Incremental Gradient, Subgradient, and Proximal Methods for Convex Optimization

Dimitri P. Bertsekas

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

April 2016
Recall the Classical Subgradient and Proximal Algorithms

**Convex Optimization Problem**

\[
\begin{align*}
\text{minimize} & \quad f(x) \quad \text{subject to} \quad x \in X, \\
\text{where} & \quad f: \mathbb{R}^n \mapsto \mathbb{R} \text{ is convex, and } X \text{ is closed and convex.}
\end{align*}
\]

**Classical subgradient projection algorithm:** Typical iteration

\[
x_{k+1} = P_X \left( x_k - \alpha_k \tilde{\nabla} f(x_k) \right)
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

**Classical proximal algorithm:** Typical iteration

\[
x_{k+1} = \arg\min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

where \( \alpha_k \) is a positive parameter.

- Proximal has more solid convergence properties, but requires more overhead.
- Proximal algorithm \( \Leftrightarrow \) augmented Lagrangian method.
Recall the Classical Subgradient and Proximal Algorithms

**Convex Optimization Problem**

\[
\begin{align*}
\text{minimize} & \quad f(x) \quad \text{subject to} \quad x \in X, \\
\text{where} \quad f : \mathbb{R}^n \mapsto \mathbb{R} \text{ is convex, and } X \text{ is closed and convex.}
\end{align*}
\]

**Classical subgradient projection algorithm: Typical iteration**

\[
x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f(x_k))
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

**Classical proximal algorithm: Typical iteration**

\[
x_{k+1} = \arg \min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\}
\]

where \( \alpha_k \) is a positive parameter.

- Proximal has more solid convergence properties, but requires more overhead.
- Proximal algorithm \( \iff \) augmented Lagrangian method.
Recall the Classical Subgradient and Proximal Algorithms

Convex Optimization Problem

\[ \text{minimize } f(x) \quad \text{subject to } \quad x \in X, \]
where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is convex, and \( X \) is closed and convex.

Classical subgradient projection algorithm: Typical iteration

\[ x_{k+1} = P_X(x_k - \alpha_k \nabla f(x_k)) \]
where \( \alpha_k \) is a positive stepsize and \( \nabla f(x_k) \) denotes (any) subgradient.

Classical proximal algorithm: Typical iteration

\[ x_{k+1} = \arg\min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\} \]
where \( \alpha_k \) is a positive parameter.

- Proximal has more solid convergence properties, but requires more overhead.
- Proximal algorithm \( \iff \) augmented Lagrangian method.
Recall the Classical Subgradient and Proximal Algorithms

Convex Optimization Problem

\[
\begin{align*}
\text{minimize} & \quad f(x) \quad \text{subject to} \quad x \in X, \\
\text{where} & \quad f : \mathbb{R}^n \mapsto \mathbb{R} \text{ is convex, and } X \text{ is closed and convex.}
\end{align*}
\]

Classical subgradient projection algorithm: Typical iteration

\[
x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f(x_k))
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

Classical proximal algorithm: Typical iteration

\[
x_{k+1} = \arg\min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\}
\]

where \( \alpha_k \) is a positive parameter.

- Proximal has more solid convergence properties, but requires more overhead.
- Proximal algorithm duality \( \iff \) augmented Lagrangian method.
Recall the Classical Subgradient and Proximal Algorithms

**Convex Optimization Problem**

\[
\text{minimize } f(x) \quad \text{subject to } \quad x \in X,
\]

where \( f : \mathbb{R}^n \to \mathbb{R} \) is convex, and \( X \) is closed and convex.

**Classical subgradient projection algorithm: Typical iteration**

\[
x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f(x_k))
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

**Classical proximal algorithm: Typical iteration**

\[
x_{k+1} = \arg \min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

where \( \alpha_k \) is a positive parameter.

- Proximal has more solid convergence properties, but requires more overhead.
- Proximal algorithm duality \( \iff \) augmented Lagrangian method.
Problems with Many Additive Cost Components

\[
\text{minimize } \sum_{i=1}^{m} f_i(x) \quad \text{subject to } x \in X,
\]

where \( f_i : \mathbb{R}^n \rightarrow \mathbb{R} \) are convex, and \( X \) is closed and convex.

Incremental algorithms (long history, early 90s-present): Typical iteration

- Choose an index \( i_k \subset \{1, \ldots, m\} \).
- Perform a subgradient iteration or a proximal iteration:

\[
x_{k+1} = P_X \left(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)\right)
\]

or

\[
x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

Motivation is to avoid processing all the cost components at each iteration.
Problems with Many Additive Cost Components

\[
\begin{align*}
\text{minimize} \quad & \sum_{i=1}^{m} f_i(x) \\
\text{subject to} \quad & x \in X,
\end{align*}
\]

where \( f_i : \mathbb{R}^n \to \mathbb{R} \) are convex, and \( X \) is closed and convex.

Incremental algorithms (long history, early 90s-present): Typical iteration

- Choose an index \( i_k \subset \{1, \ldots, m\} \).
- Perform a subgradient iteration or a proximal iteration:

\[
x_{k+1} = P_X \left( x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k) \right)
\]

or

\[
x_{k+1} = \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\}
\]

Motivation is to avoid processing all the cost components at each iteration.
Problems with Many Additive Cost Components

\[
\text{minimize } \sum_{i=1}^{m} f_i(x) \quad \text{subject to } \quad x \in X,
\]

where \( f_i : \mathbb{R}^n \rightarrow \mathbb{R} \) are convex, and \( X \) is closed and convex.

**Incremental algorithms (long history, early 90s-present): Typical iteration**

- Choose an index \( i_k \subset \{1, \ldots, m\} \).
- Perform a subgradient iteration or a proximal iteration:

\[
x_{k+1} = P_X \left( x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k) \right)
\]

or

\[
x_{k+1} = \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

Motivation is to avoid processing all the cost components at each iteration.
Problems with Many Additive Cost Components

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{m} f_i(x) \\
\text{subject to} & \quad x \in X,
\end{align*}
\]

where \( f_i : \mathbb{R}^n \rightarrow \mathbb{R} \) are convex, and \( X \) is closed and convex.

Incremental algorithms (long history, early 90s-present): Typical iteration

- Choose an index \( i_k \subset \{1, \ldots, m\} \).
- Perform a subgradient iteration or a proximal iteration:

\[
\begin{align*}
x_{k+1} &= P_X \left( x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k) \right) \\
or
\end{align*}
\]

\[
\begin{align*}
x_{k+1} &= \text{arg min}_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\end{align*}
\]

Motivation is to avoid processing all the cost components at each iteration.
Separable Convex Optimization Problem

\[
\begin{align*}
\text{minimize} \quad & \sum_{i=1}^{m} f_i(x^i) & \text{subject to} & \quad x^i \in X_i, \ i = 1, \ldots, m, & \sum_{i=1}^{m} h_i(x^i) = 0 \\
\end{align*}
\]

where \( f_i : \mathbb{R}^{n_i} \to \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \to \mathbb{R}^r \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

Dual problem decomposes

\[
\begin{align*}
\text{maximize} \quad & \sum_{i=1}^{m} q_i(\lambda) & \text{subject to} & \quad \lambda \in \mathbb{R}^r \\
\end{align*}
\]

where \( q_i \) is a “component” dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization Problem

minimize $\sum_{i=1}^{m} f_i(x^i)$ subject to $x^i \in X_i$, $i = 1, \ldots, m$, $\sum_{i=1}^{m} h_i(x^i) = 0$

where $f_i : \mathbb{R}^{n_i} \mapsto \mathbb{R}$ are convex, $h_i : \mathbb{R}^{n_i} \mapsto \mathbb{R}^r$ are linear, $X_i \subset \mathbb{R}^{n_i}$ are closed and convex.

