Alternating Maximization: An Unified View of EM and Blahut-Arimoto Algorithms

Jie Wang
Fourth Year Undergraduate Student
Pure Mathematics Major
Network Coding Lab

September 16, 2019

Table of Content

Table of Contents

Motivation

- Alternating Optimization is applicable to a special class of problems: Those here varibales can be split into two sets, i.e., $v = (x, y)$ such that

$$\max_x f(x,y) \qquad \text{is easy for fixed } y$$
$$\max_y f(x,y) \qquad \text{is easy for fixed } x$$

- Alternating Optimization Framework:

$$x_+ = \arg\max_x f(x,y)$$
$$y_+ = \arg\max_y f(x_+,y)$$

- Comment:
  - "big" global steps, and much faster than "local" gradient descent
  - No step size to be tuned / chosen

Motivation

- Alternating Optimization is applicable to a special class of problems: Those here varibales can be split into two sets, i.e., $v = (x, y)$ such that

$$\max_x f(x,y) \qquad \text{is easy for fixed } y$$
$$\max_y f(x,y) \qquad \text{is easy for fixed } x$$

- Alternating Optimization Framework:

$$x_+ = \arg\max_x f(x,y)$$
$$y_+ = \arg\max_y f(x_+,y)$$

- Comment:
  - "big" global steps, and much faster than "local" gradient descent
  - No step size to be tuned / chosen

Motivation

- Alternating Optimization is applicable to a special class of problems: Those here varibales can be split into two sets, i.e., $v = (x, y)$ such that

$$\max_x f(x,y) \qquad \text{is easy for fixed } y$$
$$\max_y f(x,y) \qquad \text{is easy for fixed } x$$

- Alternating Optimization Framework:

$$x_+ = \arg\max_x f(x,y)$$
$$y_+ = \arg\max_y f(x_+,y)$$

- Comment:
  - “big” global steps, and much faster than “local” gradient descent
  - No step size to be tuned / chosen

Problem Setting

- Consider optimization for groups of variables:

$$\max_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} f(x,y)$$

where $\mathcal{X},\mathcal{Y}$ are *convex*, and the objective function f is such that:

  - $f:\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ is bounded above;
  - f is continuous and has continuous partial derivatives on $\mathcal{X}\times\mathcal{Y}$;
  - For any fixed $y\in\mathcal{Y}$, there exists an unique $c_1(y)\in\mathcal{X}$ such that

$$f(c_1(y),y)=\max_{x\in\mathcal{X}} f(x,y)$$

For any fixed $x\in\mathcal{X}$, there exists an unique $c_2(x)\in\mathcal{Y}$ such that

$$f(x,c_2(x))=\max_{y\in\mathcal{Y}} f(x,y)$$

Problem Setting

- Consider optimization for groups of variables:

$$\max_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} f(x, y)$$

where $\mathcal{X}, \mathcal{Y}$ are *convex*, and the objective function f is such that:

  - $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is bounded above;
  - f is continuous and has continuous partial derivatives on $\mathcal{X} \times \mathcal{Y}$;
  - For any fixed $y \in \mathcal{Y}$, there exists an unique $c_1(y) \in \mathcal{X}$ such that

$$f(c_1(y), y) = \max_{x \in \mathcal{X}} f(x, y)$$

For any fixed $x \in \mathcal{X}$, there exists an unique $c_2(x) \in \mathcal{Y}$ such that

$$f(x, c_2(x)) = \max_{y \in \mathcal{Y}} f(x, y)$$

Problem Setting

- Consider optimization for groups of variables:

$$\max_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} f(x,y)$$

where $\mathcal{X},\mathcal{Y}$ are *convex*, and the objective function f is such that:

  - $f:\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ is bounded above;
  - f is continuous and has continuous partial derivatives on $\mathcal{X}\times\mathcal{Y}$;
  - For any fixed $y\in\mathcal{Y}$, there exists an unique $c_1(y)\in\mathcal{X}$ such that

$$f(c_1(y),y)=\max_{x\in\mathcal{X}} f(x,y)$$

For any fixed $x\in\mathcal{X}$, there exists an unique $c_2(x)\in\mathcal{Y}$ such that

$$f(x,c_2(x))=\max_{y\in\mathcal{Y}} f(x,y)$$

Alternating Optimization (AO) and its Variants

Algorithm 1 Alternating Optimization

Require: Objective function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$;

Ensure: A near-optimal point $(\hat{x}, \hat{y}) \in \mathcal{X} \times \mathcal{Y}$;

$(x^1, y^1) \leftarrow$ INITALIZE$()$;

for $t = 1, 2, \ldots, T$ **do**

$\quad x^{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} f(x, y^t)$;

$\quad y^{t+1} \leftarrow \arg\max_{y \in \mathcal{Y}} f(x^{t+1}, y)$;

end for

return (x^T, y^T)

1. Question 1: What point does AO Algorithm converge to?
2. Question 2: How fast for the convergence of AO Algorithm?
3. Question 3: Is the limit point guaranteed to be (*global*) optimal?

Alternating Optimization (AO) and its Variants

Algorithm 2 Alternating Optimization

Require: Objective function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$;

Ensure: A near-optimal point $(\hat{x}, \hat{y}) \in \mathcal{X} \times \mathcal{Y}$;

1: $(x^1, y^1) \leftarrow$ INITALIZE$()$;

2: **for** $t = 1, 2, \ldots, T$ **do**

3: $\quad x^{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} f(x, y^t)$;

4: $\quad y^{t+1} \leftarrow \arg\max_{y \in \mathcal{Y}} f(x^{t+1}, y)$;

5: **end for**

6: **return** (x^T, y^T)

1. Question 1: What point does AO Algorithm converge to?
2. Question 2: How fast for the convergence of AO Algorithm?
3. Question 3: Is the limit point guaranteed to be (*global*) optimal?

Remarks for Alternating Optimization

- Alternating Optimization Algorithm converges to *Bistable Point*.

Definition (Bistable Point)

A point $(x^*, y^*) \in \mathcal{X} \times \mathcal{Y}$ is said to be a *bistable point* if

$$y^* = \arg\max_{y \in \mathcal{Y}} f(x^*, y), \qquad \text{and} \qquad x^* = \arg\max_{x \in \mathcal{X}} f(x, y^*)$$

- All bistable points are *globally optimal* for convex problems.
  For marginally-convex problems, bi-stable points are equivalent to FOSP.
- Alternating Optimization has sublinear convergence rate.

