Universidad de la República
Facultad de Ingeniería

Teoría y Algoritmia de Optimización

Año 2023

Tarea Final

JUAN VARELA

Autor:

· Juan Manuel Varela

Docentes:
Ignacio Ramírez
Matías Valdes

19 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 - Puntos de Fekete Logarítmico

a) Se tiene:

$$E(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}) = -\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \log(||\mathbf{x_i} - \mathbf{x_j}||)$$

Modificando la distancia:

$$E_i(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}) = -\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \log(\sqrt{||\mathbf{x_i} - \mathbf{x_j}||^2})$$

Se saca la raiz fuera del logaritmo:

$$E_i(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}) = -\sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{1}{2} \log(||\mathbf{x_i} - \mathbf{x_j}||^2)$$

El gradiente de $\frac{1}{2}\log(||\mathbf{x_i} - \mathbf{x_j}||^2)$ respecto a $\mathbf{x_i}$ es:

$$\nabla_i \left[\frac{1}{2} \log(||\mathbf{x_i} - \mathbf{x_j}||^2) \right] = \frac{1}{2} \frac{1}{||\mathbf{x_i} - \mathbf{x_j}||^2} * 2(\mathbf{x_i} - \mathbf{x_j}) = \frac{\mathbf{x_i} - \mathbf{x_j}}{||\mathbf{x_i} - \mathbf{x_j}||^2}$$

Este término ($\frac{1}{2}\log(||\mathbf{x_i} - \mathbf{x_j}||^2)$) aparece 2 veces en la función de energía para cada combinación $\mathbf{x_i} - \mathbf{x_j}$ posible. Por lo tanto, para un $\mathbf{x_i}$ particular, si se quiere tener en cuenta todos los términos hay que multiplicar el gradiente obtenido por 2 y sumar sobre todos los j posibles.

$$\Rightarrow \nabla_i E(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}) = -\sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{2(\mathbf{x_i} - \mathbf{x_j})}{||\mathbf{x_i} - \mathbf{x_j}||^2}$$

b) Se implementa y prueba el método descripto en la letra. Para esto, se utilizan como puntos iniciales $\mathbf{x_1} = (0, -1, 0)$, $\mathbf{x_2} = (1, 0, 0)$ y $\mathbf{x_3} = (0, 0, 1)$. Con $\alpha = 0,01$ y $\epsilon = 0,001$

En primera instancia se verifica numéricamente la fórmula del gradiente obtenida en la parte anterior. Para hacer esto, se varían una a una las coordenadas de $\mathbf{x_1}$ y se calcula la derivada parcial numérica. Luego se compara con el resultado de aplicar la fórmula anterior a este mismo punto. En ambos casos el resultado es $\nabla_1 E(\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}) = (1, 2, 1)$

Luego, se aplica el método, llegando a los puntos finales.

- $\mathbf{x_1} = (-0{,}38125908, -0{,}8421941, -0{,}38124876)$
- $\mathbf{x_2} = (0{,}84451995, 0{,}37866813, -0{,}37867731)$
- $\mathbf{x_3} = (-0{,}37609914, 0{,}37609613, 0{,}84681824)$

En la Figura 1 se muestra la evolución de la función de energía en función de las iteraciones. Esta gráfica tiene la forma esperada, por lo que se concluye que la implementación funciona correctamente.

