Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Supplementary Material: Stochastic compositional gradient methods & applications

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2024)

License Information for Mathematics of Data Slides

- This work is released under a Creative Commons License with the following terms:
- **Attribution**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.
- **Non-Commercial**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor's permission.
- **Share Alike**
  - The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor's work.
- Full Text of the License

*Outline

- Reminder - Stochastic Optimization
- A more intricate matter - Stochastic Compositional problems
- Application: Model Agnostic Meta Learning
- Algorithms for stochastic compositional problems

* PhD material

Recall Lecture 3 - Statistical learning with streaming data

○ Recall that statistical learning seeks to find a $h^\star \in \mathcal{H}$ that minimizes the *expected* risk,

$$h^\star \in \arg\min_{h \in \mathcal{H}} \left\{ R(h) := \mathbb{E}_{(\mathbf{a},b)} \left[L(h(\mathbf{a}), b) \right] \right\}.$$

Abstract gradient method

$$h^{k+1} = h^k - \alpha_k \nabla R(h^k) = h^k - \alpha_k \mathbb{E}_{(\mathbf{a},b)}[\nabla L(h^k(\mathbf{a}), b)].$$

Not implementable in practice as the distribution of $(\mathbf{a}, b)$ is unknown $\Longrightarrow$ cannot compute $\mathbb{E}_{(\mathbf{a},b)}$.

Recall Lecture 3 - Statistical learning with streaming data

- Recall that statistical learning seeks to find a $h^\star \in \mathcal{H}$ that minimizes the *expected* risk,

$$h^\star \in \arg\min_{h\in\mathcal{H}} \left\{ R(h) := \mathbb{E}_{(\mathbf{a},b)} \left[L(h(\mathbf{a}), b) \right] \right\}.$$

Abstract gradient method

$$h^{k+1} = h^k - \alpha_k \nabla R(h^k) = h^k - \alpha_k \mathbb{E}_{(\mathbf{a},b)}[\nabla L(h^k(\mathbf{a}), b)].$$

Not implementable in practice as the distribution of $(\mathbf{a}, b)$ is unknown $\Longrightarrow$ cannot compute $\mathbb{E}_{(\mathbf{a},b)}$.

- In practice, data can arrive in a *streaming* way.

A parametric example: Markowitz portfolio optimization

$$\mathbf{x}^\star := \min_{\mathbf{x}\in\mathcal{X}} \left\{ \mathbb{E}\left[|b - \langle \mathbf{x}, \mathbf{a} \rangle|^2 \right] \right\}$$

- $h_{\mathbf{x}}(\cdot) = \langle \mathbf{x}, \cdot \rangle$
- $b \in \mathbb{R}$ is the desired return & $\mathbf{a} \in \mathbb{R}^p$ are the stock returns
- $\mathcal{X}$ is intersection of the standard simplex and the constraint: $\langle \mathbf{x}, \mathbb{E}[\mathbf{a}] \rangle \geq \rho$.

Recall Lecture 3 - Stochastic programming

Problem (**Mathematical formulation**)

Consider the following convex minimization problem:

$$f^\star = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) := \mathbb{E}[f(\mathbf{x},\theta)] \right\}$$

- θ *is a random vector whose probability distribution is supported on set* Θ.
- $f(\mathbf{x}) := \mathbb{E}[f(\mathbf{x},\theta)]$ *is nonconvex, differentiable, bounded from below and* L*-smooth.*
- *The solution set* $\mathcal{S}^\star := \{\mathbf{x}^\star \in \mathrm{dom}\,(f) : f(\mathbf{x}^\star) = f^\star\}$ *is nonempty.*

Recall Lecture 3 - Stochastic programming

Problem (Mathematical formulation)

Consider the following convex minimization problem:

$$f^\star = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ f(\mathbf{x}) := \mathbb{E}[f(\mathbf{x},\theta)] \right\}$$

- *θ is a random vector whose probability distribution is supported on set Θ.*
- *$f(\mathbf{x}) := \mathbb{E}[f(\mathbf{x},\theta)]$ is nonconvex, differentiable, bounded from below and L-smooth.*
- *The solution set $\mathcal{S}^\star := \{\mathbf{x}^\star \in \operatorname{dom}(f) : f(\mathbf{x}^\star) = f^\star\}$ is nonempty.*

Stochastic gradient descent (SGD)

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ and $(\alpha_k)_{k\in\mathbb{N}} \in (0,+\infty)^{\mathbb{N}}$.
2. For $k = 0, 1, \ldots$ perform:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k G(\mathbf{x}^k, \theta_k).$$