Remarks for Alternating Optimization

- Alternating Optimization Algorithm converges to *Bistable Point*.

Definition (Bistable Point)

A point $(x^*, y^*) \in \mathcal{X} \times \mathcal{Y}$ is said to be a *bistable point* if

$$y^* = \arg\max_{y \in \mathcal{Y}} f(x^*, y), \qquad \text{and} \qquad x^* = \arg\max_{x \in \mathcal{X}} f(x, y^*)$$

- All bistable points are *globally optimal* for convex problems.
  For marginally-convex problems, bi-stable points are equivalent to FOSP.
- Alternating Optimization has sublinear convergence rate.

Remarks for Alternating Optimization

- Alternating Optimization Algorithm converges to *Bistable Point*.

Definition (Bistable Point)

A point $(x^*, y^*) \in \mathcal{X} \times \mathcal{Y}$ is said to be a *bistable point* if

$$y^* = \arg\max_{y \in \mathcal{Y}} f(x^*, y), \qquad \text{and} \qquad x^* = \arg\max_{x \in \mathcal{X}} f(x, y^*)$$

- All bistable points are *globally optimal* for convex problems.
  For marginally-convex problems, bi-stable points are equivalent to FOSP.
- Alternating Optimization has sublinear convergence rate.

Concluding Remarks

1. Successful for a collection of machine learning problems:
   - Phase retrieval
   - Matrix sensing
   - Robust PCA
   - Matrix Completion
   - Mixed linear regression
2. Special initialization is designed, such as *spectral initialization*
3. Empirically: memory efficient and parallelly faster than convex methods such as trace-norm minimization
4. **Open Problem:** a more general theory of AltMin and its convergence ...

Table of Contents

Problem Setting

- An statistical model $\mathcal{F}$ is generating $X \in \mathcal{X}, Z \in \mathcal{Z}$:

$$\mathcal{F} = \{f_\theta = p(\cdot, \cdot \mid \theta) : \theta \in \Theta\}$$

- f_{θ^*} generates sample pair $(x_i, z_i)_{i=1}^n$, but only $\{x_i\}_{i=1}^n$ is observed.
- The goal is to recover θ^*, by using only samples $\{x_i\}_{i=1}^n$.
  The most popular method is by likelihood maximization:

$$\mathcal{L}(\theta; x_1, \dots, x_n) \triangleq p(x_1, \dots, x_n \mid \theta) = \prod_{i=1}^n \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta)$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \log \mathcal{L}(\theta; x_1, \dots, x_n) \triangleq \sum_{i=1}^n \log \mathcal{L}(\theta; x_i)$$

- Intractability: The objective contains $|\mathcal{Z}|^n$ conditional probability terms!

Problem Setting

- An statistical model $\mathcal{F}$ is generating $X \in \mathcal{X}, Z \in \mathcal{Z}$:

$$\mathcal{F} = \{f_\theta = p(\cdot, \cdot \mid \theta) : \theta \in \Theta\}$$

- f_{θ^*} generates sample pair $(x_i, z_i)_{i=1}^n$, but only $\{x_i\}_{i=1}^n$ is observed.
- The goal is to recover θ^*, by using only samples $\{x_i\}_{i=1}^n$. The most popular method is by likelihood maximization:

$$\mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq p(x_1, \ldots, x_n \mid \theta) = \prod_{i=1}^n \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta)$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^n \log \mathcal{L}(\theta; x_i)$$

- Intractability: The objective contains $|\mathcal{Z}|^n$ conditional probability terms!

Problem Setting

- An statistical model $\mathcal{F}$ is generating $X \in \mathcal{X}, Z \in \mathcal{Z}$:

$$\mathcal{F} = \{f_\theta = p(\cdot, \cdot \mid \theta) : \theta \in \Theta\}$$

- f_{θ^*} generates sample pair $(x_i, z_i)_{i=1}^n$, but only $\{x_i\}_{i=1}^n$ is observed.
- The goal is to recover θ^*, by using only samples $\{x_i\}_{i=1}^n$.
  The most popular method is by likelihood maximization:

$$\mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq p(x_1, \ldots, x_n \mid \theta) = \prod_{i=1}^n \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta)$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^n \log \mathcal{L}(\theta; x_i)$$

- Intractability: The objective contains $|\mathcal{Z}|^n$ conditional probability terms!

Problem Setting

- An statistical model $\mathcal{F}$ is generating $X \in \mathcal{X}, Z \in \mathcal{Z}$:

$$\mathcal{F} = \{f_\theta = p(\cdot, \cdot \mid \theta) : \theta \in \Theta\}$$

- f_{θ^*} generates sample pair $(x_i, z_i)_{i=1}^n$, but only $\{x_i\}_{i=1}^n$ is observed.
- The goal is to recover θ^*, by using only samples $\{x_i\}_{i=1}^n$.
  The most popular method is by likelihood maximization:

$$\mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq p(x_1, \ldots, x_n \mid \theta) = \prod_{i=1}^n \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta)$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^n \log \mathcal{L}(\theta; x_i)$$

- Intractability: The objective contains $|\mathcal{Z}|^n$ conditional probability terms!

Problem Setting

- An statistical model $\mathcal{F}$ is generating $X \in \mathcal{X}, Z \in \mathcal{Z}$:

$$\mathcal{F} = \{f_\theta = p(\cdot, \cdot \mid \theta) : \theta \in \Theta\}$$

- f_{θ^*} generates sample pair $(x_i, z_i)_{i=1}^n$, but only $\{x_i\}_{i=1}^n$ is observed.
- The goal is to recover θ^*, by using only samples $\{x_i\}_{i=1}^n$.
  The most popular method is by likelihood maximization:

$$\mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq p(x_1, \ldots, x_n \mid \theta) = \prod_{i=1}^n \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta)$$

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^n \log \mathcal{L}(\theta; x_i)$$

- Intractability: The objective contains $|\mathcal{Z}|^n$ conditional probability terms!

