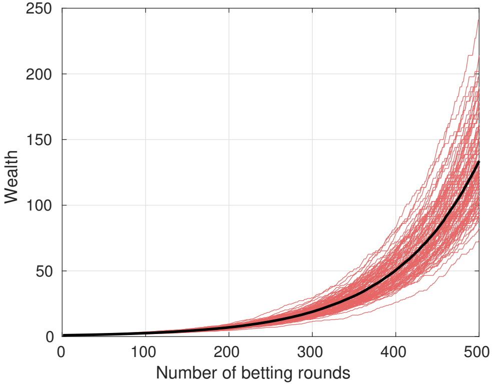
Aim: Maximize gain on all sequences where the number of tails and head are different

Optimal Betting Strategy for a Stochastic Coin: Kelly Betting (1956)

- Known problem in economics
- Assume the probability of tail, p, is bigger than 0.5
- Bet a fraction of your money equal to $2p - 1$ on tail at each round
- Asymptotically any other strategy will do worse

Kelly Betting in Practice

- $p = 0.51$, then bet a 2% of your current money at each round
- Wealth increases exponentially



Non-Stochastic setting, but Knowing the Future

- Non-stochastic setting, T rounds
- Assume to bet a fixed fraction of money at each round
- What is the optimal fraction?

[McMahan&Abernethy, NIPS'13; Orabona&Pal, NIPS'16]

Non-Stochastic setting, but Knowing the Future

- Non-stochastic setting, T rounds
- Assume to bet a fixed fraction of money at each round
- What is the optimal fraction?
- The optimal bet at each time step t is $\frac{\sum_{i=1}^{T} g_i}{T} \cdot Wealth_{t-1}$
- $$\text{Winnings} > \exp\left(\frac{(\sum_{t=1}^{T} g_t)^2}{2T}\right)$$
- Knowing something about the future allows to grow your money exponentially!

[McMahan&Abernethy, NIPS'13; Orabona&Pal, NIPS'16]

Worst case Optimal Betting Strategy: Krichevsky-Trofimov Bettor

- What if we don't know the future?

[Orabona&Pal, NIPS'16]

Worst case Optimal Betting Strategy: Krichevsky-Trofimov Bettor

- What if we don't know the future?
- Estimate probability of head with Krichevsky-Trofimov (KT) estimator: $\frac{\frac{1}{2} + \text{\# of heads in } t \text{ rounds}}{t+1}$
  - No stochastic assumptions: KT has optimal regret w.r.t. the log loss

[Orabona&Pal, NIPS'16]

Worst case Optimal Betting Strategy: Krichevsky-Trofimov Bettor

- What if we don't know the future?
- Estimate probability of head with Krichevsky-Trofimov (KT) estimator: $\frac{\frac{1}{2} + \#\text{ of heads in } t \text{ rounds}}{t+1}$
  - No stochastic assumptions: KT has optimal regret w.r.t. the log loss
- Hence, on round t bet a fraction of your money equal to $\frac{|\sum_{i=1}^{t-1} g_i|}{t}$, on the side that appeared more often

[Orabona&Pal, NIPS'16]

Worst case Optimal Betting Strategy: Krichevsky-Trofimov Bettor

- What if we don't know the future?
- Estimate probability of head with Krichevsky-Trofimov (KT) estimator: $\frac{\frac{1}{2}+\#\text{ of heads in } t \text{ rounds}}{t+1}$
    - No stochastic assumptions: KT has optimal regret w.r.t. the log loss
- Hence, on round t bet a fraction of your money equal to $\frac{|\sum_{i=1}^{t-1} g_i|}{t}$, on the side that appeared more often
- Almost same guarantee than before

$$\text{Winnings of KT Bettor} \geq \frac{\text{Winnings knowning the future}}{2\sqrt{T}}$$

[Orabona&Pal, NIPS'16]

