Algorithms for L1 minimization

Table of Contents

Table of Contents

Subdifferential

- The subdifferential of a convex function $F : \mathbb{R}^N \to (-\infty, \infty]$ at a point $x \in \mathbb{R}^N$ is defined by

$$\partial F(x) = \{v \in \mathbb{R}^N : F(z) \geq F(x) + \langle v, z - x \rangle \ \textit{for all}\ z \in \mathbb{R}^N\}$$

- The elements of $\partial F(x)$ are called subgradients of F at x
- $\partial |x| = \begin{cases} \{sgn(x)\} & if\, x \neq 0 \\ [-1, 1] & if\, x = 0 \end{cases}$
- A vector x is a minimum of a convex function F if and only if $0 \in \partial F(x)$

Table of Contents

Fundamental Concepts

- Objective : $x^\sharp = \underset{x\in\mathbb{R}^N}{argmin}\,\|x\|_1$ subject to Ax=y, $A \in \mathbb{R}^{m\times N}$ and $y \in \mathbb{R}^m$
- Basis pursuit denoising : $x_\lambda = \underset{x\in\mathbb{R}^N}{argmin}\,\frac{1}{2}\|Ax - y\|_2^2 + \lambda\|x\|_1$
- $\lim_{\lambda\to 0^+} x_\lambda = x^\sharp$
- Iteratively construct λ_n and x_n such that $\lambda_n \to 0^+$ and $x_n = x_{\lambda_n}$
  $\to$ in this way, x_n will converge to $x^\sharp$
- Let $F_{\lambda_n} = \frac{1}{2}\|Ax - y\|_2^2 + \lambda_n\|x\|_1$
- $x_n = \underset{x\in\mathbb{R}^N}{argmin}\,F_{\lambda_n}(x) \Leftrightarrow 0 \in \partial F_{\lambda_n}(x_n) = A^*(Ax_n - y) + \lambda_n\partial\|x_n\|_1$
  - $(A^*(Ax_n - y))_\ell = -\lambda_n sgn((x_n)_\ell)$, if $(x_n)_\ell \neq 0$ - (1)
  - $|(A^*(Ax_n - y))_\ell| \leq \lambda_n$, if $(x_n)_\ell = 0$ - (2)

Algorithms

- Let $c^{(j)} = A^*(Ax_{j-1} - y)$
- Initialization :
  1. $x_0 = 0 \Rightarrow \lambda_0 = \|A^* y\|_\infty$
  2. $\ell_1 = \underset{\ell \in [N]}{argmax} |c_\ell^{(1)}|$, $S_1 = \{\ell_1\}$
- Step 1 :
  1. compute $d^{(1)} \Rightarrow d_{\ell_1}^{(1)} = \frac{sgn((A^*y)_{\ell_1})}{\|a_{\ell_1}\|_2^2} = \frac{-sgn(c_{\ell_1}^{(1)})}{\|a_{\ell_1}\|_2^2}$, $d_\ell = 0$, $\ell \notin S_1$
  2. $x_1 = x_0 + \gamma_1 d^{(1)} = \gamma_1 d^{(1)}$
  3. $\lambda_1 = \lambda_0 - \gamma_1$

  Up to now, we have $(A^*(Ax_1 - y))_\ell = (\lambda_0 - \gamma_1) sgn(c_{\ell_1}^{(1)})$

Algorithms

- Step 1 :

  $(A^*(Ax_1 - y))_\ell = -(\lambda_0 - \gamma_1) sgn((x_1)_\ell)$, $\ell \in S_1$ - (1)

  $|(A^*(Ax_1 - y))_\ell| \leq (\lambda_0 - \gamma_1)$, $\ell \notin S_1$ - (2)

  should be satisfied

  (1) is satisfied

  4. $\gamma_1 = \min_{\ell \notin S_1} \{ \frac{\lambda_0 + c_\ell^{(1)}}{1 - (A^* A d^{(1)})_\ell}, \frac{\lambda_0 - c_\ell^{(1)}}{1 + (A^* A d^{(1)})_\ell} \} \Rightarrow$ to satisfy (2)
  5. $\ell_2 = \underset{\ell \notin S_1}{argmin} \{ \frac{\lambda_0 + c_\ell^{(1)}}{1 - (A^* A d^{(1)})_\ell}, \frac{\lambda_0 - c_\ell^{(1)}}{1 + (A^* A d^{(1)})_\ell} \}$
  6. $S_2 = \{\ell_1, \ell_2\}$

Algorithms

- Step j=2,3,... :
  1. compute $d^{(j)} \Rightarrow A^*_{S_j} A_{S_j} d^{(j)}_{S_j} = -sgn(c^{(j)}_{S_j})$, $d_\ell = 0$, $\ell \notin S_j$
  2. $x_j = x_{j-1} + \gamma_j d^{(j)}$
  3. $\lambda_j = \lambda_{j-1} - \gamma_j$

