Convergence Rates of Inexact Proximal-Gradient Methods for Convex Optimization

Mark Schmidt, Nicolas Le Roux, Francis Bach

INRIA - SIERRA Project - Team
Laboratoire d’Informatique de l’École Normale Supérieure
(CNRS/ENS/UMR 8548)

December 2011
Outline

1. Motivation and Overview of Contribution
2. Related work on Inexact Algorithms
3. Convergence Rates for Convex Optimization
4. Numerical Experiments on a Structured Sparsity Problem
Composite Convex Optimization Problems

- We consider composite optimization problems

\[
\min_{\mathbf{x} \in \mathbb{R}^d} f(\mathbf{x}) := g(\mathbf{x}) + h(\mathbf{x}),
\]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.
Composite Convex Optimization Problems

- We consider composite optimization problems

\[
\min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x),
\]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.

- Typically, \( g \) is a data-fitting term, and \( h \) is a regularizer,

\[
\min_{x \in \mathbb{R}^d} \sum_{i=1}^{N} l_i(x) + \lambda r(x)
\]

- The most well-studied example is \( \ell_1 \)-regularized least squares,

\[
\min_{x \in \mathbb{R}^d} \| Ax - b \|^2 + \lambda \| x \|_1.
\]
Fast Convergence Rates of Proximal-Gradient Methods

- We consider composite optimization problems

\[
\min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x),
\]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.
Fast Convergence Rates of Proximal-Gradient Methods

- We consider composite optimization problems

\[ \min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x), \]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.

- Convergence rates of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>( O(1/\sqrt{k}) )</td>
<td>( O(1/k) )</td>
</tr>
</tbody>
</table>
Motivation and Overview of Contribution
Related work on Inexact Algorithms
Convergence Rates for Convex Optimization
Numerical Experiments on a Structured Sparsity Problem

Fast Convergence Rates of Proximal-Gradient Methods

- We consider composite optimization problems

\[
\min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x),
\]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.

- Convergence rates of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>( O(1/\sqrt{k}) )</td>
<td>( O(1/k) )</td>
</tr>
<tr>
<td>Proximal-Gradient</td>
<td>( O(1/k) )</td>
<td>( O((1 − \gamma)^k) )</td>
</tr>
</tbody>
</table>
We consider composite optimization problems

\[ \min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x), \]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.

**Convergence rates** of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>( O(1/\sqrt{k}) )</td>
<td>( O(1/k) )</td>
</tr>
<tr>
<td>Proximal-Gradient</td>
<td>( O(1/k) )</td>
<td>( O((1 - \gamma)^k) )</td>
</tr>
<tr>
<td>Accelerated Proximal-Gradient</td>
<td>( O(1/k^2) )</td>
<td>( O((1 - \sqrt{\gamma})^k) )</td>
</tr>
</tbody>
</table>
Fast Convergence Rates of Proximal-Gradient Methods

- We consider **composite** optimization problems

\[
\min_{x \in \mathbb{R}^d} f(x) := g(x) + h(x),
\]

where \( g \) and \( h \) are convex but \( h \) is non-smooth.

- **Convergence rates** of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>( O(1/\sqrt{k}) )</td>
<td>( O(1/k) )</td>
</tr>
<tr>
<td>Proximal-Gradient</td>
<td>( O(1/k) )</td>
<td>( O((1 - \gamma)^k) )</td>
</tr>
<tr>
<td>Accelerated Proximal-Gradient</td>
<td>( O(1/k^2) )</td>
<td>( O((1 - \sqrt{\gamma})^k) )</td>
</tr>
</tbody>
</table>

- Proximal-gradient methods have the **same convergence rates** as [accelerated] gradient methods for smooth optimization.