Dual problem decomposes

maximize $\sum_{i=1}^{m} q_i(\lambda)$ subject to $\lambda \in \mathbb{R}^r$

where $q_i$ is a “component” dual function:

$$q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\}$$

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization Problem

minimize \[ \sum_{i=1}^{m} f_i(x^i) \] subject to \[ x^i \in X_i, \ i = 1, \ldots, m, \ \sum_{i=1}^{m} h_i(x^i) = 0 \]

where \( f_i : \mathbb{R}^{n_i} \to \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \to \mathbb{R} \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

Dual problem decomposes

maximize \[ \sum_{i=1}^{m} q_i(\lambda) \] subject to \[ \lambda \in \mathbb{R}^r \]

where \( q_i \) is a “component” dual function:

\[ q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\} \]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization Problem

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{m} f_i(x^i) \quad \text{subject to} \quad x^i \in X_i, \ i = 1, \ldots, m, \quad \sum_{i=1}^{m} h_i(x^i) = 0 \\
\end{align*}
\]

where \( f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^r \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

Dual problem decomposes

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{m} q_i(\lambda) \quad \text{subject to} \quad \lambda \in \mathbb{R}^r \\
\end{align*}
\]

where \( q_i \) is a "component" dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization Problem

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{m} f_i(x^i) \\
\text{subject to} & \quad x^i \in X_i, \quad i = 1, \ldots, m, \\
& \quad \sum_{i=1}^{m} h_i(x^i) = 0
\end{align*}
\]

where \( f_i : \mathbb{R}^{n_i} \to \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \to \mathbb{R}^{r_i} \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

**Dual problem decomposes**

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{m} q_i(\lambda) \\
\text{subject to} & \quad \lambda \in \mathbb{R}^{r_i}
\end{align*}
\]

where \( q_i \) is a “component" dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \{ f_i(x^i) + \lambda' h_i(x^i) \}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
References for this Overview Talk

- Focus on convergence, rate of convergence, component formation, and component selection.

- Work on incremental subgradient methods with A. Nedic, 2000-2010.
- Work on incremental proximal methods, 2010-2012 (DPB).
- Work on incremental augmented Lagrangian methods 2015 (DPB).

General references:
Joint and individual works with A. Nedic and M. Wang.
Focus on convergence, rate of convergence, component formation, and component selection.

Work on incremental subgradient methods with A. Nedic, 2000-2010.
Work on incremental proximal methods, 2010-2012 (DPB).
Work on incremental augmented Lagrangian methods 2015 (DPB).

General references:
References for this Overview Talk

- Focus on convergence, rate of convergence, component formation, and component selection.

- Work on **incremental subgradient methods** with A. Nedic, 2000-2010.
- Work on **incremental proximal methods**, 2010-2012 (DPB).
- Work on **incremental augmented Lagrangian methods** 2015 (DPB).

General references:

- **Convex Optimization Algorithms** book 2015 (DPB).
- **Nonlinear Programming: 3rd edition** 2016 (DPB).
1. Incremental Algorithms
2. Aggregated Incremental Algorithms
3. Incremental Augmented Lagrangian Algorithms
4. Incremental Treatment of Constraints
5. Convergence Analysis
Problem: \( \min_{x \in X} \sum_{i=1}^{m} f_i(x) \), where \( f_i \) and \( X \) are convex

Long history: LMS (Widrow-Hoff, 1960, for linear least squares w/out projection), former Soviet Union literature 1960s, stochastic approximation literature 1960s, neural network literature literature 1970s

Basic incremental subgradient method

\[ x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \]

- Stepsize selection possibilities:
  - \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \)
  - \( \alpha_k \): Constant
  - Dynamically chosen (based on estimate of optimal cost)

- Index \( i_k \) selection possibilities:
  - Cyclically
  - Fully randomized/equal probability \( 1/m \)
  - Reshuffling/randomization within a cycle (frequent practical choice)
**Problem:** \( \min_{x \in X} \sum_{i=1}^{m} f_i(x) \), where \( f_i \) and \( X \) are convex

**Long history:** LMS (Widrow-Hoff, 1960, for linear least squares w/out projection), former Soviet Union literature 1960s, stochastic approximation literature 1960s, neural network literature literature 1970s

**Basic incremental subgradient method**

\[ x_{k+1} = P_X(x_k - \alpha_k \nabla f_{i_k}(x_k)) \]

- **Stepsize selection possibilities:**
  - \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \)
  - \( \alpha_k \): Constant
  - Dynamically chosen (based on estimate of optimal cost)

- **Index \( i_k \) selection possibilities:**
  - Cyclically
  - Fully randomized/equal probability \( 1/m \)
  - Reshuffling/randomization within a cycle (frequent practical choice)
**Problem:** \( \min_{x \in X} \sum_{i=1}^{m} f_i(x) \), where \( f_i \) and \( X \) are convex

**Long history:** LMS (Widrow-Hoff, 1960, for linear least squares w/out projection), former Soviet Union literature 1960s, stochastic approximation literature 1960s, neural network literature literature 1970s

**Basic incremental subgradient method**

\[
x_{k+1} = P_X (x_k - \alpha_k \tilde{\nabla} f_{i_k} (x_k))
\]

**Stepsize selection possibilities:**
- \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \)
- \( \alpha_k \): Constant
- Dynamically chosen (based on estimate of optimal cost)

**Index \( i_k \) selection possibilities:**
- Cyclically
- Fully randomized/equal probability \( 1/m \)
- Reshuffling/randomization within a cycle (frequent practical choice)
Problem: \( \min_{x \in X} \sum_{i=1}^{m} f_i(x) \), where \( f_i \) and \( X \) are convex

Long history: LMS (Widrow-Hoff, 1960, for linear least squares w/out projection), former Soviet Union literature 1960s, stochastic approximation literature 1960s, neural network literature literature 1970s

Basic incremental subgradient method

\[
x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k))
\]

Stepsizes selection possibilities:

- \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \)
- \( \alpha_k \): Constant
- Dynamically chosen (based on estimate of optimal cost)

Index \( i_k \) selection possibilities:

- Cyclically
- Fully randomized/equal probability \( \frac{1}{m} \)
- Reshuffling/randomization within a cycle (frequent practical choice)
Quadratic One-Dimensional Example:  
\[
\min_{x \in \mathbb{R}} \sum_{i=1}^{m} (c_i x - b_i)^2
\]

Conceptually, the idea generalizes to higher dimensions, but is hard to treat/quantify analytically.