  Up to now, we have $(A^*(Ax_j - y))_\ell = (\lambda_{j-1} - \gamma_j) sgn(c^{(j)}_\ell)$, $\ell \in S_j$

  $(A^*(Ax_j - y))_\ell = -(\lambda_{j-1} - \gamma_j) sgn((x_j)_\ell)$, $\ell \in S_j$ - (1)

  $|(A^*(Ax_j - y))_\ell| \leq (\lambda_{j-1} - \gamma_j)$, $\ell \notin S_j$ - (2)

  should be satisfied

Algorithms

- Step j=2,3,... :
  4. $\gamma_-^{(j)} = \min_{\ell \in S_j, d_\ell^{(j)} \neq 0} \{-(x_{j-1})_\ell / d_\ell^{(j)}\} \Rightarrow$ to satisfy (1)

     $\gamma_+^{(j)} = \min_{\ell \notin S_j} \{\frac{\lambda_{j-1}+c_\ell^{(j)}}{1-(A^*Ad^{(j)})_\ell}, \frac{\lambda_{j-1}-c_\ell^{(j)}}{1+(A^*Ad^{(j)})_\ell}\} \Rightarrow$ to satisfy (2)

     $\gamma = \min\{\gamma_-^{(j)}, \gamma_+^{(j)}\}$
  5. $\ell_-^{(j)} = \mathit{argmin}_{\ell \in S_j, d_\ell^{(j)} \neq 0} \{-(x_{j-1})_\ell / d_\ell^{(j)}\}$

     $\ell_+^{(j)} = \mathit{argmin}_{\ell \notin S_j} \{\frac{\lambda_{j-1}+c_\ell^{(j)}}{1-(A^*Ad^{(j)})_\ell}, \frac{\lambda_{j-1}-c_\ell^{(j)}}{1+(A^*Ad^{(j)})_\ell}\}$
  6. if $\ell^{(j)} = \mathit{argmin}\{\gamma_-^{(j)}, \gamma_+^{(j)}\} = \ell_-^{(j)} \Rightarrow S_{j+1} = S_j \backslash \{\ell_-^{(j)}\}$

     if $\ell^{(j)} = \mathit{argmin}\{\gamma_-^{(j)}, \gamma_+^{(j)}\} = \ell_+^{(j)} \Rightarrow S_{j+1} = S_j \cup \{\ell_+^{(j)}\}$

Analysis

- Assume the minimizer of the origin ℓ_1-minimization problem is unique and the minimizer $\ell^{(j)}$ in each step is unique

  $\Rightarrow$ The algorithm stops when $\lambda_j = \|c^{(j+1)}\|_\infty = 0$, i.e., when the residual vanishes, and it outputs $x^\sharp = x_j$
- It is observed empirically in sparse recovery problems that the homotopy method merely removes elements from the active set

  $\rightarrow$ if we do not consider $\gamma_-^{(j)}$, then homotopy method reduces to the LARS (least angle regression) algorithm
- The homotopy and LARS methods are very efficient when the solution is very sparse. However, they only apply to the real case

Table of Contents

Table of Contents

Pseudo Inverse

$$A = U\Sigma V^* = \sum_{j=1}^{r} \sigma_j(A) u_j v_j^*, \; A \in \mathbb{C}^{m\times N} \text{ is rank r}$$

$$A^\dagger = V\Sigma^{-1}U^* = \sum_{j=1}^{r} \sigma_j^{-1}(A) v_j u_j^*$$

- $A^\dagger$ is rank r
- If A is invertible $\Rightarrow A^\dagger = A^{-1}$
- $\sigma_{max}(A^\dagger) = \|A^\dagger\|_{2\to 2} = \sigma_r^{-1}(A)$
- If $A^*A \in \mathbb{C}^{n\times n}$ is invertible $(m \geq n) \Rightarrow A^\dagger = (A^*A)^{-1}A^*$
- If $AA^* \in \mathbb{C}^{m\times m}$ is invertible $(n \geq m) \Rightarrow A^\dagger = A^*(AA^*)^{-1}$

Least Squares Problem 1

- Objective : $\underset{x}{minimize} \|Ax - y\|_2$, $A \in \mathbb{C}^{m \times n}$, $m \geq n$, full rank n
- Method 1 :

