Laboratorio 1

Optimización No Lineal, Optimización 2

Thomas Martinod[1]

[1]Ingeniería Matemática, Universidad EAFIT, Colombia, *tmartinods@eafit.edu.co*

Resumen

El presente documento corresponde a la entrega del primer laboratorio del curso de Optimización 2 impartido en la Universidad EAFIT. En el presente, se resuelven computacionalmente dos problemas de optimización no lineal, el primero sin restricciones y el segundo con restricciones. Para el primer problema se exponen los métodos de Newton, Gradiente Descendiente y Levenberg-Marquardt, mientras que para el segundo se emplea un método de barrera interior con penalización logarítmica. Se verifica que todos los métodos alcanzan los óptimos teóricos bajo los parámetros de cada modelo.

Nota: En este documento no se exponen los algoritmos desarrollados por cuestiones de longitud. Estos códigos desarrollados en Jupyter Notebooks `.ipynb` se encuentran en el repositorio de Github https://github.com/thomas-martinod/Optimizacion-2 en la carpeta **lab1**. Los resultados de este laboratorio siguiendo estos códigos son replicables.

1. Optimización sin restricciones

Problema 1. Minimizar la función $f(x) = x_1 e^{-{x_1}^2-{x_2}^2}$ con el método de Newton y comparar sus resultados con dos métodos de los revisados en clase que consideren la búsqueda basada en el gradiente.

La gráfica de la función que se pretende minimizar en $x = (x_1, x_2) \in \mathbb{R}^2$ se enseña en la figura 1. Como anotaciones, el mínimo global se halló teóricamente (verificado con WolframAlpha) en $x^* = (-1/\sqrt{2}, 0)$ donde la función alcanza un valor de $f(x^*) = -1/\sqrt{2e}$.

A continuación se expone el método de Newton para resolver este PPNL, para luego comparar los resultados de este algoritmo con los obtenidos por los métodos del gradiente descendiente y Levenberg-Marquardt.

1.1. Método de Newton

Sean $x^0 \in \mathbb{R}^n$ un punto de búsqueda inicial (idealmente cercano al óptimo), ε una tolerancia, y J el número de iteraciones máximas permitidas por el algoritmo. La descripción del algoritmo de Newton en cualquier iteración $k \in \mathbb{N} \cup \{0\}$ viene dada por:

- **Paso 1.** Calcular el gradiente $\nabla f(x^k)$, la norma del gradiente $|\nabla f(x^k)|$, la matriz Hessiana $\mathbf{H}[f(x^k)]$ y su inversa $\mathbf{H}[f(x^k)]^{-1}$ para la k-ésima iteración.
- **Paso 2. Criterios de parada:** Si $\varepsilon > |\nabla f(x^k)|$ o $k > J$ el algoritmo se detiene y $x^k = x^*$ es el mínimo. En otro caso, ir al paso 3.

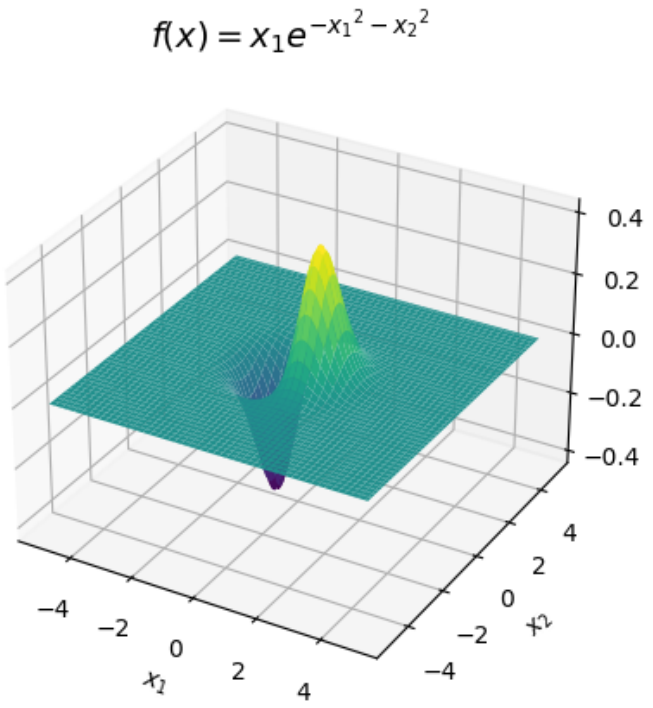

Figura 1: Gráfica de $f(x) = x_1 e^{-x_1^2 - x_2^2}$

- **Paso 3.** Calcular la relación de recurrencia,

$$x^{k+1} = x^k - \mathbf{H}[f(x^k)]^{-1} \nabla f(x^k) \tag{1}$$

  y continuar la iteración siguiente con $k := k + 1$.