Adapting the stepsize \(\alpha_k\) to the farout and confusion regions is an important issue.

Shaping the confusion region is an important issue.
Convergence Mechanism

Quadratic One-Dimensional Example: \( \min_{x \in \mathbb{R}} \sum_{i=1}^{m} (c_i x - b_i)^2 \)

- Conceptually, the idea generalizes to higher dimensions, but is hard to treat/quantify analytically.
- Adapting the stepsize \( \alpha_k \) to the farout and confusion regions is an important issue.
- Shaping the confusion region is an important issue.
Convergence Mechanism

Quadratic One-Dimensional Example: \[ \min_{x \in \mathbb{R}} \sum_{i=1}^{m} (c_i x - b_i)^2 \]

- Conceptually, the idea generalizes to higher dimensions, but is hard to treat/quantify analytically.
- Adapting the stepsize \( \alpha_k \) to the farout and confusion regions is an important issue.
- Shaping the confusion region is an important issue.
Convergence Mechanism

Quadratic One-Dimensional Example:

\[ \min_{x \in \mathbb{R}} \sum_{i=1}^{m} (c_i x - b_i)^2 \]

- Conceptually, the idea generalizes to higher dimensions, but is hard to treat/quantify analytically.
- Adapting the stepsize \( \alpha_k \) to the farout and confusion regions is an important issue.
- Shaping the confusion region is an important issue.
Select index $i_k$ and set

$$x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

Many similarities with incremental subgradient
- Similar stepsize choices
- Similar index selection schemes
- Can be written as

$$x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_{k+1}))$$

where $\tilde{\nabla} f_{i_k}(x_{k+1})$ is a special subgradient at $x_{k+1}$ (index advanced by 1)

Compared to incremental subgradient
- Likely more stable
- May be harder to implement
Select index $i_k$ and set

$$x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

Many similarities with incremental subgradient

- Similar stepsize choices
- Similar index selection schemes
- Can be written as

$$x_{k+1} = P_X (x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_{k+1}))$$

where $\tilde{\nabla} f_{i_k}(x_{k+1})$ is a special subgradient at $x_{k+1}$ (index advanced by 1)

Compared to incremental subgradient

- Likely more stable
- May be harder to implement
Select index $i_k$ and set

$$
x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
$$

Many similarities with incremental subgradient

- Similar stepsize choices
- Similar index selection schemes
- Can be written as

$$
x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_{k+1}))
$$

where $\tilde{\nabla} f_{i_k}(x_{k+1})$ is a special subgradient at $x_{k+1}$ (index advanced by 1)

Compared to incremental subgradient

- Likely more stable
- May be harder to implement
Typical iteration

Choose $i_k \in \{1, \ldots, m\}$ and do a subgradient or a proximal iteration

$$x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \quad \text{or} \quad x_{k+1} = \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

where $\alpha_k$ is a positive stepsize and $\tilde{\nabla}$ denotes (any) subgradient.

- Idea: Use proximal when easy to implement; use subgradient otherwise
- A very flexible implementation
- The proximal iterations still require diminishing $\alpha_k$ for convergence
Incremental Subgradient-Proximal Methods

Typical iteration

Choose $i_k \in \{1, \ldots, m\}$ and do a subgradient or a proximal iteration

$$x_{k+1} = P_X(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \quad \text{or} \quad x_{k+1} = \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

where $\alpha_k$ is a positive stepsize and $\tilde{\nabla}$ denotes (any) subgradient.

- Idea: Use proximal when easy to implement; use subgradient otherwise
- A very flexible implementation
- The proximal iterations still require diminishing $\alpha_k$ for convergence
Under Lipschitz continuity-type assumptions (Nedic and Bertsekas, 2000):

- Convergence to the optimum for **diminishing** stepsize.
- Convergence to a neighborhood of the optimum for **constant** stepsize.
- Faster convergence for randomized index selection (relative to a worst-case cyclic choice).
Convergence Analysis

Under Lipschitz continuity-type assumptions (Nedic and Bertsekas, 2000):

- Convergence to the optimum for **diminishing** stepsize.
- Convergence to a neighborhood of the optimum for **constant** stepsize.
- Faster convergence for randomized index selection (relative to a worst-case cyclic choice).
Convergence Analysis

Under Lipschitz continuity-type assumptions (Nedic and Bertsekas, 2000):

- Convergence to the optimum for **diminishing** stepsize.
- Convergence to a neighborhood of the optimum for **constant** stepsize.
- Faster convergence for randomized index selection (relative to a worst-case cyclic choice).
Outline

1 Incremental Algorithms

2 Aggregated Incremental Algorithms

3 Incremental Augmented Lagrangian Algorithms

4 Incremental Treatment of Constraints

5 Convergence Analysis
Incremental Aggregated Gradient Method

\[ x_{k+1} = P_X \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a "delayed" subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \ i, k. \]

- Key idea: Replace current subgradient components with earlier computed versions
- Only one component subgradient may be computed per iteration
- Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
- Key Work (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
- This is a gradient method with error proportional to the stepsize.
- A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
- Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
\[ x_{k+1} = P_X \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \ i, k. \]

- **Key idea:** Replace current subgradient components with earlier computed versions
  - Only one component subgradient may be computed per iteration
  - Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
  - **Key Work** (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
  - This is a gradient method with error proportional to the stepsize.
  - A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
  - Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
\[ x_{k+1} = P_X \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with \( k - b \leq \ell_i \leq k, \quad \forall \ i, k. \)

- Key idea: Replace current subgradient components with earlier computed versions
- Only one component subgradient may be computed per iteration
- Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
- Key Work (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
- This is a gradient method with error proportional to the stepsize.
- A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
- Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
\[ x_{k+1} = P_X \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \ i, k. \]

Key idea: Replace current subgradient components with earlier computed versions

Only one component subgradient may be computed per iteration

Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)

Key Work (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.

This is a gradient method with error proportional to the stepsize.

A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).

Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
\[ x_{k+1} = P_X \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with
\[
k - b \leq \ell_i \leq k, \quad \forall \ i, k.
\]

- Key idea: Replace current subgradient components with earlier computed versions
- Only one component subgradient may be computed per iteration
- Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
- **Key Work** (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
- This is a gradient method with error proportional to the stepsize.
- A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
- Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
Incremental Aggregated Gradient Method

\[ x_{k+1} = P_x \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed" subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \ i, k. \]

Key idea: Replace current subgradient components with earlier computed versions

Only one component subgradient may be computed per iteration

Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)

Key Work (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.

This is a gradient method with error proportional to the stepsize.

A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).

Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
Incremental Aggregated Gradient Method

\[ x_{k+1} = P_x \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right) \]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a "delayed" subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \ i, k. \]

- Key idea: Replace current subgradient components with earlier computed versions
- Only one component subgradient may be computed per iteration
- Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
- Key Work (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
- This is a gradient method with error proportional to the stepsize.
- A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
- Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
Incremental Aggregated Gradient Method

\[
x_{k+1} = P_x \left( x_k - \alpha_k \sum_{i=1}^{m} \tilde{\nabla} f_i(x_{\ell_i}) \right)
\]

where \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[ k - b \leq \ell_i \leq k, \quad \forall \; i, k. \]

- Key idea: Replace current subgradient components with earlier computed versions
- Only one component subgradient may be computed per iteration
- Proposed for nondifferentiable \( f_i \) and diminishing stepsize by Bertsekas, Nedic, and Borkar (2001)
- **Key Work** (Blatt, Hero, and Gauchman, 2008): Differentiable strongly convex \( f_i \), no constraints, constant stepsize, and linear convergence.
- This is a gradient method with error proportional to the stepsize.
- A fundamentally different convergence mechanism (relies on differentiability and aims at cost function descent (no region of confusion).
- Intense recent activity by many researchers (Gurbuzbalaban, Ozdaglar, Parrilo, 2015).
Select index $i_k$ and set

$$x_{k+1} \in \arg \min_{x \in X} \left\{ f_{i_k}(x) + \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i})'(x - x_k) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

and $\tilde{\nabla} f_i(x_{\ell_i})$ is a “delayed” subgradient of $f_i$ at some earlier iterate $x_{\ell_i}$ with

$$k - b \leq \ell_i \leq k, \quad \forall \; i, k.$$ 

Equivalently,

$$x_{k+1} \in \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - z_k\|^2 \right\},$$

where

$$z_k = x_k - \alpha_k \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i}).$$

If $f$ is differentiable and strongly convex, linear convergence can be shown with constant but sufficiently small $\alpha_k$ (DPB 2015).
Select index \( i_k \) and set

\[
x_{k+1} \in \arg \min_{x \in X} \left\{ f_{i_k}(x) + \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i})' (x - x_k) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

and \( \tilde{\nabla} f_i(x_{\ell_i}) \) is a “delayed” subgradient of \( f_i \) at some earlier iterate \( x_{\ell_i} \) with

\[
k - b \leq \ell_i \leq k, \quad \forall \ i, k.
\]

Equivalently,

\[
x_{k+1} \in \arg \min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - z_k\|^2 \right\},
\]

where

\[
z_k = x_k - \alpha_k \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i}).
\]

If \( f \) is differentiable and strongly convex, linear convergence can be shown with constant but sufficiently small \( \alpha_k \) (DPB 2015).
Select index $i_k$ and set

$$x_{k+1} \in \arg\min_{x \in X} \left\{ f_{i_k}(x) + \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i})' (x - x_k) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

and $\tilde{\nabla} f_i(x_{\ell_i})$ is a “delayed” subgradient of $f_i$ at some earlier iterate $x_{\ell_i}$ with

$$k - b \leq \ell_i \leq k, \quad \forall \, i, k.$$ 

Equivalently,

$$x_{k+1} \in \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - z_k\|^2 \right\},$$

where

$$z_k = x_k - \alpha_k \sum_{i \neq i_k} \tilde{\nabla} f_i(x_{\ell_i}).$$

If $f$ is differentiable and strongly convex, linear convergence can be shown with constant but sufficiently small $\alpha_k$ (DPB 2015).
Outline

1. Incremental Algorithms
2. Aggregated Incremental Algorithms
3. Incremental Augmented Lagrangian Algorithms
4. Incremental Treatment of Constraints
5. Convergence Analysis
Separable Convex Optimization: A Summary

The subgradient method exploits the separable structure (Lagrangian relaxation)
The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization: A Summary

minimize $\sum_{i=1}^{m} f_i(x^i)$ subject to $x^i \in X_i$, $i = 1, \ldots, m$, $\sum_{i=1}^{m} h_i(x^i) = 0$

where $f_i : \mathbb{R}^n_i \mapsto \mathbb{R}$ are convex, $h_i : \mathbb{R}^n_i \mapsto \mathbb{R}^r$ are linear, $X_i \subset \mathbb{R}^n_i$ are closed and convex.

Dual problem decomposes

maximize $\sum_{i=1}^{m} q_i(\lambda)$ subject to $\lambda \in \mathbb{R}^r$

where $q_i$ is a “component” dual function:

$$q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^T h_i(x^i) \right\}$$

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization: A Summary

minimize \( \sum_{i=1}^{m} f_i(x^i) \) \hspace{1em} \text{subject to} \hspace{1em} x^i \in X_i, \ i = 1, \ldots, m, \ \sum_{i=1}^{m} h_i(x^i) = 0

where \( f_i : \mathbb{R}^{n_i} \mapsto \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \mapsto \mathbb{R}^r \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

Dual problem decomposes

maximize \( \sum_{i=1}^{m} q_i(\lambda) \) \hspace{1em} \text{subject to} \hspace{1em} \lambda \in \mathbb{R}^r

where \( q_i \) is a "component" dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization: A Summary

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{m} f_i(x^i) \\
\text{subject to} & \quad x^i \in X_i, \ i = 1, \ldots, m, \\
& \quad \sum_{i=1}^{m} h_i(x^i) = 0
\end{align*}
\]

where \( f_i : \mathbb{R}^n \to \mathbb{R} \) are convex, \( h_i : \mathbb{R}^n_i \to \mathbb{R}^r \) are linear, \( X_i \subset \mathbb{R}^n_i \) are closed and convex.

Dual problem decomposes

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{m} q_i(\lambda) \\
\text{subject to} & \quad \lambda \in \mathbb{R}^r
\end{align*}
\]

where \( q_i \) is a "component" dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \{ f_i(x^i) + \lambda^t h_i(x^i) \}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Separable Convex Optimization: A Summary

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{m} f_i(x^i) \\
\text{subject to} & \quad x^i \in X_i, \ i = 1, \ldots, m, \quad \sum_{i=1}^{m} h_i(x^i) = 0
\end{align*}
\]

where \( f_i : \mathbb{R}^{n_i} \to \mathbb{R} \) are convex, \( h_i : \mathbb{R}^{n_i} \to \mathbb{R}^r \) are linear, \( X_i \subset \mathbb{R}^{n_i} \) are closed and convex.

