Nonnegative Least Squares

PGD, accelerated PGD and with restarts

Andersen Ang

Mathématique et recherche opérationnelle
UMONS, Belgium

manshun.ang@umons.ac.be   Homepage: angms.science

First draft : August 4, 2017
Last update : February 19, 2020

Overview

Nonnegative Least Squares

Non**n**egative **L**east **S**quares (NNLS) : given $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$, find $\mathbf{x} \in \mathbb{R}^n_+$ by solving

$$(\mathcal{P}) \;\; : \;\; \underset{\mathbf{x} \geq 0}{\operatorname{argmin}} \; f(\mathbf{x}) := \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2.$$

A constrained optimization problem: $\mathbf{x}$ has to be nonnegative.

- $\mathbf{x}$ is the *coefficient* of columns of $\mathbf{A}$
- $\mathbf{x}_{\text{NNLS}}$ tells the contributions of each columns $\mathbf{a}_i$ towards $\mathbf{b}$
- $\mathbf{x}_{\text{LS}}$ is less interpretable as coefficient $\mathbf{x}_{\text{LS}}$ can has mixed signs, leading to mutual elimination

Equivalent constrained QP formulation of NNLS

Expand the function $\frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$:

$$\begin{aligned} f(\mathbf{x}) &= \frac{1}{2}(\mathbf{A}\mathbf{x}-\mathbf{b})^\top(\mathbf{A}\mathbf{x}-\mathbf{b}) \\ &= \frac{1}{2}\Big(\mathbf{x}^\top\mathbf{A}^\top\mathbf{A}\mathbf{x}-\mathbf{x}^\top\mathbf{A}^\top\mathbf{b}-\mathbf{b}^\top\mathbf{A}\mathbf{x}+\mathbf{b}^\top\mathbf{b}\Big) \\ &= \frac{1}{2}\Big(\mathbf{x}^\top\mathbf{A}^\top\mathbf{A}\mathbf{x}-2\mathbf{b}^\top\mathbf{A}\mathbf{x}+\|\mathbf{b}\|_2^2\Big) \\ &= \frac{1}{2}\mathbf{x}^\top\mathbf{A}^\top\mathbf{A}\mathbf{x}-\mathbf{b}^\top\mathbf{A}\mathbf{x}+\frac{1}{2}\|\mathbf{b}\|_2^2. \end{aligned}$$

Let $\mathbf{Q}=\mathbf{A}^\top\mathbf{A}$, $\mathbf{p}=(\mathbf{b}^\top\mathbf{A})^\top=\mathbf{A}^\top\mathbf{b}$ and $c=\frac{1}{2}\|\mathbf{b}\|_2^2$, NNLS becomes a constrained quadratic programming (QP) problem

$$\min_{\mathbf{x}\geq 0}\frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x}-\mathbf{p}^\top\mathbf{x}+c.$$

In the following, we ignore the constant c.

NNLS(NNQP) is a convex problem

$$\min_{\mathbf{x} \in \mathbb{R}^n_+} \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{p}^\top \mathbf{x},\ \mathbf{Q} = \mathbf{A}^\top \mathbf{A},\ \mathbf{p} = \mathbf{A}^\top \mathbf{b}.$$

- matrix $\mathbf{Q} = \mathbf{A}^\top \mathbf{A}$ is always positive-semidefinite and symmetric
- If $\mathbf{A}$ is full rank then $\mathbf{Q}$ is positive-definite
- NNLS(NNQP) is a convex optimization problem
  - the function convex : it is quadratic
  - the constraint set is convex : it is the nonnegative orthant

Solving NNLS by pseudo inverse and projection

The simplest (but wrong) way to solve NNLS is to modify the solution obtained from the corresponding ordinary least squares: if $\mathbf{A}^\top\mathbf{A}$ is invertible, set the gradient $\nabla f(\mathbf{x}) = \mathbf{A}^\top\mathbf{A}\mathbf{x} - \mathbf{A}^\top\mathbf{b}$ zero gives

$$\mathbf{x}_{\mathsf{LS}} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}.$$

Now we have a two-step method to solve the NNLS

1. $\mathbf{y} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}$ (solution of ordinary least squares)
2. $\mathbf{x} = \mathcal{P}_{\mathbb{R}^n_+}(\mathbf{y}) = \max(\mathbf{y}, 0)$ (projection onto nonnegative orthant) where $\mathcal{P}_{\mathbb{R}^n_+}$ is the projection operator.

