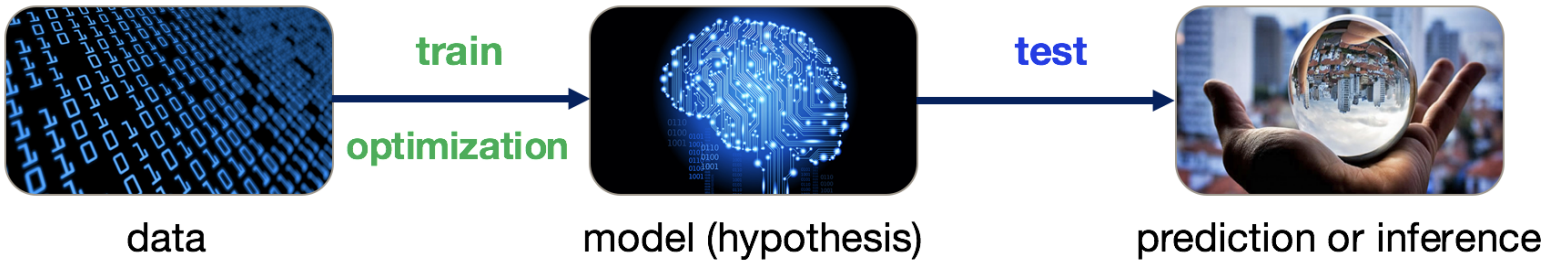
ISEN 689 – Learning and Optimization over Networks (2nd part)

Shahin Shahrampour

Industrial & Systems Engineering
Texas A&M University

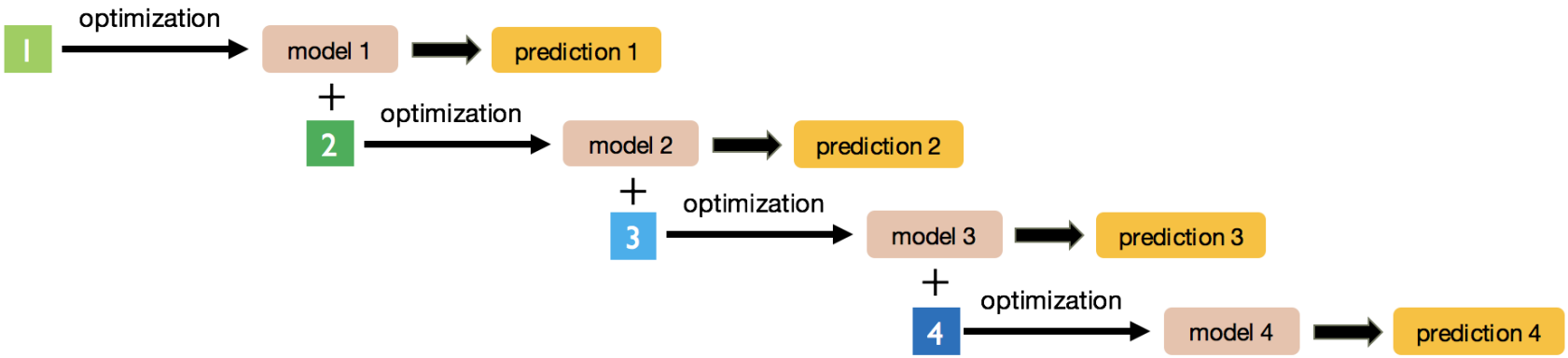
- For a lot of interesting problems in **machine learning**, we need to solve an **optimization** to extract some model parameters from data (examples in the next slides). This is done during the training process.



- Thanks to advances in computing power, many practical problems that we are currently dealing with are **large-scale**. What does that mean?
- Roughly speaking: the number of data samples is **huge**; the model is **high-dimensional**.
- In this part of the course we mostly deal with the first problem (i.e., massive data).
- How? **Online Optimization** & **Distributed Optimization**.

Online optimization

- Data samples are received sequentially.
- At each time (iteration) we use the new information (data) to update the previous model.

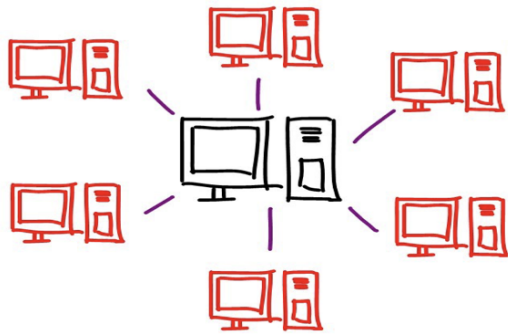


- Why online? How can it help?
  - Sometimes nature of the problem is online (ad placement).
  - We don't have to store all the data.
- **Key question**: How far we are off from offline optimization?

Distributed optimization

- Data samples are distributed. Why?
  - Sometimes nature of the problem is distributed (e.g., sensor networks).
  - We don't want to store the data in one place (e.g., memory issues in computer networks).

DARPA's 'Ocean of things' illustration:
Image by DARPA



Courtesy of https://shubheksha.com/

- **Key question**: How can we learn the **global** data model using **local** interaction?
- For distributed optimization, we build on **consensus** models, covered in the first part of the course.

Outline 1: Offline (and stochastic) optimization basics

- General offline optimization problem (convex vs. non-convex)
- Introduce some of the classical optimization algorithms
  - Gradient descent (GD)
  - Nesterov's accelerated gradient descent
  - Newton's method
- Finite-sum problems
- Examples (regression, logistic regression, support vector machine)
- Introduce more modern optimization algorithms
  - Stochastic gradient descent (SGD)
  - Adaptive gradient descent (AdaGrad)
  - Adam: A method for stochastic optimization
- Summary

General offline optimization

We are interested in solving the problem

$$\min_{\boldsymbol{\theta} \in \Theta} \; f(\boldsymbol{\theta})$$

- f is the **objective** function.
- $\boldsymbol{\theta} \in \mathbb{R}^d$ is multi-variate (vector).
- $\Theta \subseteq \mathbb{R}^d$ is the **constraint** set. If $\Theta = \mathbb{R}^d$, the problem is unconstrained.

– The problem is **convex** if the objective function **and** the constraint set are both convex. Otherwise, the problem is **non-convex**.

– Convex problems are easy to solve and theoretically very well-understood.

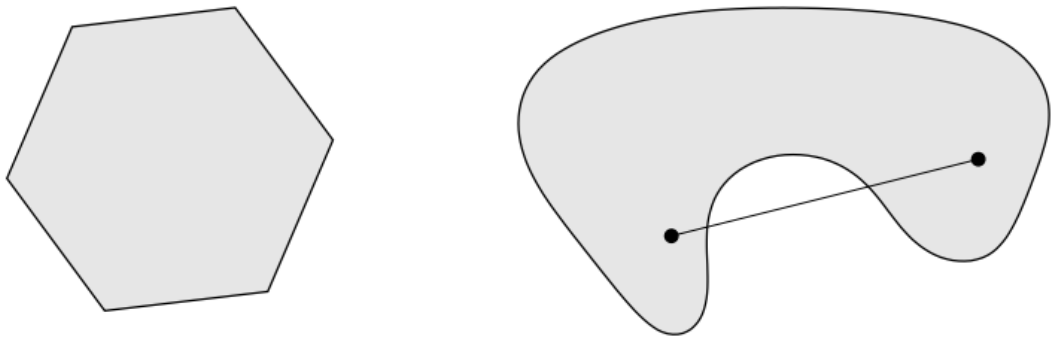
– In this course, the focus in on convex, but non-convex is also very important (e.g., deep learning).