Remarks:

- ∘ $G(\mathbf{x}^k, \theta_k)$ is an unbiased estimate of $\nabla f(\mathbf{x}^k)$, i.e. $\mathbb{E}[G(\mathbf{x}^k, \theta_k)] = \nabla f(\mathbf{x}^k)$.
- ∘ Cost of computing $G(\mathbf{x}^k, \theta_k) <$ cost of $\nabla f(\mathbf{x}^k)$.
- ∘ Sample complexity for $\mathbb{E}\left[\|\nabla F(\mathbf{x})\|^2\right] \leq \epsilon$ is $\mathcal{O}(\epsilon^{-2})$ [3].

A more intricate template – Stochastic Compositional problems

A nested problem (**Mathematical formulation**)

$$f^\star = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ F(\mathbf{x}) := \mathbb{E}[f(\mathbb{E}[g(\mathbf{x},\theta_g)],\theta_f)] \right\}$$

- θ_f, θ_g are independent random vectors whose probability distribution is supported on set Θ.
- The solution set $\mathcal{S}^\star := \{\mathbf{x}^\star \in \mathrm{dom}\,(f) : f(\mathbf{x}^\star) = f^\star\}$ is nonempty.

Questions:

- Why is this template relevant?
- How can we solve it?
- What convergence guarantees can we get?

Motivation (running example 1) – Meta Learning

Meta Learning (Learning to learn fast)

Make use of previous tasks to learn new tasks **quickly**. Simplistically, a meta learning task can be defined:

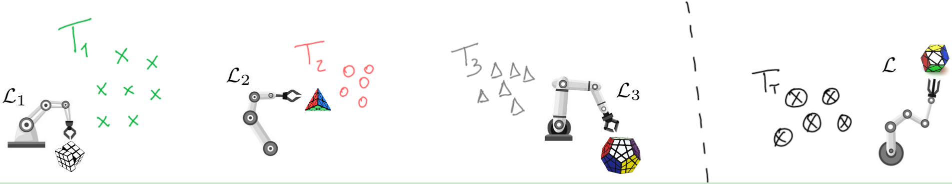
$$\theta^* = \arg\min_{\theta} \mathbb{E}_{D\sim p(D)}[L_\theta(D)]$$

where, $p(D)$ is a distribution of datasets, L is a defined loss function and D is a dataset. A task $\mathcal{T}$ is defined as the combination of dataset and specific loss function.

○ *Why it is important (examples):* Healthcare situations where data is sparse. Deployment of robotic workers which need fast adaptation.

Training Tasks

Objective



Probabilistic view of meta learning

Extract prior knowledge from a set of tasks that allows efficient learning of new tasks.

Applications of stochastic compositional optimization problems

Optimization based meta learning

Goal: Use optimization to extract meta knowledge about the task we are trying to solve.

Learn a meta-initialization model parameter

Consider a model represented by f_θ. The function's optimal parameters, θ'_i, change depending on the task, $\mathcal{T}_i$, its trying to solve.
Concretely, having a distribution of tasks $p(\mathcal{T})$, we optimize the models parameters as:

$$\min_{\theta} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}(f_{\theta'_i}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_\theta L_{\mathcal{T}_i}(f_\theta)})$$

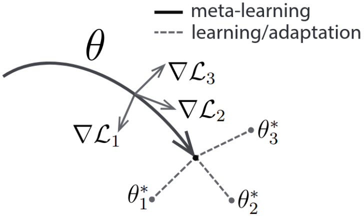
where α, β are hyperparameters.

Model Agnostic Meta Learning (MAML) [2]

1. Randomly initialize θ and sample batch $\mathcal{T}_i \sim p(\mathcal{T})$:
2. For all $\mathcal{T}_i$ perform:

$$\theta'_i = \underbrace{\theta - \alpha \nabla_\theta L_{\mathcal{T}_i(f(\theta))}}_{\text{gradient step}}$$

3. Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} L_{\mathcal{T}_i}(f_{\theta'_i})$



Applications of stochastic compositional optimization problems

Reducing MAML to a stochastic compositional optimization problem

Remembering our formulation:

$$f^\star = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ F(\mathbf{x}) := \mathbb{E}[f(\mathbb{E}[g(\mathbf{x},\theta_g)],\theta_f)] \right\}$$

Looking at MAML in the following way:

$$\min_\theta L(f_\theta) := \frac{1}{M}\sum_{i=1}^{M} L_i(f_{\theta_i'}) \text{ with } \theta_i' = \theta - \alpha\nabla_\theta L_{\mathcal{T}_i(f(\theta))}$$

Specifically:

$$\begin{aligned} \mathbb{E}[f(\ldots,\theta_f)] &\rightarrow L(f_\theta) \\ \mathbb{E}[g(\mathbf{x},\theta_g)] &\rightarrow \theta_i' = \theta - \alpha\nabla_\theta L_{\mathcal{T}_i(f(\theta))} \end{aligned}$$

This is a multi level problem that stochastic compositional methods can address.

