Deep Learning

Lecture 5: Non-convex Optimization for Learning (Part 2)

University of Agder,
Kristiansand, Norway

Prepared by: Hadi Ghauch*, Hossein S. Ghadikolaei†

* Telecom ParisTech

† Royal Institute of Technology, KTH

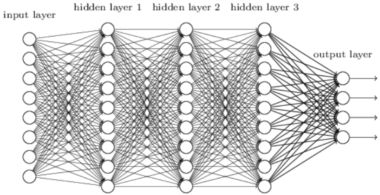
https://sites.google.com/view/fundl/home

Outline

Outline

Motivation

Resurgence of AI is due to **Deep Neural Networks (DNNs)**
and countless variants (covered later)



DNNs is a composition of **non-linear layers**, $\boldsymbol{W}_1, ..., \boldsymbol{W}_J$:

$$\boldsymbol{y}_i = \sigma_J(\boldsymbol{W}_J \cdots \sigma_2(\boldsymbol{W}_2 \sigma_1(\boldsymbol{W}_1 \boldsymbol{x}_i))),\ i \in [N]$$

Let $\boldsymbol{w} \in \mathbb{R}^d$ be the total number of weights in DNN (from all layers)
$d \geq 10^6$ in current deep learning application

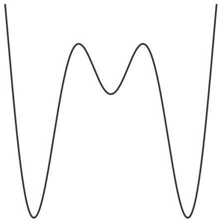
The resulting optimization DNN training

$$\min_{\boldsymbol{w} \in \mathcal{D}} \frac{1}{N} \sum_{i=1}^{N} f_i((\boldsymbol{x}_i, \boldsymbol{y}_i); \boldsymbol{w}) + \lambda \|\boldsymbol{w}\|_2^2 := f(\boldsymbol{w})$$

non-convex optimization problem

Recap: Non-convex Optimization

$$\min_{\boldsymbol{w}\in\mathcal{D}} f(\boldsymbol{w}) \qquad (1)$$



$f(\boldsymbol{w}) : \mathcal{D} \subseteq \mathbb{R}^d \to \mathbb{R}$ is non-convex
$\mathcal{D}$: domain of f is convex
f is Lipschitz continuous with constant
L: $\|f(\boldsymbol{w}_2) - f(\boldsymbol{w}_1)\|_2 \leq L\|\boldsymbol{w}_2 - \boldsymbol{w}_1\|_2$
f is differentiable
less restrictive assumptions than convex

Local optimality may not necessarily imply global optimality

Proper initialization is very important in nonconvex optimization

First-order criteria ($\nabla f(\boldsymbol{w}) = 0$)
 necessary and sufficient conditions for convex
 only necessary condition for nonconvex

Roadmap for Non-convex Optimization

No generic method for all non-convex problems ($\neq$ convex case)
solution approach depends on **structure of the optimization**

1. **No structure on $f(w)$ nor constraint set (Recap from Lec 4):**
   first-order (GD and SCG) methods may be used.
   necessary conditions for GD/SGD to convergence to local min
   successive approximation: successively approx (1) with linear/convex bound

2. $f(\boldsymbol{w})$ **is coordinate separable:** use **Coordinate Descent (CD)**
   if $f(\boldsymbol{w}) = f(w_1, ..., w_d)$, where $(w_1, ..., w_d)$ are **coordinates**
   f strongly convex in each coordinate, f non-convex jointly in all coordinates

3. $f(\boldsymbol{w})$ **is block-coordinate separable:** use **Block-Coordinate Descent(BCD)**
   if $f(\boldsymbol{w}) = f(\boldsymbol{w}_1, ..., \boldsymbol{w}_d)$, where $(\boldsymbol{w}_1, ..., \boldsymbol{w}_d)$ **block of coordinates**
   f strongly convex in each block of coordinates,
   f non-convex jointly in all coordinates

4. $f(\boldsymbol{w})$ **is block-coordinate separable**
   if $f(\boldsymbol{w}) = f(\boldsymbol{w}_1, ..., \boldsymbol{w}_d)$, where $(\boldsymbol{w}_1, ..., \boldsymbol{w}_d)$ **block of coordinates**
   f not necessarily convex in each block of coordinates,
   f non-convex jointly in all coordinates
   use **Block-Successive Upperbound minimization (BSUM)**