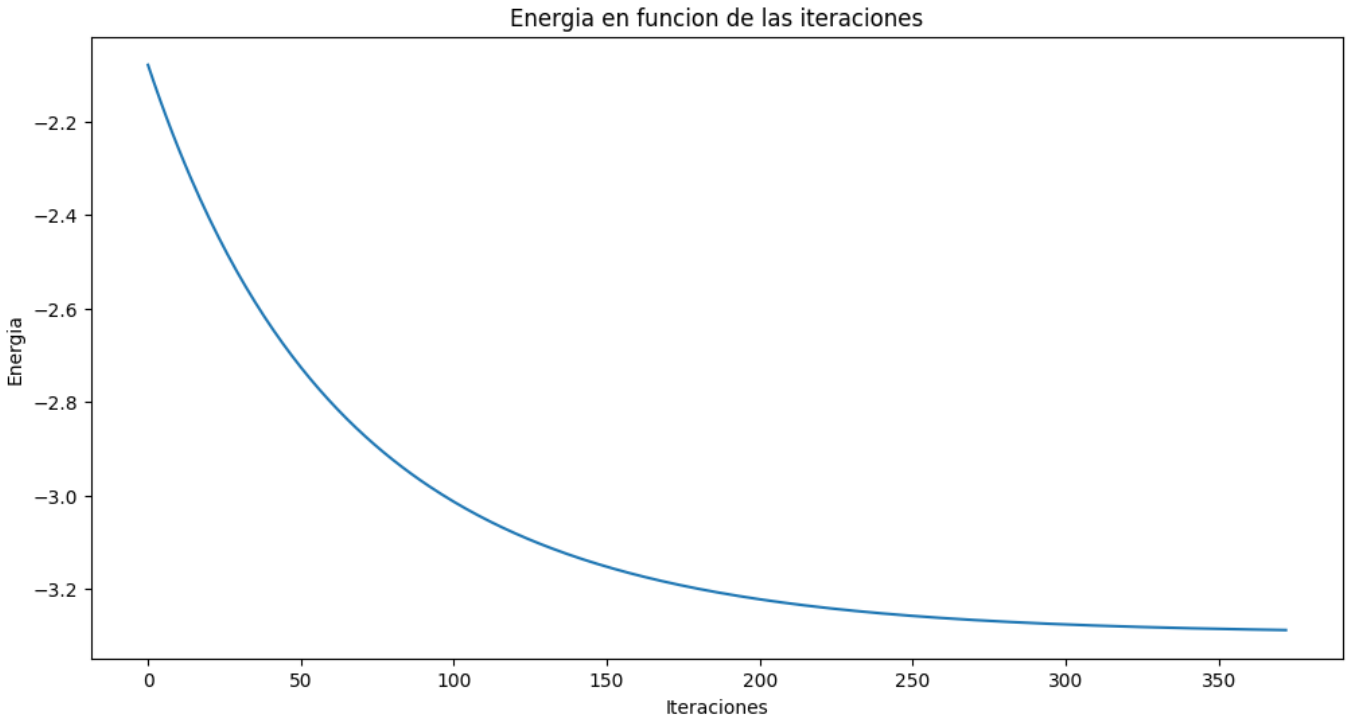

Figura 1: Valor de la energía en cada iteración, prueba.

c) Se prueba el método implementado en la parte anterior con $n = 100$, $\alpha = 0,01$ y $\epsilon = 0{,}01$. Los puntos iniciales se generan con la función proporcionada inicializar_fekete.

- Energía inicial: -1937.2
- Energía final: -2160.7
- Tiempo de ejecución: $\sim$ 36 segundos

d) En la Figura 2 se muestran los puntos al inicio de la ejecución (izquierda) y al final de la ejecución (derecha). Mientras que en la Figura 3 se muestra la evolución de la energía en función de las iteraciones.