Dual problem decomposes

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{m} q_i(\lambda) \\
\text{subject to} & \quad \lambda \in \mathbb{R}^r
\end{align*}
\]

where \( q_i \) is a “component” dual function:

\[
q_i(\lambda) = \inf_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^t h_i(x^i) \right\}
\]

- The subgradient method exploits the separable structure (Lagrangian relaxation)
- The proximal algorithm yields the augmented Lagrangian method but destroys the separable structure
- Incremental versions of the proximal algorithm yield incremental augmented Lagrangian methods that exploit the separable structure
Proximal - Augmented Lagrangian Relation

Proximal Algorithm for the Dual Problem

\[
\lambda_{k+1} \in \arg \max_{\lambda \in \mathbb{R}^r} \left\{ \sum_{i=1}^{m} q_i(\lambda) - \frac{1}{2\alpha_k} \|\lambda - \lambda_k\|^2 \right\}
\]

Dualization using Fenchel duality $\rightarrow$ augmented Lagrangian method

Introduce the augmented Lagrangian function

\[
L_\alpha(x, \lambda) = \sum_{i=1}^{m} f_i(x^i) + \lambda' \sum_{i=1}^{m} h_i(x^i) + \frac{\alpha}{2} \left\| \sum_{i=1}^{m} h_i(x^i) \right\|^2
\]

where $\alpha > 0$ is a parameter. For a sequence $\{\alpha_k\}$ and a starting $\lambda_0$, set

\[
x_{k+1} \in \arg \min_{x^i \in X_i, i=1,\ldots,m} L_{\alpha_k}(x, \lambda_k)
\]

Update $\lambda$ according to

\[
\lambda_{k+1} = \lambda_k + \alpha_k \sum_{i=1}^{m} h_i(x_{k+1}^i)
\]

A major flaw: $\min$ of $L_{\alpha_k}(x, l_k)$ is not separable.
Proximal Algorithm for the Dual Problem

\[ \lambda_{k+1} \in \arg \max_{\lambda \in \mathbb{R}^r} \left\{ \sum_{i=1}^{m} q_i(\lambda) - \frac{1}{2\alpha_k} \| \lambda - \lambda_k \|^2 \right\} \]

Dualization using Fenchel duality $\rightarrow$ augmented Lagrangian method

Introduce the augmented Lagrangian function

\[ L_{\alpha}(x, \lambda) = \sum_{i=1}^{m} f_i(x^i) + \lambda' \sum_{i=1}^{m} h_i(x^i) + \frac{\alpha}{2} \left\| \sum_{i=1}^{m} h_i(x^i) \right\|^2 \]

where $\alpha > 0$ is a parameter. For a sequence $\{\alpha_k\}$ and a starting $\lambda_0$, set

\[ x_{k+1} \in \arg \min_{x^i \in X_i, i=1,\ldots,m} L_{\alpha_k}(x, \lambda_k) \]

Update $\lambda$ according to

\[ \lambda_{k+1} = \lambda_k + \alpha_k \sum_{i=1}^{m} h_i(x^i_{k+1}) \]

A major flaw: $\min L_{\alpha_k}(x, l_k)$ is not separable.
At iteration $k$, pick index $i_k$, and set

$$
\lambda_{k+1} \in \arg\max_{\lambda \in \mathbb{R}^r} \left\{ q_{i_k}(\lambda) - \frac{1}{2\alpha_k} \|\lambda - \lambda_k\|^2 \right\}
$$

Dualization using Fenchel duality $\rightarrow$ Incremental augmented Lagrangian method

Pick index $i_k$, and update the single component $x_{i_k}^{i_k}$ according to

$$
x_{k+1}^{i_k} \in \arg\min_{x^{i_k} \in x_{i_k}} \left\{ f_{i_k}(x_{i_k}^{i_k}) + \lambda_k' h_{i_k}(x_{i_k}^{i_k}) + \frac{\alpha_k}{2} \|h_{i_k}(x_{i_k}^{i_k})\|^2 \right\},
$$

while keeping the others unchanged, $x_{k+1}^i = x_k^i$ for all $i \neq i_k$. Update $\lambda$ according to

$$
\lambda_{k+1} = \lambda_k + \alpha_k h_{i_k}(x_{k+1}^{i_k})
$$
Incremental Augmented Lagrangian Method

Incremental Proximal Algorithm for the Dual Problem

At iteration $k$, pick index $i_k$, and set

$$
\lambda_{k+1} \in \arg \max_{\lambda \in \mathbb{R}^r} \left\{ q_{i_k}(\lambda) - \frac{1}{2\alpha_k} \| \lambda - \lambda_k \|^2 \right\}
$$

Dualization using Fenchel duality $\rightarrow$ Incremental augmented Lagrangian method

Pick index $i_k$, and update the single component $x_{i_k}$ according to

$$
x_{k+1}^{i_k} \in \arg \min_{x_{i_k}^{i_k} \in X_{i_k}} \left\{ f_{i_k}(x_{i_k}^{i_k}) + \lambda_{k}^{i_k} h_{i_k}(x_{i_k}^{i_k}) + \frac{\alpha_k}{2} \| h_{i_k}(x_{i_k}^{i_k}) \|^2 \right\},
$$

while keeping the others unchanged, $x_{k+1}^{i} = x_{k}^{i}$ for all $i \neq i_k$. Update $\lambda$ according to

$$
\lambda_{k+1} = \lambda_k + \alpha_k h_{i_k}(x_{k+1}^{i_k})
$$
Incremental Aggregated Proximal Algorithm for the Dual Problem

At iteration $k$, pick index $i_k$, and set

$$
\lambda_{k+1} \in \arg \max_{\lambda \in \mathbb{R}^r} \left\{ q_{i_k}(\lambda) - \frac{1}{2\alpha_k} \| \lambda - z_k \|^2 \right\},
$$

where

$$
z_k = \lambda_k + \alpha_k \sum_{i \neq i_k} \tilde{\nabla}q_i(\lambda_{\ell_i})
$$
Implementation of IAAL

Dualization using Fenchel duality → Incremental aggregated augmented Lagrangian method

- Pick index $i_k$, and update the single component $x_{i_k}^k$ according to

$$
x_{i_k}^{k+1} \in \arg \min_{x_{i_k}^k \in X_{i_k}} \left\{ f_{i_k}(x_{i_k}^k) + \lambda_k' h_{i_k}(x_{i_k}^k) + \frac{\alpha_k}{2} \left\| h_{i_k}(x_{i_k}^k) + \sum_{i \neq i_k} h_i(x_{i_{\ell_i}}^i) \right\|^2 \right\}
$$

while keeping the others unchanged, $x_{i_k}^{k+1} = x_i^i$ for all $i \neq i_k$.

- Update $\lambda$ according to

$$
\lambda_{k+1} = \lambda_k + \alpha_k \left( h_{i_k}(x_{i_k}^{k+1}) + \sum_{i \neq i_k} h_i(x_{i_{\ell_i}}^i) \right)
$$

Here $h_i(x_{i_{\ell_i}}^i)$, $i \neq i_k$, come from earlier iterations.
Implementation of IAAL

Dualization using Fenchel duality \(\rightarrow\) Incremental aggregated augmented Lagrangian method

- Pick index \(i_k\), and update the single component \(x_{i_k}^k\) according to

\[
x_{i_k}^{k+1} \in \arg \min_{x_{i_k}^k \in X_{i_k}} \left\{ f_i(x_{i_k}^k) + \lambda_k' h_i(x_{i_k}^k) + \frac{\alpha_k}{2} \left\| h_k(x_{i_k}^k) + \sum_{i \neq i_k} h_i(x_{i_{\ell_i}}^i) \right\|^2 \right\}
\]

while keeping the others unchanged, \(x_{k+1}^i = x_k^i\) for all \(i \neq i_k\).

