Non-Smooth Optimization

Jason Hartford (with slides from Mark Schmidt)

October 2015

Where we're at...

- We've seen optimisation is hard, but we can use gradient methods to solve high-dimensional problems
- Nesterov-style and Newton-like methods allow better performance.

Where we're at...

- We've seen optimisation is hard, but we can use gradient methods to solve high-dimensional problems
- Nesterov-style and Newton-like methods allow better performance.
- To achieve linear convergence rates we made strong assumptions:
  - You can go through the entire dataset on every iteration.
  - The objective is smooth/unconstrained.

Where we're at...

- We've seen optimisation is hard, but we can use gradient methods to solve high-dimensional problems
- Nesterov-style and Newton-like methods allow better performance.
- To achieve linear convergence rates we made strong assumptions:
  - ~~You can go through the entire dataset on every iteration.~~
  - The objective is smooth/unconstrained.
- Juliette showed us how we could use stochastic sub-gradient methods to relax this.
- Mohammed showed how we could relax this and still achieve linear convergence using SAG / SVRG

Where we're at...

- We've seen optimisation is hard, but we can use gradient methods to solve high-dimensional problems
- Nesterov-style and Newton-like methods allow better performance.
- To achieve linear convergence rates we made strong assumptions:
  - ~~You can go through the entire dataset on every iteration.~~
  - The objective is smooth/unconstrained. Today!
- Juliette showed us how we could use stochastic sub-gradient methods to relax this.
- Mohammed showed how we could relax this and still achieve linear convergence using SAG / SVRG

Outline

Motivating example: Sparse Regularization

- Consider ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda \|x\|_1$$

- Regularizes and encourages sparsity in x

Motivating example: Sparse Regularization

- Consider ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda\|x\|_1$$

- Regularizes and encourages sparsity in x
- The objective is non-differentiable when any $x_i = 0$.

Motivating example: Sparse Regularization

- Consider ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda\|x\|_1$$

- Regularizes and encourages sparsity in x
- The objective is non-differentiable when any $x_i = 0$.
- More generally: the regularized empirical risk minimization problem:

$$\underbrace{\min_{x \in \mathbb{R}^P} \frac{1}{N}\sum_{i=1}^{N} L(x, a_i, b_i)}_{\text{data fitting term}} \quad + \quad \underbrace{\lambda r(x)}_{\text{regularizer}}$$

- Often, regularizer r is used to encourage sparsity pattern in x.
- Subgradient methods are optimal (slow) black-box methods.

Motivating example: Sparse Regularization

- Consider ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda \|x\|_1$$

- Regularizes and encourages sparsity in x
- The objective is non-differentiable when any $x_i = 0$.
- More generally: the regularized empirical risk minimization problem:

$$\min_{x \in \mathbb{R}^P} \frac{1}{N} \sum_{i=1}^{N} L(x, a_i, b_i) \quad + \quad \lambda r(x)$$

$$\text{data fitting term} \quad + \quad \text{regularizer}$$

- Often, regularizer r is used to encourage sparsity pattern in x.
- Subgradient methods are optimal (slow) black-box methods.
- Are there faster methods for specific non-smooth problems?

Outline

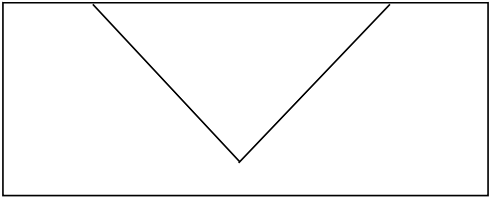
Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.

Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

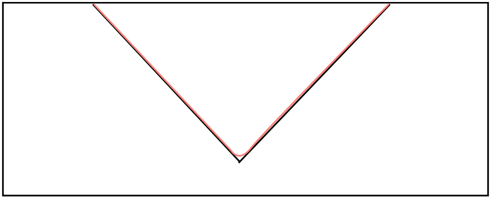
$$|x| \approx \sqrt{x^2 + \nu}.$$



Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$



Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$

- Smooth approximation to the max function:

$$\max\{a, b\} \approx \log(\exp(a) + \exp(b))$$

- Smooth approximation to the hinge loss:

$$\max\{0, 1 - x\} \approx \begin{cases} 0 & x \geq 1 \\ 1 - x^2 & t < x < 1 \\ (1-t)^2 + 2(1-t)(t-x) & x \leq t \end{cases}$$

Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$

- Smooth approximation to the max function:

$$\max\{a, b\} \approx \log(\exp(a) + \exp(b))$$

- Smooth approximation to the hinge loss:

$$\max\{0, 1 - x\} \approx \begin{cases} 0 & x \geq 1 \\ 1 - x^2 & t < x < 1 \\ (1-t)^2 + 2(1-t)(t-x) & x \leq t \end{cases}$$

- Generic smoothing strategy: strongly-convex regularization of convex conjugate [Nesterov, 2005].

Discussion of Smoothing Approach

- Nesterov [2005] shows that:
  - Gradient method on smoothed problem has $O(1/\sqrt{t})$ subgradient rate.
  - Accelerated gradient method has faster $O(1/t)$ rate.

Discussion of Smoothing Approach

- Nesterov [2005] shows that:
  - Gradient method on smoothed problem has $O(1/\sqrt{t})$ subgradient rate.
  - Accelerated gradient method has faster $O(1/t)$ rate.
- No results showing improvement in stochastic case.
- In practice:
  - Slowly decrease level of smoothing (often difficult to tune).
  - Use faster algorithms like L-BFGS, SAG, or SVRG.

Outline

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x f(x) + \lambda \|x\|_1,$$

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x f(x) + \lambda \|x\|_1,$$

is equivalent to the problem

$$\min_{x^+ \geq 0, x^- \geq 0} \quad f(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x f(x) + \lambda \|x\|_1,$$

is equivalent to the problem

$$\min_{x^+ \geq 0, x^- \geq 0} \quad f(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

or the problems

$$\min_{-y \leq x \leq y} \quad f(x) + \lambda \sum_i y_i, \qquad \min_{\|x\|_1 \leq \gamma} \quad f(x) + \lambda\gamma$$

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x f(x) + \lambda \|x\|_1,$$

is equivalent to the problem

$$\min_{x^+ \geq 0, x^- \geq 0} \quad f(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

or the problems

$$\min_{-y \leq x \leq y} \quad f(x) + \lambda \sum_i y_i, \qquad \min_{\|x\|_1 \leq \gamma} \quad f(x) + \lambda\gamma$$

- These are smooth objective with 'simple' constraints.

$$\min_{x \in \mathcal{C}} f(x).$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^{t+1} = \underset{y}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha_t} \|y - x^t\|^2 \right\}.$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^{t+1} = \underset{y}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T(y - x^t) + \frac{1}{2\alpha_t}\|y - x^t\|^2 \right\}.$$

- Consider minimizing subject to simple constraints:

$$x^{t+1} = \underset{y \in \mathcal{C}}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T(y - x^t) + \frac{1}{2\alpha_t}\|y - x^t\|^2 \right\}.$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^{t+1} = \underset{y}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha_t} \|y - x^t\|^2 \right\}.$$

- Consider minimizing subject to simple constraints:

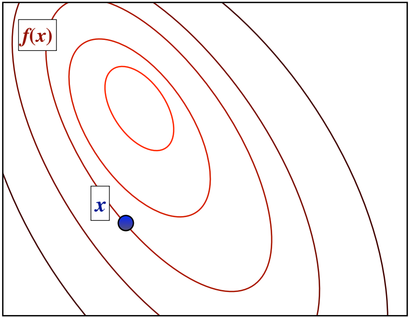
$$x^{t+1} = \underset{y \in \mathcal{C}}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha_t} \|y - x^t\|^2 \right\}.$$

- Called projected gradient algorithm:

