Proyecto final. Optimización Numérica Polígono chico de área máxima

Roberto Iván López López, Adolfo Martínez, Erneso Ulloa Pérez, José Alonso Solís Lemus

***Abstract*—Este trabajo sirve como proyecto final para el curso de Optimización Numérica. Consiste en resolver un problema de optimización numérica con base en los métodos y algoritmos utilizados durante el curso. El problema tomado de [2] consiste en encontrar el polígono de área máxima que tenga un diámetro acotado en 1. Inicialmente se describe y plantea el problema, después se especifica el algoritmo utilizado para la resolución y las adecuaciones hechas al algoritmo para mejorar su desempeño dependiendo el tamaño del problema. Finalmente se muestra una comparación de la solución al problema con MATLAB.**

***Index Terms*—Optimización Numérica, Puntos Interiores, Polígono chico de área máxima.**

I. Introducción

Para este trabajo, se tomó el problema 1 de [2] para resolver con el método de puntos interiores. En esta sección discutimos el problema y deducimos con precisión la función objetivo, así como las restricciones. Después, se plantea el problema como uno de minimización y se presentan las variables como se van a trabajar en los métodos (sección II). El método con el cual se solucionó el problema, así como las adecuacionesy sutilezas dentro del método que se tomaron en cuenta se encuentran en la sección III. Por último se presentan las gráficas de los resultados y se concluye el trabajo con algunos puntos que se notaron durante el desarrollo del proyecto.

A. Descripción del Problema

El problema consiste en encontrar el polígono de área máxima, entre todos los polígonos posibles dados n lados y diámetro $d \leq 1$. Se define al diámetro de un polígono como la máxima distancia entre dos de sus vértices. Este se considera un problema clásico, en el cual se mostró [2] que la solución óptima es regular para n impar y no regular para n par, además de que conforme la $n \to \infty$, esperaríamos que el área del polígono se aproxima al área de un círculo unitario $\pi/4 \approx 0.7854$.

El número de mínimos locales para este problema es al menos $O(n!)$, por lo que se espera que los algoritmos sólo encuentren soluciones locales.

B. Deducción de la función objetivo y las restricciones

Para el planteamiento del problema vamos a tomar los n vértices del polígono los vamos a escribir en coordenadas polares (r_i, θ_i). El área del polígono se calcula sumando las áreas de los $n-2$ triángulos que se forman con los vértices $\{(0,0), (r_i, \theta_i), (r_{i+1}, \theta_{i+1})\}$. El ángulo que subtiende a dichos vértices está dado por $\theta_{i+1} - \theta_i$. En la figura 1 se muestra un ejemplo de

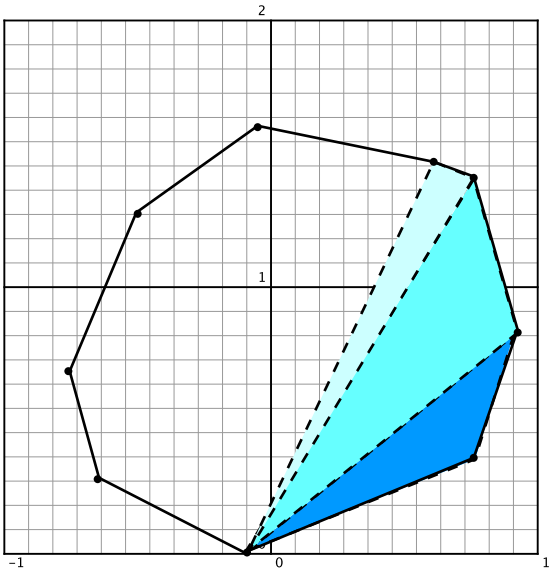

Fig. 1. Ejemplo del cálculo del área.