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^T g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^T w_t\right) - F(w^*)$$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - F(w^*) \overset{\text{Jensen}}{\leq} \frac{1}{T}\sum_{t=1}^{T} F(w_t) - F(w^*)$$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - F(w^*) \overset{\text{Jensen}}{\leq} \frac{1}{T}\sum_{t=1}^{T} F(w_t) - F(w^*) \overset{\text{Convexity}}{\leq} \frac{1}{T}\left(\sum_{t=1}^{T} g_t w^* - \sum_{t=1}^{T} g_t w_t\right)$$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^T g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^T w_t\right) - F(w^*) \overset{\text{Jensen}}{\leq} \frac{1}{T}\sum_{t=1}^T F(w_t) - F(w^*) \overset{\text{Convexity}}{\leq} \frac{1}{T}\left(\sum_{t=1}^T g_t w^* - \sum_{t=1}^T g_t w_t\right)$$

$$\overset{\text{Def. } H}{\leq} \frac{1}{T} + \frac{1}{T}\left(\sum_{t=1}^T g_t w^* - H\left(\sum_{t=1}^T g_t\right)\right)$$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^T g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^T w_t\right) - F(w^*) \overset{\text{Jensen}}{\leq} \frac{1}{T}\sum_{t=1}^T F(w_t) - F(w^*) \overset{\text{Convexity}}{\leq} \frac{1}{T}\left(\sum_{t=1}^T g_t w^* - \sum_{t=1}^T g_t w_t\right)$$

$$\overset{\text{Def. } H}{\leq} \tfrac{1}{T} + \tfrac{1}{T}\left(\sum_{t=1}^T g_t w^* - H\left(\sum_{t=1}^T g_t\right)\right) \overset{\text{Max}}{\leq} \tfrac{1}{T} + \tfrac{1}{T}\max_v \ v w^* - H(v)$$

1D Optimization through Betting

- Assume that there exists a function $H(\cdot)$ such that our betting strategy will guarantee that the wealth after T rounds will be at least $H(\sum_{t=1}^{T} g_t)$ for any arbitrary sequence $g_1, \cdots, g_T$
- We want to minimize $F(w) = |w - 10|$
- Let's set a betting game: bet w_t dollars on $g_t = -\partial F(w_t) \in \{-1, +1\}$
- Claim: the average of the bets will converge to the minimum of $F(w)$ at a rate that depends on how good is our betting strategy!

$$F\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - F(w^*) \overset{\text{Jensen}}{\leq} \frac{1}{T}\sum_{t=1}^{T} F(w_t) - F(w^*) \overset{\text{Convexity}}{\leq} \frac{1}{T}\left(\sum_{t=1}^{T} g_t w^* - \sum_{t=1}^{T} g_t w_t\right)$$

$$\overset{\text{Def. } H}{\leq} \tfrac{1}{T} + \tfrac{1}{T}\left(\sum_{t=1}^{T} g_t w^* - H\left(\sum_{t=1}^{T} g_t\right)\right) \overset{\text{Max}}{\leq} \tfrac{1}{T} + \tfrac{1}{T}\max_{v}\ v w^* - H(v)$$

$$\overset{\text{Def } H^*}{=} \frac{H^*(w^*)+1}{T}$$

KT as an Optimization Algorithm

- $g_t \in \partial(-F(w_t))$ and assume $g_t \in \{-1, +1\}$
- $w_t = \frac{\sum_{i=1}^{t-1} g_i}{t} Wealth_t = \frac{\sum_{i=1}^{t-1} g_i}{t} (\sum_{i=1}^{t-1} g_i \cdot w_i + 1)$

Theorem (Orabona&Pal, NIPS'16)

KT betting in 1-d guarantees

$$F\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - F(w^*) \leq \tilde{O}\left(\frac{|w^*|}{\sqrt{T}}\right)$$