- Update \(\lambda\) according to

\[
\lambda_{k+1} = \lambda_k + \alpha_k \left( h_k(x_{k+1}^i) + \sum_{i \neq i_k} h_i(x_{i_{\ell_i}}^i) \right)
\]

Here \(h_i(x_{i_{\ell_i}}^i), i \neq i_k\), come from earlier iterations.
**Comparison with Alternating Direction Methods of Multipliers (ADMM)**

**ADMM Iteration for Separable Problems (DPB 1989)**

Perform a separate augmented Lagrangian minimization over $x^i$, for each $i = 1, \ldots, m$, 

$$x^i_{k+1} \in \arg \min_{x^i \in X_i} \left\{ f_i(x^i) + \lambda^i_k h_i(x^i) + \frac{\alpha}{2} \left\| h_i(x^i) - h_i(x^i_k) + \frac{1}{m} \sum_{j=1}^{m} h_j(x^i_k) \right\|^2 \right\},$$

and then update $\lambda_k$ according to

$$\lambda_{k+1} = \lambda_k + \frac{\alpha}{m} \sum_{i=1}^{m} h_i(x^i_{k+1})$$

**Comparison with Incremental Aggregated Augmented Lagrangian**

- The two methods involve fairly similar operations.
- ADMM has guaranteed convergence for any constant $\alpha$, and under weaker conditions (dual differentiability and strong convexity are not required).
- IAAL has stepsize restrictions.
- At each iteration, all components $x^i$ are updated in ADMM, but a single component $x^i$ is updated in IAAL ($m$ times greater overhead per iteration).
ADMM Iteration for Separable Problems (DPB 1989)

Perform a separate augmented Lagrangian minimization over $x^i$, for each $i = 1, \ldots, m$,

$$x_{k+1}^i \in \arg \min_{x^i \in X_i} \left\{ f_i(x^i) + \lambda_k h_i(x^i) + \frac{\alpha}{2} \left\| h_i(x^i) - h_i(x_k^i) + \frac{1}{m} \sum_{j=1}^{m} h_j(x_k^j) \right\|^2 \right\},$$

and then update $\lambda_k$ according to

$$\lambda_{k+1} = \lambda_k + \frac{\alpha}{m} \sum_{i=1}^{m} h_i(x_{k+1}^i).$$

Comparison with Incremental Aggregated Augmented Lagrangian

- The two methods involve fairly similar operations
- ADMM has guaranteed convergence for any constant $\alpha$, and under weaker conditions (dual differentiability and strong convexity are not required)
- IAAL has stepsize restrictions
- At each iteration, all components $x^i$ are updated in ADMM, but a single component $x^i$ is updated in IAAL ($m$ times greater overhead per iteration)
ADMM Iteration for Separable Problems (DPB 1989)

Perform a separate augmented Lagrangian minimization over $x^i$, for each $i = 1, \ldots, m$,

$$
x_{k+1}^i \in \arg \min_{x^i \in X_i} \left\{ f_i(x^i) + \lambda_k h_i(x^i) + \frac{\alpha}{2} \left\| h_i(x^i) - h_i(x_k^i) + \frac{1}{m} \sum_{j=1}^{m} h_j(x_k^i) \right\|^2 \right\},
$$

and then update $\lambda_k$ according to

$$
\lambda_{k+1} = \lambda_k + \frac{\alpha}{m} \sum_{i=1}^{m} h_i(x_{k+1}^i)
$$

Comparison with Incremental Aggregated Augmented Lagrangian

- The two methods involve fairly similar operations
- ADMM has guaranteed convergence for any constant $\alpha$, and under weaker conditions (dual differentiability and strong convexity are not required)
- IAAL has stepsize restrictions
- At each iteration, all components $x^i$ are updated in ADMM, but a single component $x^i$ is updated in IAAL ($m$ times greater overhead per iteration)
ADMM Iteration for Separable Problems (DPB 1989)

Perform a separate augmented Lagrangian minimization over \( x^i \), for each \( i = 1, \ldots, m \),

\[
x^i_{k+1} \in \arg \min_{x^i \in X_i} \left\{ f_i(x^i) + \lambda_k^i h_i(x^i) + \frac{\alpha}{2} \left\| h_i(x^i) - h_i(x^i_k) + \frac{1}{m} \sum_{j=1}^{m} h_j(x^j_k) \right\|^2 \right\},
\]

and then update \( \lambda_k \) according to

\[
\lambda_{k+1} = \lambda_k + \frac{\alpha}{m} \sum_{i=1}^{m} h_i(x^i_{k+1})
\]

Comparison with Incremental Aggregated Augmented Lagrangian

- The two methods involve fairly similar operations
- ADMM has guaranteed convergence for any constant \( \alpha \), and under weaker conditions (dual differentiability and strong convexity are not required)
- IAAL has stepsize restrictions
- At each iteration, all components \( x^i \) are updated in ADMM, but a single component \( x^i \) is updated in IAAL (\( m \) times greater overhead per iteration)
### ADMM Iteration for Separable Problems (DPB 1989)

Perform a separate augmented Lagrangian minimization over $x^i$, for each $i = 1, \ldots, m$, 

$$
\begin{align*}
x_{k+1}^i & \in \arg \min_{x^i \in X_i} \left\{ f_i(x^i) + \lambda_k^i h_i(x^i) + \frac{\alpha}{2} \left\| h_i(x^i) - h_i(x_k^i) + \frac{1}{m} \sum_{j=1}^{m} h_j(x_k^i) \right\|^2 \right\}, 
\end{align*}
$$

and then update $\lambda_k$ according to

$$
\lambda_{k+1} = \lambda_k + \frac{\alpha}{m} \sum_{i=1}^{m} h_i(x_{k+1}^i)
$$

### Comparison with Incremental Aggregated Augmented Lagrangian

- The two methods involve fairly similar operations
- ADMM has guaranteed convergence for any constant $\alpha$, and under weaker conditions (dual differentiability and strong convexity are not required)
- IAAL has stepsize restrictions
- At each iteration, all components $x^i$ are updated in ADMM, but a single component $x^i$ is updated in IAAL ($m$ times greater overhead per iteration)
Outline

1. Incremental Algorithms
2. Aggregated Incremental Algorithms
3. Incremental Augmented Lagrangian Algorithms
4. Incremental Treatment of Constraints
5. Convergence Analysis
Incremental Methods with Constraint Projection

\[
\text{minimize} \quad \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x \in \bigcap_{\ell=1}^{q} X_{\ell},
\]

where \( f_i : \mathbb{R}^n \mapsto \mathbb{R} \) are convex, and the sets \( X_{\ell} \) are closed and convex.

Incremental constraint projection algorithm

- Choose indexes \( i_k \in \{1, \ldots, m\} \) and \( \ell_k \in \{1, \ldots, q\} \).
- Perform a subgradient iteration or a proximal iteration

\[
x_{k+1} = P_{X_{\ell_k}} (x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \quad \text{or} \quad x_{k+1} = \arg \min_{x \in X_{\ell_k}} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

Connection to feasibility/alternating projection methods.
Incremental Methods with Constraint Projection

minimize \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x \in \bigcap_{\ell=1}^{q} X_\ell,

where \( f_i : \mathbb{R}^n \mapsto \mathbb{R} \) are convex, and the sets \( X_\ell \) are closed and convex.