[Beck & Teboulle, 2009, Nesterov, 2007]
Overview of the Basic Gradient Method

- We want to solve a smooth optimization problem,

\[
\min_{x \in \mathbb{R}^d} g(x).
\]
Overview of the Basic Gradient Method

- We want to solve a smooth optimization problem,
  \[ \min_{x \in \mathbb{R}^d} g(x). \]

- At iteration \( x_k \) we use a \textit{quadratic upper bound} on \( g \),
  \[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \| x - x_k \|^2. \]
Overview of the Basic Gradient Method

- We want to solve a smooth optimization problem,

\[
\min_{x \in \mathbb{R}^d} g(x).
\]

- At iteration \(x_k\) we use a \textit{quadratic upper bound} on \(g\),

\[
x_{k+1} = \arg \min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2.
\]

- We can equivalently write this as the quadratic optimization

\[
x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \frac{1}{2} \|x - (x_k - \alpha_k g'(x_k))\|^2.
\]
Overview of the Basic Gradient Method

- We want to solve a smooth optimization problem,
  \[ \min_{x \in \mathbb{R}^d} \; g(x). \]

- At iteration \( x_k \) we use a *quadratic upper bound* on \( g \),
  \[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \; g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2. \]

- We can equivalently write this as the quadratic optimization
  \[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \frac{1}{2} \|x - (x_k - \alpha_k g'(x_k))\|^2. \]

- The solution is the gradient algorithm:
  \[ x_{k+1} = x_k - \alpha_k g'(x_k). \]
Overview of the Basic Proximal-Gradient Method

- We want to solve a smooth optimization problem,
  \[
  \min_{x \in \mathbb{R}^d} g(x).
  \]

- At iteration $x_k$ we use a quadratic upper bound on $g$,
  \[
  x_{k+1} = \arg \min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \|x - x_k\|^2.
  \]

- We can equivalently write this as the quadratic optimization
  \[
  x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \frac{1}{2} \|x - (x_k - \alpha_k g'(x_k))\|^2.
  \]

- The solution is the gradient algorithm:
  \[
  x_{k+1} = x_k - \alpha_k g'(x_k).
  \]
Overview of the Basic Proximal-Gradient Method

- We want to solve a composite optimization problem,

\[ \min_{x \in \mathbb{R}^d} \ g(x) + h(x). \]

- At iteration \( x_k \) we use a quadratic upper bound on \( g \),

\[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \ g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \| x - x_k \|^2. \]

- We can equivalently write this as the quadratic optimization

\[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \ \frac{1}{2} \| x - (x_k - \alpha_k g'(x_k)) \|^2. \]

- The solution is the gradient algorithm:

\[ x_{k+1} = x_k - \alpha_k g'(x_k). \]
Overview of the Basic Proximal-Gradient Method

- We want to solve a composite optimization problem,

\[ \min_{x \in \mathbb{R}^d} g(x) + h(x). \]

- At iteration \( x_k \) we use a quadratic upper bound on \( g \),

\[ x_{k+1} = \arg\min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \| x - x_k \|^2 + h(x). \]

- We can equivalently write this as the quadratic optimization

\[ x_{k+1} = \arg\min_{x \in \mathbb{R}^d} \frac{1}{2} \| x - (x_k - \alpha_k g'(x_k)) \|^2. \]

- The solution is the gradient algorithm:

\[ x_{k+1} = x_k - \alpha_k g'(x_k). \]
Overview of the Basic \textit{Proximal}-Gradient Method

- We want to solve a \textit{composite} optimization problem,
  \[
  \min_{x \in \mathbb{R}^d} g(x) + h(x).
  \]
- At iteration $x_k$ we use a \textit{quadratic upper bound} on $g$,
  \[
  x_{k+1} = \arg \min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \| x - x_k \|^2 + h(x).
  \]
- We can equivalently write this as the \textit{proximal} optimization
  \[
  x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \frac{1}{2} \| x - (x_k - \alpha_k g'(x_k)) \|^2 + \alpha_k h(x).
  \]
- The solution is the gradient algorithm:
  \[
  x_{k+1} = x_k - \alpha_k g'(x_k).
  \]
Overview of the Basic *Proximal*-Gradient Method

- We want to solve a composite optimization problem,
  \[ \min_{x \in \mathbb{R}^d} g(x) + h(x). \]

- At iteration \( x_k \) we use a quadratic upper bound on \( g \),
  \[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} g(x_k) + \langle g'(x_k), x - x_k \rangle + \frac{1}{2\alpha_k} \| x - x_k \|^2 + h(x). \]

- We can equivalently write this as the proximal optimization
  \[ x_{k+1} = \arg \min_{x \in \mathbb{R}^d} \frac{1}{2} \| x - (x_k - \alpha_k g'(x_k)) \|^2 + \alpha_k h(x). \]