Computacionalmente se tomaron $\varepsilon = 10^{-15}$, $J = 100$ y $x^0 = (-0.6, -0.3)$ como los parámetros del algoritmo. Las iteraciones que presenta este a la hora de hallar el mínimo se presentan en la figura 2.

	Iteration	x1	x2	f(x)	|∇f(x)|
0	0	-0.600000	-3.000000e-01	-0.382577	2.908032e-01
1	1	-0.726126	8.738739e-02	-0.425314	8.090296e-02
2	2	-0.706530	-1.485944e-03	-0.428881	1.613678e-03
3	3	-0.707107	8.538448e-09	-0.428882	3.980727e-07
4	4	-0.707107	-1.837995e-21	-0.428882	6.528111e-14
5	5	-0.707107	0.000000e+00	-0.428882	1.110223e-16

Figura 2: Iteraciones del método de Newton para hallar x^*.

El algoritmo finalmente arroja $x^* = (-0.707107, 0) \approx (-1/\sqrt{2}, 0)$ cuando $|\nabla f(x^*)| < \varepsilon = 10^{-15}$.

Conclusiones

- El método de Newton, aunque garantiza convergencia cuadrática (y por ende solo tarda 6 iteraciones en converger), requiere de la hipótesis fuerte de que el punto inicial x^0 esté cercano, o en

una vecindad de x^* (o en general del óptimo local que se busca). Se vió lo delicado que es el algoritmo en cuanto a la selección del punto inicial cuando se seleccionaron $(1,1),(-1,-1),(-1,0.5)$ y demás como posibles puntos iniciales. En estos casos el algoritmo converge a una solución diferente, que no resultan ni máximos ni mínimos locales.

- Ahora, el método converge y no diverge por la forma de la función. Al tratarse del producto de dos exponenciales negativas centradas en el origen, que trazan una superficie que conforme el algoritmo se aleja del origen, $f(x)$ se va a 0 con pendientes muy pequeñas en cualquier dirección. Como esta pendiente es muy pequeña, se cumple el criterio $|\nabla f(x^k)| < \varepsilon$ sin haber ningún óptimo, y sin evaluar que en cualquier bola de radio $r > 0$ centrada en x^k hay puntos en los que $f(x') < f(x^k)$.
- Finalmente, el método de Newton también es excepcional en casos en que la función sea diferenciable (de clase C^2), en todo su dominio, puesto que requiere del cálculo del gradiente y la matriz Hessiana. En caso de que $f \notin C^2$, deben buscarse métodos alternativos para la resolución del problema.

1.2. Gradiente Descendiente

Sean $x^0 \in \mathbb{R}^n$ un punto de búsqueda inicial, ε una tolerancia pre-especificada, y J el número de iteraciones máximas permitidas por el algoritmo. La descripción del método del gradiente descendiente en cualquier iteración $k \in \mathbb{N} \cup \{0\}$ viene dada por:

- **Paso 1.** Calcular el gradiente de f en x^k, $\nabla f(x^k)$, el valor de f en ese punto $f(x^k)$ y definir la dirección de búsqueda $s^k = -\nabla f(x^k)$.
- **Paso 2.** Hallar la longitud de paso de la k-ésima iteración λ^k que minimice $g(\lambda^k) = f(x^k + \lambda^k s^k)$. (Line-search)
- **Paso 3.** Usar la relación de recurrencia,

$$x^{k+1} = x^k + \lambda^k s^k \tag{2}$$

  para calcular el punto siguente.
- **Paso 4. Criterio de parada:** Si $|f(x^k) - f(x^{k+1})| < \varepsilon$ entonces el algoritmo se detiene y x^{k+1} es el óptimo sujeto a la tolerancia ε. Además, si $k+1 > J$, el algoritmo no converge y se detiene. Finalmente, en otro caso, se hace $k := k+1$ y $x^k := x^{k+1}$ y volvemos al paso 1.

Similar al caso de Newton, se tomaron $\varepsilon = 10^{-15}$, $J = 100$ y $x^0 = (-0.6, -0.3)$ como los parámetros del algoritmo. Las iteraciones que presenta el algoritmo para de hallar el mínimo se presentan en la figura 3.

