Universidad de la República
Facultad de Ingeniería

Teoría y Algoritmia de Optimización

Año 2023

Entregable 4

JUAN VARELA

Autor:
· Juan Manuel Varela

Docentes:
Ignacio Ramírez
Matías Valdes

13 de noviembre de 2023

i) He leído y estoy de acuerdo con las Instrucciones especificadas en la carátula obligatorio. ii) He resuelto por mi propia cuenta los ejercicios, sin recurrir a informes de otros compañeros, o soluciones existentes. iii) Soy el único autor de este trabajo. El informe y todo programa implementado como parte de la resolución del obligatorio son de mi autoría y no incluyen partes ni fragmentos tomados de otros informes u otras fuentes, salvo las excepciones mencionadas.

Ejercicio 1 - Métodos proximales

a) Primero se plantea el operador proximal de la función:

$$prox_{\lambda I_C}(\mathbf{x}) = argmin_{\mathbf{z}}\ I_C(\mathbf{z}) + \frac{1}{2\lambda}||\mathbf{x}-\mathbf{z}||_2^2$$

Para minimizar $I_C(\mathbf{z}) + \frac{1}{2\lambda}||\mathbf{x}-\mathbf{z}||_2^2$, $\mathbf{z}$ debe pertenecer a C, ya que de otra forma $I_C(\mathbf{z})$ valdría $+\infty$. Entonces $I_C(\mathbf{z}) = 0$ y el operador proximal pasa a ser:

$$prox_{\lambda I_C}(\mathbf{x}) = argmin_{\mathbf{z}\in C}\ \frac{1}{2\lambda}||\mathbf{x}-\mathbf{z}||_2^2$$

Como la distancia entre dos puntos siempre es positiva, elevarla al cuadrado y multiplicarla por un factor positivo no cambia la ubicación del mínimo. Entonces minimizar la función $\frac{1}{2\lambda}||\mathbf{x}-\mathbf{z}||_2^2$ es equivalente a minimizar $||\mathbf{x}-\mathbf{z}||_2$

Por lo tanto $prox_{\lambda I_C}(\mathbf{x}) = argmin_{\mathbf{z}\in C}\ ||\mathbf{x}-\mathbf{z}||_2 = \Pi_C(\mathbf{x})$

Ejercicio 2 - LASSO

a) Se tiene $g(\mathbf{x}) = \lambda||\mathbf{x}||_1$

Se sabe que si $g(\mathbf{x}) = \sum_{i=1}^n g_i(x_i) \Rightarrow (prox_{\alpha g}(\mathbf{v}))_i = prox_{\alpha g_i}(v_i)$

Esto es válido en particular para este caso, donde $\lambda||\mathbf{x}||_1 = \sum_{i=1}^n \lambda|x_i|$, entonces basta con hallar el operador proximal de la función $g_i(x) = \lambda|x|$

Se procede a calcular $prox_{\alpha g_i}(v)$ tomando $g_i(x) = \lambda|x|$:

Para esto se plantea la condición de optimalidad de $x = prox_{\alpha g_i}(v)$, que es $0 \in \partial g_i(x) + \frac{1}{\alpha}(x - v) \Leftrightarrow v \in x + \alpha\partial g_i(x)$.

Se quiere hallar x, que dado un v, cumpla esa condición. Es decir, se quiere hallar un x, que dado un v, forme un conjunto $x + \alpha\partial g_i(x)$ tal que v pertenezca a él.

Primero se identifica el conjunto $x + \alpha\partial g_i(x)$ para todos los casos posibles:

$$\text{Si } x > 0 \Rightarrow \partial g_i(x) = \{\lambda\} \Rightarrow x + \alpha\partial g_i(x) = \{\alpha\lambda + x\}$$
$$\text{Si } x < 0 \Rightarrow \partial g_i(x) = \{-\lambda\} \Rightarrow x + \alpha\partial g_i(x) = \{-\alpha\lambda + x\}$$
$$\text{Si } x = 0 \Rightarrow \partial g_i(x) = [-\lambda, \lambda] \Rightarrow x + \alpha\partial g_i(x) = [-\alpha\lambda, \alpha\lambda]$$

Entonces se separa por casos posibles de v:

- Si $v > \alpha\lambda$ el único caso posible es que x sea mayor a 0, ya que en los otros casos el conjunto $x + \alpha\partial g_i(x)$ siempre tiene valores menores o iguales a $\alpha\lambda$. Por lo tanto $x + \alpha\partial g_i(x)$ tiene un único elemento que es $\alpha\lambda + x = v$, entonces $x = v - \alpha\lambda$

- Si $v < -\alpha\lambda$ el único caso posible es que x sea menor a 0, ya que en los otros casos el conjunto $x + \alpha\partial g_i(x)$ siempre tiene valores mayores o iguales a $-\alpha\lambda$. Por lo tanto $x + \alpha\partial g_i(x)$ tiene un único elemento que es $-\alpha\lambda + x = v$, entonces $x = v + \alpha\lambda$
- Si $v \in [-\alpha\lambda, \alpha\lambda]$ el único caso posible es que $x = 0$, ya que en los otros casos el conjunto $x + \alpha\partial g_i(x)$ siempre tiene valores fuera del conjunto $[-\alpha\lambda, \alpha\lambda]$.