Convex set

- A set Θ is **convex** if for any two points in the set, the line segment connecting the two is inside the set:

$$\forall\, \boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta \quad \text{and} \quad 0 \leq \lambda \leq 1 \Rightarrow \lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta}' \in \Theta$$

- Examples from Stephen Boyd's slides (left is convex, right is non-convex)



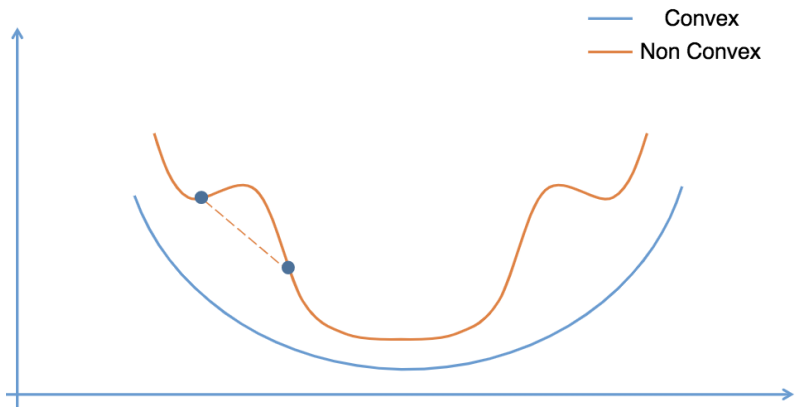
- Is the surface of a sphere convex? How about the full sphere?

Convex function

- A function $f : \mathbb{R}^d \to \mathbb{R}$ is **convex**, if its domain Θ is a convex set, and

$$f(\lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta}') \leq \lambda f(\boldsymbol{\theta}) + (1-\lambda) f(\boldsymbol{\theta}')$$

for any $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$ and $0 \leq \lambda \leq 1$.



Courtesy of https://fmin.xyz/docs/theory/Theory/

- f is concave if $-f$ is convex.

Convex function (when differentiable)

- A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is **convex**, if its domain Θ is a convex set, and

$$f(\boldsymbol{\theta}') \geq f(\boldsymbol{\theta}) + \nabla f(\boldsymbol{\theta})^\top (\boldsymbol{\theta}' - \boldsymbol{\theta})$$

for any $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$.

- A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is μ-**strongly convex**, if its domain Θ is a convex set, and

$$f(\boldsymbol{\theta}') \geq f(\boldsymbol{\theta}) + \nabla f(\boldsymbol{\theta})^\top (\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{\mu}{2} \left\| \boldsymbol{\theta}' - \boldsymbol{\theta} \right\|^2$$

for any $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$ and some $\mu > 0$.

Gradient descent

- One of the most basic (yet efficient) optimization methods, where we run the following update to converge to (local) minimum of a convex function

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \nabla f(\boldsymbol{\theta}_t)$$

- $\{\alpha_t\}$ is the step-size sequence, chosen based on the problem setting.

Theorem

Suppose that f is convex, differentiable, and its gradient is Lipschitz continuous, i.e., $\left\|\nabla f(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\theta}')\right\| \leq L \left\|\boldsymbol{\theta} - \boldsymbol{\theta}'\right\|$. If we run gradient descent with a fixed step-size $\alpha_t = \alpha \leq 1/L$, we have

$$f(\boldsymbol{\theta}_t) - f(\boldsymbol{\theta}^\star) \leq \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^\star\|^2}{\alpha t},$$

where $\boldsymbol{\theta}^\star \in \operatorname{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$.

Proof

Due to the assumption that gradient is Lipschitz continuous (with constant L), we have for any θ, θ' that

$$f(\theta') \leq f(\theta) + \nabla f(\theta)^\top (\theta' - \theta) + \frac{L}{2} \left\| \theta' - \theta \right\|^2$$

Now, let $\theta' = \theta_{t+1}$ and $\theta = \theta_t$, and recall that based on the gradient descent update $\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t)$. Then,

$$\begin{aligned} f(\theta_{t+1}) &\leq f(\theta_t) + \nabla f(\theta_t)^\top (\theta_{t+1} - \theta_t) + \frac{L}{2} \left\| \theta_{t+1} - \theta_t \right\|^2 \\ &= f(\theta_t) - \alpha \nabla f(\theta_t)^\top \nabla f(\theta_t) + \frac{L\alpha^2}{2} \left\| \nabla f(\theta_t) \right\|^2 \\ &= f(\theta_t) - \alpha (1 - \frac{L\alpha}{2}) \left\| \nabla f(\theta_t) \right\|^2 \\ &\leq f(\theta_t) - \frac{\alpha}{2} \left\| \nabla f(\theta_t) \right\|^2, \end{aligned}$$

where the last line is due to $\alpha \leq 1/L$.

Proof

Now since f is convex

$$f(\boldsymbol{\theta}_t) \leq f(\boldsymbol{\theta}^\star) + \nabla f(\boldsymbol{\theta}_t)^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star),$$

which (combined with the previous slide) implies

$$\begin{aligned} f(\boldsymbol{\theta}_{t+1}) - f(\boldsymbol{\theta}^\star) &\leq \nabla f(\boldsymbol{\theta}_t)^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) - \frac{\alpha}{2} \|\nabla f(\boldsymbol{\theta}_t)\|^2 \\ &= \frac{1}{2\alpha} \left(2\alpha \nabla f(\boldsymbol{\theta}_t)^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) - \alpha^2 \|\nabla f(\boldsymbol{\theta}_t)\|^2 \right) \\ &= \frac{1}{2\alpha} \left(\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star - \alpha \nabla f(\boldsymbol{\theta}_t)\|^2 \right) \\ &= \frac{1}{2\alpha} \left(\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2 \right). \end{aligned}$$

Summing above, the right-hand side telescopes and

$$\sum_{s=0}^{t-1} \left(f(\boldsymbol{\theta}_{s+1}) - f(\boldsymbol{\theta}^\star) \right) \leq \frac{\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^\star\|^2}{2\alpha}.$$

As $f(\boldsymbol{\theta}_t) - f(\boldsymbol{\theta}^\star) \leq \frac{1}{t} \sum_{s=0}^{t-1} \left(f(\boldsymbol{\theta}_{s+1}) - f(\boldsymbol{\theta}^\star) \right)$ (**why?**), we are done.

Gradient descent for strongly convex functions

- The previous rate was for **convex** functions. Can we do better for **strongly convex** functions?

Theorem

Suppose that f is μ-strongly convex, differentiable, and its gradient is Lipschitz continuous, i.e., $\|\nabla f(\theta) - \nabla f(\theta')\| \leq L\|\theta - \theta'\|$. If we run gradient descent with a fixed step-size $\alpha_t = \alpha = 1/L$, we have

$$f(\theta_{t+1}) - f(\theta^\star) \leq \left(1 - \frac{\mu}{L}\right)^t (f(\theta_0) - f(\theta^\star)),$$

where $\theta^\star = \text{argmin}_\theta f(\theta)$.

- Convergence rate is **linear** (exponentially fast).
- For proof see Week 5, Gradient Descent, Theorem 3 of **Yaron Singer's course**.