  Project y to range of A

  $$\langle y - Ax, Ax \rangle = 0 \Rightarrow \langle A^* y - A^* Ax, x \rangle = 0$$
  $$\Rightarrow A^* y = A^* Ax \therefore x = (A^* A)^{-1} A^* y = A^\dagger y$$
- Method 2 :

  $$\underset{x}{argmin} \|Ax - y\|_2 = \underset{x}{argmin} \langle A^* Ax, x \rangle - 2\langle Ax, y \rangle$$
  $$\Rightarrow 2A^* Ax - 2A^* y = 0 \Rightarrow x = (A^* A)^{-1} A^* y = A^\dagger y$$

Least Squares Problem 2

- Objective : $\underset{x}{minimize}\,\|x\|_2$ subject to $Ax = y$
- Method :

  $x^\sharp = \underset{x}{argmin}\,\|x\|_2^2 + \lambda^T(Ax - y) \Rightarrow x^\sharp = -\frac{1}{2}A^*\lambda$

  $\because Ax^\sharp = y \therefore -\frac{1}{2}AA^*\lambda = y \Rightarrow \lambda = -2(AA^*)^{-1}y \Rightarrow x^\sharp = A^\dagger y$

Table of Contents

- Objective : $\underset{x\in\mathbb{C}^N}{minimize}\|x\|_1$ subject to Ax=y, $A\in\mathbb{C}^{m\times N}$ $m\le N$
- Key : $|t| = |t|^2/|t|$ for $t\neq 0$

$\rightarrow$ naive idea : $\underset{x}{minimize}\sum_{j=1}^{N}|x_j|^2|x_j^\sharp|^{-1}$ subject to Ax=y

$\Rightarrow$ advantage : we can minimize a quadractic function

disadvantage :

  - $x^\sharp$ is unknown
  - $x^\sharp$ is sparse $\Rightarrow$ inverse will becomes infinity

- Mature objective function :

$$J(x, w, \epsilon) = \frac{1}{2}[\sum_{j=1}^{N} |x_j|^2 w_j + \sum_{j=1}^{N}(\epsilon^2 w_j + w_j^{-1})]$$

$\Rightarrow$ substitute the role of $x^\sharp$ with w_j

$\rightarrow$ When $x^\sharp = 0,\ w_j = |x_j^\sharp|^{-1} \rightarrow \infty$

$\Rightarrow$ we use $\epsilon^2 w_j$ to regularize w_j from being too large while also using the regularization term w_j^{-1} to prevent w_j from being too small

Iteratively reweighted least squares (IRLS)

Input: $\mathbf{A} \in \mathbb{C}^{m \times N}$, $\mathbf{y} \in \mathbb{C}^m$.
Parameter: $\gamma > 0$, $s \in [N]$.
Initialization: $\mathbf{w}^0 = [1, 1, \ldots, 1]^\top \in \mathbb{R}^N$, $\varepsilon_0 = 1$.
Iteration: repeat until $\varepsilon_n = 0$ or a stopping criterion is met at $n = \bar{n}$:

$$\mathbf{x}^{n+1} := \underset{\mathbf{z} \in \mathbb{C}^N}{\operatorname{argmin}} \mathcal{J}(\mathbf{z}, \mathbf{w}^n, \varepsilon_n) \quad \text{subject to } \mathbf{A}\mathbf{z} = \mathbf{y}, \tag{IRLS$_1$}$$

$$\varepsilon_{n+1} := \min\{\varepsilon_n, \gamma\, (\mathbf{x}^{n+1})^*_{s+1}\}, \tag{IRLS$_2$}$$

$$\mathbf{w}^{n+1} := \underset{\mathbf{w} > 0}{\operatorname{argmin}} \mathcal{J}(\mathbf{x}^{n+1}, \mathbf{w}, \varepsilon_{n+1}). \tag{IRLS$_3$}$$

Output: A solution $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$ of $\mathbf{A}\mathbf{x} = \mathbf{y}$, approximating the sparsest one.

- $IRLS_1$: $x^{n+1} = \underset{z}{argmin}\, J(z, w^n, \epsilon_n)$ subject to Az=y

  $\Rightarrow x^{n+1} = \underset{z}{argmin}\, \frac{1}{2}[\sum_{j=1}^{N} |z_j|^2 w_j^n]$ subject to Az=y

  Let $D_{W^n} = diag[w_1^n, w_2^n, \cdots, w_n^n]$ and make a substitution $x = D_{W^n}^{1/2} z$

  $\Rightarrow D_{W^n}^{1/2} x^{n+1} = \underset{x}{argmin}\, \|x\|_2$ subject to $A D_{W^n}^{-1/2} x = y$

  $$\begin{aligned} \Rightarrow x^{n+1} &= D_{W^n}^{-1/2} (A D_{W^n}^{-1/2})^{\dagger} y \\ &= D_{W^n}^{-1} A^* (A D_{W^n}^{-1} A^*)^{-1} y \\ &= D_{W^n}^{-1} v, \text{ where } (A D_{W^n}^{-1} A^*) v = y \end{aligned}$$