- The solution is the proximal-gradient algorithm:
  \[ x_{k+1} = \text{prox}_{\alpha_k} \left[ x_k - \alpha_k g'(x_k) \right]. \]
Special case of Projected-Gradient Methods

- Projected-gradient methods are a special case:

\[ h(x) = \begin{cases} 
0 & \text{if } x \in C \\
\infty & \text{if } x \notin C. \end{cases} \]
Special case of Projected-Gradient Methods

- **Projected-gradient** methods are a special case:

\[ h(x) = \begin{cases} 
0 & \text{if } x \in C \\
\infty & \text{if } x \notin C. 
\end{cases} \]
Special case of Projected-Gradient Methods

Projected-gradient methods are a special case:

\[ h(x) = \begin{cases} 
0 & \text{if } x \in C \\
\infty & \text{if } x \notin C.
\end{cases} \]
Special case of Projected-Gradient Methods

- **Projected-gradient** methods are a special case:

\[
h(x) = \begin{cases} 
0 & \text{if } x \in \mathcal{C} \\
\infty & \text{if } x \notin \mathcal{C}.
\end{cases}
\]
**Special case of Projected-Gradient Methods**

- **Projected-gradient** methods are a special case:

\[
  h(x) = \begin{cases} 
0 & \text{if } x \in C \\
\infty & \text{if } x \notin C.
\end{cases}
\]
Special case of Iterative Soft-Thresholding Methods

- **Iterative Soft-Thresholding** methods are a special case:

\[ h(x) = \lambda \| x \|_1. \]
Special case of Iterative Soft-Thresholding Methods

- **Iterative Soft-Thresholding** methods are a special case:
  \[ h(x) = \lambda \|x\|_1. \]

- In this case \( \text{prox}_{\alpha_k} [x]_i \) shrinks \( |x_i| \) by \( \min\{\alpha_k \lambda, |x_i|\} \)
Special case of Iterative Soft-Thresholding Methods

- **Iterative Soft-Thresholding** methods are a special case:

  \[ h(x) = \lambda \|x\|_1. \]

- In this case \( \text{prox}_{\alpha_k} [x]_i \) shrinks \( |x_i| \) by \( \min\{\alpha_k \lambda, |x_i|\} \)
Iterative Soft-Thresholding methods are a special case:

\[ h(x) = \lambda \|x\|_1. \]

In this case \( \text{prox}_{\alpha_k} [x]_i \) shrinks \(|x_i|\) by \( \min\{\alpha_k \lambda, |x_i|\} \)
Iterative Soft-Thresholding methods are a special case:

\[ h(x) = \lambda \|x\|_1. \]

In this case \( \text{prox}_{\alpha_k} [x]_i \) shrinks \( |x_i| \) by \( \min \{ \alpha_k \lambda, |x_i| \} \)
Special case of Iterative Soft-Thresholding Methods

- **Iterative Soft-Thresholding** methods are a special case:
  \[ h(x) = \lambda \| x \|_1. \]

- In this case \( \text{prox}_{\alpha_k} [x]_i \) shrinks \( |x_i| \) by \( \min\{\alpha_k \lambda, |x_i|\} \)
Accelerated (Proximal-)Gradient Methods

- Proximal-gradient methods have the same convergence rates as gradient methods for smooth optimization.
Accelerated (Proximal-)Gradient Methods

- Proximal-gradient methods have the same convergence rates as gradient methods for smooth optimization.
- But for smooth problems accelerated gradient methods have faster rates [Nesterov, 1983]:

  \[
  x_{k+1} = y_k - \alpha_k g'(y_k), \\
  y_{k+1} = x_{k+1} + \beta_k (x_{k+1} - x_k).
  \]
Accelerated (Proximal-)Gradient Methods

- Proximal-gradient methods have the same convergence rates as gradient methods for smooth optimization.
- But for smooth problems accelerated gradient methods have faster rates [Nesterov, 1983]:

\[
x_{k+1} = y_k - \alpha_k g'(y_k),
\]
\[
y_{k+1} = x_{k+1} + \beta_k (x_{k+1} - x_k).
\]

- For composite problems accelerated proximal-gradient methods have these same rates:

\[
x_{k+1} = \text{prox}_{\alpha_k} [y_k - \alpha_k g'(y_k)],
\]
\[
y_{k+1} = x_{k+1} + \beta_k (x_{k+1} - x_k).
\]
For what problems can we apply proximal-gradient methods?
For what problems can we apply proximal-gradient methods?