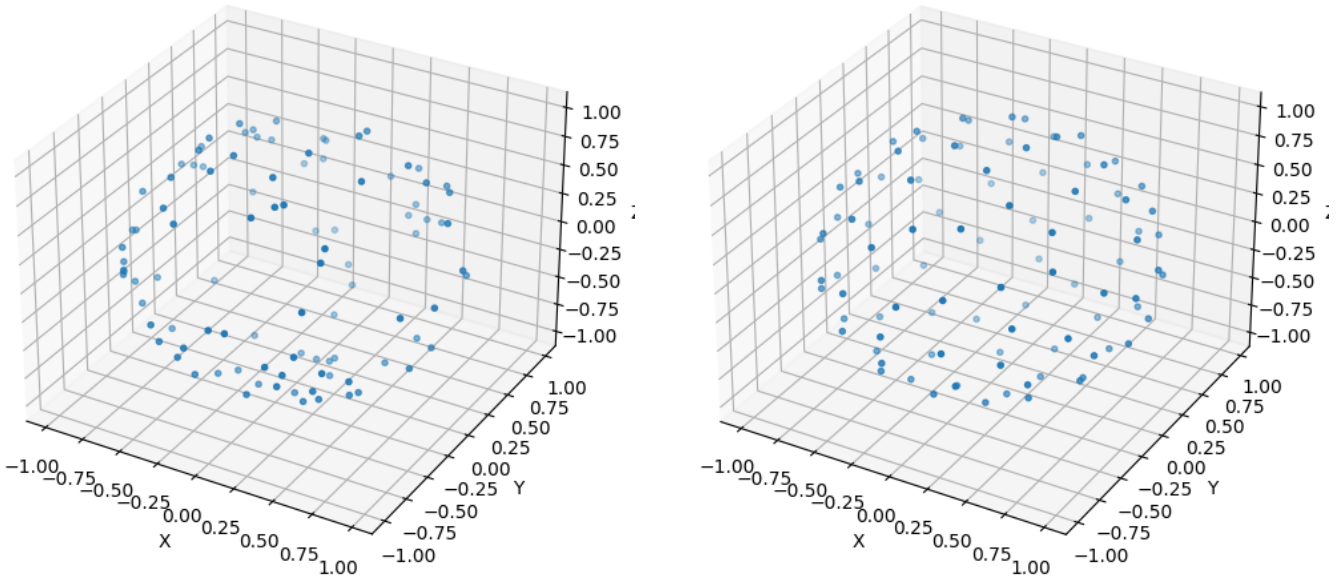

Figura 2: Puntos al inicio y final de la ejecución

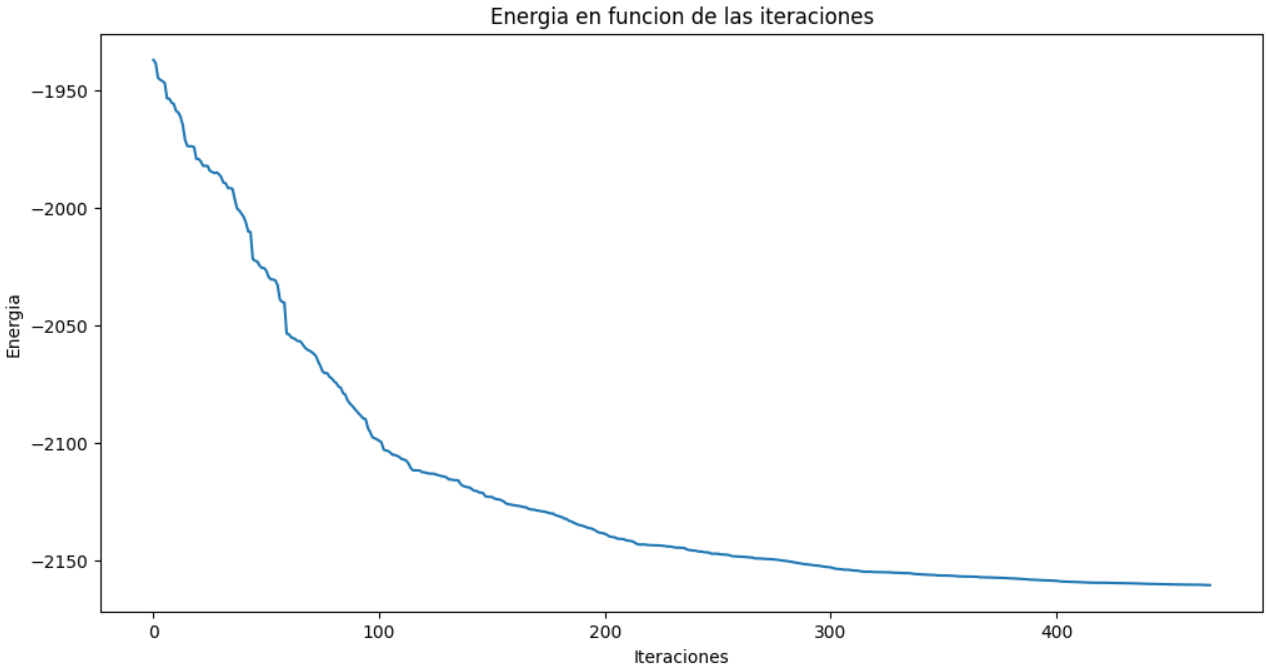

Figura 3: Valor de la energía en cada iteración.

La Figura 3 muestra que la energía disminuye bastante rápido al principio y luego empieza a estabilizarse a medida que se avanza en las iteraciones. Esto indica que el método está logrando llegar a un conjunto de puntos que resulte en un mínimo local de energía.

En la configuración inicial de los puntos, se puede ver que están acumulados en ciertas zonas, mientras que otras están más vacías. En cambio, en la configuración final, los puntos se ven más

uniformemente distribuidos en la esfera. Como la función de energía crece exponencialmente cuanto más cerca están los puntos entre sí, intuitivamente se esperaría que la energía sea menor cuanto más uniforme es la distribución de los puntos, esto es coherente con los resultados obtenidos.