Incremental constraint projection algorithm

- Choose indexes \( i_k \in \{1, \ldots, m\} \) and \( \ell_k \in \{1, \ldots, q\} \).
- Perform a subgradient iteration or a proximal iteration

\[
x_{k+1} = P_{X_{\ell_k}} (x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \quad \text{or} \quad x_{k+1} = \arg \min_{x \in X_{\ell_k}} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\}
\]

where \( \alpha_k \) is a positive stepsize and \( \tilde{\nabla} \) denotes (any) subgradient.

Connection to feasibility/alternating projection methods.
minimize $\sum_{i=1}^{m} f_i(x)$ subject to $x \in \cap_{\ell=1}^{q} X_\ell$,

where $f_i : \mathbb{R}^n \mapsto \mathbb{R}$ are convex, and the sets $X_\ell$ are closed and convex.

**Incremental constraint projection algorithm**

- Choose indexes $i_k \in \{1, \ldots, m\}$ and $\ell_k \in \{1, \ldots, q\}$.
- Perform a subgradient iteration or a proximal iteration

$$x_{k+1} = P_{X_{\ell_k}} (x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)) \quad \text{or} \quad x_{k+1} = \arg \min_{x \in X_{\ell_k}} \left\{ \frac{1}{2\alpha_k} \| x - x_k \|^2 \right\}$$

where $\alpha_k$ is a positive stepsize and $\tilde{\nabla}$ denotes (any) subgradient.

**Connection to feasibility/alternating projection methods.**
1. Incremental Algorithms
2. Aggregated Incremental Algorithms
3. Incremental Augmented Lagrangian Algorithms
4. Incremental Treatment of Constraints
5. Convergence Analysis
Problem

\[
\text{minimize } \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x \in X = \bigcap_{\ell=1}^{q} X_\ell,
\]

Typical iteration

- Choose indexes \( i_k \in \{1, \ldots, m\} \) and \( \ell_k \in \{1, \ldots, q\} \).
- Set
  \[
  x_{k+1} = P_{X_{\ell_k}} (x_k - \alpha_k \tilde{\nabla} f_{i_k} (\bar{x}_k))
  \]
- \( \bar{x}_k = x_k \) (subgradient iteration) or \( \bar{x} = x_{k+1} \) (proximal iteration).
- \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \) (diminishing stepsize is essential).

Two-way progress

- Progress to feasibility: The projection \( P_{X_{\ell_k}} (\cdot) \).
- Progress to optimality: The “subgradient/proximal” iteration \( x_k - \alpha_k \tilde{\nabla} f_{i_k} (\bar{x}_k) \).
Problem

\begin{equation}
\text{minimize} \quad \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x \in X = \cap_{\ell=1}^{q} X_{\ell},
\end{equation}

Typical iteration

- Choose indexes $i_k \in \{1, \ldots, m\}$ and $\ell_k \in \{1, \ldots, q\}$.
- Set
  \begin{equation}
  x_{k+1} = P_{X_{\ell_k}}(x_k - \alpha_k \tilde{\nabla} f_{i_k}(\tilde{x}_k))
  \end{equation}
- $\tilde{x}_k = x_k$ (subgradient iteration) or $\tilde{x} = x_{k+1}$ (proximal iteration).
- $\sum_{k=0}^{\infty} \alpha_k = \infty$ and $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$ (diminishing stepsize is essential).

Two-way progress

- Progress to feasibility: The projection $P_{X_{\ell_k}}(\cdot)$.
- Progress to optimality: The “subgradient/proximal” iteration $x_k - \alpha_k \tilde{\nabla} f_{i_k}(\tilde{x}_k)$.
**Problem**

\[
\text{minimize } \sum_{i=1}^{m} f_i(x) \quad \text{subject to} \quad x \in X = \bigcap_{\ell=1}^{q} X_\ell,
\]

**Typical iteration**

- Choose indexes \( i_k \in \{1, \ldots, m\} \) and \( \ell_k \in \{1, \ldots, q\} \).
- Set
  \[
x_{k+1} = P_{X_{\ell_k}} (x_k - \alpha_k \tilde{\nabla} f_{i_k}(\bar{x}_k))
\]
- \( \bar{x}_k = x_k \) (subgradient iteration) or \( \bar{x} = x_{k+1} \) (proximal iteration).
- \( \sum_{k=0}^{\infty} \alpha_k = \infty \) and \( \sum_{k=0}^{\infty} \alpha_k^2 < \infty \) (diminishing stepsize is essential).

**Two-way progress**

- **Progress to feasibility**: The projection \( P_{X_{\ell_k}}(\cdot) \).
- **Progress to optimality**: The “subgradient/proximal” iteration \( x_k - \alpha_k \tilde{\nabla} f_{i_k}(\bar{x}_k) \).
Progress to feasibility should be faster than progress to optimality. Gradient stepsizes $\alpha_k$ should be $<<$ than the feasibility stepsize of 1.
Sampling Schemes for Constraint Index $\ell_k$

**Nearly independent sampling**

\[
\inf_{k \geq 0} \text{Prob}(\ell_k = X_\ell \mid \mathcal{F}_k) > 0, \quad \ell = 1, \ldots, q,
\]

where $\mathcal{F}_k$ is the history of the algorithm up to time $k$.

**Cyclic sampling**

Deterministic or random reshuffling every $q$ iterations.

**Most distant constraint sampling**

\[
\ell_k = \arg \max_{\ell = 1, \ldots, q} \| x_k - P_{x_\ell}(x_k) \|
\]

**Markov sampling**

Generate $\ell_k$ as the state of an ergodic Markov chain with states 1, \ldots, $q$. 
Nearly independent sampling

\[ \inf_{k \geq 0} \text{Prob}(\ell_k = X_\ell | \mathcal{F}_k) > 0, \quad \ell = 1, \ldots, q, \]

where \( \mathcal{F}_k \) is the history of the algorithm up to time \( k \).

Cyclic sampling

Deterministic or random reshuffling every \( q \) iterations.

Most distant constraint sampling

\[ \ell_k = \arg \max_{\ell = 1, \ldots, q} \| x_k - P_{X_\ell}(x_k) \| \]

Markov sampling

Generate \( \ell_k \) as the state of an ergodic Markov chain with states 1, \ldots, \( q \).
Sampling Schemes for Constraint Index $\ell_k$

**Nearly independent sampling**

$$\inf_{k \geq 0} \text{Prob}(\ell_k = X_\ell \mid \mathcal{F}_k) > 0, \quad \ell = 1, \ldots, q,$$

where $\mathcal{F}_k$ is the history of the algorithm up to time $k$.

**Cyclic sampling**

Deterministic or random reshuffling every $q$ iterations.