In fact, this method can produce a wrong solution. For example, if all components in $\mathbf{x}_{\mathsf{LS}}$ are negative, then this method basically output a zero vector, while the true $\mathbf{x}$ which is non-zero may exists.

Solving NNLS by Projected Gradient Descent

For $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{p}^\top \mathbf{x}$, the gradient and the projection are

$$\nabla f = \mathbf{Q}\mathbf{x} - \mathbf{p}, \qquad \mathcal{P}_{\mathbb{R}^n_+}(\mathbf{x}) = \big[\mathbf{x}\big]_+ := \max(\mathbf{x}, 0).$$

The Projected Gradient Descent (PGD) algorithm for solving NNLS is :

Algorithm 1: PGD for NNLS

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$
Initialization Set $\mathbf{x}_0 \in \mathbb{R}^n_+$, $\mathbf{p} = \mathbf{A}^\top \mathbf{b}$, $\mathbf{Q} = \mathbf{A}^\top \mathbf{A}$, $k = 1$
while *stopping condition is not met* **do**

$\quad \mathbf{x}_k = \big[\mathbf{x}_{k-1} - t_k(\mathbf{Q}\mathbf{x}_{k-1} - \mathbf{p})\big]_+$
$\quad k = k + 1$

end

---

where stepsize t_k can be set as $\frac{1}{L}$, where L is the Lipschitz constant of $\nabla f(\mathbf{x})$. The next slide shows $L = \|\mathbf{Q}\|_2$.

A lemma

Fact 1. For all matrix $\mathbf{A}$ and all vector $\mathbf{x}$, we have $\|\mathbf{A}\mathbf{x}\|_2 \leq \|\mathbf{A}\|_2\|\mathbf{x}\|_2$.
Remark : this is operator norm inequality, which is a immediate consequence of the definition of operator norm.

Lemma 1. $f(\mathbf{x}) = \dfrac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2$ is L-smooth with $L = \|\mathbf{A}^\top\mathbf{A}\|_2$.

i.e. $\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\|_2 \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|_2$ and $L = \|\mathbf{Q}\|_2 = \|\mathbf{A}^\top\mathbf{A}\|_2$.

Proof (Direct proof).

$$\begin{aligned}
\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\|_2 &= \|(\mathbf{A}^\top\mathbf{A}\mathbf{x}_1 - \mathbf{A}^\top\mathbf{b}) - (\mathbf{A}^\top\mathbf{A}\mathbf{x}_2 - \mathbf{A}^\top\mathbf{b})\|_2 \\
&= \|\mathbf{A}^\top\mathbf{A}\mathbf{x}_1 - \mathbf{A}^\top\mathbf{A}\mathbf{x}_2\|_2 \\
&= \|\mathbf{A}^\top\mathbf{A}(\mathbf{x}_1 - \mathbf{x}_2)\|_2 \\
&\overset{\text{fact 1}}{\leq} \|\mathbf{A}^\top\mathbf{A}\|_2\|\mathbf{x}_1 - \mathbf{x}_2\|_2 \qquad \square
\end{aligned}$$

Soving NNLS by PGD

With $L = \|\mathbf{A}^\top\mathbf{A}\|_2$, step size $t = \dfrac{1}{L} = \dfrac{1}{\|\mathbf{A}^\top\mathbf{A}\|_2}$, the PGD algorithm for solving NNLS becomes:

Algorithm 2: PGD (constant step size) for NNLS

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$
Initialization Set $\mathbf{x}_0 \in \mathbb{R}^n_+$, $\mathbf{p} = \mathbf{A}^\top\mathbf{b}$, $\mathbf{Q} = \mathbf{A}^\top\mathbf{A}$, $t = \frac{1}{\|\mathbf{Q}\|_2}$, $k = 1$
while *stopping condition is not met* **do**
  $\mathbf{x}_k = [\mathbf{x}_{k-1} - t(\mathbf{Q}\mathbf{x}_{k-1} - \mathbf{p})]_+$
  $k = k + 1$
end

From the theory of gradient descent, PGD converges at rate $\mathcal{O}(\frac{1}{k})$, where k is the iteration number.