- We can efficiently compute the proximity operator for:
  1. $\ell_1$-Regularization.
  2. Group $\ell_1$-Regularization.
  3. Lower and upper bound constraints.
  4. Hyper-plane and half-space constraints.
  5. Simplex constraints.
  6. Euclidean cone constraints.
For what problems can we apply proximal-gradient methods?

We can efficiently compute the proximity operator for:

1. $\ell_1$-Regularization.
2. Group $\ell_1$-Regularization.
3. Lower and upper bound constraints.
4. Hyper-plane and half-space constraints.
5. Simplex constraints.
6. Euclidean cone constraints.

But for many problems we \textit{can not efficiently compute the proximity operator}.
Inexact Proximal-Gradient Methods

- We can efficiently approximate the proximity operator for:
  1. Total-variation regularization and generalizations like the graph-guided fused-LASSO.
  2. Nuclear-norm regularization and other regularizers on the singular values of matrices.
  3. Overlapping group $\ell_1$-regularization with general groups.
  5. Combinations of simple functions.
We can efficiently **approximate** the proximity operator for:

1. Total-variation regularization and generalizations like the graph-guided fused-LASSO.
2. Nuclear-norm regularization and other regularizers on the singular values of matrices.
3. Overlapping group $\ell_1$-regularization with general groups.
5. Combinations of simple functions.
Many recent works use **inexact proximal-gradient** methods:

- Cai et al. [2010], Liu & Ye [2010], Schmidt & Murphy [2010], Barbero & Sra [2011], Fadili & Peyré [2011], Ma et al. [2011].
Many recent works use **inexact proximal-gradient** methods:

- Cai et al. [2010], Liu & Ye [2010], Schmidt & Murphy [2010], Barbero & Sra [2011], Fadili & Peyré [2011], Ma et al. [2011].

Our question:

- Can **inexact proximal-gradient** methods achieve the fast convergence rates?

Our contribution:

- Inexact proximal-gradient methods can achieve the fast convergence rates, if the errors are appropriately controlled.

- We also allow an error in the gradient, and compare various inexact strategies on a structured sparsity problem.
Summary of Contribution

Many recent works use **inexact proximal-gradient** methods:

- Cai et al. [2010], Liu & Ye [2010], Schmidt & Murphy [2010], Barbero & Sra [2011], Fadili & Peyré [2011], Ma et al. [2011].

Our question:

- Can **inexact proximal-gradient** methods achieve the fast convergence rates?

Our contribution:

- **Inexact proximal-gradient methods can achieve the fast convergence rates, if the errors are appropriately controlled.**
Summary of Contribution

Many recent works use **inexact proximal-gradient** methods:

- Cai et al. [2010], Liu & Ye [2010], Schmidt & Murphy [2010], Barbero & Sra [2011], Fadili & Peyré [2011], Ma et al. [2011].

Our question:

- Can **inexact proximal-gradient methods** achieve the fast convergence rates?

Our contribution:

- **Inexact proximal-gradient methods can achieve the fast convergence rates, if the errors are appropriately controlled.**

We also allow an **error in the gradient**, and compare various inexact strategies on a structured sparsity problem.
Outline

1 Motivation and Overview of Contribution

2 Related work on Inexact Algorithms
   - Stochastic Proximal-Gradient Methods
   - Inexact Projected-Gradient Methods
   - Inexact Proximal-Gradient Methods

3 Convergence Rates for Convex Optimization

4 Numerical Experiments on a Structured Sparsity Problem
Prior Work: Stochastic Proximal-Gradient Methods

Proximal-gradient methods with zero-mean random error:
[Duchi & Singer, 2009, Langford et al., 2009]
Prior Work: Stochastic Proximal-Gradient Methods

Proximal-gradient methods with zero-mean random error:
[Duchi & Singer, 2009, Langford et al., 2009]

- Same slow convergence rates as sub-gradient methods.
Prior Work: Stochastic Proximal-Gradient Methods

Proximal-gradient methods with zero-mean random error:
[Duchi & Singer, 2009, Langford et al., 2009]
- Same slow convergence rates as sub-gradient methods.