Ejercicio 2 - Otra vez LASSO

a) Se reformula el problema (P0) como un problema de programación cuadrática con restricciones lineales introduciendo las variables t_i con $i = 1, \ldots, p$ y agregando las restricciones necesarias de forma tal que $\theta \sum_{i=1}^{p} t_i$ represente el término de regularización $\theta||\mathbf{x}||_1$. El término de ajuste a datos $\frac{1}{2}||\mathbf{Ax} - \mathbf{y}||_2^2$ se mantiene porque ya es cuadrático.

La no linealidad del término de regularización se da porque toma los valores absolutos de las componentes de $\mathbf{x}$, por lo tanto, las restricciones serán dos por cada valor absoluto (se divide en dos funciones lineales):

$$t_i >= x_i$$
$$t_i >= -x_i$$

Con estas restricciones, es claro que si se minimizan los t_i, se está minimizando el valor absoluto de $\mathbf{x}$.

De esta forma, el problema pasa a ser:

$$(\mathbf{P1})\ \mathbf{x}^* = \arg\min_{\mathbf{x},t_i}\ \frac{1}{2}||\mathbf{Ax} - \mathbf{y}||_2^2 + \theta \sum_{i=1}^{p} t_i$$

sujeto a:

$$t_i >= x_i$$
$$t_i >= -x_i$$
$$\text{con } i = 1, \ldots, p$$

b) Se resuelve numéricamente el problema (P1) para $\mathbf{A}$ e $\mathbf{y}$ generadas con las funciones proporcionadas. Se utiliza CVXPY para la resolución.

El valor de la función objetivo con el $\mathbf{x}^*$ obtenido es 430.0886282518154.

El tiempo de ejecución fue de 0.5832428932189941 segundos.

c) Para utilizar el método de descenso por coordenadas, se actualiza en cada iteración una coordenada del vector $\mathbf{x}$ resolviendo el problema de optimización unidimensional que resulta de fijar todas las otras coordenadas. Para simplificar esto, se separa el problema (P0) en los términos que dependen de la coordenada en cuestión y los que no.

Se tiene:

$$x_i^k = \arg\min_{\zeta} \frac{1}{2}||\mathbf{A}(x_1^{k-1}, \ldots, x_{i-1}^{k-1}, \zeta, x_{i+1}^{k-1}, \ldots, x_p^{k-1}) - \mathbf{y}||_2^2 + \theta||(x_1^{k-1}, \ldots, x_{i-1}^{k-1}, \zeta, x_{i+1}^{k-1}, \ldots, x_p^{k-1})||_1$$

Separando los términos que dependen de ζ:

$$x_i^k = \arg\min_{\zeta} \frac{1}{2}||\sum_{j \neq i} \mathbf{A_j} x_j^{k-1} + \mathbf{A_i}\zeta - \mathbf{y}||_2^2 + \theta \sum_{i \neq j} |x_j^{k-1}| + \theta|\zeta|$$

Como $\theta \sum_{i \neq j} |x_j^{k-1}|$ es una constante se puede eliminar sin modificar la solución al problema. Si además se define el vector constante $\mathbf{c_i} = \mathbf{y} - \sum_{j \neq i} \mathbf{A_j} x_j^{k-1}$, el problema pasa a ser:

$$x_i^k = \arg\min_{\zeta} \frac{1}{2}||\mathbf{A_i}\zeta - \mathbf{c_i}||_2^2 + \theta|\zeta|$$

Para hallar el mínimo de la parte diferenciable, se calcula su gradiente y se iguala a 0. El gradiente es $\mathbf{A_i^T}(\mathbf{A_i}\zeta - \mathbf{c_i})$.

$$\Rightarrow \zeta' = \frac{\mathbf{A_i^T c_i}}{\mathbf{A_i^T A_i}}$$

Este mínimo no tiene en cuenta el término de regularización $\theta|\zeta|$, para tenerlo en cuenta, se aplica a ζ' el operador proximal de la función valor absoluto, que fue calculado en el obligatorio 4 (soft-thresholding con parámetro θ).

$$\Rightarrow \zeta = signo(\zeta').max(|\zeta| - \theta, 0)$$

A partir de esto, se resuelve numéricamente el problema (P0) para $\mathbf{A}$ e $\mathbf{y}$ generadas anteriormente con las funciones proporcionadas. Se utiliza el método de descenso por coordenadas para la resolución, actualizando en cada iteración una coordenada (hallada según los cálculos detallados previamente). Para que el algoritmo no se detenga antes de lo esperado, se verifica el criterio de parada una vez que se actualizaron todas las coordenadas del vector.