Implementation issue – more compact form

Rewrite the update in compact form

$$\mathbf{x}_k = \left[(\mathbf{I}_n - t\mathbf{Q})\mathbf{x}_{k-1} + t\mathbf{p}\right]_+$$

Fix constants can be pre-computed outside the loop, we have

Algorithm 3: PGD (constant step size) for NNLS (compact form)

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$

Initialization Set $\mathbf{x}_0 \in \mathbb{R}^n_+$, $\Theta_1 = \mathbf{I}_n - \dfrac{\mathbf{A}^\top\mathbf{A}}{\|\mathbf{A}^\top\mathbf{A}\|_2}$, $\theta_2 = \dfrac{\mathbf{A}^\top\mathbf{b}}{\|\mathbf{A}^\top\mathbf{A}\|_2}$, $k = 1$

while *stopping condition is not met* **do**

$\quad \mathbf{x}_{k+1} = [\Theta_1\mathbf{x}_k + \theta_2]_+$

$\quad k = k + 1$

end

Nesterov's Acceleration

With Nesterov's acceleration, the accelerated PGD (APGD, with constant step size) algorithm is

Algorithm 4: APGD for NNLS

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$

Initialization Set $\mathbf{y}_0 = \mathbf{x}_0 \in \mathbb{R}^n_+$, $\Theta_1 = \mathbf{I}_n - \dfrac{\mathbf{A}^\top\mathbf{A}}{\|\mathbf{A}^\top\mathbf{A}\|_2}$, $\theta_2 = \dfrac{\mathbf{A}^\top\mathbf{b}}{\|\mathbf{A}^\top\mathbf{A}\|_2}$,

$k = 1$, set $\alpha_0 \in (0\ 1)$

while *stopping condition is not met* **do**

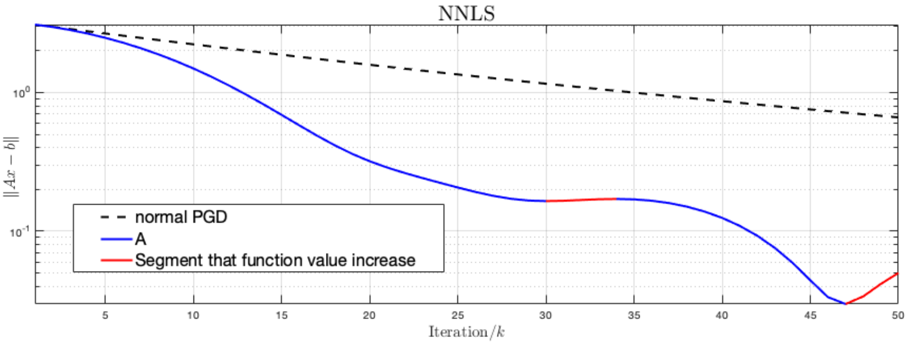
- $\mathbf{x}_k = [\Theta_1\mathbf{y}_{k-1} + \theta_2]_+$ (projected gradient step)
- $\alpha_k = \frac{1}{2}(\sqrt{\alpha_{k-1}^4 + 4\alpha_{k-1}^2} - \alpha_{k-1}^2)$, $\beta_k = \frac{\alpha_{k-1}(1-\alpha_{k-1})}{\alpha_{k-1}^2+\alpha_k}$
- $\mathbf{y}_k = \mathbf{x}_k + \beta_k(\mathbf{x}_k - \mathbf{x}_{k-1})$ (extrapolation)
- $k = k + 1$

end

The items in blue are the modifications from Nesterov's acceleration.

PGD is monotone but APGD is not

Recall, PGD is a *monotone*[1] method : for all k, $f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k)$. However, Nesterov's acclerated method is not monotone : in some iterations, the objective value actually increases.



An illustrative example $(m,n) = 100, 5$.

[1]In Nesterov's wording, relaxation sequence.

Accelerated Projected Gradient Descent with restarts

To make the scheme monotone, we can apply *adaptive restarts* : if error increases, we switch to gradient descent, and reset all parameters.