Por lo tanto, el operador proximal de $g_i(x)$ con parámetro α es:

$$prox_{\alpha g_i}(v) \begin{cases} v - \alpha\lambda & \text{si } v > \alpha\lambda, \\ 0 & \text{si } |v| \leq \alpha\lambda, \\ v + \alpha\lambda & \text{si } v < -\alpha\lambda \end{cases}$$

Finalmente, por la propiedad vista anteriormente se llega a que:

$$prox_{\alpha g}(\mathbf{v}) = (prox_{\alpha g_i}(v_1), ..., prox_{\alpha g_i}(v_n))$$

b) Se tiene $f(\mathbf{x}) = \frac{1}{2}||A\mathbf{x} - b||_2^2$

La condición de optimalidad de $\mathbf{x} = prox_{\alpha f}(\mathbf{v})$ es $0 \in \partial f(\mathbf{x}) + \frac{1}{\alpha}(\mathbf{x} - \mathbf{v}) \Leftrightarrow \mathbf{v} \in \mathbf{x} + \alpha\partial f(\mathbf{x})$.

Como $f(x)$ es diferenciable, el subgradiente de la condición es el gradiente, entonces pasa a ser $\mathbf{v} = \mathbf{x} + \alpha\nabla f(\mathbf{x})$

El gradiente de $f(\mathbf{x})$ es:

$$\nabla f(\mathbf{x}) = \mathbf{A^T}(\mathbf{Ax} - \mathbf{b})$$

Entonces se tiene:

$$\mathbf{v} = \mathbf{x} + \alpha\mathbf{A^T}(\mathbf{Ax} - \mathbf{b})$$

Desarrollando y despejando $\mathbf{x}$:

$$\mathbf{x} + \alpha\mathbf{A^T}\mathbf{Ax} = \alpha\mathbf{A^T}\mathbf{b} + \mathbf{v}$$
$$(\mathbf{I} + \alpha\mathbf{A^T}\mathbf{A})\mathbf{x} = \alpha\mathbf{A^T}\mathbf{b} + \mathbf{v}$$
$$\mathbf{x} = (\mathbf{I} + \alpha\mathbf{A^T}\mathbf{A})^{-1}(\alpha\mathbf{A^T}\mathbf{b} + \mathbf{v})$$

$$\Rightarrow prox_{\alpha f}(\mathbf{v}) = (\mathbf{I} + \alpha\mathbf{A^T}\mathbf{A})^{-1}(\alpha\mathbf{A^T}\mathbf{b} + \mathbf{v})$$

c) Se calcula numéricamente la solución del LASSO con los datos proporcionados utilizando *Proximal Gradient Method*. Para esto se utiliza un paso $\alpha = \frac{1}{||A^T A||_2}$ y se actualiza en cada iteración:

$$\mathbf{x^{k+1}} = prox_{\alpha g}(\mathbf{x^k} - \alpha \nabla f(\mathbf{x^k}))$$

Utilizando como condición inicial $\mathbf{x^0} = \mathbf{0}$ y como condición de parada que la diferencia de la función objetivo entre iteraciones consecutivas sea menor a 0.0001.

El resultado después de 32 iteraciones, en un tiempo de 0.00178 segundos es:

$$\mathbf{x}^* = \begin{bmatrix} -0{,}13283252 \\ 0{,}12737118 \end{bmatrix}$$

El valor final de la función objetivo es 6,71263045

d) Se calcula numéricamente la solución del LASSO con los datos proporcionados utilizando el método ADMM. Para esto se plantea el problema equivalente:

$$min_{x,z}\ \frac{1}{2}||A\mathbf{x} - b||_2^2 + \lambda||\mathbf{z}||_1$$
$$\text{s.a: } \mathbf{x} = \mathbf{z}$$

Y se actualiza en cada iteración:

$$\mathbf{x^{k+1}} = prox_{\alpha f}(\mathbf{z^k} - \mathbf{u^k})$$
$$\mathbf{z^{k+1}} = prox_{\alpha g}(\mathbf{x^{k+1}} + \mathbf{u^k})$$
$$\mathbf{u^{k+1}} = \mathbf{u^k} + \mathbf{x^{k+1}} - \mathbf{z^{k+1}}$$

Utilizando como condición inicial $\mathbf{x^0} = \mathbf{z^0} = \mathbf{u^0} = \mathbf{0}$ y como condición de parada que la diferencia de la función objetivo entre iteraciones consecutivas sea menor a 0.0001.

Los resultados obtenidos para los distintos valores de α utilizados se muestran en la Tabla 1.