This is different than our scenario:
- We consider a decreasing sequence of errors.
- This leads to faster convergence rates.
- Analysis applies for deterministic (and adversarial) errors.
Prior Work: Projected-Gradient Methods (Fixed Error)

Projected-gradient methods with **fixed error magnitude**:

Prior Work: Projected-Gradient Methods (Fixed Error)

Projected-gradient methods with **fixed error magnitude**:

- Fast convergence rate but **only up to some fixed error level**.
Prior Work: Projected-Gradient Methods (Fixed Error)

Projected-gradient methods with **fixed error magnitude**:

- Fast convergence rate but **only up to some fixed error level**.

We allow the error magnitude to change on every iteration:
- We achieve **convergence to an optimal solution**.
- We allow a **larger error in early iterations**.
Prior Work: Projected-Gradient Methods (Variable Error)

Projected-gradient methods with decreasing error magnitude:

- These works either do not consider acceleration, assume an exact projection, or require that the domain is compact.
Prior Work: Projected-Gradient Methods (Variable Error)

Projected-gradient methods with decreasing error magnitude:

- These works either do not consider acceleration, assume an exact projection, or require that the domain is compact.

In contrast:
- We do not have these restrictions.
- We generalize to proximal-gradient methods.
Prior Work: Proximal-Gradient Methods

Inexact proximal-gradient methods are globally convergent under:
Inexact proximal-gradient methods are globally convergent under:

- Closedness and descent assumptions [Patriksson, 1995].
- Summability of the sequence of errors [Combettes, 2004].
Inexact proximal-gradient methods are globally convergent under:

- Closedness and descent assumptions [Patriksson, 1995].
- Summability of the sequence of errors [Combettes, 2004].

But there was no prior work on convergence rates.
Outline

1 Motivation and Overview of Contribution

2 Related work on Inexact Algorithms

3 Convergence Rates for Convex Optimization
   - Problem Setting, Algorithm, and Assumptions
   - Analysis for Convex Objectives
   - Analysis for Strongly Convex Objectives

4 Numerical Experiments on a Structured Sparsity Problem
Problem Setting and Algorithm

- We consider the problem

\[
\min_{x \in \mathbb{R}^d} g(x) + h(x).
\]

- The \textit{basic} proximal-gradient method uses

\[
x_k = \text{prox}_{\alpha_k} [x_{k-1} - \alpha_k g'(x_{k-1})].
\]
Problem Setting and Algorithm

• We consider the problem

\[ \min_{x \in \mathbb{R}^d} g(x) + h(x). \]

• The basic proximal-gradient method uses

\[ x_k = \text{prox}_{\alpha_k} [x_{k-1} - \alpha_k g'(x_{k-1})]. \]

• The accelerated proximal-gradient method uses

\[ x_k = \text{prox}_{\alpha_k} [y_{k-1} - \alpha_k g'(y_{k-1})], \]

where

\[ y_k = x_k + \beta_k (x_k - x_{k-1}), \]

and the sequence \( \{\beta_k\} \) is chosen to give a faster rate.
Central Assumptions and Notation

- In all our results we assume:
  - \( g \) is **convex** and \( g' \) is **\( L \)-Lipschitz continuous**, 
    \[
    \|g'(x) - g'(y)\| \leq L\|x - y\|, \quad \forall x, y.
    \]
  (if **twice-differentiable**, equivalent to \( 0 \preceq g''(x) \preceq LI, \quad \forall x \))
Central Assumptions and Notation

In all our results we assume:

- $g$ is convex and $g'$ is $L$-Lipschitz continuous,

$$\|g'(x) - g'(y)\| \leq L\|x - y\|, \forall x, y.$$  

(if twice-differentiable, equivalent to $0 \leq g''(x) \leq L I, \forall x$)

- $h$ is a lower semi-continuous proper convex function
  (includes all real-valued functions, and indicator functions).
Central Assumptions and Notation

- In all our results we assume:
  - $g$ is convex and $g'$ is $L$-Lipschitz continuous,
    \[
    \|g'(x) - g'(y)\| \leq L \|x - y\|, \forall x, y.\]
    (if twice-differentiable, equivalent to $0 \leq g''(x) \leq LI, \forall x$)
  - $h$ is a lower semi-continuous proper convex function (includes all real-valued functions, and indicator functions).
  - $g + h$ attains its minimum at a certain $x_*$.
  - The step size $\alpha_k$ is set to $1/L$. 