Description of EM Algorithm

- The likelihood maximization is to solve

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta\in\Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^{n} \underbrace{\log \sum_{z_i\in\mathcal{Z}} p(x_i, z_i \mid \theta)}_{\log \mathcal{L}(\theta, x_i)}$$

- The EM Algorithm is separated into two steps:
  1. E-step: Construct the Q-function $Q(\theta \mid \theta^t)$;
     (A lower bound of the objective function for current iteration)
  2. M-step: Maximize the Q-function such that

$$\theta^{t+1} \leftarrow \arg\max Q(\theta \mid \theta^t).$$

Description of EM Algorithm

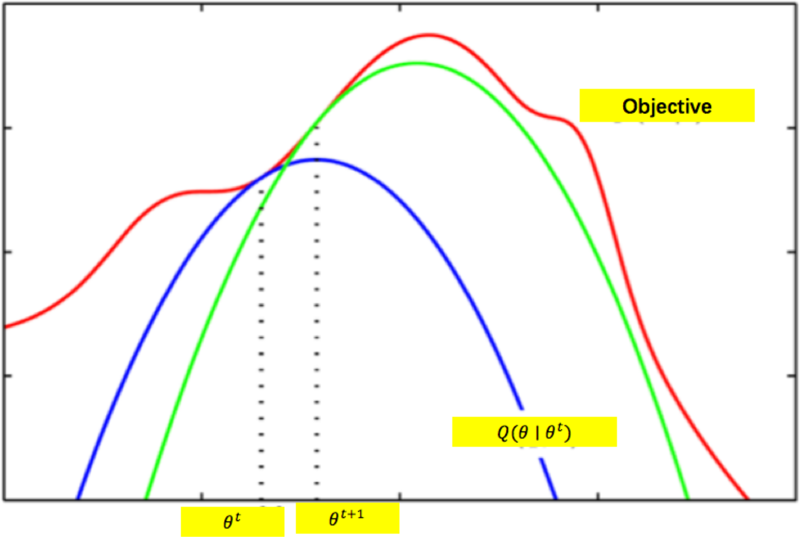
- The likelihood maximization is to solve

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta\in\Theta} \log \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq \sum_{i=1}^{n} \underbrace{\log \sum_{z_i\in\mathcal{Z}} p(x_i, z_i \mid \theta)}_{\log \mathcal{L}(\theta, x_i)}$$

- The EM Algorithm is separated into two steps:
  1. E-step: Construct the Q-function $Q(\theta \mid \theta^t)$;
     (A lower bound of the objective function for current iteration)
  2. M-step: Maximize the Q-function such that

$$\theta^{t+1} \leftarrow \arg\max Q(\theta \mid \theta^t).$$

EM is a successive approximation method



EM Algorithm in Detail

Key inequality:

$$\log \mathcal{L}(\theta; x_i) \triangleq \log \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta) \geq Q_{x_i}(\theta \mid \theta^t) \triangleq \mathbb{E}_{z \sim p(\cdot \mid x_i, \theta^t)}[\log p(x_i, z \mid \theta)]$$

$$\text{Objective} \triangleq \sum_{i=1}^{n} \log \mathcal{L}(\theta; x_i) \geq Q(\theta \mid \theta^t) \triangleq \sum_{i=1}^{n} Q_{x_i}(\theta \mid \theta^t)$$

EM Algorithm Description

- E-step: Compute the distribution $p(Z \mid x_i, \theta^t)$.
  (Then the Q-function can be constructed)
- M-Step: Choose θ^{t+1} maximizing the objective function above.

Remark: The construction of Q-function only requires $p(\cdot \mid x, \theta^t)$ and $p(\cdot, \cdot \mid \theta)$.

EM Algorithm in Detail

Key inequality:

$$\log \mathcal{L}(\theta; x_i) \triangleq \log \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta) \geq Q_{x_i}(\theta \mid \theta^t) \triangleq \mathbb{E}_{z \sim p(\cdot \mid x_i, \theta^t)}[\log p(x_i, z \mid \theta)]$$

$$\text{Objective} \triangleq \sum_{i=1}^{n} \log \mathcal{L}(\theta; x_i) \geq Q(\theta \mid \theta^t) \triangleq \sum_{i=1}^{n} Q_{x_i}(\theta \mid \theta^t)$$

EM Algorithm Description

- E-step: Compute the distribution $p(Z \mid x_i, \theta^t)$.
  (Then the Q-function can be constructed)
- M-Step: Choose θ^{t+1} maximizing the objective function above.

Remark: The construction of Q-function only requires $p(\cdot \mid x, \theta^t)$ and $p(\cdot, \cdot \mid \theta)$.

EM Algorithm in Detail

Key inequality:

$$\log \mathcal{L}(\theta; x_i) \triangleq \log \sum_{z_i \in \mathcal{Z}} p(x_i, z_i \mid \theta) \geq Q_{x_i}(\theta \mid \theta^t) \triangleq \mathbb{E}_{z \sim p(\cdot \mid x_i, \theta^t)}[\log p(x_i, z \mid \theta)]$$

$$\text{Objective} \triangleq \sum_{i=1}^{n} \log \mathcal{L}(\theta; x_i) \geq Q(\theta \mid \theta^t) \triangleq \sum_{i=1}^{n} Q_{x_i}(\theta \mid \theta^t)$$

EM Algorithm Description

- E-step: Compute the distribution $p(Z \mid x_i, \theta^t)$.
  (Then the Q-function can be constructed)
- M-Step: Choose θ^{t+1} maximizing the objective function above.

Remark: The construction of Q-function only requires $p(\cdot \mid x, \theta^t)$ and $p(\cdot, \cdot \mid \theta)$.