  $\rightarrow$ can use conjugate gradient method to solve

- $IRLS_3 : w^{n+1} = \underset{w>0}{argmin}\, J(x^{n+1}, w, \epsilon_{n+1})$

$$= \underset{w>0}{argmin}\, \frac{1}{2}[\sum_{j=1}^{N} |x_j^{n+1}|^2 w_j + \sum_{j=1}^{N} \epsilon_{n+1}^2 w_j + w_j^{-1}]$$

$\Rightarrow w_j^{n+1} = \frac{1}{\sqrt{|x_j^{n+1}|^2 + \epsilon_{n+1}^2}}, j \in [N]$

$\rightarrow$ we find that ϵ_{n+1} can effectively prevent w_j^{n+1} from exploding; however, we hope that ϵ_{n+1} can tend to zero

- $IRLS_2 : \epsilon_{n+1} := \min\{\epsilon_n, \gamma(x^{n+1})^*_{s+1}\}$
  - ϵ is nonincreasing
  - As x tends to s-sparse, ϵ can also decrease.

Table of Contents

Table of Contents

- Objective : $\min_{x \in \mathbb{R}^N} F(Ax) + G(x)$

$$\equiv \min_{x \in \mathbb{R}^N, z \in \mathbb{R}^m} F(z) + G(x) \text{ subject to } Ax - z = 0$$

  ($A \in \mathbb{C}^{m \times N}$; $F : \mathbb{C}^m \to (-\infty, \infty]$, $G : \mathbb{C}^N \to (-\infty, \infty]$ are two convex functions)

Primal–Dual Algorithm

Input: $\mathbf{A} \in \mathbb{C}^{m\times N}$, convex functions F, G.
Parameters: $\theta \in [0,1]$, $\tau, \sigma > 0$ such that $\tau\sigma\|\mathbf{A}\|_{2\to 2}^2 < 1$.
Initialization: $\mathbf{x}^0 \in \mathbb{C}^N$, $\boldsymbol{\xi}^0 \in \mathbb{C}^m$, $\bar{\mathbf{x}}^0 = \mathbf{x}^0$.
Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\boldsymbol{\xi}^{n+1} := P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \mathbf{A}\bar{\mathbf{x}}^n), \qquad (\mathrm{PD}_1)$$

$$\mathbf{x}^{n+1} := P_G(\tau; \mathbf{x}^n - \tau \mathbf{A}^* \boldsymbol{\xi}^{n+1}), \qquad (\mathrm{PD}_2)$$

$$\bar{\mathbf{x}}^{n+1} := \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n). \qquad (\mathrm{PD}_3)$$

Output: Approximation $\boldsymbol{\xi}^\sharp = \boldsymbol{\xi}^{\bar{n}}$ to a solution of the dual problem (15.16),
Approximation $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$ to a solution of the primal problem (15.15).

Convex Conjugate

- Given a function $F : \mathbb{R}^N \to (-\infty, \infty]$, the convex conjugate function of F is the function $F^* : \mathbb{R}^N \to (-\infty, \infty]$ defined by

$$F^*(y) := \sup_{x \in \mathbb{R}^N} \{\langle x, y \rangle - F(x)\}$$

- The convex conjugate F^* is always a convex function
- By the definition of convex conjugate, we can get the Fenchel (or Young, or Fenchel-Young) inequality

$$\langle x, y \rangle \leq F(x) + F^*(y) \ \forall x, y \in R^N$$

  - If $x \in \partial F^*(y)$ (equivalently, $y \in \partial F(x)$), then equality holds

Primal–Dual Algorithm

Input: $\mathbf{A} \in \mathbb{C}^{m \times N}$, convex functions F, G.
Parameters: $\theta \in [0,1]$, $\tau, \sigma > 0$ such that $\tau\sigma\|\mathbf{A}\|_{2\to 2}^2 < 1$.
Initialization: $\mathbf{x}^0 \in \mathbb{C}^N$, $\boldsymbol{\xi}^0 \in \mathbb{C}^m$, $\bar{\mathbf{x}}^0 = \mathbf{x}^0$.
Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\boldsymbol{\xi}^{n+1} := P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \mathbf{A}\bar{\mathbf{x}}^n), \qquad (\mathrm{PD}_1)$$

$$\mathbf{x}^{n+1} := P_G(\tau; \mathbf{x}^n - \tau \mathbf{A}^* \boldsymbol{\xi}^{n+1}), \qquad (\mathrm{PD}_2)$$

$$\bar{\mathbf{x}}^{n+1} := \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n). \qquad (\mathrm{PD}_3)$$

Output: Approximation $\boldsymbol{\xi}^\sharp = \boldsymbol{\xi}^{\bar{n}}$ to a solution of the dual problem (15.16),
Approximation $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$ to a solution of the primal problem (15.15).