El valor de la función objetivo con el x^* obtenido es 430.0886271828191.

El tiempo de ejecución fue de 0.047321319580078125 segundos.

El número de iteraciones fue 800.

d) Se resuelve numéricamente el problema (P0) para $\mathbf{A}$ e $\mathbf{y}$ generadas anteriormente con las funciones proporcionadas. Se utiliza el ADMM implementado en el Obligatorio 4, probando con distintos valores del parámetro λ con el fin de obtener el mejor rendimiento posible en este método.

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

λ	**tiempo(s)**	**# iteraciones**	**Valor objetivo**
0,001	4.888044118881226	2000 (Max)	432.1942853132777
0,01	3.2495710849761963	820	430.08947458928264
0,1	0.17964720726013184	105	430.08871019795487
1	0.031450510025024414	19	430.0886397891583
10	0.07867074012756348	45	430.0887173772972
100	0.4307820796966553	231	430.089437067307
1000	1.6273303031921387	946	430.0942846547439

Tabla 1: Tiempo de ejecución, número de iteraciones y valor final de la función objetivo para cada *lambda* utilizado en ADMM.

A partir del tiempo de ejecución, el número de iteraciones y el valor final de la función objetivo, se puede ver que el mejor resultado se da con $\lambda = 1$, este será el resultado que se comparará con los demás metodos.

e) En la Tabla 2 se muestra un resumen con los resultados obtenidos para los tres métodos.

Método	**tiempo(s)**	**Valor objetivo**	# Ceros $\mathbf{x}^*$	Norma L1 $\mathbf{x}^*$
CVXPY	0.3931558132171631	430.0886282518154	6	89.81871483741263
CDM	0.05101656913757324	430.0886271828192	6	89.81834373763051
ADMM	0.034265756607055664	430.0886397891583	6	89.81322126277186

Tabla 2: Tiempo de ejecución, valor final de la función objetivo, número de ceros y norma L1 de la solución para cada método.

En la Tabla 3 se muestran los primeros 10 elementos de $\mathbf{x}^*$ para cada método.

Indice	**CVXPY**	**CDM**	**ADMM**
0	0.297682688	0.29765056	0.297597172
1	0.442441129	0.44246285	0.442260553
2	0.000012959	0.	0.000043998
3	-0.258671811	-0.25861608	-0.258624776
4	0.310903242	0.3108792	0.310894793
5	1.09124969	1.09127633	1.09130836
6	0.266780651	0.26671502	0.266546090
7	-2.28603573	-2.28600907	-2.28567663
8	0.178214204	0.17809847	0.178288998
9	0.474904305	0.47492698	0.474745010

Tabla 3: Primeros 10 elementos de $\mathbf{x}^*$ para cada método.

Analizando los tiempos de ejecución se puede ver que el método ADMM es el más rápido, seguido de cerca por el método de descenso por coordenadas, y finalmente la resolución mediante CVXPY es la que insume más tiempo, de un orden mayor a los otros dos métodos.

Si se compara los valores finales de la función de costo, se puede ver que la solución alcanzada en todos los casos es muy similar, ya que las diferencias entre estos valores son mínimas. De todas formas, cabe mencionar que la solución obtenida mediante descenso por coordenadas fue la de valor más bajo.

Para evaluar la esparcidad de las soluciones, se contó el número de entradas nulas. En todos los casos son 6 (es esperable ya que las soluciones son muy similares como se puede ver para las primeras 10 entradas en la Tabla 3). Para realizar este conteo, se consideraron nulas las entradas con valor menor a 10^-3, ya que en realidad el único caso con entradas estrictamente nulas es la solución obtenida mediante descenso por coordenadas.

También se compara la norma L1 de las soluciones (que es parte de lo que se quiere reducir). Se puede ver que en los tres casos es muy similar, siendo la obtenida mediante ADMM la menor.

A partir de estos resultados, se puede concluir que el método ADMM es muy útil para la resolución de problemas de optimización de este tipo, debido a la buena calidad de soluciones obtenidas y el bajo tiempo de ejecución. De todas formas, el método de descenso por coordenadas tuvo un desempeño sorprendentemente bueno a pesar del alto número de iteraciones requeridas.