	$\alpha = \mathbf{0{,}0001}$	$\alpha = \mathbf{0{,}001}$	$\alpha = \mathbf{0{,}01}$	$\alpha = \mathbf{0{,}1}$
tiempo(s)	0.02492	0.00424	0.00641	0.00236
# iteraciones	37	8	4	3
$\mathbf{x}^*[\mathbf{0}]$	-0.13283263	-0.13283194	-0.13283185	-0.13283183
$\mathbf{x}^*[\mathbf{1}]$	0.1272908	0.12777082	0.12783212	0.12783336
Valor objetivo	6.71269473	6.71246335	6.71246018	6.71246018

Tabla 1: Tiempo de ejecución, número de iteraciones, punto donde se da el mínimo y valor final de la función objetivo para cada alfa utilizado en ADMM.

e) En la Figura 1 se grafica la evolución de la función objetivo de ambos métodos en función de las iteraciones.

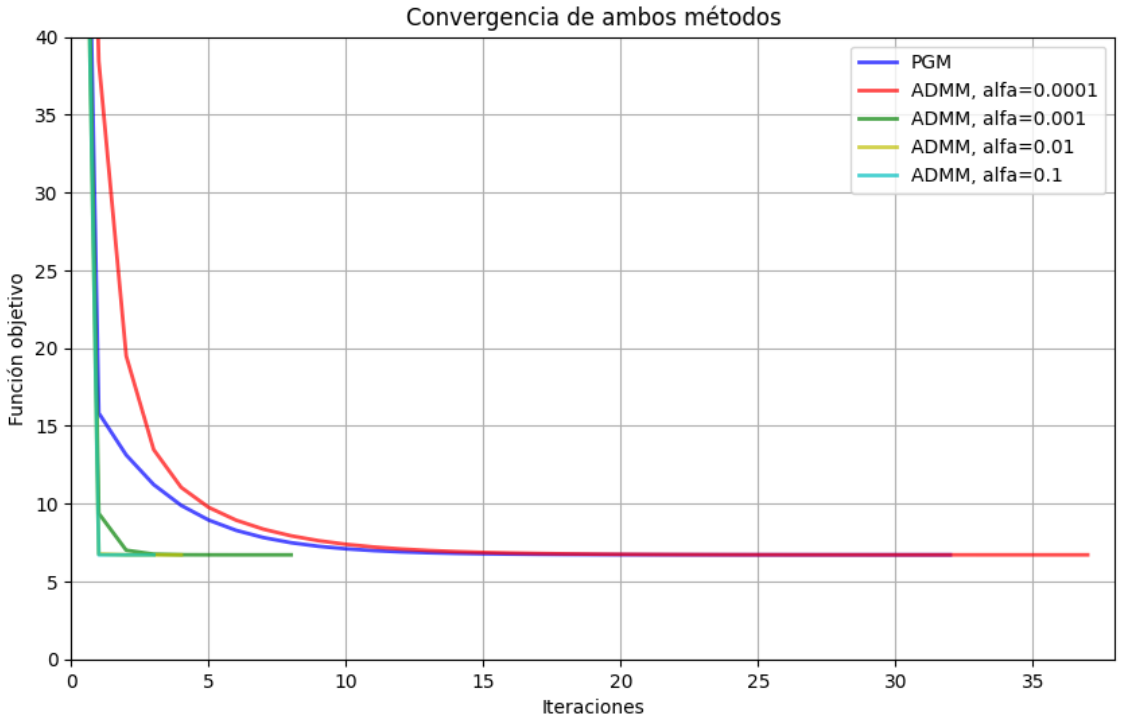

Figura 1: Valor de la función objetivo en cada iteración, por método y alfa

Se puede ver que el PGM converge a la solución suavemente, pero es el que le toma más iteraciones para hacerlo. De todas formas estas iteraciones no son costosas computacionalmente en este caso, y es por esto que el tiempo de ejecución es el menor de todos.

Si se observan los resultados de utilizar el método ADMM, se puede ver que varían mucho en función del parámetro α. Partiendo desde una convergencia muy lenta cuando α es muy chico, con más iteraciones y un tiempo de ejecución un orden mayor a PGM, acelerando la convergencia a medida que se aumenta α, y llegando a una convergencia muy rápida en solo 3 iteraciones cuando α es más grande, pero con un tiempo de ejecución que es algo mayor a PGM, aunque del mismo orden.

Se concluye entonces que si se selecciona un α demasiado chico el desempeño de ADMM no es muy bueno en comparación con PGM, pero si se prueba con varios valores de α se puede hallar uno con el que se converja en muchas menos iteraciones, alcanzando un óptimo igual o mejor que el hallado por PGM. Esto se percibe en el valor final de la función objetivo, que para $\alpha = 0{,}0001$ es algo mayor que el obtenido con PGM, pero para los siguientes α es menor y va decreciendo.