KT as an Optimization Algorithm

- $g_t \in \partial(-F(w_t))$ and assume $g_t \in [-1, +1]$
- $w_t = \frac{\sum_{i=1}^{t-1} g_i}{t} Wealth_t = \frac{\sum_{i=1}^{t-1} g_i}{t}(\sum_{i=1}^{t-1} g_i \cdot w_i + 1)$

Theorem (Orabona&Pal, NIPS'16)

KT betting in 1-d guarantees

$$F\left(\frac{1}{T}\sum_{t=1}^{T} w_t\right) - F(w^*) \leq \tilde{\mathcal{O}}\left(\frac{|w^*|}{\sqrt{T}}\right)$$

Proof idea: worst gradients are $\{-1, +1\}$

KT as an Optimization Algorithm

- $\boldsymbol{g}_t \in \partial(-F(\boldsymbol{w}_t))$ and assume $\|\boldsymbol{g}_t\| \leq 1$
- $\boldsymbol{w}_t = \frac{\sum_{i=1}^{t-1} \boldsymbol{g}_i}{t} Wealth_t = \frac{\sum_{i=1}^{t-1} \boldsymbol{g}_i}{t} (\sum_{i=1}^{t-1} \langle \boldsymbol{g}_i, \boldsymbol{w}_i \rangle + 1)$

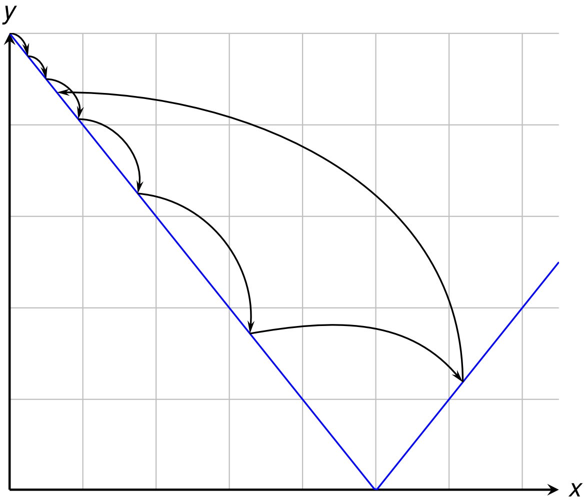
Theorem (Orabona&Pal, NIPS'16)

KT betting in Hilbert spaces guarantees

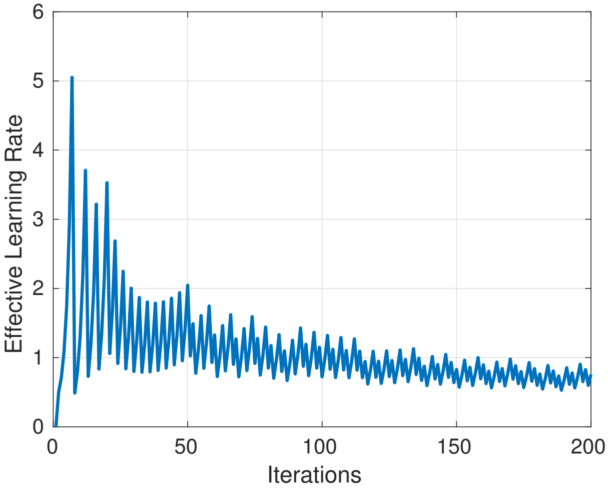
$$F\left(\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{w}_t\right) - F(\boldsymbol{w}^*) \leq \tilde{\mathcal{O}}\left(\frac{\|w^*\|}{\sqrt{T}}\right)$$

Proof idea: worst direction for gradient at time t is parallel to $\sum_{i=1}^{t-1} \boldsymbol{g}_i$

How the Betting Approach Work?