Algorithm 5: APGD for NNLS

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$

Initialization Set $\mathbf{y}_0 = \mathbf{x}_0 \in \mathbb{R}^n_+$, $\Theta_1 = \mathbf{I}_n - \frac{\mathbf{A}^\top \mathbf{A}}{\|\mathbf{A}^\top \mathbf{A}\|_2}$, $\theta_2 = \frac{\mathbf{A}^\top \mathbf{b}}{\|\mathbf{A}^\top \mathbf{A}\|_2}$, $k = 1$, set $\alpha_0 \in (0\ 1)$

while *stopping condition is not met* **do**

- $\mathbf{x}_k = [\Theta_1 \mathbf{y}_{k-1} + \theta_2]_+$ (projected gradient step)
- $\alpha_k = \frac{1}{2}(\sqrt{\alpha_{k-1}^4 + 4\alpha_{k-1}^2} - \alpha_{k-1}^2)$, $\beta_k = \frac{\alpha_{k-1}(1-\alpha_{k-1})}{\alpha_{k-1}^2 + \alpha_k}$
- $\mathbf{y}_k = \mathbf{x}_k + \beta_k(\mathbf{x}_k - \mathbf{x}_{k-1})$ (extrapolation)
- **if** error increases **do**
  - $\mathbf{x}_{k+1} = [\Theta_1 \mathbf{x}_k + \theta_2]_+$ (perform normal projected gradient step)
  - $\mathbf{y}_{k+1} = \mathbf{x}_{k+1}$ (restart)
  - $\alpha_k = \alpha_0$ (reset parameter)
- **endif**
- $k = k + 1$

end

Accelerated Projected Gradient Descent with restart

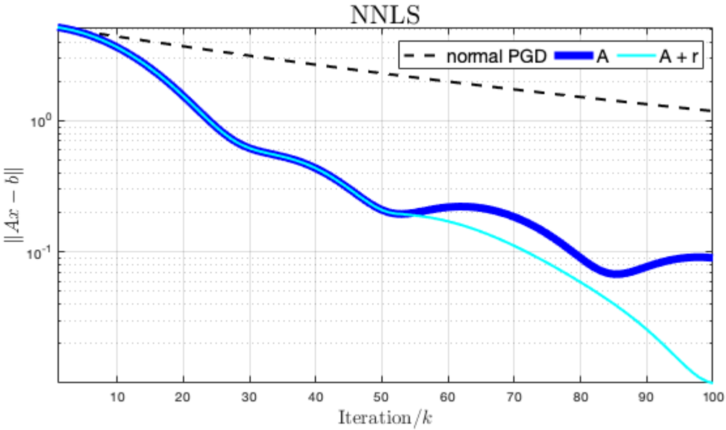


Figure: An illustrative example $(m, n) = 100, 10$.

MATLAB code (click me)

Nesterov's Acceleration other β

The parameters

$$\alpha_{k+1} = \frac{1}{2}(\sqrt{\alpha_k^4 + 4\alpha_k^2} - \alpha_k^2), \quad \beta_k = \frac{\alpha_k(1-\alpha_k)}{\alpha_k^2 + \alpha_{k+1}}$$

are so "complicated". Is there a simpler one ?

The answer is : Yes. Paul Tseng gave $\beta_k = \dfrac{k-1}{k+2}$:

Algorithm 6: APGD for NNLS using Paul Tseng's β

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$

Initialization Set $\mathbf{y}_0 = \mathbf{x}_0 \in \mathbb{R}_+^n$, $\Theta_1 = \mathbf{I}_n - \frac{\mathbf{A}^\top\mathbf{A}}{\|\mathbf{A}^\top\mathbf{A}\|_F}$, $\theta_2 = \frac{\mathbf{A}^\top\mathbf{b}}{\|\mathbf{A}^\top\mathbf{A}\|_F}$

while *stopping condition is not met* **do**

$\quad \mathbf{x}_k = [\Theta_1 \mathbf{y}_{k-1} + \theta_2]_+$ (projected gradient step)

$\quad \mathbf{y}_k = \mathbf{x}_k + \dfrac{k-1}{k+2}(\mathbf{x}_k - \mathbf{x}_{k-1})$ (extrapolation)

end

(Note that, for simplicity, the update of k is not shown in the algorithm)

APGD with constant β

Note that the function $f(\mathbf{x})$ in NNLS is smooth, and it is strongly convex if $\mathbf{A}$ is full rank.

- Strongly convex : recall a function $f(\mathbf{x})$ is strongly convex iff $\nabla^2 f(\mathbf{x}) - \mu\mathbf{I} \geq 0$. As $\nabla^2 f(\mathbf{x}) = \mathbf{Q} = \mathbf{A}^\top\mathbf{A}$, we have

$$\mathbf{Q} - \mu\mathbf{I} \geq 0.$$

  Here, μ can be taken as $\lambda_{\text{min}}(\mathbf{Q}) = \sigma_{\text{min}}(\mathbf{A})$.
- L-Smooth : as f is twice differentiable, f is L-smooth iff $\nabla^2 f(\mathbf{x}) - L\mathbf{I} \leq 0$. We have $L \leq \lambda_{\text{max}}(\mathbf{Q}) = \sigma_{\text{max}}(\mathbf{A})$.