EM as maximization-maximization

Consider the following function:

$$F(\theta, q) \equiv \mathbb{E}_q[\log p(z, x_{1:n} \mid \theta)] + H(q), \quad q \text{ is some distribution of } z.$$

Or equivalently,

$$F(\theta, q) = -D(q \| q_\theta) + \mathcal{L}(\theta; x_{1:n}), \qquad q_\theta(z) = p(z \mid x_{1:n}, \theta)$$

Theorem (Alternative Maximimzation of F reduces to EM)

1. *For fixed θ, the unique maximizer of $F(\theta, q)$ is given by*

$$q = q_\theta \triangleq p(z \mid x_{1:n}, \theta).$$

2. *For fixed $q = q_\theta$, the M-step is equivalent to maximizing $F(\theta, q_\theta)$.*

EM as maximization-maximization

Consider the following function:

$$F(\theta, q) \equiv \mathbb{E}_q[\log p(z, x_{1:n} \mid \theta)] + H(q), \quad q \text{ is some distribution of } z.$$

Or equivalently,

$$F(\theta, q) = -D(q \| q_\theta) + \mathcal{L}(\theta; x_{1:n}), \qquad q_\theta(z) = p(z \mid x_{1:n}, \theta)$$

Theorem (Alternative Maximimzation of F reduces to EM)

1. *For fixed θ, the unique maximizer of $F(\theta, q)$ is given by*

$$q = q_\theta \triangleq p(z \mid x_{1:n}, \theta).$$

2. *For fixed $q = q_\theta$, the M-step is equivalent to maximizing $F(\theta, q_\theta)$.*

Landscape of Alternating Maximization

Alternating Maximimzation & EM Algorithm

Suppose that the function $F(\theta, q)$ has a local maxima (θ^*, q^*):

$$\begin{aligned}(\theta^*, q^*) &= \arg\max F(\theta, q)\\ &:= \arg\max \mathbb{E}_q[\log p(z, x_{1:n} \mid \theta)] + H(q)\\ &:= \arg\max -D(q \| q_\theta) + \mathcal{L}(\theta; x_{1:n}).\end{aligned}$$

Then the point θ^* is a local maxima of objective function $\mathcal{L}(\theta; x_{1:n})$:

$$\theta^* = \arg\max \mathcal{L}(\theta; x_1, \ldots, x_n) \triangleq p(x_1, \ldots, x_n \mid \theta).$$

Proof.

$$\text{local maxima } (\theta^*, q^*) \implies q^* = q_\theta \implies \theta^* = \arg\max \mathcal{L}(\theta; x_{1:n})$$

□

Landscape of Alternating Maximization

Alternating Maximimzation & EM Algorithm

Suppose that the function $F(\theta, q)$ has a local maxima (θ^*, q^*):

$$\begin{aligned}(\theta^*, q^*) &= \arg\max F(\theta, q) \\ &:= \arg\max \mathbb{E}_q[\log p(z, x_{1:n} \mid \theta)] + H(q) \\ &:= \arg\max -D(q \| q_\theta) + \mathcal{L}(\theta; x_{1:n}).\end{aligned}$$

Then the point θ^* is a local maxima of objective function $\mathcal{L}(\theta; x_{1:n})$:

$$\theta^* = \arg\max \mathcal{L}(\theta; x_1, \dots, x_n) \triangleq p(x_1, \dots, x_n \mid \theta).$$

Proof.

$$\text{local maxima } (\theta^*, q^*) \implies q^* = q_\theta \implies \theta^* = \arg\max \mathcal{L}(\theta; x_{1:n})$$

□

EM Algorithm as Alternating Maximization

Algorithm 3 Expectation Maximization as Alternating Maximization

1: **Input:** Solvers of $\max\ F(\theta, q)$ for fixed q and $\max\ F(\theta, q)$ for fixed θ;
2: **Output:** A good parameter $\hat{\theta} \in \Theta$;
3: $\theta^1 \leftarrow$ INITALIZE();
4: **for** $t = 1, 2, \ldots$ **do**
5: $\quad q^t \leftarrow \arg\max F(\theta^{t-1}, q)$;
6: $\quad \theta^t \leftarrow \arg\max F(\theta, q^t)$;
7: **end for**

Comment:

1. The Alternating Maximization blindly searches for FOSP.
2. Pay attention to Initialization, unless for unique bi-stable point;

EM Algorithm as Alternating Maximization

Algorithm 4 Expectation Maximization as Alternating Maximization

Input: Solvers of $\max\ F(\theta, q)$ for fixed q and $\max\ F(\theta, q)$ for fixed θ;
Output: A good parameter $\hat{\theta} \in \Theta$;
$\theta^1 \leftarrow$ INITALIZE();
for $t = 1, 2, \ldots$ **do**
    $q^t \leftarrow \arg\max F(\theta^{t-1}, q)$;
    $\theta^t \leftarrow \arg\max F(\theta, q^t)$;
end for

Comment:

1. The Alternating Maximization blindly searches for FOSP.
2. Pay attention to Initialization, unless for unique bi-stable point;

Table of Contents

Incremental Variants of EM Algorithm (IVEM)

Key Observation:

$$\begin{aligned} F(\theta, q) &= \mathbb{E}_q[\log p(z, x_{1:n} \mid \theta)] + H(q) \\ &= \sum_{i=1}^{n} \mathbb{E}_q[\log p(z_i, x_i \mid \theta)] + H(q_i) \\ &\triangleq \sum_{i=1}^{n} F_i(\theta, q_i), \end{aligned}$$

Incremental Variants of EM Algorithm (IVEM)

$$F(\theta, q) = \sum_{i=1}^{n} F_i(\theta, q_i)$$

Perform the maximization of $q := (q_i)_{i=1}^{n}$ *lazily*:

Algorithm 5 An Incremental Variant of EM Algorithm

```
Input: Solvers of max F_i(θ, q) for fixed θ;
Output: A good parameter θ̂ ∈ Θ;
θ^1 ← INITALIZE();
for t = 1, 2, ... do
    Pick Index i_t from {1, ..., n};
    q^t_{i_t} ← arg max F_{i_t}(θ^{t−1}, q_{i_t});
    q^t_j ← q^{t−1}_j for any j ≠ i_t;
    θ^t ← arg max F(θ, q^t) = Σ_{i=1}^{n} F_i(θ, q^t_i)        (Not Memory Efficient);
end for
```

- **Input:** Solvers of $\max \; F_i(\theta, q)$ for fixed θ;
- **Output:** A good parameter $\hat{\theta} \in \Theta$;
- $\theta^1 \leftarrow$ INITALIZE$()$;
- **for** $t = 1, 2, \ldots$ **do**
  - Pick Index i_t from $\{1, \ldots, n\}$;
  - $q_{i_t}^t \leftarrow \arg\max F_{i_t}(\theta^{t-1}, q_{i_t})$;
  - $q_j^t \leftarrow q_j^{t-1}$ for any $j \neq i_t$;
  - $\theta^t \leftarrow \arg\max F(\theta, q^t) = \sum_{i=1}^{n} F_i(\theta, q_i^t)$ (Not Memory Efficient);
- **end for**