Mark Schmidt, Nicolas Le Roux, Francis Bach
Central Assumptions and Notation

- In all our results we assume:
  - $g$ is convex and $g'$ is $L$-Lipschitz continuous,
    $$||g'(x) - g'(y)|| \leq L||x - y||, \forall x, y.$$
    (if twice-differentiable, equivalent to $0 \leq g''(x) \leq LI, \forall x$)
  - $h$ is a lower semi-continuous proper convex function (includes all real-valued functions, and indicator functions).
  - $g + h$ attains its minimum at a certain $x^*$.
  - The step size $\alpha_k$ is set to $1/L$.
  - The gradient $g'$ is computed with an error $e_k$.
  - $x_k$ is an $\varepsilon_k$-approximate solution of the proximity operator,
    $$\frac{L}{2}||x_k - y||^2 + h(x_k) \leq \varepsilon_k + \min_{x \in \mathbb{R}^d} \left\{ \frac{L}{2}||x - y||^2 + h(x) \right\}.$$  
    (we can use a duality gap to check this condition)
Fast Convergence Rates of Proximal-Gradient Methods

- Convergence rates of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>$O(1/\sqrt{k})$</td>
<td>$O(1/k)$</td>
</tr>
<tr>
<td>Proximal-Gradient</td>
<td>$O(1/k)$</td>
<td>$O((1 - \mu/L)^k)$</td>
</tr>
<tr>
<td>Accelerated Proximal-Gradient</td>
<td>$O(1/k^2)$</td>
<td>$O((1 - \sqrt{\mu/L})^k)$</td>
</tr>
</tbody>
</table>
Fast Convergence Rates of Proximal-Gradient Methods

- Convergence rates of methods for composite optimization:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Convex</th>
<th>Strongly Convex</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Gradient</td>
<td>$O(1/\sqrt{k})$</td>
<td>$O(1/k)$</td>
</tr>
<tr>
<td>Proximal-Gradient</td>
<td>$O(1/k)$</td>
<td>$O((1 - \mu/L)^k)$</td>
</tr>
<tr>
<td>Accelerated Proximal-Gradient</td>
<td>$O(1/k^2)$</td>
<td>$O((1 - \sqrt{\mu/L})^k)$</td>
</tr>
</tbody>
</table>

- We give conditions on the sequences of gradient errors $\{e_k\}$ and proximity errors $\{\varepsilon_k\}$ that preserve these rates.
Convexity - Basic Proximal-Gradient Method

**Proposition 1.** If the sequences $\{\|e_k\|\}$ and $\{\sqrt{\varepsilon_k}\}$ are summable then the basic proximal-gradient method achieves

$$f \left( \frac{1}{k} \sum_{i=1}^{k} x_i \right) - f(x_*) = O(1/k).$$
Proposition 1. If the sequences \( \{\|e_k\|\} \) and \( \{\sqrt{\varepsilon_k}\} \) are summable then the basic proximal-gradient method achieves

\[
f \left( \frac{1}{k} \sum_{i=1}^{k} x_i \right) - f(x_*) = O(1/k).
\]

- E.g., \( \|e_k\| \) and \( \sqrt{\varepsilon_k} \) could decrease as \( O(1/k^{1+\delta}) \) for \( \delta > 0 \).
**Proposition 1.** If the sequences $\{\|e_k\|\}$ and $\{\sqrt{\varepsilon_k}\}$ are summable then the basic proximal-gradient method achieves

$$f\left(\frac{1}{k} \sum_{i=1}^{k} x_i\right) - f(x_*) = O(1/k).$$

- E.g., $\|e_k\|$ and $\sqrt{\varepsilon_k}$ could decrease as $O(1/k^{1+\delta})$ for $\delta > 0$.
- If they decrease as $O(1/k)$, then we get $O((\log k)^2/k)$.
  (see the paper for the constant factor)
- Bound also holds for the best iterate.
Proposition 2. If the sequences \( \{k \parallel e_k \parallel\} \) and \( \{k \sqrt{\varepsilon_k}\} \) are summable then the accelerated proximal-gradient method achieves

\[
f(x_k) - f(x^*) = O\left(\frac{1}{k^2}\right),
\]

with \( \beta_k = (k - 1)/(k + 2) \).
Proposition 2. If the sequences \( \{k\|e_k\|\} \) and \( \{k\sqrt{\varepsilon_k}\} \) are summable then the accelerated proximal-gradient method achieves

\[
f(x_k) - f(x^*) = O(1/k^2),
\]

with \( \beta_k = (k - 1)/(k + 2) \).

