Accelerated Proximal Algorithms

Karol Chojnacki

March 2025

Optimization Problem:

$$\min_x f(x)$$

Update Rule:

$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

Convergence Rate: $\mathcal{O}(1/t)$

Optimization Problem:

$$\min_x f(x)$$

Update Rule:

$$y_{t+1} = x_t - \eta \nabla f(x_t)$$

$$x_{t+1} = y_{t+1} + \frac{k-1}{k+2}(y_{t+1} - y_t)$$

Convergence Rate: $\mathcal{O}(1/t^2)$

Proximal Operator: For a closed, proper, convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the (scaled) proximal operator is defined as:

$$\mathrm{prox}_{\lambda f}(v) = \arg\min_x \left(f(x) + \frac{1}{2\lambda} \|x - v\|^2 \right),$$

where $\lambda > 0$ scales the problem.

Motivation.

When we linearize $f(x)$ by $\langle \boldsymbol{x} - \boldsymbol{x}_k, \nabla f(\boldsymbol{x}_k)\rangle$, prox iterating becomes GD.

$$x^* = \text{argmin } f(x) \iff x^* = \text{argmin } f(x) + \delta||x - x^*||_2^2$$

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \underset{\boldsymbol{x}\in\mathbb{R}^N}{\arg\min}\left(f(\boldsymbol{x}_k) + \langle \boldsymbol{x} - \boldsymbol{x}_k, \nabla f(\boldsymbol{x}_k)\rangle + \frac{1}{2\alpha_k}\|\boldsymbol{x} - \boldsymbol{x}_k\|_2^2\right) \\
&= \underset{\boldsymbol{x}\in\mathbb{R}^N}{\arg\min}\left(\frac{\alpha_k}{2}\|\nabla f(\boldsymbol{x}_k)\|_2^2 + \langle \boldsymbol{x} - \boldsymbol{x}_k, \nabla f(\boldsymbol{x}_k)\rangle + \frac{1}{2\alpha_k}\|\boldsymbol{x} - \boldsymbol{x}_k\|_2^2\right) \\
&= \underset{\boldsymbol{x}\in\mathbb{R}^N}{\arg\min}\left(\frac{1}{2\alpha_k}\|\boldsymbol{x} - \boldsymbol{x}_k + \alpha_k\nabla f(\boldsymbol{x}_k)\|_2^2\right) \\
&= \boldsymbol{x}_k - \alpha_k\nabla f(\boldsymbol{x}_k)
\end{aligned}$$

Optimization Problem:

$$\min_{x} g(x) + h(x)$$

where $h(x)$ is a regularization term, e.g., $h(x) = \lambda \|x\|_1$.

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x} \in \mathbb{R}^N} \left(g\left(\boldsymbol{x}_k\right) + \langle \boldsymbol{x} - \boldsymbol{x}_k, \nabla g\left(\boldsymbol{x}_k\right) \rangle + h(\boldsymbol{x}) + \frac{1}{2\alpha_k} \left\| \boldsymbol{x} - \boldsymbol{x}_k \right\|_2^2 \right) \\
&= \arg\min_{x \in \mathbb{R}^N} \left(h(\boldsymbol{x}) + \frac{1}{2\alpha_k} \left\| \boldsymbol{x} - \boldsymbol{x}_k + \alpha_k \nabla g\left(\boldsymbol{x}_k\right) \right\|_2^2 \right) \\
&= \operatorname{prox}_{\alpha_k h}\left(\boldsymbol{x}_k - \alpha_k \nabla g\left(\boldsymbol{x}_k\right)\right).
\end{aligned}$$

ISTA

Update Rule:

$$x_{t+1} = \mathrm{prox}_{\lambda\eta h}(x_t - \eta\nabla g(x_t)) = T(x_t - \eta\nabla g(x_t))$$