Motivation (running example 2) – Reinforcement Learning

Goal of Reinforcement Learning

In on-policy reinforcement learning, we aim to learn the value function $V^\pi(s)$, representing the expected return starting from state s and following policy π:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s, a_t \sim \pi\right]$$

where $\gamma \in [0,1]$ is the discount factor, and r_t is the reward at time t.

Bellman Equation Formulation

The value function V^π satisfies the Bellman equation:

$$V^\pi = r^\pi + \gamma P^\pi V^\pi$$

where P^π is the transition matrix and r^π is the expected reward. Solving the Bellman equation requires estimating P^π and r^π, both of which are often unknown.

Applications of stochastic compositional optimization problems

Reinforcement Learning (Learning State Values)

On-policy reinforcement learning seeks to estimate the value-per-state in a stochastic system by interacting with it and observing outcomes.

- We aim to solve a Bellman equation for the value function V^π:

$$\gamma P^\pi V^\pi + r^\pi = V^\pi,$$

  where P^π is the transition probability matrix and r^π is the reward vector, both unknown.
- On-policy learning aims to solve Bellman equation via blackbox simulation.
- Reformulate it as a stochastic compositional optimization problem:

$$\min_x \mathbb{E}[f_v(\mathbb{E}[g_w(x)])] \leftrightarrow \min_{x \in \mathbb{R}^S} \|\mathbb{E}[A]x - \mathbb{E}[b]\|^2,$$

  where $\mathbb{E}[A] = I - \gamma P^\pi$ and $\mathbb{E}[b] = r^\pi$.

Probabilistic View of Reinforcement Learning

Treating RL as a stochastic optimization problem enables using sampling-based methods for value estimation.

Applications of stochastic compositional optimization problems

Optimization in Reinforcement Learning

Goal: Apply stochastic optimization to learn the value function efficiently in large or complex environments.

Compositional Formulation of RL

To estimate the value function V^{π}, we can minimize a compositional objective:

$$\min_x \mathbb{E}_{f_v}\left[f_v\left(\mathbb{E}_{g_w}\left[g_w(x)\right]\right)\right] \leftrightarrow \min_x \|\mathbb{E}[A]x - \mathbb{E}[b]\|^2,$$

where:

$$\begin{aligned} f_v(x) &= \|Ax - b\|^2, \\ g_w(x) &= \text{sampled estimates based on } \gamma P^{\pi} \text{ and } r^{\pi}. \end{aligned}$$

This objective involves nested expectations, suitable for stochastic compositional optimization.

- **Outer expectation** $\mathbb{E}_{f_v}$: Represents the dependency on policy-driven transition dynamics.
- **Inner expectation** $\mathbb{E}_{g_w}$: Represents the stochastic transitions and reward estimates.

Solving stochastic compositional optimization problems - First attempt

A first, natural idea: apply the machinery of SGD

1. Use the following unbiased gradient estimator:

$$G(\mathbf{x}^k, \theta_k) = \nabla g\left(\mathbf{x}^k, \theta_g^k\right) \nabla f\left(\mathbb{E}_{\theta_g}\left[g(\mathbf{x}^k, \theta_g)\right], \theta_f^k\right),$$

and replace update in step 2. of SGD (slide 5) become

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \nabla g\left(\mathbf{x}^k, \theta_g^k\right) \nabla f\left(\mathbb{E}_{\theta_g}\left[g(\mathbf{x}^k, \theta_g)\right], \theta_f^k\right).$$

Problem: we don't know the distribution of θ_g and even if we do, the inner expectation is costly.

2. What about removing the inner $\mathbb{E}$ by using a stochastic sample

$$G(\mathbf{x}^k, \theta_k) = \nabla g\left(\mathbf{x}^k, \theta_g^k\right) \nabla f\left(g(\mathbf{x}^k, \theta_g^k), \theta_f\right)?$$

Problem: gradient estimator is biased $\Longrightarrow$ theoretical machinery of SGD doesn't give guarantees.