Si tomamos como base de estos triángulos al lado de longitud r_i, entonces es fácil ver que la altura para diche base va a ser $r_{i+1}\sin(\theta_{i+1} - \theta_i)$, por lo que el área de dicho triángulo será

$$A_i = \frac{1}{2} r_{i+1} r_i \sin(\theta_{i+1} - \theta_i) \tag{1}$$

La función objetivo -a maximzar- para este problema queda entonces como sigue:

$$f(r, \theta) = \frac{1}{2} \sum_{i=1}^{n-1} r_{i+1} r_i \sin(\theta_{i+1} - \theta_i) \tag{2}$$

Las restricciones las podemos deducir fácilmente a partir de la descripción de las vriables que hicimos anteriormente. En primer lugar, vamos a requerir que el diámetro sea menos o igual a uno. Entonces, sea $j > i$, y tomamos el triángulo de vértices $\{(0,0), (r_i, \theta_i), (r_j, \theta_j)\}$, el ángulo que sale del origen es $\theta_j - \theta_i$. Si llamamos d_{ij} a la distancia entre los vértices i y j, utilizando la ley de cosenos obtenemos que el cuadrado entre cualesquiera dos vértices va a estar dada por $d_{ij}^2 = r_i^2 + r_j^2 - 2 r_i r_j \cos(\theta_j - \theta_i)$. De donde nuestra primera restricción queda como

$$r_i^2 + r_j^2 - 2 r_i r_j \cos(\theta_j - \theta_i) \leq 1 \tag{r1}$$

Como se muestra en la figura 1, vamos a posicionar nuestro polígono en el primer y segundo cuadrantes. Esto agrega un par de restricciones, ya que

$$0 \leq \theta \leq \pi \tag{r2}$$

$$0 \leq r \leq 1 \tag{r3}$$

Por último, los ángulos deben incrementarse conforme la i crece, por lo que agregamos la restricción

$$\theta_i \leq \theta_{i+1} \tag{r4}$$

II. Planteamiento del Problema

Tradicionalmente, los algoritmos de optimización se dedican al siguiente tipo de problema

$$\begin{aligned} \underset{x}{\text{minimizar}} \quad & f(x) \\ \text{sujeto a} \quad & g(x) \geq 0 \end{aligned} \tag{3}$$

En I-B se encontró la función objetivo y las restricciones, de manera que el problema quedaría escrito de la siguiente forma

$$\begin{aligned} \underset{(r,\theta)}{\text{Maximizar}} \quad & \frac{1}{2}\sum_{i=1}^{n-1} r_{i+1} r_i \sin(\theta_{i+1} - \theta_i) \\ \text{sujeto a} \quad & r_i^2 + r_j^2 - 2r_i r_j \cos(\theta_j - \theta_i) \leq 1 \\ & 0 \leq \theta \leq \pi \\ & 0 \leq r \leq 1 \end{aligned} \tag{4}$$

Es claro que tenemos que reescribir el problema para que se acoplen a los algoritmos, por lo que finalmente, el problema con el que vamos a tratar es el siguiente:

$$\begin{aligned} \underset{(r,\theta)}{\text{minimizar}} \quad & -\frac{1}{2}\sum_{i=1}^{n-1} r_{i+1} r_i \sin(\theta_{i+1} - \theta_i) \\ \text{sujeto a} \quad & 1 - (r_i^2 + r_j^2 - 2r_i r_j \cos(\theta_j - \theta_i)) \geq 0 \\ & \theta_{i+1} - \theta_i \geq 0 \; 1 \leq i < n \\ & \theta \geq 0 \\ & \pi - \theta \geq 0 \\ & r \geq 0 \\ & 1 - r \geq 0 \end{aligned} \tag{5}$$

Para la primera restricción, i.e. la del diámetro menor a 1, vemos que debe hacerse para todas las i, j que cumplan $1 \leq i < j \leq n$. Finalmente, vamos a fijar el último punto $(r_n, \theta_n) = (0, \pi)$, no se agrega como restricción, sino que se toma en cuenta en cada iteración. En la tabla I vemos los datos para este problema.