Proximal Mapping

- $P_F(z) := \underset{x \in R^N}{argmin}\, F(x) + \frac{1}{2}\|x - z\|_2^2$

  $\rightarrow$ the proximal mapping associated with F
- $x = P_F(z)$ if and only if $z \in x + \partial F(x) \Rightarrow P_F = (Id + \partial F)^{-1}$
- Moreau's identity : $P_F(z) + P_{F^*}(z) = z$
- $P_G(\tau; z) := P_{\tau G}(z)$; $P_{F^*}(\sigma; z) := P_{\sigma F^*}(z)$

Primal–Dual Algorithm

Input: $\mathbf{A} \in \mathbb{C}^{m \times N}$, convex functions F, G.
Parameters: $\theta \in [0,1]$, $\tau, \sigma > 0$ such that $\tau\sigma\|\mathbf{A}\|_{2\to 2}^2 < 1$.
Initialization: $\mathbf{x}^0 \in \mathbb{C}^N$, $\boldsymbol{\xi}^0 \in \mathbb{C}^m$, $\bar{\mathbf{x}}^0 = \mathbf{x}^0$.
Iteration: repeat until a stopping criterion is met at $n = \bar{n}$:

$$\boldsymbol{\xi}^{n+1} := P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \mathbf{A}\bar{\mathbf{x}}^n), \qquad (\mathrm{PD}_1)$$

$$\mathbf{x}^{n+1} := P_G(\tau; \mathbf{x}^n - \tau \mathbf{A}^* \boldsymbol{\xi}^{n+1}), \qquad (\mathrm{PD}_2)$$

$$\bar{\mathbf{x}}^{n+1} := \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n). \qquad (\mathrm{PD}_3)$$

Output: Approximation $\boldsymbol{\xi}^\sharp = \boldsymbol{\xi}^{\bar{n}}$ to a solution of the dual problem (15.16),
Approximation $\mathbf{x}^\sharp = \mathbf{x}^{\bar{n}}$ to a solution of the primal problem (15.15).

Optimization problems with a composite objective function

- Primal problem : $\underset{x\in\mathbb{R}^N}{min}\, F(Ax) + G(x)$

$$\equiv \underset{x\in\mathbb{R}^N, z\in\mathbb{R}^m}{min} F(z) + G(x) \text{ subject to } Ax - z = 0$$

- Lagrange function : $L(x, z, \xi) = F(z) + G(x) + \langle \xi, Ax - z\rangle$
- Lagrange dual function : $H(\xi) = \underset{x,z}{inf}\, L(x, z, \xi)$

$$= -F^*(\xi) - G^*(-A^*\xi)$$

- Dual problem : $\underset{\xi\in\mathbb{R}^m}{max}(-F^*(\xi) - G^*(-A^*\xi))$
- By strong duality : it is equivalent to solving a saddle-point problem

$$\min_{x\in\mathbb{R}^N} \max_{\xi\in\mathbb{R}^m} Re\langle Ax, \xi\rangle + G(x) - F^*(\xi)$$

Fixed-Point Interpretation

- Fix $x = x^\sharp$:

  the saddle-point problem becomes
  $\max_{\xi \in \mathbb{R}^m} Re\langle Ax^\sharp, \xi\rangle + G(x^\sharp) - F^*(\xi) = \min_{\xi \in \mathbb{R}^m} -Re\langle Ax^\sharp, \xi\rangle + G(x^\sharp) + F^*(\xi)$

  $\Rightarrow \xi^\sharp$ is a minimizer iff $0 \in -Ax^\sharp + \partial F^*(\xi^\sharp)$
- Fix $\xi = \xi^\sharp$:

  the saddle-point problem becomes

  $\min_{x \in \mathbb{R}^m} Re\langle Ax, \xi^\sharp\rangle + G(x) - F^*(\xi^\sharp)$

  $\Rightarrow x^\sharp$ is a minimizer iff $0 \in A^*\xi^\sharp + \partial G(x^\sharp)$

Fixed-Point Interpretation

- Fixed $x = x^\sharp$ to iterate ξ :

  $0 \in -Ax^\sharp + \partial F^*(\xi^\sharp)$

  $\Rightarrow \sigma A x^\sharp \in \sigma \partial F^*(\xi^\sharp)$

  $\Rightarrow \xi^\sharp + \sigma A x^\sharp \in \xi^\sharp + \sigma \partial F^*(\xi^\sharp)$