Outline

Coordinate Descent (CD) Methods

Assumes optimization problem is **coordinate-separable**:

$$\min_{\boldsymbol{w}\in\mathcal{D}} f(\boldsymbol{w}) = f(w_1, \cdots, w_d),$$

w_i is the ith coordinate
domain **convex and separable** $\mathcal{D} = \prod_{i=1}^{d} \ D_i$

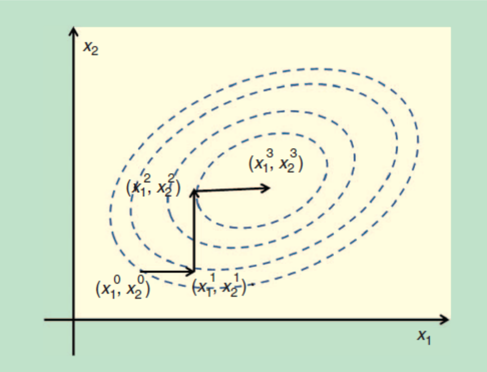
Idea: Minimize coordinate w_i, while fixing all other ones
at iteration k, optimization for w_i

$$w_i^{k+1} := \arg\min_{w_i\in\mathcal{D}_i} f(w_1^k, \cdots, w_{i-1}^k, w_i, w_{i+1}^k \cdots, w_d^k) \tag{2}$$

assume that each **subproblem**, (2), **strongly convex problem**
iteratively minimize d subproblems instead of joint problem

Update rule ? **cyclic update:** $i = (k \mod d), i \in \mathbb{N}_+$
many other update rules: **random**, **greedy**

Coordinate Descent Methods: Example



- descent direction along each coordinate (unlike GD)
- f convex in x_1 and x_2 separately (not jointly)

Outline

Block Coordinate Descent Methods

Block Coordinate Descent (BCD) generalize CD:

- from minimizing coordinates to blocks of coordinates

$$\arg\min_{\boldsymbol{w}_1,\cdots,\boldsymbol{w}_d} f(\boldsymbol{w}) = f(\boldsymbol{w}_1,\cdots,\boldsymbol{w}_d)$$

$\boldsymbol{w}_i$ is the ith block of coordinates,
domain convex and separable: $\mathcal{D} = \prod_{i=1}^{d} D_i$

Low-rank MF: Factorize large matrix $\boldsymbol{X} \in \mathbb{R}^{M\times N}$ with low-rank matrices: $\boldsymbol{P} \in \mathbb{R}^{M\times d}$, $\boldsymbol{Q} \in \mathbb{R}^{N\times d}$, $d \ll (M, N)$ such that $\boldsymbol{X} \approx \boldsymbol{P}\boldsymbol{Q}^T$

$$\min_{\mathbf{P},\mathbf{Q}} \|\mathbf{X} - \mathbf{P}\mathbf{Q}^T\|_F^2 \quad \mathrm{s.t.} \quad \|\boldsymbol{P}\|_F^2 \le \rho_1, \|\boldsymbol{Q}\|_F^2 \le \rho_2$$

Not jointly convex in $\boldsymbol{P}, \boldsymbol{Q}$. But strongly convex in $\boldsymbol{P}, \boldsymbol{Q}$ **separately**.
constraints convex and separable

Similar optim problems for **training auto-encoders**, and **deep linear networks**

Block Coordinate Descent Methods

Idea: Minimize block of coordinates, $\boldsymbol{w}_i$, while fixing all other ones
optimization for block $\boldsymbol{w}_i$, at iteration k

$$\begin{aligned} \boldsymbol{w}_i^{k+1} &:= \arg\min_{\boldsymbol{w}_i \in \mathcal{D}_i} f(\boldsymbol{w}_1^k, \cdots, \boldsymbol{w}_{i-1}^k, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}^k, \cdots, \boldsymbol{w}_d^k) \\ &= \arg\min_{\boldsymbol{w}_i \in \mathcal{D}_i} f(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k) \end{aligned} \tag{3}$$

assume that (3) **strongly convex problem**
iteratively minimize d subproblems instead of joint problem

- $\boldsymbol{w}_{-i}^k$ is the block of fixed variables at iteration k:
  $\boldsymbol{w}_{-i}^k := (\boldsymbol{w}_1^k, \cdots, \boldsymbol{w}_{i-1}^k, \boldsymbol{w}_{i+1}^k, \cdots, \boldsymbol{w}_d^k)$
- $f(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$ is the function when “looking” at block $\boldsymbol{w}_i$ only
- same type of update rules as CD

Block Coordinate Descent Methods

Convergence of BCD and CD

A1) **solution for each subproblem(block) is unique**
$\Leftrightarrow f$ **strongly convex** in each of its blocks.