Nesterov's accelerated gradient descent

- Vanilla gradient descent has a convergence rate of $O(1/t)$ after t iterations (for convex problems). Nesterov's accelerated gradient descent is a modified version of the algorithm that achieves $O(1/t^2)$ convergence rate.
- The algorithm follows the two intertwined updates:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}'_t - \frac{1}{L}\nabla f(\boldsymbol{\theta}'_t)$$
$$\boldsymbol{\theta}'_{t+1} = (1-\alpha_t)\boldsymbol{\theta}_{t+1} + \alpha_t \boldsymbol{\theta}_t$$

  where $\alpha_t = \frac{1-\lambda_t}{\lambda_{t+1}}$ and $\lambda_t = \frac{1+\sqrt{1+4\lambda_{t-1}^2}}{2}$ with $\lambda_0 = 0$.
- See **Sebastien Bubeck's blog** for proof and more information.
- So far a key assumption was L-smoothness (Lipschitz continuity of gradients). Without that, other sorts of guarantees are possible (see Chapter 2 of **Elad Hazan's book**).

Newton's method

- Gradient descent is a **first-order** method, i.e., it uses only gradient (first-order) information. Can we speed up the convergence using **second-order** information?

- Newton's method does so using the update

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \mathbf{H}_t^{-1} \nabla f(\boldsymbol{\theta}_t),$$

  where $\mathbf{H}_t \triangleq \nabla^2 f(\boldsymbol{\theta}_t)$ is the Hessian of function f evaluated at $\boldsymbol{\theta}_t$.

- This update uses Hessian information, and we can expect to achieve faster rates, but that comes at the cost of more expensive operations! In particular, the Hessian inversion is expensive when d is large.

Newton's method

Under the assumptions that:

1) The Hessian is Lipschitz continuous with parameter M, i.e.,

$$\left\|\nabla^2 f(\boldsymbol{\theta}) - \nabla^2 f(\boldsymbol{\theta}')\right\| \leq M \left\|\boldsymbol{\theta} - \boldsymbol{\theta}'\right\|.$$

2) There exists a local minimum of function f with positive definite Hessian, i.e., $\nabla^2 f(\boldsymbol{\theta}^\star) \succcurlyeq \mu \mathbf{I}_d$ for $\mu > 0$ and $\boldsymbol{\theta}^\star \in \operatorname{argmin}_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$.

3) The starting point $\boldsymbol{\theta}_0$ is chosen such that $\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^\star\| < \frac{2\mu}{3M}$.

We have the following **quadratic** convergence (double exponential):

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\| \leq \frac{M \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2}{2(\mu - M \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|)}$$

- We can use gradient method to get close to a local minimum, and do the final job by applying the Newton's method.
- See Pages 36-37 of **Nesterov's book** for the proof.

Finite-sum problems

- What we have seen so far are classical and textbook optimization algorithms.
- In many practical (and modern) machine learning problems, the objective function can we written as a sum,

$$f(\theta) = \frac{1}{T} \sum_{t=1}^{T} f_t(\theta)$$

  where f_t is formed based on a portion of data (rather than the whole data) and T can be potentially large.

- Example 1: empirical risk minimization (e.g., regression and classification), where f_t's are statistically the same.

- Example 2: distributed optimization

Linear regression

- We are given a dataset $\{\mathbf{x}_t, y_t\}_{t=1}^{T}$, and we believe that the generating model is $y = \mathbf{x}^\top \theta^\star + \epsilon$, where ϵ is a zero-mean, finite-variance noise.

- $\theta^\star$ is unknown. We are interested in finding the estimate of $\theta^\star$ by solving a least-square problem

$$\min_{\theta} \left\{ f(\theta) = \frac{1}{T} \sum_{t=1}^{T} (y_t - \theta^\top \mathbf{x}_t)^2 \right\}$$

- Example 1: predicting house price based on number of bedrooms, square footage, neighborhood, etc.

- Example 2: predicting car insurance premium based on driver's age and experience, city, etc.

- Many other examples where y is a continuous variable...

Logistic regression

- We are given a dataset $\{\mathbf{x}_t, y_t\}_{t=1}^{T}$, where $y_t \in \{-1, 1\}$ is binary.
- We are interested in finding the estimate of $\theta^\star$ by minimizing the logistic loss

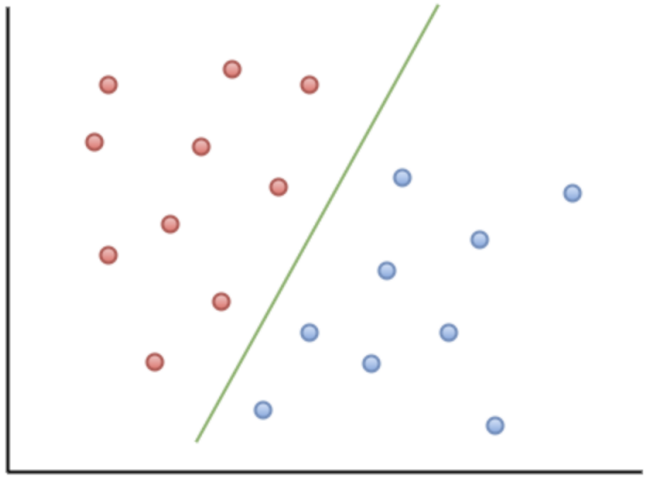
$$\min_{\theta} \left\{ f(\theta) = \frac{1}{T} \sum_{t=1}^{T} \log \left[1 + e^{-y_t \theta^\top \mathbf{x}_t} \right] \right\}$$

- Examples: spam filtering, image classification, face recognition, ...
- Many other examples where y is a discrete variable. Multi-class logistic regression is also possible.

Support vector machine

- We are interested in finding the estimate of $\theta^\star$ by minimizing the hinge loss

$$\min_{\theta} \left\{ f(\theta) = \frac{1}{T} \sum_{t=1}^{T} \max \left\{ 1 - y_t \theta^\top \mathbf{x}_t, 0 \right\} \right\}$$



Courtesy of https://geohackweek.github.io/machine-learning/02-svm/

- **What happens if T is large in finite-sum problems?**

Stochastic gradient descent (SGD)

- For finite-sum problems calculating the exact gradient is difficult when T is large. So, in practice we can use randomized approximations of the true gradient. SGD does so by randomly selecting one of the functions in the sum. That is,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha_t \nabla f_{j_t}(\boldsymbol{\theta}_t),$$

  where j_t is an index selected uniformly at random from $\{1, \ldots, T\}$.

- This selection ensures that $\mathbb{E}[\nabla f_{j_t}(\cdot)] = \nabla f(\cdot)$ (unbiased estimation).

- Since we approximate the gradient based on only one of the summands, the computation does not scale with T and we can avoid memory issues.

- But how does the convergence rate change?

Stochastic gradient descent (SGD)

- First, notice that the convergence guarantees for SGD are probabilistic (e.g., in expectation) due to the stochastic nature of the algorithm.
- Generally speaking, the convergence rates are slower ("**not faster**" to be accurate) than gradient descent (GD).
- For example, GD achieves a linear (exponentially fast) rate for strongly convex functions with Lipschitz continuous gradients. However, with the same conditions, we can only get the following guarantee

$$\mathbb{E}[f(\boldsymbol{\theta}_t)] - f(\boldsymbol{\theta}^\star) = O(1/t),$$

  much slower than GD.
- For the proof, see Theorem 4.7 of **this paper**.
- SGD is popular in large-scale machine learning (even for non-convex problems).