  $\Rightarrow \xi^{n+1} := P_{F^*}(\sigma; \xi^n + \sigma A \bar{x}^n)$

- Fixed $\xi = \xi^\sharp$ to iterate x :

  $0 \in A^* \xi^\sharp + \partial G(x^\sharp)$

  $\Rightarrow -\tau A^* \xi^\sharp \in \tau \partial G(x^\sharp)$

  $\Rightarrow x^\sharp - \tau A^* \xi^\sharp \in x^\sharp + \tau \partial G(x^\sharp)$

  $\Rightarrow x^{n+1} := P_G(\tau; x^n - \tau A^* \xi^{n+1})$

Algorithm Settings

- Initialization : $x^0 = \bar{x}^0 = A^* y$, $\xi^0 = 0$
- $\|A\|_{2\to 2} = \sigma_{max}(A) \therefore$ choose τ, σ such that $\tau\sigma < \sigma_{max}(A)^{-2}$
- A practical stopping criterion can be based on the primal-dual gap
  $E(x, \xi) = F(Ax) + G(x) + G^*(-A^*\xi) + F^*(\xi) \geq 0$
  i.e., stops when $E(x^n, \xi^n) \leq \eta$ for a prescribed tolerance $\eta > 0$

Table of Contents

ℓ_1-minimization problem

- Objective : $\min_{x \in \mathbb{C}^N} \|x\|_1$ subject to Ax=y
- $$\begin{cases} F(z) = \chi_{\{y\}}(z) = \begin{cases} 0, & z = y \\ \infty & otherwise \end{cases} \\ G(x) = \|x\|_1 \end{cases}$$
- $F^*(\xi) = \langle y, \xi \rangle \Rightarrow P_{F^*}(\sigma, z) = z - \sigma y$
- $P_G(\tau, z) = S_\tau(z)$, where $S_\tau(z)$ is the soft thresholding operator operated entrywise on z

$$S_\tau(z_\ell) = \begin{cases} sgn(z_\ell)(|z_\ell - \tau|) & |z_\ell| \geq \tau \\ 0 & otherwise \end{cases}$$

ℓ_1-minimization problem

- The primal-dual algorithm :

$$
\begin{aligned}
\xi^{n+1} &= \xi^n + \sigma(\mathbf{A}\bar{\mathbf{x}}^n - \mathbf{y}),\\
\mathbf{x}^{n+1} &= \mathcal{S}_\tau(\mathbf{x}^n - \tau \mathbf{A}^* \xi^{n+1}),\\
\bar{\mathbf{x}}^{n+1} &= \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n).
\end{aligned}
$$

quadractically constrained ℓ_1-minimization problem

- Objective : $\min_{x \in \mathbb{C}^N} \|x\|_1$ subject to $\|Ax - y\|_2 \leq \eta$
- $$\begin{cases} F(z) = \begin{cases} 0, & \|z - y\|_2 \leq \eta \\ \infty & otherwise \end{cases} \\ G(x) = \|x\|_1 \end{cases}$$
- $F^*(\xi) = Re\langle \xi, y \rangle + \eta \|\xi\|_2$
- $$P_{F^*}(\sigma, \xi) = \begin{cases} 0 & \|\xi - \sigma y\|_2 \leq \eta\sigma \\ (1 - \frac{\eta\sigma}{\|\xi - \sigma y\|_2})(\xi - \sigma y) & otherwise \end{cases}$$

quadractically constrained ℓ_1-minimization problem

- The primal-dual algorithm :

$$
\begin{aligned}
\boldsymbol{\xi}^{n+1} &= P_{F^*}(\sigma; \boldsymbol{\xi}^n + \sigma \mathbf{A}\bar{\mathbf{x}}^n) \\
&= \begin{cases} 0 & \text{if } \|\sigma^{-1}\boldsymbol{\xi}^n + \mathbf{A}\bar{\mathbf{x}}^n - \mathbf{y}\|_2 \le \eta, \\ \left(1 - \dfrac{\eta\sigma}{\|\boldsymbol{\xi}^n + \sigma(\mathbf{A}\bar{\mathbf{x}}^n - \mathbf{y})\|_2}\right)(\boldsymbol{\xi}^n + \sigma(\mathbf{A}\bar{\mathbf{x}}^n - \mathbf{y})) & \text{otherwise,} \end{cases} \\
\mathbf{x}^{n+1} &= \mathcal{S}_\tau(\mathbf{x}^n - \tau \mathbf{A}^* \boldsymbol{\xi}^{n+1}), \\
\bar{\mathbf{x}}^{n+1} &= \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n).
\end{aligned}
$$