- E.g., \( \|e_k\| \) and \( \sqrt{\varepsilon_k} \) could decrease as \( O(1/k^{2+\delta}) \) for \( \delta > 0 \).
Proposition 2. If the sequences \( \{ k \|e_k\| \} \) and \( \{ k \sqrt{\epsilon_k} \} \) are summable then the accelerated proximal-gradient method achieves

\[
f(x_k) - f(x_*) = O(1/k^2),
\]

with \( \beta_k = (k - 1)/(k + 2) \).

- E.g., \( \|e_k\| \) and \( \sqrt{\epsilon_k} \) could decrease as \( O(1/k^{2+\delta}) \) for \( \delta > 0 \).
- If they decrease as \( O(1/k^2) \), then we get \( O((\log k)^2/k^2) \).
- Our analysis indicates the accelerated method is more sensitive to errors.
We also consider the case where \( g \) is strongly convex.
We also consider the case where $g$ is strongly convex.

A function $g$ is strongly convex if the function

$$g(x) - \mu \|x\|^2,$$

is convex for some $\mu > 0$.

For twice-differentiable functions, equivalent to $g''(x) \geq \mu I, \forall x$. 
We also consider the case where $g$ is strongly convex.

A function $g$ is strongly convex if the function

$$g(x) - \mu \|x\|^2,$$

is convex for some $\mu > 0$.

For twice-differentiable functions, equivalent to $g''(x) \geq \mu I, \forall x$.

Here, we can obtain exponential rates.
Proposition 3. If the sequences $\{||e_k||\}$ and $\{\sqrt{\varepsilon_k}\}$ are in $O(\rho^k)$ for $\rho < (1 - \mu/L)$ then the basic proximal-gradient method achieves

$$||x_k - x_*|| = O((1 - \mu/L)^k).$$
Proposition 3. If the sequences $\{|e_k|\}$ and $\{|\sqrt{\varepsilon_k}|\}$ are in $O(\rho^k)$ for $\rho < (1 - \mu/L)$ then the basic proximal-gradient method achieves

$$\|x_k - x_*\| = O((1 - \mu/L)^k).$$

- If they converge with $\rho > (1 - \mu/L)$, the rate is $O(\rho^k)$.
- If they converge with $\rho = (1 - \mu/L)$, the rate is $O(k(1 - \mu/L)^k)$. 
Proposition 4. If the sequences $\{||e_k||^2\}$ and $\{\varepsilon_k\}$ are in $O(\rho^k)$ for $\rho < (1 - \sqrt{\mu/L})$ then the accelerated proximal-gradient method achieves

$$f(x_k) - f(x_*) = O((1 - \sqrt{\mu/L})^k),$$

with $\beta_k = (1 - \sqrt{\mu/L})/(1 + \sqrt{\mu/L})$. 
Proposition 4. If the sequences \( \{\|e_k\|^2\} \) and \( \{\epsilon_k\} \) are in \( O(\rho^k) \) for \( \rho < (1 - \sqrt{\mu/L}) \) then the accelerated proximal-gradient method achieves

\[
f(x_k) - f(x_*) = O((1 - \sqrt{\mu/L})^k),
\]

with \( \beta_k = (1 - \sqrt{\mu/L})/(1 + \sqrt{\mu/L}) \).