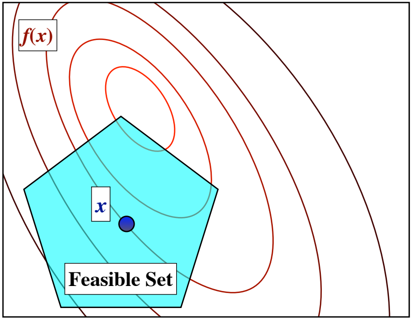
$$x_t^{GD} = x^t - \alpha_t \nabla f(x^t),$$

$$x^{t+1} = \underset{y \in \mathcal{C}}{\operatorname{argmin}} \left\{ \|y - x_t^{GD}\| \right\},$$

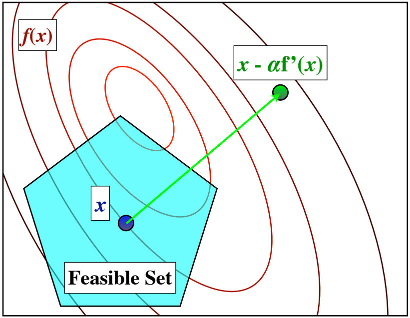
Gradient Projection



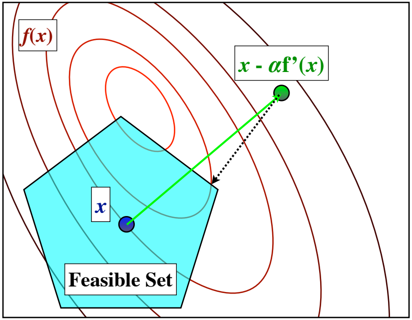
Gradient Projection



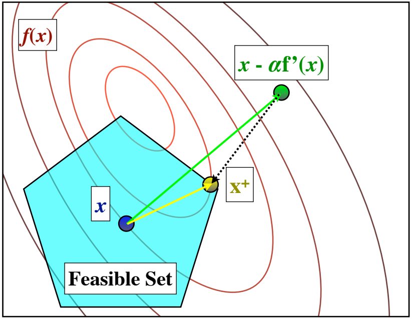
Gradient Projection



Gradient Projection



Gradient Projection



Projection Onto Simple Sets

Projections onto simple sets:

- Bound constraints ($l \leq x \leq u$)
- Small number of linear equalities/inequalities. ($a^T x = b$ or $a^T x \leq b$)
- Norm-balls and norm-cones ($\|x\| \leq \tau$ or $\|x\| \leq x_0$).
- Probability simplex ($x \geq 0, \sum_i x_i = 1$).
- Intersection of disjoint simple sets.

We can solve large instances of problems with these constraints.

Projection Onto Simple Sets

Projections onto simple sets:

- Bound constraints ($l \leq x \leq u$)
- Small number of linear equalities/inequalities. ($a^T x = b$ or $a^T x \leq b$)
- Norm-balls and norm-cones ($\|x\| \leq \tau$ or $\|x\| \leq x_0$).
- Probability simplex ($x \geq 0, \sum_i x_i = 1$).
- Intersection of disjoint simple sets.

We can solve large instances of problems with these constraints.

Intersection of non-disjoint simple sets: Dykstra's algorithm.

Discussion of Projected Gradient

- Projected gradient has same rate as gradient method!

Discussion of Projected Gradient

- Projected gradient has same rate as gradient method!
- Can do many of the same tricks (i.e. line-search, acceleration, Barzilai-Borwein, SAG, SVRG).

Discussion of Projected Gradient

- Projected gradient has same rate as gradient method!
- Can do many of the same tricks (i.e. line-search, acceleration, Barzilai-Borwein, SAG, SVRG).
- Projected Newton needs expensive projection under $\|\cdot\|_{H_t}$:
  - Two-metric projection methods are efficient Newton-like strategy for bound constraints.
  - Inexact Newton methods allow Newton-like like strategy for optimizing costly functions with simple constraints.