Adaptive gradient method (AdaGrad)

- GD and SGD use the same learning rate (step-size) for all of the components of the vector that is being updated (i.e., $\boldsymbol{\theta}_t$). Can we adjust the rate **adaptively** for each component?
- Yes! **Duchi et al.** proposed an adaptive (sub-)gradient method of the following form

$$\theta_{i,t+1} = \theta_{i,t} - \frac{\alpha}{\sqrt{\sum_{s=1}^{t} \nabla_{i,s}^2 + \epsilon}} \nabla_{i,t},$$

  where $\nabla_{i,t}$ the partial derivative of the objective function w.r.t. to the parameter θ_i. $\epsilon > 0$ is a small constant to avoid zero in the denominator if gradients are small.
- Each dimension has a specific learning rate for its own.
- Efficiently deals with sparse features and gradients.

Remarks for AdaGrad

- It performs larger updates for infrequent and smaller updates for frequent parameters. For this reason, it is well-suited for dealing with sparse data.
- Step-size tuning is one of the most challenging problems in practical optimization.
- One of AdaGrad's main benefits is that it eliminates the need to manually tune the step-size. Most implementations use a default value of 0.01 and leave it at that.
- AdaGrad's main weakness is its accumulation of the squared gradients in the denominator: Since every added term is positive, the accumulated sum keeps growing during training. This in turn causes the learning rate to shrink and eventually become infinitesimally small, at which point the algorithm is no longer able to acquire additional knowledge.
- Source: **this survey paper**.

Adam: A method for stochastic optimization

- Another algorithm that **adaptively** chooses the learning rate by moment estimation.
- **The paper** proposed a method that estimates the moments of stochastic gradient $\mathbf{g}_t$ as

$$\mathbf{m}_{t+1} = \beta_1 \mathbf{m}_t + (1-\beta_1)\mathbf{g}_t \qquad \mathbf{v}_{t+1} = \beta_2 \mathbf{v}_t + (1-\beta_2)\mathbf{g}_t \odot \mathbf{g}_t$$

  where $\odot$ is component-wise product, and $\beta_1, \beta_2 \in [0, 1)$.
- And it corrects the estimates by

$$\widehat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t} \qquad \widehat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t}$$

- Then, we have the optimization update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\alpha \widehat{\mathbf{m}}_t}{\sqrt{\widehat{\mathbf{v}}_t}+\epsilon}$.

Summary

- This was a quick introduction of some optimization algorithms. You can also find more modern algorithms in **this survey paper**.
- Methods that we have seen so far are neither online nor distributed! But some of them do have online and/or distributed variants.
- Remember that in online setting, we receive data points (loss functions) sequentially, so we can't choose a loss function purely at random (to estimate the gradient).
- In distributed setting, there is a networks of agents, and each agent has access to a specific portion of data points (loss functions). They communicate with each other to find the global solution.

Outline 2: First-order online optimization

This part closely follows Chapter 3 of **Elad Hazan's book**:

- Online optimization and regret definition
- Online gradient descent (OGD)
- Regret bound for convex functions
- Optimality (lower bound)
- Improved regret for strongly convex functions
- Application of OGD to stochastic optimization
- Summary

Online optimization

- In statistical learning, we deal with finite-sum problems in which individual loss functions (summands) are **statistically** the same. This is due to the assumption that data points are independent and identically distributed (i.i.d.) samples from a fixed, unknown distribution.

- Online learning deals with finite-sum problems in which loss functions are not necessarily the same (statistically). The functions may even be chosen in an adversarial way (under some constraints). The other distinction is that the loss functions become available sequentially.

- We discuss the most simple and basic algorithms for online convex optimization and introduce a new measure – **regret** – for understanding their theoretical performance.

- Regret is a measure that tells us how well the online algorithm performs versus an offline benchmark.

Regret

- Regret is defined as follows:

$$\text{regret} \triangleq \sum_{t=1}^{T} f_t(\theta_t) - \min_{\theta \in \Theta} \sum_{t=1}^{T} f_t(\theta),$$

  where $\{\theta_t\}_{t=1}^{T}$ is generated by an **online** algorithm, i.e., an algorithm that generates θ_t based **only** on information from prior observations $\{f_s\}_{s=1}^{t-1}$.

- Notice that the functions are **time-varying** (dependence on t).

- $\min_{\theta \in \Theta} \sum_{t=1}^{T} f_t(\theta)$ is the offline benchmark: what we could have achieved had we known all the functions in advance. Regret tells us how far we are from this benchmark. Ideally, we want algorithms that scale sub-linearly with time (i.e., $o(T)$) so that on average the regret goes to zero, i.e., $\text{regret}/T \to 0$ as $T \to \infty$.

Regret

- A more accurate definition of regret also has a **supremum** over all functions, which means they can be chosen adversarially as well...
- Regret definition can also be useful when functions are time-invariant (i.e., when $f_t = f$).
- Let us define $\bar{\boldsymbol{\theta}}_T \triangleq \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\theta}_t$. Then, due to convexity

$$f(\bar{\boldsymbol{\theta}}_T) - f(\boldsymbol{\theta}^\star) \leq \frac{1}{T}\sum_{t=1}^{T}\Big(f(\boldsymbol{\theta}_t) - f(\boldsymbol{\theta}^\star)\Big) = \frac{\text{regret}}{T}.$$

- So instead of a guarantee for $\boldsymbol{\theta}_t$ (as we had before), we will get a guarantee for $\bar{\boldsymbol{\theta}}_t$.

Online gradient descent

- Online gradient descent (OGD) is pretty much like the usual gradient descent. Here, since we are looking at a constrained problem ($\boldsymbol{\theta} \in \Theta$), we also need a projection:

$$\boldsymbol{\theta}'_{t+1} = \boldsymbol{\theta}_t - \alpha_t \nabla f_t(\boldsymbol{\theta}_t) \qquad \boldsymbol{\theta}_{t+1} = \Pi_\Theta(\boldsymbol{\theta}'_{t+1})$$

- $\Pi_\Theta(\cdot) \triangleq \operatorname{argmin}_{\boldsymbol{\theta} \in \Theta} \|\cdot - \boldsymbol{\theta}\|$ is the Euclidean projection.

- The algorithm takes a step from the previous point in the direction of the gradient of the previous loss. This step may result in a point outside of the underlying convex set. In such cases, the algorithm projects the point back to the convex set, i.e., finds its closest point in the convex set.

- This algorithm was analyzed by **Zinkevich 2003**.

Online gradient descent

- We assume bounded diameter and gradient in the sense that for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$, we have that $\left\|\boldsymbol{\theta} - \boldsymbol{\theta}'\right\| \leq D$ and $\|\nabla f(\boldsymbol{\theta})\| \leq G$.

- The following theorem suggests that online gradient descent achieves a sub-linear regret. For its proof, we don't need Lipschitz continuity of gradients, but of course the rate is slower than $O(1/t)$ of GD.