TABLE I
DATOS DEL PROBLEMA

Variables	$2(n-1)$
Restricciones	(n/2 + 1)(n-1) -1
Cotas	2(n-1)
Restricciones de igualdad	0
Restricciones lineales de desigualdad	n-2
Restricciones no lineales de desigualdad	n/2(n-1)

III. Solución con puntos interiores

El método general de puntos interiores consiste en analizar las condiciones necesarias de primer orden (CNPO) de algún problema en particular. Entonces, sean $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$ con $f, g \in \mathcal{C}^3(\mathbb{R}^n)$ que definen el problema (3), Entonces, ya que la función $g(x)$ puede presentar no linealidades muy marcadas, vamos a introducir la variable $z \in \mathbb{R}^p$ reescribiendo el problema (3) de la siguiente manera

$$\begin{aligned} \underset{x}{\text{minimizar}} \quad & f(x) \\ \text{sujeto a} \quad & g(x) - z = 0 \\ & z \geq 0 \end{aligned} \tag{6}$$

El problema se ha simplificado en el sentido que las restricciones de desigualdad son lineales. Por notación, vamos a definir las siguientes variables

$$Z = \operatorname{diag}(z) \tag{7a}$$

$$U = \operatorname{diag}(\mu) \tag{7b}$$

$$\omega = (x, z, \mu)^T \tag{7c}$$

$$e = (1, 1, \cdots, 1)^T \in \mathbb{R}^p \tag{7d}$$

Entonces, definimos la función $F : \mathbb{R}^{n+2p} \to \mathbb{R}^{n+2p}$, $F \in \mathcal{C}^2(\mathbb{R}^{n+2p})$ de la siguiente forma:

$$F(\omega) = \begin{pmatrix} \nabla_x \mathcal{L}(\omega) \\ z - g(x) \\ UZe \end{pmatrix} \tag{8}$$

Las CNPO nos dicen que vamos a buscar en cada iteración vamos a buscar una ω_k tal que $F(\omega_k) = 0$, esto nos sugiere que una forma de atacar el problema por medio del método de Newton, de tal forma que el paso que se dé en la iteración k-ésima $\Delta\omega_k$ va a estar dado por la ecuación

$$F'(\omega_k)\Delta\omega_k = -F(\omega_k) \tag{9}$$

La matriz $F'(\omega)$ se ve de la siguiente forma

$$F'(\omega) = \begin{pmatrix} \nabla^2_{xx}\mathcal{L}(\omega) & -\nabla g(x)^T & 0 \\ \nabla g(x) & 0 & I_p \\ 0 & Z & U \end{pmatrix} \tag{10}$$

El método de puntos interiores consiste en empezar con un punto $x_0 \in \Omega^0 = \{x \in \mathbb{R} | g(x) > 0\}$, y dado que el paso de Newton podría causar que la matriz (12) se vuelva mal condicionada, introducimos el parámetro $\gamma \in (0, 1)$ a la función F

$$F_\gamma(\omega) = \begin{pmatrix} \nabla_x \mathcal{L}(\omega) \\ z - g(x) \\ UZe - \gamma e \end{pmatrix} \tag{11}$$

La idea es que conforme las iteraciones avanzan ($k \to \infty$), el parámetro $\gamma \to 0$ y en consecuencia, se llegue al óptimo $\omega^\star$ en el cual $F(\omega^\star) = 0$.

Con las consideraciones anteriores, escribimos el algoritmo en la tabla II. Cabe mencionar que se va generar una trayectoria

de puntos óptimos a los problemas que se vayan resolviendo en cada iteración $\omega^\star(\gamma_k)$ tal que

$$\lim_{k\to\infty} \omega^\star(\gamma_k) = \omega^\star$$

TABLE II
PSEUDO CÓDIGO DEL ALGORITMO UTILIZADO

In:		$x_0 \in \Omega^0, f : \mathrm{R}^n \to \mathrm{R}, g : \mathrm{R}^n \to \mathrm{R}^p$
1.		$z_0 \leftarrow e,\ \mu_0 \leftarrow e, \gamma_0 \leftarrow 1,\ e = (1, 1, \cdots, 1)^T$ $\omega_0 = (x_0, z_0, \mu_0)^T$
2.		**while** !CNPO
	2.1	Resolver: $F'_\gamma(\omega_k)\Delta\omega_k = -F_\gamma(\omega_k)$
	2.2	Escoger $\rho_k \in (0, 1]$, tal que $z_k + \rho_k\Delta z_k > 0$ y $\mu_k + \rho_k\Delta\mu_k > 0$
	2.3	Actualizar: $\omega_{k+1} \leftarrow \omega_k + \rho_k\Delta\omega_k$ $\gamma_{k+1} \leftarrow \gamma_k/10$