We also obtain a bound on the iterates because

\[
\frac{\mu}{2} \|x_k - x_*\|^2 \leq f(x_k) - f(x_*).
\]
Outline

1 Motivation and Overview of Contribution

2 Related work on Inexact Algorithms

3 Convergence Rates for Convex Optimization

4 Numerical Experiments on a Structured Sparsity Problem
   - Experimental Set-Up
   - Experiments Results
   - Discussion and Summary
CUR-like factorization with the $\ell_2$-norm

We consider the factorization of Mairal et al. [2011] to approximate a matrix $W$ using a subsets of rows and columns:

$$\min_X \frac{1}{2} \|W - WXW\|_F^2 + \lambda_{\text{row}} \sum_{i=1}^{n_r} \|X^i\|_p + \lambda_{\text{col}} \sum_{j=1}^{n_c} \|X_j\|_p.$$
CUR-like factorization with the $\ell_2$-norm

We consider the factorization of Mairal et al. [2011] to approximate a matrix $W$ using a subsets of rows and columns:

$$\min_X \frac{1}{2} \|W - WXW\|_F^2 + \lambda_{\text{row}} \sum_{i=1}^{n_r} \|X^i\|_p + \lambda_{\text{col}} \sum_{j=1}^{n_c} \|X_j\|_p.$$

- For appropriate $p$, yields sparse rows and sparse columns.
- Previous work used $p = \infty$, since there is no known exact algorithm for $p = 2$. 
CUR-like factorization with the $\ell_2$-norm

We consider the factorization of Mairal et al. [2011] to approximate a matrix $W$ using a subsets of rows and columns:

$$\min_{X} \frac{1}{2} \| W - WXW \|_F^2 + \lambda_{\text{row}} \sum_{i=1}^{n_r} \| X^i \|_p + \lambda_{\text{col}} \sum_{j=1}^{n_c} \| X_j \|_p.$$

- For appropriate $p$, yields sparse rows and sparse columns.
- Previous work used $p = \infty$, since there is no known exact algorithm for $p = 2$.
- We use the proximal-Dykstra algorithm to compute an approximate proximity operator with $p = 2$.
- Duality gap ensures $\varepsilon_k$-optimality of approximate proximity.
Comparison against a fixed prox solution accuracy

Using an optimal $\varepsilon_k$ sequence compared to a fixed precision for the approximate proximity:

![Graph showing convergence rates with different $\varepsilon_k$ sequences and comparison to $1/k^3$ sequence.](image-url)
Comparison against a fixed number of prox iterations

Using an optimal $\varepsilon_k$ sequence compared to running a fixed number of proximal iterations:

![Graph showing convergence rates with different objective values and number of proximal iterations.

Mark Schmidt, Nicolas Le Roux, Francis Bach

Convergence Rates of Inexact Proximal-Gradient Methods
Comparison of different prox accuracy decays

Using different $\varepsilon_k$ sequences ($1/k^3$ has optimal rate):
Discussion

- Inexact proximal-gradient methods **may be useful in other applications**: total-variation or nuclear-norm regularization.
- Our analysis also allows errors in the gradient: undirected graphical models, kernel methods, and SDPs.
Inexact proximal-gradient methods may be useful in other applications: total-variation or nuclear-norm regularization.

Our analysis also allows errors in the gradient: undirected graphical models, kernel methods, and SDPs.

We would like to handle an unknown $L$ and $\mu$.

We would like to adaptively update $\|e_k\|$ and $\varepsilon_k$.

We would like to analyze proximal-Newton methods.
Inexact proximal-gradient methods may be useful in other applications: *total-variation or nuclear-norm regularization*. Our analysis also allows errors in the gradient: *undirected graphical models, kernel methods, and SDPs*. We would like to handle an unknown $L$ and $\mu$. We would like to adaptively update $||e_k||$ and $\varepsilon_k$. We would like to analyze proximal-Newton methods. Villa et al. [2011] and Jiang et al. [2011] have independently analyzed accelerated proximal-gradient methods (convex $g$).
Summary

- Proximal-gradient methods are appealing because of their good theoretical and empirical convergence rates.
Summary

- Proximal-gradient methods are appealing because of their good theoretical and empirical convergence rates.
- But, they require the calculation of the proximity operator.
Summary

- Proximal-gradient methods are appealing because of their good theoretical and empirical convergence rates.
- But, they require the calculation of the proximity operator.
- Many authors have recently applied these methods under an inexact proximity operator.
Summary

- Proximal-gradient methods are appealing because of their good theoretical and empirical convergence rates.
- But, they require the calculation of the proximity operator.
- Many authors have recently applied these methods under an inexact proximity operator.
- We show that the convergence rates are preserved if the inexactness is appropriately controlled.