Theorem

Suppose that $\{f_t\}_{t=1}^T$ are convex and differentiable. If we run online gradient descent with the diminishing step-size $\alpha_t = \frac{D}{G\sqrt{t}}$, we have

$$\textit{regret} = \sum_{t=1}^{T} f_t(\boldsymbol{\theta}_t) - \min_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} f_t(\boldsymbol{\theta}) \leq \frac{3}{2} GD\sqrt{T}.$$

Proof

Let $\boldsymbol{\theta}^\star \triangleq \text{argmin}_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} f_t(\boldsymbol{\theta})$ and $\nabla_t \triangleq \nabla f_t(\boldsymbol{\theta}_t)$. Now since f_t is convex

$$f_t(\boldsymbol{\theta}_t) - f_t(\boldsymbol{\theta}^\star) \leq \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star),$$

First, we need to find an upper bound for the right-hand side. By the OGD update (and the Pythagorean theorem) we have that

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2 &= \|\Pi_\Theta(\boldsymbol{\theta}_t - \alpha_t \nabla_t) - \boldsymbol{\theta}^\star\|^2 \leq \|\boldsymbol{\theta}_t - \alpha_t \nabla_t - \boldsymbol{\theta}^\star\|^2 \\ &= \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 + \alpha_t^2 \|\nabla_t\|^2 - 2\alpha_t \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star),\end{aligned}$$

which implies

$$2\nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) \leq \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2}{\alpha_t} + \alpha_t G^2$$

since the gradients are bounded by G. We can now immediately see that

$$2 * \text{regret} \leq \sum_{t=1}^{T} 2\nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) \leq \sum_{t=1}^{T} \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2}{\alpha_t} + \alpha_t G^2$$

Proof

To find an upper bound for the right-hand side of the previous equation, observe that

$$G^2 \sum_{t=1}^{T} \alpha_t \leq 2GD\sqrt{T}, \qquad (I)$$

by the choice of step-size in theorem. Also, defining $1/\alpha_0 = 0$, we get

$$\begin{aligned}
\sum_{t=1}^{T} \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2}{\alpha_t} &\leq \sum_{t=1}^{T} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 \left(\frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}}\right) \\
&\leq D^2 \sum_{t=1}^{T} \frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}} \\
&= D^2 \frac{1}{\alpha_T} = GD\sqrt{T} \qquad (II)
\end{aligned}$$

Combining (I) and (II) finishes the proof.

Optimality (lower bound)

- Can the previous bound be improved? Is there any other algorithm that can outperform OGD by giving a better regret rate for convex problems?
- The answer is surprisingly "no"!
- Up to small constant factors, OGD is optimal, which means we can always construct problem environments where no algorithm would do better than OGD (in terms of regret).

Theorem

Any algorithm for online convex optimization incurs $\Omega(GD\sqrt{T})$ regret in the worst case. This is true even if the cost functions are generated from a fixed stationary distribution.

Proof sketch

- We can consider a problem where the convex set is the d-dimensional hypercube, i.e., $\Theta = \{\boldsymbol{\theta} \in \mathbb{R}^d : \|\boldsymbol{\theta}\|_\infty \leq 1\}$.
- The loss functions are chosen linearly such that $f_t(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{v}_t$, and $\mathbf{v}_t \in \mathbb{R}^d$ is a d-dimensional vector whose components are chosen from $\{-1, 1\}$ uniformly at random.
- For this problem setup, one can show that regret is $\Omega(d\sqrt{T})$.
- **Exercise:** For a part of proof you need to show that for i.i.d. random variables $\{s_t\}_{t=1}^T$, where s_t is uniformly chosen from $\{-1, 1\}$, we have

$$\mathbb{E}\Big|\sum_{t=1}^{T} s_t\Big| \leq \sqrt{T}$$

- For more details, see Theorem 3.2 of **Elad Hazan's book**.

Online gradient descent for strongly convex functions

- We previously saw that for gradient descent, **strong convexity** can improve the convergence rate. Will we get this advantage (i.e., improved regret bound) in the online setup with OGD?

- Yes! This is possible only by changing the step-size. Using this simple modification, we can improve the regret rate from $O(\sqrt{T})$ to $O(\log T)$.

Theorem

Suppose that $\{f_t\}_{t=1}^T$ are μ-strongly convex and differentiable. If we run online gradient descent with the diminishing step-size $\alpha_t = \frac{1}{\mu t}$, we have

$$\textit{regret} \leq \frac{G^2}{2\mu}(1 + \log T).$$

Proof

Let $\boldsymbol{\theta}^\star \triangleq \text{argmin}_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} f_t(\boldsymbol{\theta})$ and $\nabla_t \triangleq \nabla f_t(\boldsymbol{\theta}_t)$. By strong convexity of f_t

$$2(f_t(\boldsymbol{\theta}_t) - f_t(\boldsymbol{\theta}^\star)) \leq 2\nabla_t^\top(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) - \mu \left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\right\|^2,$$

and from the previous proof, we know that

$$2\nabla_t^\top(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) \leq \frac{\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\right\|^2 - \left\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\right\|^2}{\alpha_t} + \alpha_t G^2$$

We can now immediately see that

$$\begin{aligned}
2 * \text{regret} &\leq \sum_{t=1}^{T} 2\nabla_t^\top(\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star) - \mu \left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\right\|^2 \\
&\leq \sum_{t=1}^{T} \frac{\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\right\|^2 - \left\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\right\|^2}{\alpha_t} + \alpha_t G^2 - \mu \left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\right\|^2
\end{aligned}$$

Proof

To find an upper bound for the right-hand side of the previous equation, observe that

$$G^2 \sum_{t=1}^{T} \alpha_t \leq \frac{G^2}{\mu}(1 + \log T),$$

by the choice of step-size in theorem. Also, defining $1/\alpha_0 = 0$, we get

$$\sum_{t=1}^{T} \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^\star\|^2}{\alpha_t} - \mu \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 \leq \sum_{t=1}^{T} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^\star\|^2 \left(\frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}} - \mu\right),$$

and

$$\frac{1}{\alpha_t} - \frac{1}{\alpha_{t-1}} - \mu = 0.$$

Therefore,

$$\text{regret} \leq \frac{G^2}{2} \sum_{t=1}^{T} \alpha_t \leq \frac{G^2}{2\mu}(1 + \log T),$$

and we are done.

Application: stochastic gradient descent (SGD)

- We previously discussed the SGD algorithm, where we deal with a time-invariant function, and at each iteration SGD receives a stochastic (and unbiased) gradient $\tilde{\nabla}_t(\boldsymbol{\theta})$ of the function, i.e., $\mathbb{E}\left[\tilde{\nabla}_t(\boldsymbol{\theta})\right] = \nabla f(\boldsymbol{\theta})$.
- Assume that $\mathbb{E}\left[\left\|\tilde{\nabla}_t(\boldsymbol{\theta})\right\|^2\right] \leq G^2$ for any $\boldsymbol{\theta} \in \Theta$.
- Then, using the OGD regret bound for convex functions, we only need a few basic steps to prove that OGD with stochastic gradient enjoys the following rate (see Theorem 3.4 of **Elad Hazan's book**)

$$\mathbb{E}[f(\bar{\boldsymbol{\theta}}_T)] - f(\boldsymbol{\theta}^\star) \leq \frac{3GD}{2\sqrt{T}}.$$

- This shows the generality of online optimization setting, by using which we can prove stochastic optimization results.
- We can also extend the result to **strongly convex** functions to get convergence rate of $O(\log T / T)$.

Summary

- We discussed OGD, a basic algorithm for **online optimization**.
- OGD achieves $O(\sqrt{T})$ and $O(\log T)$ regret rates for convex and strongly convex problems, respectively.
- We also saw the $O(\sqrt{T})$ rate in the convex setting is optimal, i.e., no other algorithm can improve upon that.
- The generality of online optimization framework allows to study convergence properties of certain stochastic optimization problems.
- OGD is a first-order algorithm using only gradient information. Can we benefit from second-order information (as we did in the offline case) to improve regret rates?