A. Adecuaciones al método

La tabla II muestra únicamente el algoritmo de puntos interiores, sin embargo, no muestra cómo atacar al sistema $F'_\gamma(\omega_k)\Delta\omega_k = -F_\gamma(\omega_k)$, cómo escoger la siguiente ρ_k que recorta al paso. También es cierto que calcular numéricamente la Hessiana $\nabla^2_{xx}\mathcal{L}(\omega_k)$ puede llevar a malos condicionamientos de la matriz. En esta sección se muestra la forma elegida para este problema.

1) Solución alternativa al sistema lineal: Analicemos el sistema lineal $F'_\gamma(\omega_k)\Delta\omega_k = -F_\gamma(\omega_k)$ expandiendolo como sigue

$$\begin{pmatrix} \nabla^2_{xx}\mathcal{L}(\omega) & -\nabla g(x)^T & 0 \\ \nabla g(x) & 0 & I_p \\ 0 & Z & U \end{pmatrix} \begin{pmatrix} \Delta x_k \\ \Delta\mu_k \\ \Delta z_k \end{pmatrix} = \begin{pmatrix} \nabla_x\mathcal{L}(\omega) \\ z - g(x) \\ UZe - \gamma e \end{pmatrix} \tag{12}$$

Haciendo sustituciones [1], se llega a la solución a partir del siguiente sistema de ecuaciones para Δx_k

$$\begin{aligned}(B_k + C_k^T Z_k^{-1} U_k C_k)\Delta x_k = \nabla_x\mathcal{L}(\omega, k) \\ -C_k^T Z_k^{-1} U_k(z_k - g(x_k)) \\ +C_k^T Z_k^{-1}(U_k Z_k e - \gamma_k e)\end{aligned}$$

donde B_k es una matriz simétrica, positiva definida obtenida por un método cuasi-Newton $C_k = \nabla g(x_k)$, $U_k = diag(\mu_k)$, $Z_k = diag(z_k)$.

2) Actualización BFGS y el parámetro σ_n: Como se mostró en el curso, y se muestra en [1], estimar en cada iteración del método la Hessiana del lagrangiano resulta en una tarea poco eficiente y que puede llevar a soluciones poco confiables debido malos condicionamientos de la matriz. La matriz B_k se escoge con s_k, y_k que satisfacen la ecuación de la secante.

$$\begin{aligned} s_k &= x_{k+1} - x_k \\ y_k &= \nabla_x\mathcal{L}(x_{k+1}, \mu_{k+1}) - \nabla_x\mathcal{L}(x_k, \mu_{k+1}) \end{aligned}$$

Definimos $r_k = \theta_k y_k + (1 - \theta_k)B_k s_k$ donde

$$\theta_k = \begin{cases} 1 & s_k^T y_k \geq \sigma_n s_k^T B_k s_k \\ (1 - \sigma_n)\frac{s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k} & \text{e.o.c} \end{cases}$$

IV. RESULTADOS

A continuación mostramos algunas imágenes que reflejan la calidad de los resultados usando nuestro algoritmo y el de Matlab.

La norma de las CNPO sí es relativamente chica para los polígonos de tamaño 5 a 13; únicamente en el caso $n = 4$ no se obtuvo una norma pequeña.