ℓ_1-regularized least squares problem

- Objective : $\min_{x \in \mathbb{C}^N} \|x\|_1 + \frac{\gamma}{2}\|Ax - y\|_2^2$
- $\begin{cases} F(z) = \frac{\gamma}{2}\|z - y\|_2^2 \\ G(x) = \|x\|_1 \end{cases}$
- $F^*(\xi) = Re\langle y, \xi \rangle + \frac{\|\xi\|_2^2}{2\gamma}$
- $P_{F^*}(\sigma; \xi) = \frac{\gamma}{\gamma + \sigma}(\xi - \sigma y)$

ℓ_1-regularized least squares problem

- The primal-dual algorithm :

$$
\begin{aligned}
\xi^{n+1} &= \frac{\gamma}{\gamma+\sigma}\left(\xi^n + \sigma(\mathbf{A}\bar{\mathbf{x}}^n - \mathbf{y})\right), \\
\mathbf{x}^{n+1} &= \mathcal{S}_\tau(\mathbf{x}^n - \tau \mathbf{A}^* \xi^{n+1}), \\
\bar{\mathbf{x}}^{n+1} &= \mathbf{x}^{n+1} + \theta(\mathbf{x}^{n+1} - \mathbf{x}^n).
\end{aligned}
$$

Table of Contents

Table of Contents

Objective

$$\min_{x \in \mathbb{R}^N} F(x) + G(Bx)$$
$$\equiv \min_{x,y \in \mathbb{R}^N} F(x) + G(y) \text{ subject to } y = Bx$$

- F and G are both lower semicontinuous and convex
- $B^* B$ is invertible

Augmented Lagrangian of index $\tau > 0$:

$$L_\tau(x, y, \xi) = F(x) + G(y) + \frac{1}{\tau} Re\langle \xi, Bx - y \rangle + \frac{1}{2\tau} \|Bx - y\|_2^2$$

Iteration Rule

1. Fix $y = y^\sharp$, $\xi = \xi^\sharp$, minimize over x
   $\Rightarrow 0 \in \tau \partial F(x) + B^* B x + B^*(\xi^\sharp - y^\sharp)$
   Let $P_F^B(\tau; y) = \underset{z}{argmin}\{\tau F(z) + \frac{1}{2}\|Bz - y\|_2^2\}$
   $\Rightarrow x^n = P_F^B(\tau; y^n - \xi^n)$
2. Fix $x = x^\sharp$, $\xi = \xi^\sharp$, minimize over y
   $\Rightarrow 0 \in \tau \partial G(y) + y - \xi^\sharp - Bx^\sharp$
   $\Rightarrow y^{n+1} = P_G(\tau; Bx^n + \xi^n)$
3. Fix $x = x^\sharp$, $y = y^\sharp$, minimize over ξ
   $\Rightarrow 0 = Bx^\sharp - y^\sharp$
   $\Rightarrow \xi^{n+1} = \xi^n + Bx^n - y^{n+1}$

Table of Contents

Basis Pursuit

- Objective : $\underset{x}{argmin} \|x\|_1$ subject to Ax=y
- $$\begin{cases} F(x) = \chi_{\{y\}}(z) = \begin{cases} 0, & Ax = y \\ \infty & otherwise \end{cases} \\ G(x) = \|x\|_1 \Rightarrow B = I_d \end{cases}$$
- Iteration :
  1. $x^n = (y^n - \xi^n) - A^*(AA^*)^{-1}(A(y^n - \xi^n) - y)$
  2. $y^{n+1} = S_\tau(x^n + \xi^n)$
  3. $\xi^{n+1} = \xi^n + x^n - y^{n+1}$

ℓ_1-regularized least squares problem

- Objective : $\underset{x}{minimize} \frac{1}{2}\|Ax - y\|_2^2 + \lambda\|x\|_1$
- $F(x) = \frac{1}{2}\|Ax - y\|_2^2, \; G(x) = \lambda\|x\|_1$
- Iteration :
  1. $x^n = (A^*A + \tau^{-1}Id)^{-1}(A^*y + \tau^{-1}(y^n - \xi^n))$
  2. $y^{n+1} = S_{\tau\lambda}(x^n + \xi^n)$
  3. $\xi^{n+1} = \xi^n + x^n - y^{n+1}$

Table of Contents

Table of Contents

Objective

$$\underset{x}{\textit{minimize}}\ \mathrm{F(x)+G(x)}$$

- F is differentiable and convex
- ∇F is L-Lipschitz $\Rightarrow \|\nabla F(x) - \nabla F(y)\|_2 \leq L\|x - y\|_2 \ \forall x, y$
- G is lower semicontinuous and convex

Iteration Rule

- $x^\sharp = \underset{x}{argmin}\ \mathrm{F(x)+G(x)} \Rightarrow 0 \in \nabla F(x^\sharp) + \partial G(x^\sharp)$