Effective Learning Rate



Improving KT

- We want per-coordinate learning rates
- We want faster convergence with sparse gradients

Improving KT

- We want per-coordinate learning rates
  - One coin for each coordinate
- We want faster convergence with sparse gradients

Improving KT

- We want per-coordinate learning rates
  - One coin for each coordinate
- We want faster convergence with sparse gradients
  - KT strategy: $w_t = \frac{\sum_{i=1}^{t-1} g_t}{t} \, Wealth_{t-1}$

Improving KT

- We want per-coordinate learning rates
  - One coin for each coordinate
- We want faster convergence with sparse gradients
  - KT strategy: $w_t = \frac{\sum_{i=1}^{t-1} g_t}{t} \, Wealth_{t-1}$
  - COCOB strategy: $w_t = \sigma\left(\frac{\sum_{i=1}^{t-1} g_t}{L(L+\sum_{i=1}^{t-1} |g_t|)}\right) Wealth_{t-1}$

Theorem (Orabona&Tommasi, ArXiv'17)

Let $F : \mathbb{R}^d \to \mathbb{R}$ be a convex function and assume that $\boldsymbol{g}_t$ satisfy $|g_{t,i}| \leq L_i$. Then, running COCOB for T iterations guarantees

$$\mathbb{E}[F(\bar{\boldsymbol{w}}_T)] - F(\boldsymbol{w}^*) \leq \tilde{\mathcal{O}}\left(\sum_{i=1}^{d} |w_i^*| \frac{\sqrt{\mathbb{E}\left[L_i \sum_{t=1}^{T} |g_{T,i}|\right]}}{T}\right),$$

- It holds for quasi-convex functions too
- Compare with AdaGrad with initial learning rate η:

$$\mathbb{E}[F(\bar{\boldsymbol{w}}_T)] - F(\boldsymbol{w}^*) \leq \mathcal{O}\left(\sum_{i=1}^{d} \left(\frac{(w_i^*)^2}{\eta} + \eta\right) \frac{\sqrt{\mathbb{E}\left[\sum_{t=1}^{T} g_{T,i}^2\right]}}{T}\right)$$

COCOB for Deep Learning

$$w_{t,i} = \sigma\left(\frac{\sum_{i=1}^{t-1} g_{t,i}}{L_i(L_i + \sum_{i=1}^{t-1} |g_{t,i}|)}\right) Wealth_{t-1,i}$$

COCOB for Deep Learning

$$w_{t,i} = \sigma\left(\frac{\sum_{i=1}^{t-1} g_{t,i}}{L_i(L_i + \sum_{i=1}^{t-1} |g_{t,i}|)}\right) Wealth_{t-1,i}$$

- Estimate L_i over time
- Assures that the Wealth remains positive
- Remove the sigmoid
- Make sure first steps are small

COCOB for Deep Learning

$$w_{t,i} = \sigma\left(\frac{\sum_{i=1}^{t-1} g_{t,i}}{L_i(L_i + \sum_{i=1}^{t-1} |g_{t,i}|)}\right) Wealth_{t-1,i}$$

- Estimate L_i over time
- Assures that the Wealth remains positive
- Remove the sigmoid
- Make sure first steps are small

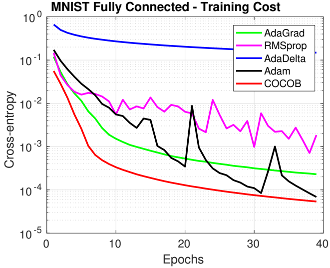
$$\begin{aligned}
L_{t,i} &= \max(L_{t-1,i}, |g_{t,i}|) \\
Wealth_{t-1,i} &= \min(Wealth_{t-1,i}, L_{t,i}) \\
w_{t,i} &= \frac{\sum_{i=1}^{t-1} g_{t,i}}{L_{t,i} \max(L_{t,i} + \sum_{i=1}^{t-1} |g_{t,i}|, 100 L_{t,i})} Wealth_{t-1,i}
\end{aligned}$$

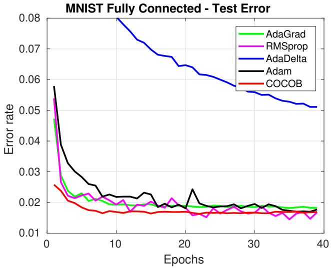
And More...