Outline 3: Second-order online optimization

This part closely follows Chapter 4 of **Elad Hazan's book**:

- Motivation for second order online optimization
- Exp-concave functions
- The online Newton step (ONS) algorithm
- Regret bound of ONS for exp-concave functions
- Implementation and running time of ONS
- Summary

Motivation

- We saw in offline optimization that using second order information can substantially speed up the convergence rate (assuming local strong convexity). Is that also true for online optimization?
- Yes! In fact, we need a condition (exp-concavity) that is somewhat weaker than strong convexity.
- Chapter 4.1 of **Elad Hazan's book** discusses an application (portfolio selection) where the objective function is exp-concave.
- In particular, the loss function $f_t(\theta) = -\log(\mathbf{r}_t^\top \theta)$ in portfolio selection is not strongly convex but is exp-concave.
- Can we get an improved regret rate (i.e., better that $O(\sqrt{T})$ of OGD with first order information) for the class of exp-concave functions?

Exp-concave functions

- A function $f : \mathbb{R}^d \to \mathbb{R}$ is β-exp-concave over $\Theta \subseteq \mathbb{R}^d$ if the function

$$g(\boldsymbol{\theta}) = \exp(-\beta f(\boldsymbol{\theta}))$$

  is concave ($\beta > 0$).

- This intuitively means that the Hessian of f is large in the direction of the gradient.

- A twice-differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is β-exp-concave at $\boldsymbol{\theta}$ if and only if

$$\nabla^2 f(\boldsymbol{\theta}) \succcurlyeq \beta \nabla f(\boldsymbol{\theta}) \nabla f(\boldsymbol{\theta})^\top$$

- Let function $f : \Theta \to \mathbb{R}$ be β-exp-concave. Let D and G denote the diameter of Θ and a bound on the gradients of f, respectively. The following holds for all $\gamma \leq 0.5 \min\{\beta, \frac{1}{4GD}\}$ for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$

$$f(\boldsymbol{\theta}') \geq f(\boldsymbol{\theta}) + \nabla f(\boldsymbol{\theta})^\top (\boldsymbol{\theta}' - \boldsymbol{\theta}) + \frac{\gamma}{2} (\boldsymbol{\theta}' - \boldsymbol{\theta})^\top \nabla f(\boldsymbol{\theta}) \nabla f(\boldsymbol{\theta})^\top (\boldsymbol{\theta}' - \boldsymbol{\theta})$$

Online Newton step (ONS)

- It's a quasi-Newton approach in the sense that it approximates Hessian based on first-order information.
- ONS uses the following updates:

$$\boldsymbol{\theta}'_{t+1} = \boldsymbol{\theta}_t - \frac{1}{\gamma}\mathbf{A}_t^{-1}\nabla f_t(\boldsymbol{\theta}_t) \qquad \boldsymbol{\theta}_{t+1} = \Pi_{\Theta}^{\mathbf{A}_t}(\boldsymbol{\theta}'_{t+1})$$

- $\mathbf{A}_t = \mathbf{A}_{t-1} + \nabla f_t(\boldsymbol{\theta}_t)\nabla f_t(\boldsymbol{\theta}_t)^\top$ is a rank-one update initialized at $\mathbf{A}_0 = \epsilon\mathbf{I}$.
- $\Pi_{\Theta}^{\mathbf{A}}(\cdot) \triangleq \operatorname{argmin}_{\boldsymbol{\theta}\in\Theta} \|\cdot - \boldsymbol{\theta}\|_{\mathbf{A}}$ is the projection step.
- $\|\cdot\|_{\mathbf{A}}$ is norm with respect to $\mathbf{A}$ (positive definite): $\|\boldsymbol{\theta}\|_{\mathbf{A}}^2 \triangleq \boldsymbol{\theta}^\top\mathbf{A}\boldsymbol{\theta}$.
- The Newton's method would move in the direction of the vector which is the inverse Hessian times the gradient. In ONS, the direction is $\mathbf{A}_t^{-1}\nabla f_t(\boldsymbol{\theta}_t)$, where $\mathbf{A}_t$ is related to Hessian.

Online Newton step (ONS)

- We assume bounded diameter and gradient in the sense that for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$, we have that $\left\|\boldsymbol{\theta} - \boldsymbol{\theta}'\right\| \leq D$ and $\|\nabla f(\boldsymbol{\theta})\| \leq G$.

- The following theorem suggests that ONS achieves a logarithmic regret, which improves upon the $O(\sqrt{T})$ rate of OGD. But the result is only for exp-concave functions.

Theorem

Suppose that $\{f_t\}_{t=1}^T$ are differentiable and β-exp-concave. If we run the online Newton step algorithm with the parameters $\gamma = 0.5 \min\{\beta, \frac{1}{4GD}\}$ and $\epsilon = \frac{1}{\gamma^2 D^2}$, we have that

$$\textit{regret} \leq 5\left(\frac{1}{\beta} + GD\right) d \log T.$$

Remarks on the proof

- The proof (for most of the part) uses properties of exp-concavity, projection, and basic linear algebra. We have seen similar steps in the OGD proofs, and for more details, see Theorem 4.4 of **Elad Hazan's book**.

- The only part that is specific to this proof, and might be of separate interest, is the following inequality:

Lemma

Let $\mathbf{A} \succcurlyeq \mathbf{B} \succ 0$ *be positive definite matrices. Then*

$$Tr\left[\mathbf{A}^{-1}(\mathbf{A}-\mathbf{B})\right] \leq \log \frac{|\mathbf{A}|}{|\mathbf{B}|},$$

where $Tr[\cdot]$ *is the trace operator and* $|\mathbf{A}|$ *denotes the determinant of* $\mathbf{A}$.

Implementation and running time

- Calculation of $\mathbf{A}_t$ requires $O(d^2)$ operations.
- While (general) matrix inversion cost is cubic in the dimension, for $\mathbf{A}_t$ we don't have that problem. $\mathbf{A}_t$ accumulates rank-one matrices and due to this structure we can get a closed-form solution for the inversion by matrix inversion lemma.
- Consider an invertible matrix $\mathbf{A}$ and any vector θ. Then,

$$(\mathbf{A} + \theta\theta^\top)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\theta\theta^\top\mathbf{A}^{-1}}{1 + \theta^\top\mathbf{A}^{-1}\theta}.$$

- So given $\mathbf{A}_{t-1}^{-1}$ and the gradient, one can obtain $\mathbf{A}_t^{-1}$ with $O(d^2)$ operations using only matrix-vector and vector-vector products.
- The projection step is a convex program which can be solved up to any degree of accuracy in polynomial time. Projection complexity largely depends on the shape of the convex set. Sometimes even closed-form solutions are available...

Summary

- We discussed ONS, a quasi-Newton algorithm for **online optimization**.
- ONS uses gradient information to approximate Hessian.
- ONS achieves $O(\log T)$ regret rate for exp-concave functions.
- The ONS algorithm can be efficiently implemented with $O(d^2)$ operations per iteration (without the need for matrix inversion).

Outline 4: Regularization

This part closely follows Chapter 5 of **Elad Hazan's book**:

- Motivation for regularization
- Regularization functions
- Regularized follow the leader (RFTL) algorithm
- Regret bound for RFTL
- Online mirror descent (OMD) algorithm
- Regret bound for OMD
- Summary

Motivation

- Recall the definition of regret

$$\text{regret} \triangleq \sum_{t=1}^{T} f_t(\boldsymbol{\theta}_t) - \min_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} f_t(\boldsymbol{\theta}),$$

- The algorithms we have seen so far (OGD and ONS) use first order information for regret minimization. But when we look at the benchmark that we want to compete with (i.e., $\min_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} f_t(\boldsymbol{\theta})$), it seems that there exists a more straightforward approach for the online algorithm. Why not going with $\boldsymbol{\theta}_{t+1} = \operatorname{argmin}_{\boldsymbol{\theta}\in\Theta} \sum_{s=1}^{t} f_s(\boldsymbol{\theta})$?