Convergence Rate: $\mathcal{O}(1/t)$
Proximal Operator for L1 Regularization:

$$\mathrm{prox}_{\lambda||x||_1}(v) := T(v) = \begin{cases} v - \lambda, & v > \lambda \\ 0, & |v| \leq \lambda \\ v + \lambda, & v < -\lambda \end{cases}$$

Optimization Problem:

$$\min_x f(x) + g(x)$$

where $g(x)$ is a regularization term.
Update Rule:

$$y_{t+1} = \text{prox}_{\lambda \eta g}(x_t - \eta \nabla f(x_t))$$

$$x_{t+1} = y_{t+1} + \frac{t-1}{t+2}(y_{t+1} - y_t)$$

Convergence Rate: $\mathcal{O}(1/t^2)$

Modification: Adaptive step size $\eta \to \eta_k$:

$$\eta_k = \frac{1}{L} + k\mu$$

Expected Convergence: Empirically close to $\mathcal{O}(1/t^2)$

Adaptive Support FISTA

Modification: Step-size depends on support structure:

$$\eta_k = c \frac{\|s_k \nabla f(x_k)\|^2}{\|\Phi(s_k \nabla f(x_k))\|^2}$$

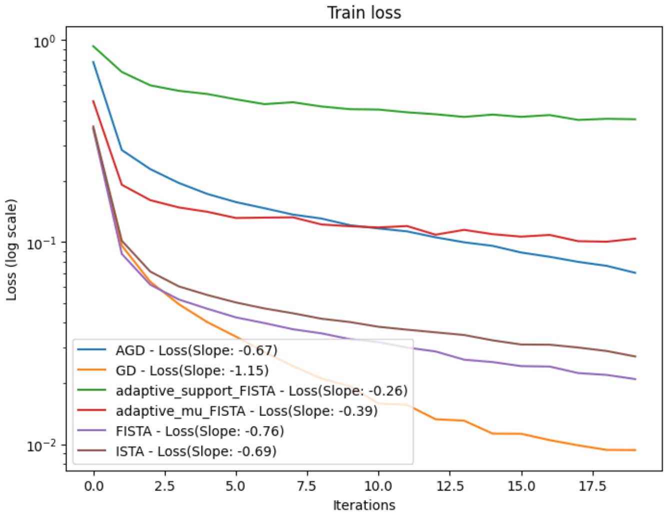
where:

- c is a scaling constant controlling the step size.
- s_k is a binary mask indicating which elements of x_k are nonzero (support of x_k).
- $\nabla f(x_k)$ is the gradient of the loss function with respect to the model parameters.
- $\Phi(\cdot)$ represents a transformation that captures structural information, such as a convolutional layer in a CNN.

Expected Convergence: Adaptive, may accelerate convergence in sparse problems to $\mathcal{O}(1/t^3)$, *Gustavo Silva and Paul Rodriguez*.

Goal: Compare optimization methods on ML problems.
Tasks: MNIST classification with CNN.
Loss function: Cross-entropy + L1.
Metrics: Convergence speed, sparsity, test accuracy.
Implementation: PyTorch, Google Colab (T4 GPU).

Losses


Train loss
Loss (log scale)
Iterations
10^0
10^{-1}
10^{-2}
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
AGD - Loss(Slope: -0.67)
GD - Loss(Slope: -1.15)
adaptive_support_FISTA - Loss(Slope: -0.26)
adaptive_mu_FISTA - Loss(Slope: -0.39)
FISTA - Loss(Slope: -0.76)
ISTA - Loss(Slope: -0.69)

Sparsity

Algorithm	Final Sparsity (%)	Test Accuracy (%)
AGD	0.064	97.73
GD	0.075	98.25
adaptive_support_FISTA	50.968	63.10
adaptive_mu_FISTA	13.115	94.81
FISTA	39.842	98.31
ISTA	43.394	98.24

Sources

- Proximal Algorithms - Stanford University, Neal Parikh
- https://sci-hub.se/https://ieeexplore.ieee.org/document/8903154
- My code for experiment, if anybody wants hit me up

Thank you very much for listening!