**Most distant constraint sampling**

$$\ell_k = \arg \max_{\ell=1,\ldots,q} \| x_k - P_{X_\ell}(x_k) \|$$

**Markov sampling**

Generate $\ell_k$ as the state of an ergodic Markov chain with states $1, \ldots, q$. 
Sampling Schemes for Constraint Index $\ell_k$

Nearly independent sampling

$$\inf_{k \geq 0} \text{Prob}(\ell_k = X_\ell | F_k) > 0, \quad \ell = 1, \ldots, q,$$

where $F_k$ is the history of the algorithm up to time $k$.

Cyclic sampling

Deterministic or random reshuffling every $q$ iterations.

Most distant constraint sampling

$$\ell_k = \arg \max_{\ell=1,\ldots,q} \| x_k - P_{X_\ell}(x_k) \|$$

Markov sampling

Generate $\ell_k$ as the state of an ergodic Markov chain with states 1, $\ldots$, $q$. 

Bertsekas (M.I.T.)
Random independent uniform sampling

Each index \( i \in \{1, \ldots, m\} \) is chosen with equal probability \( 1/m \), independently of earlier choices.

Cyclic sampling

Deterministic or random reshuffling every \( m \) iterations.

Markov sampling

Generate \( i_k \) as the state of a Markov chain with states \( 1, \ldots, m \), and steady state distribution \( \{1/m, \ldots, 1/m\} \).
Random independent uniform sampling

Each index $i \in \{1, \ldots, m\}$ is chosen with equal probability $1/m$, independently of earlier choices.

Cyclic sampling

Deterministic or random reshuffling every $m$ iterations.

Markov sampling

Generate $i_k$ as the state of a Markov chain with states $1, \ldots, m$, and steady state distribution $\{1/m, \ldots, 1/m\}$. 
### Random independent uniform sampling

Each index \( i \in \{1, \ldots, m\} \) is chosen with equal probability \( 1/m \), independently of earlier choices.

### Cyclic sampling

Deterministic or random reshuffling every \( m \) iterations.

### Markov sampling

Generate \( i_k \) as the state of a Markov chain with states \( 1, \ldots, m \), and steady state distribution \( \{1/m, \ldots, 1/m\} \).
Assuming Lipschitz continuity of the cost, linear regularity of the constraint, and nonemptiness of the optimal solution set, \( \{x_k\} \) converges to some optimal solution \( x^* \) w.p. 1, under any combination of the preceding sampling schemes.

Idea of the convergence proof

There are two convergence processes taking place:

- **Progress towards feasibility**, which is fast (geometric thanks to the linear regularity assumption).
- **Progress towards optimality**, which is slower (because of the diminishing stepsize \( \alpha_k \)).

This two-time scale convergence analysis idea is encoded in a coupled supermartingale convergence theorem, which governs the evolution of two measures of progress:

\[
E[\text{dist}^2(x_k, X)] : \text{Distance to the constraint set, which is fast}
\]

\[
E[\text{dist}^2(x_k, X^*)] : \text{Distance to the optimal solution set, which is slow}
\]
Assuming Lipschitz continuity of the cost, linear regularity of the constraint, and nonemptiness of the optimal solution set, \( \{ x_k \} \) converges to some optimal solution \( x^* \) w.p. 1, under any combination of the preceding sampling schemes.

Idea of the convergence proof

There are two convergence processes taking place:

- **Progress towards feasibility**, which is fast (geometric thanks to the linear regularity assumption).
- **Progress towards optimality**, which is slower (because of the diminishing stepsize \( \alpha_k \)).

This two-time scale convergence analysis idea is encoded in a coupled supermartingale convergence theorem, which governs the evolution of two measures of progress:

\[
E[\text{dist}^2(x_k, X)] : \text{Distance to the constraint set, which is fast}
\]

\[
E[\text{dist}^2(x_k, X^*)] : \text{Distance to the optimal solution set, which is slow}
\]
Assuming Lipschitz continuity of the cost, linear regularity of the constraint, and nonemptiness of the optimal solution set, \( \{x_k\} \) converges to some optimal solution \( x^* \) w.p. 1, under any combination of the preceding sampling schemes.

Idea of the convergence proof

There are two convergence processes taking place:

- **Progress towards feasibility**, which is fast (geometric thanks to the linear regularity assumption).
- **Progress towards optimality**, which is slower (because of the diminishing stepsize \( \alpha_k \)).

This two-time scale convergence analysis idea is encoded in a coupled supermartingale convergence theorem, which governs the evolution of two measures of progress

\[
\mathbb{E} [ \text{dist}^2(x_k, X)] : \text{Distance to the constraint set, which is fast}
\]

\[
\mathbb{E} [ \text{dist}^2(x_k, X^*)] : \text{Distance to the optimal solution set, which is slow}
\]
Incremental methods exhibit interesting convergence behavior, and can lead to great efficiencies for large-sum cost functions.

- Incremental proximal methods enhance reliability and can be combined seamlessly with incremental gradient/subgradient methods.
- Incremental proximal methods when dualized yield incremental augmented Lagrangian methods that can take advantage of constrained problem separability.
- Constraint projection variants provide flexibility and enlarge the range of potential applications.
- Incremental methods are amenable to distributed asynchronous implementation.
Concluding Remarks

- Incremental methods exhibit interesting convergence behavior, and can lead to great efficiencies for large-sum cost functions.

- Incremental proximal methods enhance reliability and can be combined seamlessly with incremental gradient/subgradient methods.

- Incremental proximal methods when dualized yield incremental augmented Lagrangian methods that can take advantage of constrained problem separability.

- Constraint projection variants provide flexibility and enlarge the range of potential applications.

- Incremental methods are amenable to distributed asynchronous implementation.
Incremental methods exhibit interesting convergence behavior, and can lead to great efficiencies for large-sum cost functions.

Incremental proximal methods enhance reliability and can be combined seamlessly with incremental gradient/subgradient methods.

Incremental proximal methods when dualized yield incremental augmented Lagrangian methods that can take advantage of constrained problem separability.

Constraint projection variants provide flexibility and enlarge the range of potential applications.

Incremental methods are amenable to distributed asynchronous implementation.
Concluding Remarks

- Incremental methods exhibit interesting convergence behavior, and can lead to great efficiencies for large-sum cost functions.
- Incremental proximal methods enhance reliability and can be combined seamlessly with incremental gradient/subgradient methods.
- Incremental proximal methods when dualized yield incremental augmented Lagrangian methods that can take advantage of constrained problem separability.
- Constraint projection variants provide flexibility and enlarge the range of potential applications.
- Incremental methods are amenable to distributed asynchronous implementation.
Incremental methods exhibit interesting convergence behavior, and can lead to great efficiencies for large-sum cost functions.

Incremental proximal methods enhance reliability and can be combined seamlessly with incremental gradient/subgradient methods.

Incremental proximal methods when dualized yield incremental augmented Lagrangian methods that can take advantage of constrained problem separability.

Constraint projection variants provide flexibility and enlarge the range of potential applications.

Incremental methods are amenable to distributed asynchronous implementation.
Thank you!