- This flavor of strategy is known as "fictitious play" in economics, and has been named "Follow the Leader" (FTL) in machine learning.

- FTL fails miserably in a worst-case sense!

Motivation

- Consider an example where $\Theta = [-1, 1]$ and $f_1(\theta) = \frac{1}{2}\theta$. Also, let f_t's for any $t > 1$ alternate between $-\theta$ and θ such that

$$\sum_{s=1}^{t} f_s(\boldsymbol{\theta}) = \begin{cases} \frac{1}{2}\theta, & \text{if } t \text{ is odd} \\ -\frac{1}{2}\theta, & \text{otherwise} \end{cases}$$

- Then, the FTL algorithm will keep shifting between $\theta_t = -1$ and $\theta_t = 1$, always making the wrong choice. This results in a linear regret.

- The intuitive FTL strategy fails in the example above because it is unstable. Can we modify the FTL strategy such that it won't change decisions often, thereby causing it to attain low regret?

- This question motivates the need for a general means of stabilizing the FTL method. Such a means is referred to as "regularization".

Regularization functions

- We denote a generic regularization function by $R : \Theta \to \mathbb{R}$ and assume that R is strongly convex, smooth, and twice differentiable over Θ.

- We denote the diameter of the set Θ relative to the function R as

$$D_R \triangleq \left(\max_{\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta} \{R(\boldsymbol{\theta}) - R(\boldsymbol{\theta}')\} \right)^{\frac{1}{2}}$$

- For any norm $\|\cdot\|$, we denote its dual norm by $\|\boldsymbol{\theta}\|^\star \triangleq \max_{\|\boldsymbol{\theta}'\| \leq 1} \boldsymbol{\theta}^\top \boldsymbol{\theta}'$.

- **Exercise:** Recall that $\|\boldsymbol{\theta}\|_{\mathbf{A}}^2 \triangleq \boldsymbol{\theta}^\top \mathbf{A} \boldsymbol{\theta}$ for a positive definite $\mathbf{A}$. Show that $\|\cdot\|_{\mathbf{A}}^\star = \|\cdot\|_{\mathbf{A}^{-1}}$.

- **Generalized Cauchy-Schwarz theorem:** $\boldsymbol{\theta}^\top \mathbf{v} \leq \|\boldsymbol{\theta}\| \, \|\mathbf{v}\|^\star$ for any primal-dual norm pair.

Bregman divergence

- **Bregman divergence** is a crucial notion in online optimization. The difference between the value of the regularization function at θ and the value of the first order Taylor approximation is known as the Bregman divergence.

- We denote by $B_R(\theta\|\theta')$ the Bregman divergence with respect to the function R, defined as

$$B_R(\theta\|\theta') \triangleq R(\theta) - R(\theta') - \nabla R(\theta')^\top(\theta - \theta')$$

- Let's use the shorthand $\|\cdot\|_\theta \triangleq \|\cdot\|_{\nabla^2 R(\theta)}$ for the norm with respect to the Hessian of the regularization function.

- Taylor expansion and the mean-value theorem warrant that

$$B_R(\theta\|\theta') = \frac{1}{2}\left\|\theta - \theta'\right\|_{\mathbf{v}}^2$$

for some $\mathbf{v}$ on the line segment connecting θ and θ'.

Regularized follow the leader (RFTL)

- We saw that FTL may vary wildly from one iteration to the next. This motivates the modification of the basic FTL strategy in order to stabilize the prediction. By adding a regularization term, we obtain regularized follow the leader (RFTL) algorithm.

- Let $\nabla_t \triangleq \nabla f_t(\boldsymbol{\theta}_t)$ and denote by $\eta > 0$ the regularization parameter.

- RFTL follows the update below:

$$\boldsymbol{\theta}_{t+1} = \text{argmin}_{\boldsymbol{\theta}\in\Theta} \left\{ \eta \sum_{s=1}^{t} \nabla_s^\top \boldsymbol{\theta} + R(\boldsymbol{\theta}) \right\}$$

  with an initialization $\boldsymbol{\theta}_1 = \text{argmin}_{\boldsymbol{\theta}\in\Theta} \{R(\boldsymbol{\theta})\}$.

- The regularization parameter can be tuned as we will see later.

Regularized follow the leader (RFTL)

Theorem

Suppose that $\{f_t\}_{t=1}^T$ *are convex and differentiable. The RFTL algorithm attains for every* $\mathbf{v} \in \Theta$*, the following bound on the regret:*

$$\textit{regret} \leq 2\eta \sum_{t=1}^{T} \|\nabla_t\|_t^{\star 2} + \frac{R(\mathbf{v}) - R(\boldsymbol{\theta}_1)}{\eta}$$

- $\|\cdot\|_t$ is the norm defined with respect to the Hessian of R evaluated at some point on the line segment connecting $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}_{t+1}$.
- If an upper bound on the local norms is known, i.e., $\|\nabla_t\|_t^{\star} \leq G_R$ for all times t, then we can further optimize over the choice of η to obtain

$$\text{regret} \leq 2G_R D_R \sqrt{2T},$$

which is optimal up to constant factors.

Proof sketch

- The proof involves several steps. We first need a lemma that shows

$$\text{regret} \leq \sum_{t=1}^{T} \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}) + \frac{R(\mathbf{v}) - R(\boldsymbol{\theta}_1)}{\eta} \qquad (I)$$

  for any $\mathbf{v} \in \Theta$.
- To show the inequality above, we can use a proof by induction.
- The second step is to show that

$$\nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}) \leq 2\eta \|\nabla_t\|_t^{\star 2} \qquad (II)$$

- Then, if we combine (I) and (II), the result follows immediately.
- Let's see the proof of (II) in the next slide as it uses properties of Bregman divergence.

Proof sketch

Let's define $\Phi_t(\boldsymbol{\theta}) \triangleq \eta \sum_{s=1}^{t} \nabla_s^\top \boldsymbol{\theta} + R(\boldsymbol{\theta})$. By the Taylor expansion (with its explicit remainder term via the mean-value theorem) at $\boldsymbol{\theta}_{t+1}$, and by the definition of the Bregman divergence

$$\Phi_t(\boldsymbol{\theta}_t) = \Phi_t(\boldsymbol{\theta}_{t+1}) + (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1})^\top \nabla \Phi_t(\boldsymbol{\theta}_{t+1}) + B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}_{t+1}) \geq \Phi_t(\boldsymbol{\theta}_{t+1}) + B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}_{t+1}),$$

where the inequality follows by the optimality condition ($\boldsymbol{\theta}_{t+1}$ is the minimum of Φ_t over Θ). We now have that

$$\begin{aligned} B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}_{t+1}) &\leq \Phi_t(\boldsymbol{\theta}_t) - \Phi_t(\boldsymbol{\theta}_{t+1}) = \Phi_{t-1}(\boldsymbol{\theta}_t) - \Phi_{t-1}(\boldsymbol{\theta}_{t+1}) + \eta \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}) \\ &\leq \eta \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}), \end{aligned}$$

since $\boldsymbol{\theta}_t$ is the minimizer of Φ_{t-1}. By the generalized Cauchy-Schwarz inequality

$$\begin{aligned} \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}) &\leq \|\nabla_t\|_t^\star \, \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}\|_t = \|\nabla_t\|_t^\star \sqrt{2 B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}_{t+1})} \\ &\leq \|\nabla_t\|_t^\star \sqrt{2\eta \nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1})}, \end{aligned}$$

and by re-arranging the terms we can see that $\nabla_t^\top (\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}) \leq 2\eta \|\nabla_t\|_t^{\star 2}$.