A2) f smooth with Lipschitz constant L

A3) The domain, $\mathcal{D}_i$, is closed and convex

Theorem 3: Suppose that A1)-A3) hold. The sequence of updates generated by the CD method, in (2), and BCD method, in (3), monotonically converges to a **stationary point** of f, as $k \to \infty$

A1) is too strong. May not be true in many problems

Outline

Block Successive Upperbound Minimization

What happens when f is not strongly convex in each block ?

- **Block Successive Upperbound Minimization (BSUM)** extends convergence of BCD

Include many known ML algo as special cases:

- Expectation Minimization
- Convex Concave Procedure
- Non-negative Matrix Factorization
- Majorisation Minimization
- Forward Backward Splitting algorithm

Block Successive Upperbound Minimization

Idea: f **not convex** in each block.
intuition: do SLA/SCA on each block of coordinates
Minimize a **strongly convex upperbound** for each block

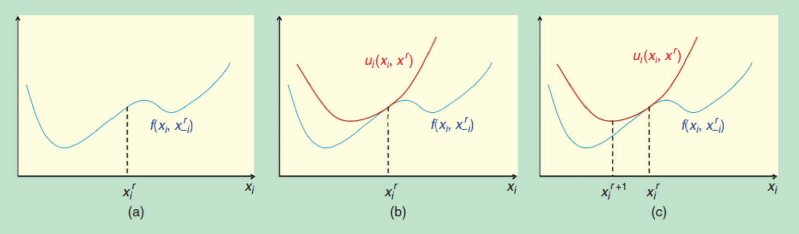
Minimize block of coordinates, $\boldsymbol{w}_i$, while fixing all other ones
optimization for block $\boldsymbol{w}_i$, at iteration k

$$(\mathcal{P}_i) \quad \boldsymbol{w}_i^{k+1} := \arg\min_{\boldsymbol{w}_i \in \mathcal{D}_i} u_i(\boldsymbol{w}_1^k, \cdots, \boldsymbol{w}_{i-1}^k, \boldsymbol{w}_i, \boldsymbol{w}_{i+1}^k, \cdots, \boldsymbol{w}_d^k)$$
$$= \arg\min_{\boldsymbol{w}_i \in \mathcal{D}_i} u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$$

$u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$ is upperbound for block $\boldsymbol{w}_i$
minimize upperbound for each subproblem (not f)

$\boldsymbol{w}_{-i}^k$ is the block of fixed variables at iteration k:
$\boldsymbol{w}_{-i}^k := (\boldsymbol{w}_1^k, \cdots, \boldsymbol{w}_{i-1}^k, \boldsymbol{w}_{i+1}^k, \cdots, \boldsymbol{w}_d^k)$
$u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$ **strongly convex** in $\boldsymbol{w}_i$

BSUM Example [Hong, 2016]



a) f non-convex in coordinate x_i
b) construct upperbound fnc, $u_i()$, at point x_i^r
c) find next iterate, x_i^{r+1}, by minimizing the upperbound $u_i()$ (not the function)

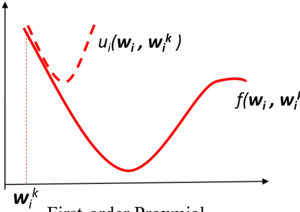
Hong etal "A Unified Algorithmic Framework for Block- Structured Optimization Involving Big Data"

BSUM: Choosing the Upperbound

Several standard choices for upperbound, $u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$

first order proximal upperbound:

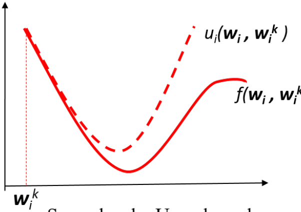
$$u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k) := f(\boldsymbol{w}_i^k) + \nabla f(\boldsymbol{w}_i^k)^T(\boldsymbol{w}_i - \boldsymbol{w}_i^k) + \gamma_i/2\ \|\boldsymbol{w}_i - \boldsymbol{w}_i^k\|_2^2$$



First-order Proxmial Upperbound

second-order upperbound:

$$u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k) := f(\boldsymbol{w}_i^k) + \nabla f(\boldsymbol{w}_i^k)^T(\boldsymbol{w}_i - \boldsymbol{w}_i^k) + (1/2)(\boldsymbol{w}_i - \boldsymbol{w}_i^k)^T \nabla^2 f(\boldsymbol{w}_i^k)(\boldsymbol{w}_i - \boldsymbol{w}_i^k)$$



Second-order Upperbound

note similarity with SLA/SCA bounds

BSUM: Convergence

When and where does BSUM converge ?

Recall def of directional derivative and regular point.

A1) $u_i(-)$ is strongly convex, smooth, same directional derivatives as f

A2) f smooth with Lipschitz constant L

A3) The domain, $\mathcal{D}_i$, is closed and convex

A4) The sequence of points, $\{\boldsymbol{w}_1^k, ..., \boldsymbol{w}_d^k\}_k$, is regular

Theorem 4: Suppose that A1)-A4) hold. Assume that the solution of each subproblem, $\mathcal{P}_i$, is unique. Then, every limit point of $\{\boldsymbol{w}_1^k, ..., \boldsymbol{w}_d^k\}_k$ is a **stationary point** of f.

Convergence of BSUM is the most general among CD/BCD

Variants of BSUM

Parallel Successive Convex Approximation [M. Razaviyayn, NIPS 2014]:

recent **variant of BSUM** to solve problems,

$$\arg\min_{\boldsymbol{w}_1,\cdots,\boldsymbol{w}_d} f(\boldsymbol{w}_1,\cdots,\boldsymbol{w}_d) + \sum_i g_i(\boldsymbol{w}_i)$$

f is smooth and block-separable

g non-smooth and strongly convex

several applications in **distributed learning**:
sparse dictionary learning, LASSO, K-SVD

current works apply this theoretical framework to show convergence several DNN training algorithm: gradient methods, backprop, Newton, AdaGrad,

Take-home Messages: BCD and BSUM

CB, BCD and BSUM ideal for optimization problems where

- f is separable by (blocks of) coordinates
- f not jointly convex in all blocks, but strongly convex in each block
- constraints are convex and separable
- strong convexity of each block **not a necessary cond** for convergence of BSUM

Achilles' heel: When does BCD and BSUM fail ?

- When the constraints are **not separable** (coupled constraints)
- Convergence to a stationary point **cannot be shown** for example:

$$\min_{x_1,x_2} \; x_1^2 + x_2^2 \;\; \text{s. t.} \;\; x_1 + x_2 = 1$$

Outline

Application: Low rank Matrix Factorization

Factorize large matrix $\boldsymbol{X} \in \mathbb{R}^{M\times N}$ with low-rank matrices:
$\boldsymbol{P} \in \mathbb{R}^{M\times d}$, $\boldsymbol{Q} \in \mathbb{R}^{N\times d}$, $d \ll (M, N)$ such that $\boldsymbol{X} \approx \boldsymbol{P}\boldsymbol{Q}^T$

$$\min_{\mathbf{P},\mathbf{Q}} \|\mathbf{X} - \mathbf{P}\mathbf{Q}^T\|_F^2 \quad \mathrm{s.t.} \quad \|\boldsymbol{P}\|_F^2 \leq \rho_1 \ , \ \|\boldsymbol{Q}\|_F^2 \leq \rho_2$$

not jointly convex in $\boldsymbol{P}, \boldsymbol{Q}$. But strongly convex in $\boldsymbol{P}, \boldsymbol{Q}$ separately.
constraints convex and separable

Solved using BCD.