Question:

- How can $\mathbb{E}_{\theta_g}\left[g(\mathbf{x}^k, \theta_g)\right]$ be estimated efficiently?

Solving stochastic compositional optimization problems - Second attempt

Idea of [4]: add new variable $\mathbf{y}^k$ to 'approximate' $\mathbb{E}_{\theta_g}\left[g(\mathbf{x}^k, \theta_g)\right]$

Replace step 2. of SGD with:

$$\mathbf{y}^{k+1} = (1-\beta_k)\mathbf{y}^k - \beta_k g\left(\mathbf{x}^k, \theta_g^k\right) \longleftarrow \text{track expectation via exponential averaging} \tag{1}$$

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_k \nabla g\left(\mathbf{x}^k, \theta_g^k\right) \nabla f\left(\mathbf{y}^{k+1}, \theta_f^k\right), \tag{2}$$

where $\{\alpha_k\}$, $\{\beta_k\}$ are stepsize sequences such that

$$\alpha_k = \frac{1}{k^{3/4}} \text{ and } \beta_k = \frac{1}{k^{1/2}}. \tag{3}$$

Remarks:

- ○ α_k and β_k decrease at different rates, with $\mathbf{x}^k$ being relatively static w.r.t $\mathbf{y}^k$.
- ○ Sample complexity when $f \circ g$ convex/nonconvex is $\mathcal{O}(\epsilon^{-4})$ (Theorems 6, 8).
- ○ Rates can be accelerated (details omitted) but are still worse than $\mathcal{O}(\epsilon^{-2})$ achieved by SGD.

Is there a better way of estimating $\mathbb{E}_{\theta_g}\left[g(\mathbf{x}^k, \theta_g)\right]$?

Correct the estimator $\mathbf{y}^k$

- Problem: Slower timescale updates for $\mathbf{x}^{k+1}$ relative to $\mathbf{y}^{k+1}$ $\Longrightarrow$ suboptimal convergence rates.

- Observation: $\mathbf{y}^{k+1}$ uses outdated information from $\mathbf{y}^k$, which demands $\alpha_k < \beta_k$.

- Idea [1]: 'Correct' $\mathbf{y}^{k+1}$ updates to alleviate the problem:

$$\mathbf{y}^{k+1} = (1-\beta_k)\left(\mathbf{y}^k + \nabla g(\mathbf{x}^k, \theta_k^g)(\mathbf{x}^k - \mathbf{x}^{k-1})\right) - \beta_k g\left(\mathbf{x}^k, \theta_g^k\right). \tag{4}$$

$\Longrightarrow$ the contribution of $\mathbf{y}^k$ to $\mathbf{y}^{k+1}$ is corrected.

Intuition – why would (4) work?

- $\mathbf{y}^{k+1}$ should be as close to $g(\mathbf{x}^k)$ as possible. By first order Taylor approximation:

$$g(\mathbf{x}^k) \approx \underbrace{g(\mathbf{x}^{k-1})}_{\approx \mathbf{y}^k} + \nabla g(\mathbf{x}^{k-1})\left(\mathbf{x}^k - \mathbf{x}^{k-1}\right),$$

so the term in blue nudges $\mathbf{y}^k$ more in the right direction.

- A more mathematically-involved explanation based on the gradient flow view is given in Section 2.2 of [1].

The algorithm and its assumptions

Stochastically Corrected Stochastic Compositional Gradient (SCSC)

```
1. Choose $\mathbf{x}^0, \mathbf{y}^0 \in \mathbb{R}^p$ and stepsizes $\alpha_0,\ \beta_0$.
2. For $k = 0, 1, \ldots$ perform:
    2.a Randomly select $\theta_g^k$ and compute $g(\mathbf{x}_k, \theta_g^k)$ and $\nabla g(\mathbf{x}^k, \theta_g^k)$
    2.b Update $\mathbf{y}^{k+1}$ via (4)
    2.c Randomly select $\theta_f^k$ and compute $\nabla f(\mathbf{y}^{k+1}, \theta_f^k)$
    2.d Update variable $\mathbf{x}^{k+1}$ via (2)
```

Assumption

A1. *Functions f and g are L_f and L_g-smooth, respectively.*

A2. *Norms of stochastic gradients $\| \nabla f(\mathbf{x}, \theta_f) \|^2$ and $\| \nabla g(\mathbf{x}, \theta_g) \|^2$ are bounded in expectation*