Online mirror descent (OMD)

- In the convex optimization literature, **mirror descent** refers to a general class of first order methods generalizing gradient descent. Online mirror descent (OMD) is the online counterpart of this class of methods.

- The generality of OMD stems from the update being carried out in a "dual" space, where the duality notion is defined by the choice of regularization.

- The gradient of the regularization function defines a mapping from $\mathbb{R}^d$ onto itself, which is a vector field. The gradient updates are then carried out in this vector field.

- For RFTL the intuition was straightforward: the regularization was used to ensure stability of the decision. For OMD, regularization has an additional purpose: it transforms the space in which gradient updates are performed. This transformation enables better bounds in terms of the geometry of the space.

Online mirror descent (OMD)

- Let $\nabla_t \triangleq \nabla f_t(\boldsymbol{\theta}_t)$ and denote by $\eta > 0$ the regularization parameter.
- Let $\boldsymbol{\theta}'_1$ be chosen such that $\nabla R(\boldsymbol{\theta}'_1) = 0$ and $\boldsymbol{\theta}_1 = \text{argmin}_{\boldsymbol{\theta}\in\Theta} \left\{ B_R(\boldsymbol{\theta} \| \boldsymbol{\theta}'_1) \right\}$.
- OMD has two versions (agile and lazy):

$$\begin{aligned} &\text{[Lazy version]} && \nabla R(\boldsymbol{\theta}'_{t+1}) = \nabla R(\boldsymbol{\theta}'_t) - \eta \nabla_t \\ &\text{[Agile version]} && \nabla R(\boldsymbol{\theta}'_{t+1}) = \nabla R(\boldsymbol{\theta}_t) - \eta \nabla_t \end{aligned}$$

- After (one of the updates) above, project according to B_R:

$$\boldsymbol{\theta}_{t+1} = \text{argmin}_{\boldsymbol{\theta}\in\Theta} \left\{ B_R(\boldsymbol{\theta} \| \boldsymbol{\theta}'_{t+1}) \right\}$$

- **Exercise:** Show that for $R(\boldsymbol{\theta}) = 0.5 \|\boldsymbol{\theta}\|^2$, Agile-OMD is equivalent to OGD.
- When loss functions are linear, one can show that RFTL and lazy-OMD are equivalent! (see Lemma 5.5 of **Elad Hazan's book**)

Theorem

Suppose that $\{f_t\}_{t=1}^T$ are convex and differentiable. The agile-OMD algorithm attains the following bound on the regret:

$$\textit{regret} \leq \frac{\eta}{2} \sum_{t=1}^{T} \|\nabla_t\|_t^{\star 2} + \frac{R(\theta^\star) - R(\theta_1)}{\eta}$$

- Agile-OMD gives roughly the same regret bound as the RFTL algorithm.
- As before, if an upper bound on the local norms is known, i.e., $\|\nabla_t\|_t^\star \leq G_R$ for all times t, then we can further optimize over the choice of η to obtain

$$\text{regret} \leq G_R D_R \sqrt{T}$$

Proof

First, since f_t is convex, for $\mathbf{v} \in \Theta$, we have that

$$f_t(\boldsymbol{\theta}_t) - f_t(\mathbf{v}) \leq \nabla_t^\top (\boldsymbol{\theta}_t - \mathbf{v}).$$

The following property of Bregman divergences follows from the definition, for any vectors $\mathbf{v}, \mathbf{w}, \mathbf{z}$:

$$(\mathbf{v} - \mathbf{w})^\top (\nabla R(\mathbf{z}) - \nabla R(\mathbf{w})) = B_R(\mathbf{v} \| \mathbf{w}) - B_R(\mathbf{v} \| \mathbf{z}) + B_R(\mathbf{w} \| \mathbf{z}).$$

Combining both observations,

$$\begin{aligned} f_t(\boldsymbol{\theta}_t) - f_t(\mathbf{v}) \leq \nabla_t^\top (\boldsymbol{\theta}_t - \mathbf{v}) &= \frac{1}{\eta} (\mathbf{v} - \boldsymbol{\theta}_t)^\top (\nabla R(\boldsymbol{\theta}'_{t+1}) - \nabla R(\boldsymbol{\theta}_t)) \\ &= \frac{1}{\eta} \Big(B_R(\mathbf{v} \| \boldsymbol{\theta}_t) - B_R(\mathbf{v} \| \boldsymbol{\theta}'_{t+1}) + B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}'_{t+1}) \Big) \\ &\leq \frac{1}{\eta} \Big(B_R(\mathbf{v} \| \boldsymbol{\theta}_t) - B_R(\mathbf{v} \| \boldsymbol{\theta}_{t+1}) + B_R(\boldsymbol{\theta}_t \| \boldsymbol{\theta}'_{t+1}) \Big), \end{aligned}$$

where the last inequality follows from the generalized Pythagorean theorem.

Proof

Then, we can conclude that (telescoping sum)

$$\begin{aligned}\text{regret} &\leq \frac{1}{\eta}\Big(B_R(\boldsymbol{\theta}^\star\|\boldsymbol{\theta}_1) - B_R(\boldsymbol{\theta}^\star\|\boldsymbol{\theta}_T)\Big) + \frac{1}{\eta}\sum_{t=1}^{T} B_R(\boldsymbol{\theta}_t\|\boldsymbol{\theta}'_{t+1}) \\ &\leq \frac{R(\boldsymbol{\theta}^\star) - R(\boldsymbol{\theta}_1)}{\eta} + \frac{1}{\eta}\sum_{t=1}^{T} B_R(\boldsymbol{\theta}_t\|\boldsymbol{\theta}'_{t+1})\end{aligned}$$

where the inequality above is due to convexity of R and the choice of $\boldsymbol{\theta}_1$. By the definition of Bregman divergence, and the generalized Cauchy-Schwartz:

$$\begin{aligned}B_R(\boldsymbol{\theta}_t\|\boldsymbol{\theta}'_{t+1}) + B_R(\boldsymbol{\theta}'_{t+1}\|\boldsymbol{\theta}_t) &= (\boldsymbol{\theta}_t - \boldsymbol{\theta}'_{t+1})^\top(\nabla R(\boldsymbol{\theta}_t) - \nabla R(\boldsymbol{\theta}'_{t+1})) \\ &= \eta\nabla_t^\top(\boldsymbol{\theta}_t - \boldsymbol{\theta}'_{t+1}) \leq \eta\,\|\nabla_t\|_t^\star\,\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}'_{t+1}\right\|_t \\ &\leq \frac{\eta^2\|\nabla_t\|_t^{\star 2}}{2} + \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}'_{t+1}\|_t^2}{2}\end{aligned}$$

So $B_R(\boldsymbol{\theta}_t\|\boldsymbol{\theta}'_{t+1}) \leq \frac{\eta^2\|\nabla_t\|_t^{\star 2}}{2}$, and the proof follows immediately.