Finite-Sum Problem Setting

Consider the optimization for a finite-sum objective:

$$\max_x \quad \sum_{i=1}^{n} f_i(x)$$

$$\begin{aligned} \text{Full GD}: \quad & x^{t+1} = x^t + \eta_t \frac{1}{n} \sum_i \nabla f_i(x^t) \quad && \mathcal{O}(n) \text{ computations per iteration} \\ \text{SGD}: \quad & x^{t+1} = x^t + \eta_t \nabla f_{i_t}(x^t) && \mathcal{O}(1) \text{ computation per iteration} \\ & && \text{(but many more iterations)} \end{aligned}$$

Stochastic Average Gradient (SAG)

- At each iteration t, update the gradient estimate lazily:
  - Pick Index i_t from $\{1, \ldots, n\}$,

$$g_{i_t}^t = \nabla f_{i_t}(x^{t-1})$$
$$g_j^t = g_j^{t-1}, \quad \text{for any } j \neq i_t$$

  - The gradient estimate for objective $\sum_{i=1}^{n} f_i(x)$ is given by

$$\sum_{i=1}^{n} g_i^t$$

- At each iteration t perform the update

$$x^t = x^{t-1} + \eta_t \sum_{i=1}^{n} g_i^t$$

Stochastic Average Gradient (SAG)

- At each iteration t, update the gradient estimate lazily:
  - Pick Index i_t from $\{1, \dots, n\}$,

$$g_{i_t}^t = \nabla f_{i_t}(x^{t-1})$$
$$g_j^t = g_j^{t-1}, \quad \text{for any } j \neq i_t$$

  - The gradient estimate for objective $\sum_{i=1}^n f_i(x)$ is given by

$$\sum_{i=1}^n g_i^t$$

- At each iteration t perform the update

$$x^t = x^{t-1} + \eta_t \sum_{i=1}^n g_i^t$$

Memoery Efficient Implementation of SAG

The gradient update of SAG only takes *contant* time, independ of n:

$$\sum_{i=1}^{n} g_i^t = g_{i_t}^t - g_{i_t}^{t-1} + \sum_{i=1}^{n} g_i^{t-1}$$

$$a^t = g_{i_t}^t - g_{i_t}^{t-1} + a^{t-1}$$

At each iteration t perform the update

$$x^t = x^{t-1} + \eta_t a^t$$

Memoery Efficient Implementation of SAG

The gradient update of SAG only takes *contant* time, independ of n:

$$\sum_{i=1}^{n} g_i^t = g_{i_t}^t - g_{i_t}^{t-1} + \sum_{i=1}^{n} g_i^{t-1}$$

$$a^t = g_{i_t}^t - g_{i_t}^{t-1} + a^{t-1}$$

At each iteration t perform the update

$$x^t = x^{t-1} + \eta_t a^t$$

Memoery Efficient Implementation of IVEM

$$\max \sum_{i=1}^{n} F_i(\theta, q_i^t) = \max \sum_{i=1}^{n} F_i(\theta, q_i^{t-1}) - F_{i_t}(\theta, q_{i_t}^{t-1}) + F_{i_t}(\theta, q_{i_t}^t)$$

Algorithm 6 Another Incremental Variant of EM Algorithm

1: **Input:** Solvers of $\max \ F_i(\theta, q)$ for fixed θ;
2: **Output:** A good parameter $\hat{\theta} \in \Theta$;
3: $\theta^1 \leftarrow$ INITALIZE();
4: **for** $t = 1, 2, \ldots$ **do**
5: Pick Index i_t from $\{1, \ldots, n\}$;
6: $q_{i_t}^t \leftarrow \arg\max F_{i_t}(\theta^{t-1}, q_{i_t})$;
7: $\theta^t \leftarrow \arg\max F(\theta, q^{t-1}) - F_{i_t}(\theta, q_{i_t}^{t-1}) + F_{i_t}(\theta, q_{i_t}^t)$;
8: **end for**

Exponentially decaying variants of IVEM

$$\max \sum_{i=1}^{n} F_i(\theta, q_i^t) \approx \max \ \gamma \sum_{i=1}^{n} F_i(\theta, q_i^{t-1}) + F_{i_t}(\theta, q_{i_t}^t)$$

Algorithm 7 Exponentially decaying version of IVEM

1: **Input:** Solvers of $\max \ F_i(\theta, q)$ for fixed θ;
2: **Output:** A good parameter $\hat{\theta} \in \Theta$;
3: $\theta^1 \leftarrow$ INITALIZE();
4: **for** $t = 1, 2, \ldots$ **do**
5: Pick Index i_t from $\{1, \ldots, n\}$;
6: $q_{i_t}^t \leftarrow \arg\max F_{i_t}(\theta^{t-1}, q_{i_t})$;
7: $\theta^t \leftarrow \arg\max \gamma F(\theta, q^{t-1}) + F_{i_t}(\theta, q_{i_t}^t)$;
8: **end for**

Table of Contents

Preliminary Definitions

- The KL-Divergence between two distributions $\boldsymbol{p}$ and $\boldsymbol{q}$ is given by:

$$D(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

  where $\boldsymbol{p}$ and $\boldsymbol{q}$ are assumed to have the same support set $\{1, \ldots, n\}$
- The mutual information between a pair of random variables (X, Y) is:

$$I(X;Y) = D(p_{X,Y} \| p_X \otimes p_Y)$$

- Suppose that the joint distribution of (X, Y) is $p_X \cdot p_{Y|X}$. For fixed conditional distribution $p_{Y|X}$, the channel capacity of (X, Y) is defined as

$$C = \max_{p_X} I(X;Y)$$

Preliminary Definitions

- The KL-Divergence between two distributions $\boldsymbol{p}$ and $\boldsymbol{q}$ is given by:

$$D(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

  where $\boldsymbol{p}$ and $\boldsymbol{q}$ are assumed to have the same support set $\{1, \ldots, n\}$
- The mutual information between a pair of random variables (X, Y) is:

$$I(X;Y) = D(p_{X,Y} \| p_X \otimes p_Y)$$

- Suppose that the joint distribution of (X, Y) is $p_X \cdot p_{Y|X}$. For fixed conditional distribution $p_{Y|X}$, the channel capacity of (X, Y) is defined as

$$C = \max_{p_X} I(X;Y)$$

Preliminary Definitions

- The KL-Divergence between two distributions $\boldsymbol{p}$ and $\boldsymbol{q}$ is given by:

$$D(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$$

  where $\boldsymbol{p}$ and $\boldsymbol{q}$ are assumed to have the same support set $\{1, \ldots, n\}$
- The mutual information between a pair of random variables (X, Y) is:

$$I(X;Y) = D(p_{X,Y} \| p_X \otimes p_Y)$$

- Suppose that the joint distribution of (X, Y) is $p_X \cdot p_{Y|X}$. For fixed conditional distribution $p_{Y|X}$, the channel capacity of (X, Y) is defined as

$$C = \max_{p_X} I(X;Y)$$

Variantional Charactization of Mutual Information

- Suppose that $X \sim \boldsymbol{p}$ and $Y \mid X \sim \boldsymbol{Q}$, then

$$I(X;Y) \triangleq I(\boldsymbol{p}, \boldsymbol{Q}) = \sum_{i=1}^{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{Y}|} p_i Q_{i,j} \log \frac{Q_{i,j}}{q_j}$$

where $Y \sim \boldsymbol{q} \triangleq \boldsymbol{p} \cdot \boldsymbol{Q}$.

Theorem (Global Tight Lower Bound of Mutual Information)

$$I(\boldsymbol{p}, \boldsymbol{Q}) \geq \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) \triangleq \sum_{i=1}^{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{Y}|} p_i Q_{i,j} \log \frac{\phi(i \mid j)}{p_i}$$

$$I(\boldsymbol{p}, \boldsymbol{Q}) = \max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi),$$

where the optimal transition matrix ϕ is given by:

$$\phi^*(i \mid j) = p_i \frac{Q_{i,j}}{q_j} \triangleq p_i \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i Q_{i,j}}$$

Variantional Charactization of Mutual Information

- Suppose that $X \sim \boldsymbol{p}$ and $Y \mid X \sim \boldsymbol{Q}$, then

$$I(X;Y) \triangleq I(\boldsymbol{p}, \boldsymbol{Q}) = \sum_{i=1}^{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{Y}|} p_i Q_{i,j} \log \frac{Q_{i,j}}{q_j}$$

where $Y \sim \boldsymbol{q} \triangleq \boldsymbol{p} \cdot \boldsymbol{Q}$.

Theorem (Global Tight Lower Bound of Mutual Information)

$$I(\boldsymbol{p}, \boldsymbol{Q}) \geq \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) \triangleq \sum_{i=1}^{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{Y}|} p_i Q_{i,j} \log \frac{\phi(i \mid j)}{p_i}$$

$$I(\boldsymbol{p}, \boldsymbol{Q}) = \max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi),$$

where the optimal transition matrix ϕ is given by:

$$\phi^*(i \mid j) = p_i \frac{Q_{i,j}}{q_j} \triangleq p_i \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i Q_{i,j}}$$

Variants of Channel Capacity Problem

- Channel Capacity Problem: for fixed transition matrix Q, we aim to solve

$$C(Q) := \max_{p_X} I_Q(X;Y)$$

- Rate Distortion Functions: for fixed distortion D,

$$R(D) := \min_{Q(y|x)} I(X;Y), \quad \text{subject to } \mathbb{E}d(X;Y) \leq D.$$

- Information Bottleneck Problem.

Comment: The method to be discussed can be applied to all those problems.

Variants of Channel Capacity Problem

- Channel Capacity Problem: for fixed transition matrix $\boldsymbol{Q}$, we aim to solve

$$C(Q) := \max_{p_X} I_Q(X;Y)$$

- Rate Distortion Functions: for fixed distortion D,

$$R(D) := \min_{Q(y|x)} I(X;Y), \quad \text{subject to } \mathbb{E}d(X;Y) \leq D.$$

- Information Bottleneck Problem.

Comment: The method to be discussed can be applied to all those problems.

Variants of Channel Capacity Problem

- Channel Capacity Problem: for fixed transition matrix $\boldsymbol{Q}$, we aim to solve

$$C(Q) := \max_{p_X} I_Q(X;Y)$$

- Rate Distortion Functions: for fixed distortion D,

$$R(D) := \min_{Q(y|x)} I(X;Y), \quad \text{subject to } \mathbb{E}d(X;Y) \leq D.$$

- Information Bottleneck Problem.

Comment: The method to be discussed can be applied to all those problems.

Table of Contents

Channel Capacity by Alternating Optimization (1)

The Channel Capacity is by maximizing over two groups of variables:

$$\mathcal{C} = \max_{\boldsymbol{p}} \max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi)$$

Key observation

1. For fixed $\boldsymbol{p}$ and $\boldsymbol{Q}$, we have

$$\arg\max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \phi^*, \quad \phi^*(i \mid j) = p_i \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i Q_{i,j}}$$

2. For fixed ϕ and $\boldsymbol{Q}$, we have

$$\arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \boldsymbol{p}^*, \quad p^*(i) = \frac{r_i}{\sum_{i=1}^{|\mathcal{X}|} r_i},$$

$$r_i \triangleq \exp\left(\sum_{j=1}^{|\mathcal{Y}|} Q_{i,j} \log \phi(i \mid j)\right)$$

Channel Capacity by Alternating Optimization (1)

The Channel Capacity is by maximizing over two groups of variables:

$$\mathcal{C} = \max_{\boldsymbol{p}} \max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi)$$

Key observation

1. For fixed $\boldsymbol{p}$ and $\boldsymbol{Q}$, we have

$$\arg\max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \phi^*, \quad \phi^*(i \mid j) = p_i \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i Q_{i,j}}$$

2. For fixed ϕ and $\boldsymbol{Q}$, we have

$$\arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \boldsymbol{p}^*, \quad p^*(i) = \frac{r_i}{\sum_{i=1}^{|\mathcal{X}|} r_i},$$

$$r_i \triangleq \exp\left(\sum_{j=1}^{|\mathcal{Y}|} Q_{i,j} \log \phi(i \mid j)\right)$$