  To iterate x

  $\Rightarrow x^\sharp - \tau \nabla F(x^\sharp) \in x^\sharp - \tau \partial G(x^\sharp)$

  $\Rightarrow x^{n+1} := P_G(\tau; x^n - \tau \nabla F(x^n))$
- Forward step : $z^n = x^n - \tau \nabla F(x^n) \rightarrow$ gradient method

  (from x^n forward to z^n)
- Backward step : $x^{n+1} = P_G(\tau; z^n) \rightarrow$ proximal point algorithm

  $\Rightarrow z^n \in x^{n+1} + \tau \partial G(x^{n+1}) \rightarrow$ subgradient step

  (from x^{n+1} backward to z^n)
- Convergence is guaranteed if $\tau < 2/L$

Accelerated Proximal Gradient Method

- Initialization : $x^0 = z^0,\ t_0 = 1$
- Iteration :
  1. $x^{n+1} = P_G(L^{-1}; z^n - \nabla F(z^n))$
  2. $t_{n+1} = \frac{1+\sqrt{4t_n^2+1}}{2},\ \lambda_n = 1 + \frac{t_n - 1}{t_{n+1}}$
  3. $z^{n+1} = x^n + \lambda_n(x^{n+1} - x^n)$

Table of Contents

ℓ_1-regularized least squares problem

- Objective : $\min_x \frac{1}{2}\|Ax - y\|_2^2 + \lambda\|x\|_1$
  - $F(x) = \frac{1}{2}\|Ax - y\|_2^2 \Rightarrow \nabla F(x) = A^*(Ax - y)$
    $\rightarrow \|\nabla F(x) - \nabla F(y)\|_2 = \|A^*A(x - z)\|_2 \leq \|A^*A\|_{2\to 2}\|x - z\|_2$
    $\therefore \nabla F(x)$ is L-Lipschitz with $L \leq \|A^*A\|_{2\to 2} = \|A\|_{2\to 2}^2$
  - $G(x) = \|x\|_1$
- Forward-Backward Algorithm : $x^{n+1} := S_{\lambda\tau}(x^n - \tau A^*(Ax^n - y))$
  $\Rightarrow$ iterative shrinkage-thresholding algorithm (ISTA) or iterative soft-thresholding
- convergence is guaranteed if $\tau < 2/\|A\|_{2\to 2}^2$

Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)

1. $x^{n+1} := S_{\lambda\|A\|_{2\to 2}^{-2}}(z^n - A^*(Az^n - y))$
2. $t_{n+1} = \frac{1+\sqrt{4t_n^2+1}}{2}$, $\lambda_n = 1 + \frac{t_n - 1}{t_{n+1}}$
3. $z^{n+1} := x^n + \lambda_n(x^{n+1} - x^n)$

Table of Contents

Table of Contents

Objective

$$\underset{x}{minimize}\ F(x) + G(x)$$

- F and G are both convex, but not necessarily need to be differentiable

Iteration Rule

- $x^\sharp = \underset{x}{argmin}\ \mathrm{F(x)+G(x)} \Rightarrow 0 \in \partial F(x^\sharp) + \partial G(x^\sharp)$
- Introduce another variable z to separately consider F and G

  Let $z^\sharp \in x^\sharp + \tau\partial F(x^\sharp) \rightarrow x = P_F(\tau; z)$

  $\Rightarrow z^\sharp - x^\sharp \in \tau\partial F(x^\sharp)$

  $\therefore 0 \in z^\sharp - x^\sharp + \tau\partial G(x^\sharp)$

  $\Rightarrow 2x^\sharp - z^\sharp \in x^\sharp + \tau\partial G(x^\sharp) \rightarrow x = P_F(\tau; z) = P_G(\tau; 2P_F(\tau; z) - z)$
- Iteration :
  1. $x^n = P_F(\tau; z^n)$
  2. $z^{n+1} = P_G(\tau; 2x^n - z^n) - x^n + z^n$

Table of Contents

Basis Pursuit

- Objective : $\min_x \|x\|_1$ subject to $Ax = y$
  - $F(x) = \begin{cases} 0 & Ax = y \\ \infty & otherwise \end{cases}$

    $\Rightarrow P_F(\tau; x) = \underset{z}{argmin}\{\|z - x\|_2 \text{ subject to Az=y } \} = x + A^\dagger(y - Ax)$
  - $G(x) = \|x\|_1$
- Iteration :
  1. $x^n = z^n + A^\dagger(y - Az^n)$
  2. $z^{n+1} = S_\tau(2x^n - z^n) - x^n + z^n$

Table of Contents

Reference

Simon Foucart , Holger Rauhut, " *A Mathematical Introduction to Compressive Sensing* " , chapter 15