Se puede ver que el algoritmo converge a la solución $x^* = (-0.707107, 0) \approx (-1/\sqrt{2}, 0)$ en 71 iteraciones, solucionando el PPNL.

Conclusiones

- El método del gradiente al ser un método de primer orden y no emplear la matriz Hessiana, claramente converge a la solución de forma mucho más lenta, con orden de convergencia lineal. Además, de forma similar al método de Newton, el gradiente descendiente es muy sensible a la elección del punto inicial x^0, y al igual que Newton, este llega a valores que son aproximadamente puntos silla (al alejarse mucho del origen).
- Como observación, en esta implementación del método del gradiente se implementó un parámetro de longitud de paso calculado mediante un *line search*. Se puede ver en la figura 3 que conforme $x^k \to x^*$, $\lambda^k \to 0$. Así, el algoritmo se regula y no se desvía del óptimo dando grandes pasos en la dirección de descenso.

	Iteration	x1	x2	f(x)	|∇f(x)|	λ
0	0	-0.600000	-3.000000e-01	-0.382577	2.908032e-01	9.266659e-01
1	1	-0.765443	-8.728743e-02	-0.422814	1.202278e-01	7.456717e-01
2	2	-0.694677	-3.224738e-02	-0.428303	3.499490e-02	8.455856e-01
3	3	-0.712845	-8.889514e-03	-0.428820	1.241814e-02	7.200949e-01
4	4	-0.705786	-3.399510e-03	-0.428875	3.694085e-03	8.464618e-01
..	...	...	...	...	...	...
67	67	-0.707107	4.134883e-14	-0.428882	1.106592e-10	1.095448e+00
68	68	-0.707107	2.495982e-15	-0.428882	9.729950e-11	-1.396333e+00
69	69	-0.707107	5.485481e-15	-0.428882	3.303754e-10	5.875089e-01
70	70	-0.707107	2.721106e-15	-0.428882	2.606028e-12	1.947564e-11
71	71	-0.707107	2.721106e-15	-0.428882	2.606028e-12	1.947564e-11

Figura 3: Iteraciones realizadas por el método del gradiente descendiente.

- Aunque el valor de λ puede ser fijo y muy pequeño, y aún así converger, implementar el *line search* mejora el rendimiento del algoritmo, haciendolo requerir menos iteraciones.
- La única ventaja del gradiente con respecto al método de Newton, además de su fácil implementación, es que el método del gradiente puede emplearse en funciones de clase C^1 que no sean dos veces diferenciables en algún punto de su dominio.

1.3. Método de Levenberg-Marquardt (LM)

El método de Levenberg-Marquardt es una técnica de optimización utilizada para resolver problemas de mínimos cuadrados no lineales. A diferencia de otros métodos como el de Newton y el Gradiente Descendiente, el enfoque de Levenberg-Marquardt combina elementos de ambos, lo que le otorga ciertas ventajas en ciertos escenarios.

Sean $x^0 \in \mathbb{R}^n$ un punto de búsqueda inicial, $\varepsilon > 0$ una tolerancia predefinida, J un número máximo de iteraciones, y β^0 un parámetro de regularización inicial, el algoritmo LM en la k-ésima iteración ejecuta los siguientes pasos:

- **Paso 1.** Se calcula el gradiente de la función objetivo f en el punto x^k, $\nabla f(x^k)$, y se calcula la matriz Hessiana $\mathbf{H}[f(x^k)]$ de f en x^k. La matriz Hessiana contiene las segundas derivadas parciales de f con respecto a las variables independientes.
- **Paso 2.** Se utiliza la relación:

$$(\mathbf{H}[f(x^k)] + \beta^k I)s^k = -\nabla f(x^k), \tag{3}$$

  donde I es la matriz identidad, para calcular la dirección de búsqueda s^k.
- **Paso 3.** Se define el siguiente punto de búsqueda usando:

$$x^{new} = x^k + \lambda^k s^k. \tag{4}$$

- **Paso 4. Criterio de parada:** El algoritmo se detiene si se cumple alguna de las siguientes condiciones:
  - Se ha alcanzado una convergencia suficiente, $|f(x^{new}) - f(x^k)| < \varepsilon$.
  - Se ha superado el número máximo de iteraciones J.