A3. *The sampling oracle satisfies i) $\mathbb{E}\left[g(\mathbf{x}, \theta_g)\right] = g(x)$ and ii) $\mathbb{E}\left[\nabla g(\mathbf{x}, \theta_g) \nabla f(\mathbf{y}, \theta_f)\right] = \nabla g(\mathbf{x}) \nabla f(\mathbf{y})$.*

A4. *Function g has bounded variance: $\exists V_g > 0$ s.t. $\mathbb{E}\left[\| g(\mathbf{x}, \theta_g) - g(\mathbf{x}) \|^2\right] \leq V_g^2$.*

Convergence guarantees of SCSC Cont'd

Convergence Theorem (simplified statement of Theorem 1 in [1])

Assuming **A1.–A4.** and choosing $\alpha_k, \beta_k \in \mathcal{O}(1/\sqrt{k})$, the iterates of SCSC satisfy:

$$\frac{1}{K}\sum_{i=0}^{K}\mathbb{E}\left[\| \nabla F(\mathbf{x}^k) \|^2\right] \in \mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$$

Remarks:

◦ α_k and β_k are of the same order $\implies$ we can get optimal sample complexity.

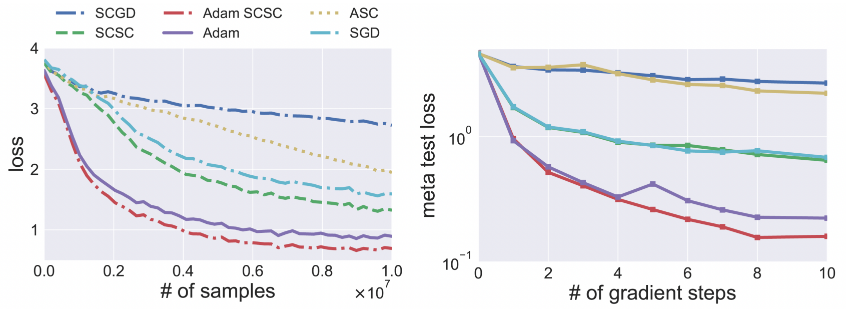
◦ SCSC can be extended to many compositions with the same convergence guarantees.

$$f^\star := \min_{\mathbf{x}\in\mathbb{R}^p}\left\{F(\mathbf{x}) := f_N(f_{N-1}(\dots f_1(\mathbf{x})\dots))\right\}$$

$$\text{with } f_n(\cdot) := \mathbb{E}\left[f_n(\cdot,\theta^n)\right], \quad \text{for } n = 1\dots N$$

◦ SCSC also works with Adam-type updates and preserves the convergence guarantees of the non-compositional counterpart.

MAML experiments [1]



- **Tasks:** regress from input to output of sine wave $s(\theta; a, \phi) = a \sin(\theta + \phi)$, where a, ϕ vary across tasks.
- Use NN with 2 hidden layers & RelU activations, weights $\mathbf{x}$ and output $\hat{s}(\theta; \mathbf{x})$.
- Define the task loss to be $F_m(\mathbf{x}) = \mathbb{E}_\theta \left[\| \hat{s}(\theta; \mathbf{x}) - s(\theta; a_m, \phi_m) \|^2 \right]$.
- Formulating it as SCSC: let $g(\mathbf{x}) = \left[g_1^T(\mathbf{x}), \ldots g_M^T(\mathbf{x}) \right]^T$ with $g_m(\mathbf{x}) = \mathbf{x} - \nabla F_m(\mathbf{x})$ tracked by $\mathbf{y}_m$ and define f as $f(\mathbf{x}) := \frac{1}{M} \sum_{m=1}^{M} F_m(\mathbf{y}_m)$.

References I

[1] Tianyi Chen, Yuejiao Sun, and Wotao Yin.
Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization.
arXiv preprint arXiv:2008.10847, 2021.
(Cited on pages 17, 19, and 20.)

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine.
Model-agnostic meta-learning for fast adaptation of deep networks.
In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.
(Cited on page 10.)

[3] Saeed Ghadimi and Guanghui Lan.
Stochastic first-and zeroth-order methods for nonconvex stochastic programming.
SIAM Journal on Optimization, 23(4):2341–2368, 2013.
(Cited on pages 6 and 7.)

[4] Mengdi Wang, Ethan X Fang, and Han Liu.
Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions.
Mathematical Programming, 161(1-2):419–449, 2017.
(Cited on page 16.)