Given $\mathbf{Q}_k$ optimize $\mathbf{P}_{k+1} := \min_{\|\boldsymbol{P}\|_F^2 \leq \rho_1} \|\mathbf{X} - \mathbf{P}\mathbf{Q}_k^T\|_F^2$

Given $\mathbf{P}_{k+1}$ update $\mathbf{Q}_{k+1} := \min_{\|\boldsymbol{Q}\|_F^2 \leq \rho_2} \|\mathbf{X} - \mathbf{P}_{k+1}\mathbf{Q}^T\|_F^2$

subproblems are strongly convex: necessary cond for convergence of BCD hold

a.k.a. Alternating Least Squares

Same BCD-based method used to train auto-encoder and deep linear nets.

More Applications

Training a deep linear network

Deep Linear Network is a composition of **linear layers**, $\boldsymbol{W}_1, ..., \boldsymbol{W}_J$ (no activation function): $\boldsymbol{y}_i = \boldsymbol{W}_J \cdots \boldsymbol{W}_2 \boldsymbol{W}_1 \boldsymbol{x}_i, \; i \in [N]$

Training may be done using BCD:

$$\frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{y}_i - \boldsymbol{W}_J \cdots \boldsymbol{W}_2 \boldsymbol{W}_1 \boldsymbol{x}_i\|_2^2 := f(\boldsymbol{W}_1, ..., \boldsymbol{W}_J)$$

notice similarity with MF optimization

Auto-encoder Training in DL : efficiently done using BCD

we will discuss these DL architectures in Lec 8.

Conclusions

Solution method depends on structure of problem

Non-convex optimization w/out structure:

- first-order methods: with tweaks to escape saddle pts
- Successive linear/convex approximation

Non-convex optimization with structure:

- If variable and constraint is separable
- if function convex in each block: CD, BCD
- if function non-convex in each block: BSUM
- BSUM is most generic framework
- applications of BSUM in ML

Useful references

- M. Hong, M. Razaviyayn, Z. Q. Luo and J. S. Pang, "A Unified Algorithmic Framework for Block-Structured Optimization Involving Big Data: With applications in machine learning and signal processing" in IEEE Signal Processing Magazine, vol. 33, no. 1, pp. 57-77, Jan. 2016.
- M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization", SIAM J. Optim., vol. 23, no. 2, pp. 1126-1153, 2013
- M. Razaviyayn, "Successive convex approximation: Analysis and applications", Ph.D. dissertation, Univ. of Minnesota, 2014;
- Stephen J. Wright, "Coordinate descent algorithm" , Math. Program., vol. 151, no. 1, pp 3-34, 2015
- M. Razaviyayn, M Hong, ZQ Luo, JS Pang, "Parallel successive convex approximation for nonsmooth nonconvex optimization", NIPS 2014

Outline

Special Cases

Many known algorithm are special cases of BSUM

Difference of Convex (DC) programming:

$\arg\min_{\boldsymbol{w}} f(\boldsymbol{w}) = g_1(\boldsymbol{w}) - g_2(\boldsymbol{w})$, where g_1 and g_2 convex

$$\boldsymbol{w}^{k+1} = \arg\min_{\boldsymbol{w}} \; g_1(\boldsymbol{w}) - (\nabla \; g_2(\boldsymbol{w}^k)^T(\boldsymbol{w} - \boldsymbol{w}^k)) - g_2(\boldsymbol{w}^k)$$

- *Convex Concave Procedure*

BCD: special case of BSUM

Select the upperbound in BSUM as the function itself:
$u_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k) = f(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^k)$
we recover BCD

More Applications

Sparse Dictionary Learning

Given a data matrix $\mathbf{D} \in \mathbb{R}^{N \times M}$, find a dictionary $\mathbf{X}\mathbf{Y}^T$, that sparsely represents the data matrix,

$$\min_{\mathbf{X},\mathbf{Y}} \|\mathbf{D} - \mathbf{X}\mathbf{Y}^T\|_F^2 + \lambda\|\mathbf{X}\|_1 \text{ s. t. } \|\mathbf{Y}\|_F \leq \beta$$

Deep Learning

Lecture 5: Non-convex Optimization for Learning (Part 2)

University of Agder,
Kristiansand, Norway

Prepared by: Hadi Ghauch[*], Hossein S. Ghadikolaei[†]

[*] Telecom ParisTech

[†] Royal Institute of Technology, KTH

https://sites.google.com/view/fundl/home