For smooth strongly convex function, the extrapolation parameter β of Nesterov's acceleration can be set to

$$\beta_k = \beta = \frac{1-\sqrt{Q}}{1+\sqrt{Q}}$$

where $Q = \dfrac{L}{\mu}$ is the (optimization) condition number of the f.

Recall the (linear algebra) condition number of a matrix $\mathbf{A}$ is $\kappa(\mathbf{A})$.

Nesterov's Acceleration with β

With constant β, we have the following

Algorithm 7: APGD for NNLS using fixed β

Result: A solution $\mathbf{x}$ that approximately solves $(\mathcal{P})$

Initialization Set $\mathbf{y}_0 = \mathbf{x}_0 \in \mathbb{R}^n_+$, $\Theta_1 = \mathbf{I}_n - \dfrac{\mathbf{A}^\top \mathbf{A}}{\|\mathbf{A}^\top \mathbf{A}\|_F}$, $\theta_2 = \dfrac{\mathbf{A}^\top \mathbf{b}}{\|\mathbf{A}^\top \mathbf{A}\|_F}$

Set $\beta = \dfrac{1-\sqrt{\kappa}}{1+\sqrt{\kappa}}$, where $\kappa = \dfrac{L}{\mu} = \dfrac{\lambda_{\text{max}}(\mathbf{Q})}{\lambda_{\text{min}}(\mathbf{Q})} = \dfrac{\sigma_{\text{max}}(\mathbf{A})}{\sigma_{\text{min}}(\mathbf{A})} = \dfrac{1}{\kappa(\mathbf{A})}$

while *stopping condition is not met* **do**

  $\mathbf{x}_k = [\Theta_1 \mathbf{y}_{k-1} + \theta_2]_+$ (projected gradient step)

  $\mathbf{y}_k = \mathbf{x}_k + \beta(\mathbf{x}_k - \mathbf{x}_{k-1})$ (extrapolation)

end

The items in blue are the modifications from the acceleration scheme with fixed β.

Comparisons

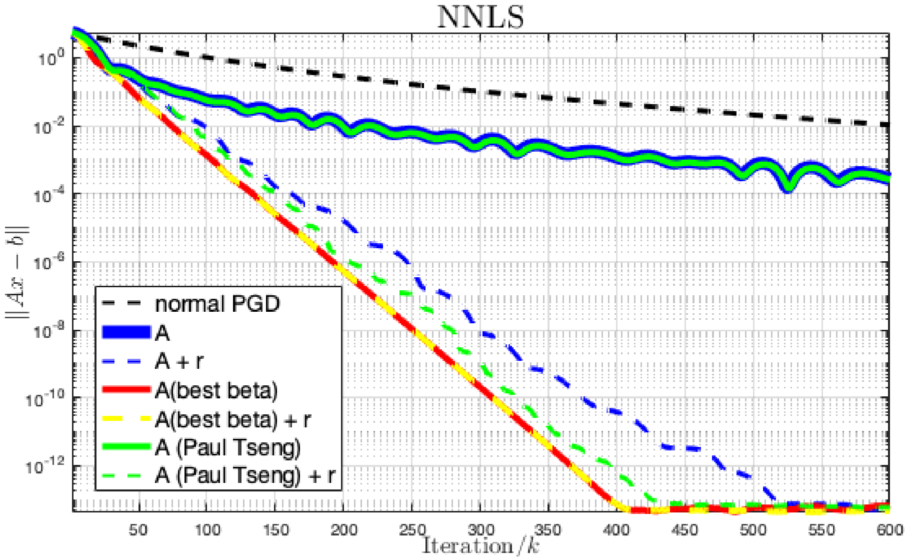


Figure: An illustrative example $(m, n) = 100, 20$. MATLAB code (click me)

Last page - summary

Summary :

- NNLS problem $\min_{\mathbf{x}\in\mathbb{R}^n_+} f(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$
- PGD algorithm for NNLS
- APGD algorithms for NNLS
- APGD algorithm with restart for NNLS

End of document