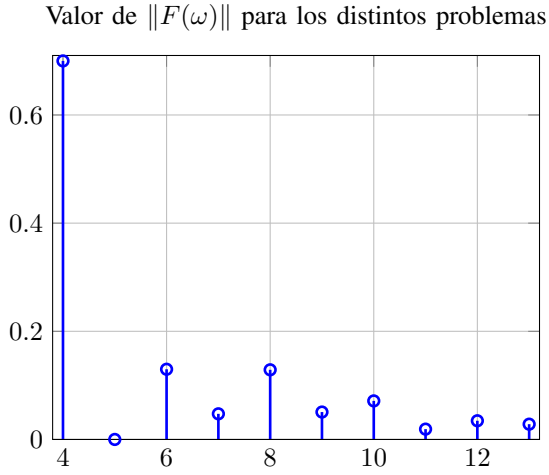

Fig. 2. Valor de las CNPO para los distintos tamaños

Los valores de la función objetivo en el punto óptimo son muy parecidos a los que MATLAB obtiene.

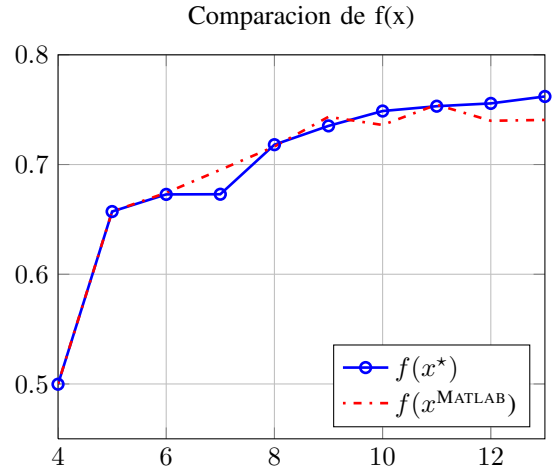

Fig. 3. Comparación de los valores de la función objetivo

A continuación mostramos las figuras de los polígonos obtenidos por nosotros (en verde) y los de la solución de MATLAB (azul). Notemos que son prácticamente los mismos salvo que se encuentran ligeramente rotados. Por último, en la figura 10 se muestra un ejemplo de cómo evolucionó la función objetivo a lo largo del método.