Channel Capacity by Alternating Optimization (1)

The Channel Capacity is by maximizing over two groups of variables:

$$\mathcal{C} = \max_{\boldsymbol{p}} \max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi)$$

Key observation

1. For fixed $\boldsymbol{p}$ and $\boldsymbol{Q}$, we have

$$\arg\max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \phi^*, \quad \phi^*(i \mid j) = p_i \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i Q_{i,j}}$$

2. For fixed ϕ and $\boldsymbol{Q}$, we have

$$\arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}, \boldsymbol{Q}; \phi) = \boldsymbol{p}^*, \quad p^*(i) = \frac{r_i}{\sum_{i=1}^{|\mathcal{X}|} r_i},$$

$$r_i \triangleq \exp\left(\sum_{j=1}^{|\mathcal{Y}|} Q_{i,j} \log \phi(i \mid j)\right)$$

Channel Capacity by Alternating Optimization (2)

The Channel Capacity can be computed using alternating maximization:

$$\phi^{t+1} = \arg\max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi)$$
$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1})$$

Key observation

1. $$\phi^{t+1}(i \mid j) = p_i^t \frac{Q_{i,j}}{\sum_{i=1}^{|\mathcal{X}|} p_i^t Q_{i,j}} \triangleq \frac{p_i^t Q_{i,j}}{q_j^{t+1}}$$
2. Substituting $\phi := \phi^{t+1}$ and for fixed $\boldsymbol{Q}$,

$$p^{t+1}(i) = \frac{r_i^{t+1}}{\sum_{i=1}^{|\mathcal{X}|} r_i^{t+1}}, \qquad \text{with } r_i^{t+1} \triangleq p_i^t \exp\left(D(\boldsymbol{Q}_i \| \boldsymbol{q}^{t+1}) \right)$$

Blahut-Arimoto Algorithm

Algorithm 8 Blahut-Arimoto Algorithm for Computing Channel Capacity

1: $\boldsymbol{p}^0 \leftarrow$ INITALIZE$()$;
2: **for** $t = 1, 2, \ldots$ **do**
3: $\quad \boldsymbol{q}^{t+1} \leftarrow \boldsymbol{p}^t \boldsymbol{Q}$;
4:

$$p_i^{t+1} \leftarrow \frac{p_i^t \exp(D(\boldsymbol{Q}_i \| \boldsymbol{q}^{t+1}))}{\sum_{i=1}^{|\mathcal{X}|} p_i^t \exp(D(\boldsymbol{Q}_i \| \boldsymbol{q}^{t+1}))}$$

5: **end for**

Table of Contents

Re-consideration of Blahut-Arimoto Algorithm

- The Blahut-Arimoto Algorithm is essentially the alternating maximization:

$$\phi^{t+1} = \arg\max_{\phi\in\Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi)$$
$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1})$$

Theorem (Reformulation for the update of $\boldsymbol{p}^{t+1}$)

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I(\boldsymbol{p}^t, \boldsymbol{Q}) + \sum_{i=1}^{N} (p_i - p_i^t) \cdot D(\boldsymbol{Q}_i \| \boldsymbol{q}^t) - D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

Each update of $\boldsymbol{p}^{t+1}$ suffices to maximize the first-order Taylor-expansion of $I(\boldsymbol{p}, \boldsymbol{Q})$, with penalty term $D(\boldsymbol{p} \| \boldsymbol{p}^t)$.

Re-consideration of Blahut-Arimoto Algorithm

- The Blahut-Arimoto Algorithm is essentially the alternating maximization:

$$\phi^{t+1} = \arg\max_{\phi\in\Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi)$$
$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1})$$

Theorem (Reformulation for the update of $\boldsymbol{p}^{t+1}$)

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I(\boldsymbol{p}^t, \boldsymbol{Q}) + \sum_{i=1}^{N} (p_i - p_i^t) \cdot D(\boldsymbol{Q}_i \| \boldsymbol{q}^t) - D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

Each update of $\boldsymbol{p}^{t+1}$ suffices to maximize the first-order Taylor-expansion of $I(\boldsymbol{p}, \boldsymbol{Q})$, with penalty term $D(\boldsymbol{p} \| \boldsymbol{p}^t)$.

Re-consideration of Blahut-Arimoto Algorithm

- The Blahut-Arimoto Algorithm is essentially the alternating maximization:

$$\phi^{t+1} = \arg\max_{\phi \in \Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi)$$
$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1})$$

Theorem (Reformulation for the update of $\boldsymbol{p}^{t+1}$)

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I(\boldsymbol{p}^t, \boldsymbol{Q}) + \sum_{i=1}^{N} (p_i - p_i^t) \cdot D(\boldsymbol{Q}_i \| \boldsymbol{q}^t) - D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

Each update of $\boldsymbol{p}^{t+1}$ suffices to maximize the first-order Taylor-expansion of $I(\boldsymbol{p}, \boldsymbol{Q})$, with penalty term $D(\boldsymbol{p} \| \boldsymbol{p}^t)$.

Re-consideration of Blahut-Arimoto Algorithm

Theorem (Reformulation for the update of $\boldsymbol{p}^{t+1}$)

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I(\boldsymbol{p}^t, \boldsymbol{Q}) + \sum_{i=1}^{N} (p_i - p_i^t) \cdot D(\boldsymbol{Q}_i \| \boldsymbol{q}^t) - D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

Proof.