- All the results hold in the more general Online Convex Optimization settings
- Learning With Expert Advice with Betting [Orabona&Pal, NIPS'16]
- Sleeping Experts [Jun el al., AISTATS'17]
  - Experts do not always output a prediction
- Online Learning with changing environments [Jun el al., AISTATS'17]
- Optimal rates of convergence for kernel SVM, without hyperparameters to tune [Orabona, NIPS'14]

Outline

Deep Learning: MNIST, fully connected

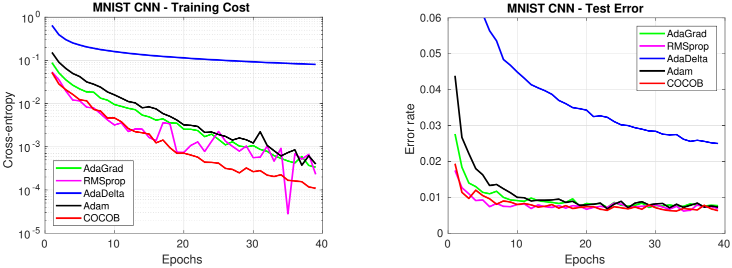




Training cost (cross-entropy) (left) and testing error rate (0/1 loss) (right) vs. the number epochs

[Orabona&Tommasi, ArXiv'17]

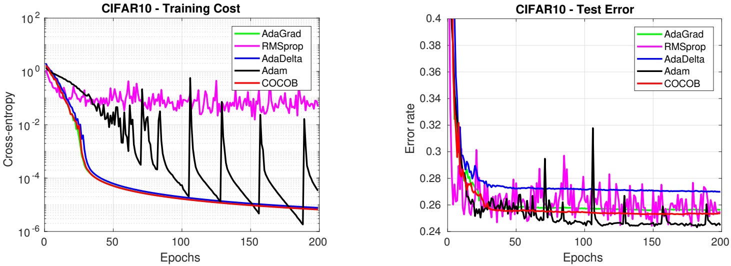
Deep Learning: MNIST, CNN



Training cost (cross-entropy) (left) and testing error rate (0/1 loss) (right) vs. the number epochs

[Orabona&Tommasi, ArXiv'17]

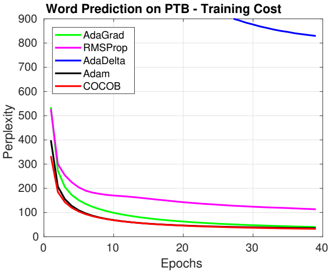
Deep Learning: CIFAR10

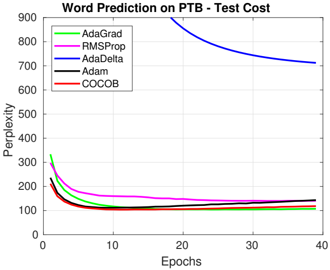


Training cost (cross-entropy) (left) and testing error rate (0/1 loss) (right) vs. the number epochs

[Orabona&Tommasi, ArXiv'17]

Deep Learning: PTB, word-level





Training cost (left) and test cost (right) measured as average per-word perplexity vs. the number epochs

[Orabona&Tommasi, ArXiv’17]

Conclusions

- Learning rates in SGD for Lipschitz functions are unnecessary
- SGD, Learning with Experts, SVMs, etc. can be reduced to betting on a coin
- Betting algorithms are easy to design, hyperparameter-free, and (most of the time) optimal

Future Work:

- Better bound, matching tuned AdaGrad
- Coin betting and Newton algorithms?

Thanks for your attention

`http://francesco.orabona.com`

TensorFlow COCOB code: `http://github.com/bremen79/cocob`

Thanks to my collaborators: Kwang-Sung Jun, David Pal, Brendan McMahan, Tatiana Tommasi, Rebecca Willett, Stephen Wright

Thanks to the support of Google through the Google Research Award