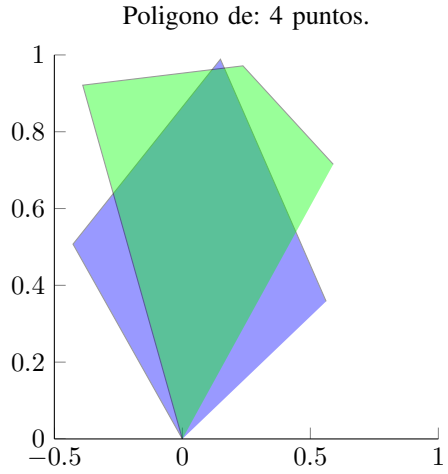

Fig. 4. Polígono $n = 4$

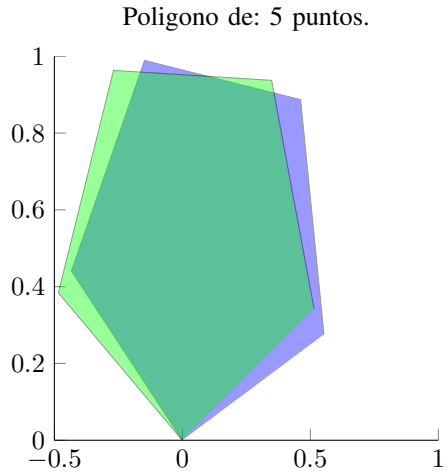

Fig. 5. Polígono $n = 5$

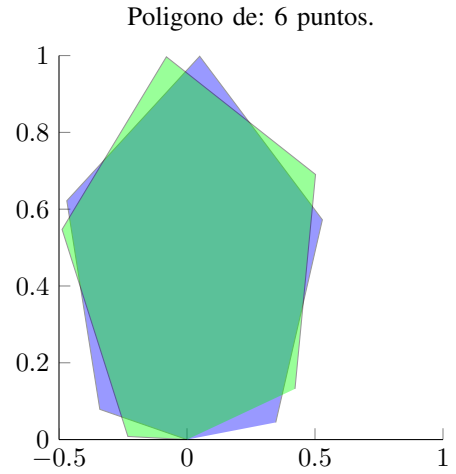

Fig. 6. Polígono $n = 6$

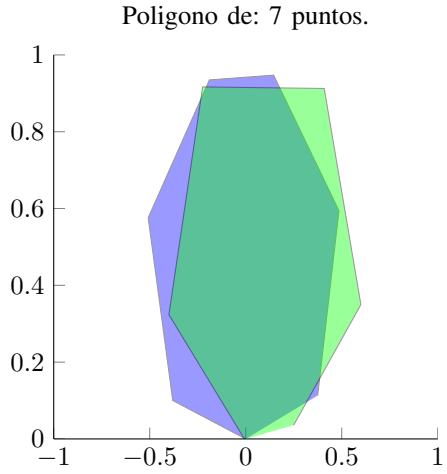

Fig. 7. Polígono $n = 7$

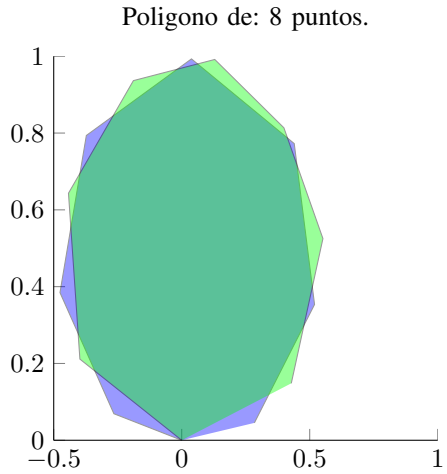

Fig. 8. Polígono $n = 8$

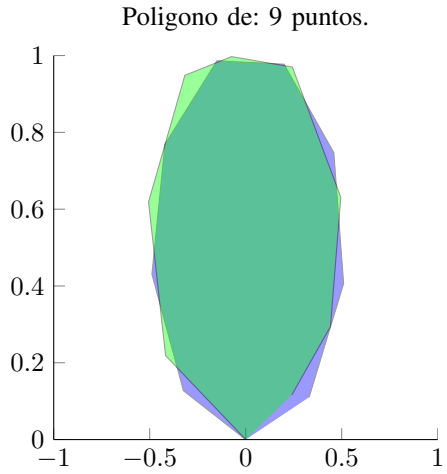

Fig. 9. Polígono $n = 9$

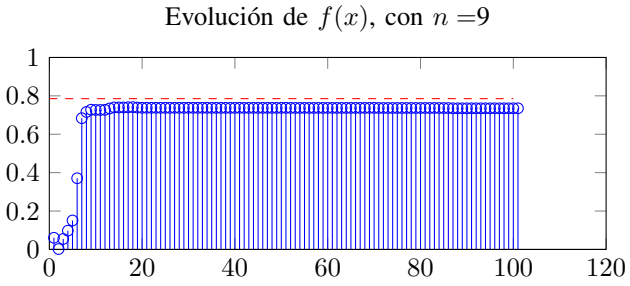

Fig. 10. Evolución de la función objetivo $n = 9$

V. Conclusiones

Durante el desarrollo del proyecto pudimos ver que la implementación del algoritmo es un proceso delicado que requiere de mucho cuidado cuando se trata de ajustar valores a los parámetros como, por ejemplo, la sucesión de $\gamma_0, ...\gamma_k$ o bien la elección del punto inicial x_0. Es importante también tratar de reducir lo más que se pueda el cálculo de operaciones ya que es muy costoso computacionalmente hacer procesos ineficientemente.

References

[1] Jorge Nocedal & Stephen J. Wright (2010) *Numerical Optimization*, 2^{nd} Edition, Springer

[2] E.D. Dolan & Jorge J. Moré & Todd S. Munson (2004) *Benchmarking Optimization Software with COPS 3.0*, Technical Report

[3] *Notas de clase* (Primavera 2013) Optimización Numérica, ITAM.