Outline

Proximal-Gradient Method

- Proximal-gradient generalizes projected-gradient for

$$\min_x f(x) + r(x),$$

where f is smooth but r is a general convex function.

Proximal-Gradient Method

- Proximal-gradient generalizes projected-gradient for

$$\min_x f(x) + r(x),$$

where f is smooth but r is a general convex function.

- Consider the update:

$$x^{t+1} = \operatorname*{argmin}_y \left\{ f(x^t) + \nabla f(x^t)^T(y - x^t) + \frac{1}{2\alpha}\|y - x^t\|^2 + r(y) \right\}$$

- Applies proximity operator of r to gradient descent on f:

$$x_t^{GD} = x^t - \alpha_t \nabla f(x_t),$$

$$x^{t+1} = \operatorname*{argmin}_y \left\{ \frac{1}{2}\|y - x_t^{GD}\|^2 + \alpha r(y) \right\},$$

- Convergence rates are still the same as for minimizing f.

Proximal-Gradient Method

- How do we derive that?

$$
\begin{aligned}
x^{t+1} &= \underset{y}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha}\|y - x^t\|^2 + r(y) \right\} \\
&= \underset{y}{\operatorname{argmin}} \left\{ < \alpha \nabla f(x^t), (y - x^t) > + \frac{1}{2}\|y - x^t\|^2 + \alpha r(y) \right\} \\
&= \underset{y}{\operatorname{argmin}} \{ < \alpha \nabla f(x^t), (y - x^t) > + \frac{1}{2}\|y - x^t\|^2 + \alpha r(y) \\
&\qquad + \frac{\alpha^2}{2}\|\nabla f(x^k)\|^2 - \frac{\alpha^2}{2}\|\nabla f(x^k)\|^2 \} \\
&= \underset{y}{\operatorname{argmin}} \left\{ \frac{1}{2}\|y - \underbrace{(x^t - \alpha \nabla f(x^t))}_{x^{GD}}\|^2 + \alpha r(y) \right\}
\end{aligned}
$$

Proximal Operator, Iterative Soft Thresholding

- The proximal operator is the solution to

$$\mathrm{prox}_r[y] = \underset{x \in \mathbb{R}^P}{\mathrm{argmin}} \; \frac{1}{2}\|x - y\|^2 + r(x).$$

Proximal Operator, Iterative Soft Thresholding

- The proximal operator is the solution to

$$\text{prox}_r[y] = \underset{x \in \mathbb{R}^P}{\text{argmin}} \ \frac{1}{2}\|x - y\|^2 + r(x).$$

- For L1-regularization, we obtain iterative soft-thresholding:

$$x^{t+1} = \text{softThresh}_{\alpha\lambda}[x^t - \alpha \nabla f(x^t)].$$

Proximal Operator, Iterative Soft Thresholding

- The proximal operator is the solution to

$$\text{prox}_r[y] = \underset{x \in \mathbb{R}^P}{\text{argmin}} \ \frac{1}{2}\|x - y\|^2 + r(x).$$

- For L1-regularization, we obtain iterative soft-thresholding:

$$x^{t+1} = \text{softThresh}_{\alpha\lambda}[x^t - \alpha \nabla f(x^t)].$$

- Example with $\lambda = 1$:

Input	Threshold	Soft-Threshold
$\begin{bmatrix} 0.6715 \\ -1.2075 \\ 0.7172 \\ 1.6302 \\ 0.4889 \end{bmatrix}$		

Proximal Operator, Iterative Soft Thresholding

- The proximal operator is the solution to

$$\text{prox}_r[y] = \underset{x \in \mathbb{R}^P}{\text{argmin}} \; \frac{1}{2}\|x - y\|^2 + r(x).$$

- For L1-regularization, we obtain iterative soft-thresholding:

$$x^{t+1} = \text{softThresh}_{\alpha\lambda}[x^t - \alpha \nabla f(x^t)].$$

- Example with $\lambda = 1$:

Input	Threshold	Soft-Threshold
$\begin{bmatrix} 0.6715 \\ -1.2075 \\ 0.7172 \\ 1.6302 \\ 0.4889 \end{bmatrix}$	$\begin{bmatrix} 0 \\ -1.2075 \\ 0 \\ 1.6302 \\ 0 \end{bmatrix}$	

Proximal Operator, Iterative Soft Thresholding

- The proximal operator is the solution to

$$\text{prox}_r[y] = \underset{x \in \mathbb{R}^P}{\text{argmin}} \ \frac{1}{2}\|x - y\|^2 + r(x).$$

- For L1-regularization, we obtain iterative soft-thresholding:

$$x^{t+1} = \text{softThresh}_{\alpha\lambda}[x^t - \alpha \nabla f(x^t)].$$

- Example with $\lambda = 1$:

$$\begin{array}{ccc} \text{Input} & \text{Threshold} & \text{Soft-Threshold} \\ \begin{bmatrix} 0.6715 \\ -1.2075 \\ 0.7172 \\ 1.6302 \\ 0.4889 \end{bmatrix} & \begin{bmatrix} 0 \\ -1.2075 \\ 0 \\ 1.6302 \\ 0 \end{bmatrix} & \begin{bmatrix} 0 \\ -0.2075 \\ 0 \\ 0.6302 \\ 0 \end{bmatrix} \end{array}$$

Exact Proximal-Gradient Methods

- For what problems can we apply these methods?

Exact Proximal-Gradient Methods

- For what problems can we apply these methods?
- We can efficiently compute the proximity operator for:
  1. L1-Regularization.
  2. Group ℓ_1-Regularization.

Exact Proximal-Gradient Methods

- For what problems can we apply these methods?
- We can efficiently compute the proximity operator for:
    1. L1-Regularization.
    2. Group ℓ_1-Regularization.
    3. Lower and upper bounds.
    4. Small number of linear constraint.
    5. Probability constraints.
    6. A few other simple regularizers/constraints.

Exact Proximal-Gradient Methods

- For what problems can we apply these methods?
- We can efficiently compute the proximity operator for:
  1. L1-Regularization.
  2. Group ℓ_1-Regularization.
  3. Lower and upper bounds.
  4. Small number of linear constraint.
  5. Probability constraints.
  6. A few other simple regularizers/constraints.
- Can solve these non-smooth/constrained problems as fast as smooth/unconstrained problems!

Exact Proximal-Gradient Methods

- For what problems can we apply these methods?
- We can efficiently compute the proximity operator for:
  1. L1-Regularization.
  2. Group ℓ_1-Regularization.
  3. Lower and upper bounds.
  4. Small number of linear constraint.
  5. Probability constraints.
  6. A few other simple regularizers/constraints.
- Can solve these non-smooth/constrained problems as fast as smooth/unconstrained problems!
- We can again do many of the same tricks (line-search, acceleration, Barzilai-Borwein, two-metric subgradient-projection, inexact proximal operators, inexact proximal Newton, SAG, SVRG).

Inexact Proximal-Gradient Methods

- What about problems where we can not efficiently compute the proximity operator?

Inexact Proximal-Gradient Methods

- What about problems where we can not efficiently compute the proximity operator?
- 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 l1 -regularization with general groups.
  4. Positive semi-definite cone.
  5. Combinations of simple functions.

Inexact Proximal-Gradient Methods

- What about problems where we can not efficiently compute the proximity operator?
- 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 l1 -regularization with general groups.
  4. Positive semi-definite cone.
  5. Combinations of simple functions.
- Can still achieve the fast convergence rates, if the errors are appropriately controlled.

Summary

- No black-box method can beat subgradient methods
- For most objectives, you can beat subgradient methods.

Summary

- No black-box method can beat subgradient methods
- For most objectives, you can beat subgradient methods.
- You just need a long list of tricks:
  - Smoothing.
  - Projected-gradient.
  - Proximal-gradient.
  - Proximal-Newton.