By substituting the term ϕ^{t+1} into the update of $\boldsymbol{p}^{t+1}$:

$$\begin{aligned}
\boldsymbol{p}^{t+1} :&= \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1}) = \arg\max_{\boldsymbol{p}} \sum_i \sum_j p_i Q_{i,j} \log \frac{\phi^{t+1}(i \mid j)}{p_i} \\
&= \arg\max_{\boldsymbol{p}} \sum_i \sum_j p_i Q_{i,j} \log \frac{p_i^t Q_{i,j}}{p_i q_j^{t+1}} \\
&= \arg\max_{\boldsymbol{p}} \left(\sum_{i=1}^{N} p_i \cdot D(\boldsymbol{Q}_i \| \boldsymbol{q}^t) - D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)
\end{aligned}$$

$\square$

Proximal Point (Mirror Descent) Algorithm

Consider the maximization of a concave function $f(x)$:

$$\max_{x \in \mathcal{X}} \ f(x)$$

- Let $f_k(x)$ be a certain concave approximation of $f(x)$.
- Let $\{t_k > 0 \mid k = 0, 1, \dots\}$ be a sequence of parameters.
- The Bregman distance is a generalized choice of Euclidean distance:

$$B(y, x) = \Phi(y) - \Phi(x) - \nabla^{\mathrm{T}}\Phi(x)(y - x),$$

where Φ is a smooth and strongly convex function

Algorithm 9 Proximal Point (Mirror Descent) Algorithm

$x^0 \leftarrow$ INITALIZE();
for $t = 0, 1, 2, \dots$ **do**

$$x^{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} f_k(x) - \frac{1}{\gamma_t} B(x, x^t).$$

end for

Proximal Point (Mirror Descent) Algorithm

Consider the maximization of a concave function $f(x)$:

$$\max_{x \in \mathcal{X}} \ f(x)$$

- Let $f_k(x)$ be a certain concave approximation of $f(x)$.
- Let $\{t_k > 0 \mid k = 0, 1, \dots\}$ be a sequence of parameters.
- The Bregman distance is a generalized choice of Euclidean distance:

$$B(y, x) = \Phi(y) - \Phi(x) - \nabla^{\mathrm{T}} \Phi(x)(y - x),$$

  where Φ is a smooth and strongly convex function

Algorithm 10 Proximal Point (Mirror Descent) Algorithm

$x^0 \leftarrow$ INITALIZE();
for $t = 0, 1, 2, \dots$ **do**

$$x^{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} f_k(x) - \frac{1}{\gamma_t} B(x, x^t).$$

end for

Proximal Point (Mirror Descent) Algorithm

Consider the maximization of a concave function $f(x)$:

$$\max_{x \in \mathcal{X}} \; f(x)$$

- Let $f_k(x)$ be a certain concave approximation of $f(x)$.
- Let $\{t_k > 0 \mid k = 0, 1, \dots\}$ be a sequence of parameters.
- The Bregman distance is a generalized choice of Euclidean distance:

$$B(y, x) = \Phi(y) - \Phi(x) - \nabla^{\mathrm{T}} \Phi(x)(y - x),$$

where Φ is a smooth and strongly convex function

Algorithm 11 Proximal Point (Mirror Descent) Algorithm

$x^0 \leftarrow$ INITALIZE();

for $t = 0, 1, 2, \dots$ **do**

$$x^{t+1} \leftarrow \arg\max_{x \in \mathcal{X}} f_k(x) - \frac{1}{\gamma_t} B(x, x^t).$$

end for

Blahut-Arimoto Algorithm as a Proximal Point Algorithm

- The Blahut-Arimoto Algorithm is essentially the proximal point algorithm:

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I_t(\boldsymbol{p}, \boldsymbol{Q}) - \frac{1}{\gamma_t} D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

  where
  - $\gamma_t \equiv 1$;
  - I_t is the first order Taylor expansion of $I(\boldsymbol{p}, \boldsymbol{Q})$ around $\boldsymbol{p}^t$.
- A suitable choice of step size γ_t could accelerate this algorithm.

Theorem

$$\gamma_t \le \frac{1}{\lambda_{KL}^2(\boldsymbol{Q})} \triangleq 1 / \left[\sup_{\boldsymbol{p} \neq \boldsymbol{p}'} \frac{D(\boldsymbol{p}\boldsymbol{Q} \| \boldsymbol{p}'\boldsymbol{Q})}{D(\boldsymbol{p} \| \boldsymbol{p}')} \right]$$

Blahut-Arimoto Algorithm as a Proximal Point Algorithm

- The Blahut-Arimoto Algorithm is essentially the proximal point algorithm:

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \left(I_t(\boldsymbol{p}, \boldsymbol{Q}) - \frac{1}{\gamma_t} D(\boldsymbol{p} \| \boldsymbol{p}^t) \right)$$

  where
  - $\gamma_t \equiv 1$;
  - I_t is the first order Taylor expansion of $I(\boldsymbol{p}, \boldsymbol{Q})$ around $\boldsymbol{p}^t$.
- A suitable choice of step size γ_t could accelerate this algorithm.

Theorem

$$\gamma_t \leq \frac{1}{\lambda_{KL}^2(\boldsymbol{Q})} \triangleq 1 / \left[\sup_{\boldsymbol{p} \neq \boldsymbol{p}'} \frac{D(\boldsymbol{p}\boldsymbol{Q} \| \boldsymbol{p}'\boldsymbol{Q})}{D(\boldsymbol{p} \| \boldsymbol{p}')} \right]$$

Concluding Remarks

- EM and Blahut-Arimoto algorithms all belong to *alternating optimization* methods.
- Blahut-Arimoto algorithm can be understood as a generalized EM Algorithm:

$$\phi^{t+1} = \arg\max_{\phi\in\Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi) \qquad \text{E-step}$$

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1}) \qquad \text{M-Step}$$

$$q^{t+1} = \arg\max -D(q \| q_{\theta^t})$$

$$\theta^{t+1} = \arg\max \mathbb{E}_{q^{t+1}}[\log p(z, x_{1:n} \mid \theta)]$$

The first step of them are searching for the poserior probability of latent variable; the second step of them are all maximzing some likelihood function based on the estimation in the first step.

Concluding Remarks

- EM and Blahut-Arimoto algorithms all belong to *alternating optimization* methods.
- Blahut-Arimoto algorithm can be understood as a generalized EM Algorithm:

$$\phi^{t+1} = \arg\max_{\phi\in\Phi} \tilde{I}(\boldsymbol{p}^t; \boldsymbol{Q}; \phi) \qquad \text{E-step}$$

$$\boldsymbol{p}^{t+1} = \arg\max_{\boldsymbol{p}} \tilde{I}(\boldsymbol{p}; \boldsymbol{Q}; \phi^{t+1}) \qquad \text{M-Step}$$

$$q^{t+1} = \arg\max -D(q\|q_{\theta^t})$$

$$\theta^{t+1} = \arg\max \mathbb{E}_{q^{t+1}}[\log p(z, x_{1:n} \mid \theta)]$$

The first step of them are searching for the poserior probability of latent variable; the second step of them are all maximzing some likelihood function based on the estimation in the first step.