- **Paso 5. Ajuste del parámetro de regularización:** Si no se ha alcanzado la convergencia, se ajusta el parámetro de regularización β^{k+1} según el desempeño de las iteraciones anteriores. En particular, se define que si $f(x^k + 1) < f(x)$, entonces $\beta^{k+1} = \beta^k/10$ y $x^{k+1} = x^{new}$, mientras que en caso contrario, $\beta^{k+1} = 10\beta^k$ y $x^{k+1} = x^k$.
- **Paso 6.** Se actualiza $x^k := x^{k+1}$ y se retorna al paso 1.

Como observación, λ^k puede ser hallado empleando *line search*, al igual que en el método del gradiente. No obstante, como el método convergió con facilidad, se usó implícitamente $\lambda^k = 1$ en todas las iteraciones.

Ahora bien, para la implementación computacional, se emplearon $x^0 = (0.6, 0.3)$, $\varepsilon = 10^{-15}$, $J = 100$, y $\beta^0 = 0.01$ como los parámetros iniciales del modelo. En la figura 4 se exponen las primeras y últimas iteraciones del método LM implementado en `Python`.

Optimization Data:

	Iteration	x1	x2	f(x)	|f(x_next) - f(x)|	β
0	0	-0.600000	-3.000000e-01	-0.382577	3.825217e-01	0.01
1	1	-0.600000	-3.000000e-01	-0.382577	3.274279e-01	0.10
2	2	-0.600000	-3.000000e-01	-0.382577	-4.023621e-02	1.00
3	3	-0.764615	-8.835218e-02	-0.422813	4.619382e-01	0.10
4	4	-0.764615	-8.835218e-02	-0.422813	-4.923016e-03	1.00
..	...	...	...	...	...	...
84	84	-0.707107	-1.473192e-35	-0.428882	-2.275957e-15	1.00
85	85	-0.707107	-2.095411e-36	-0.428882	6.360468e-13	0.10
86	86	-0.707107	-2.095411e-36	-0.428882	-1.221245e-15	1.00
87	87	-0.707107	-2.980432e-37	-0.428882	3.256839e-13	0.10
88	88	-0.707107	-2.980432e-37	-0.428882	-5.551115e-16	1.00

Figura 4: Iteraciones del algoritmo LM.

Claramente el algoritmo converge a la solución $x^* = (-0.707107, 0) \approx (-1/\sqrt{2}, 0)$, con $f(x^*) = -0.428882$. Aunque el algoritmo tardó más iteraciones que sus contrapartes, declara como solucionado el PPNL bajo los parámetros iniciales asignados.

Conclusiones

- A diferencia del gradiente descendiente, el método de Levenberg-Marquardt es un método de segundo orden, similar al método de Newton. Ahora, aunque la convergencia de este se supone mayor a la lineal, pero menor a la cuadrática, el algoritmo tarda más iteraciones en la implementación por dos motivos. El primero es que hay iteraciones (aproximadamente la mitad) en las que no se actualiza x^k, porque en el punto de avance la función es mayor a $f(x^k)$. El segundo es que se fijó $\lambda^k = 1$ que puede resultar o no conveniente en la resolución del problema.
- A diferencia del método de Newton, Levenberg-Marquardt es menos susceptible a quedar atrapado en mínimos locales o alejarse rápidamente debido a singularidades. La principal ventaja de Levenberg-Marquardt sobre el método de Newton es su capacidad para manejar problemas donde la matriz Hessiana puede volverse singular o no esté bien definida en ciertas regiones del dominio. Esto lo hace aplicable a una gama más amplia de funciones.
- Como conclusión general, en los tres métodos anteriores se aplicaron los criterios de parada $|f(x^{k+1}) - f(x^k)| < \varepsilon$ y $|\nabla f(x^k)| < \varepsilon$ casi indistintamente. Ambos criterios miden convergencia (en orden cero y orden uno respectivamente), y aunque tienen implicaciones técnicas distintas, en esta función sencilla se aplicó el criterio más exacto para cada método.

2. Optimización con Restricciones.

Problema 2. Resolver el siguiente PPNL:

$$\begin{aligned} \text{mín } & f(x) = (x-2)^2 + (y-2)^2 \\ \text{s.a.} \quad & x + y \geq 4 \\ & 2x - y \geq 0 \end{aligned} \tag{5}$$

usando el método de barrera. Este es un PPNL restricto, con la región factible convexa. En las figuras 5 y 6 se exponen la función objetivo y los planos que definen la región factible en 3D y 2D respectivamente.

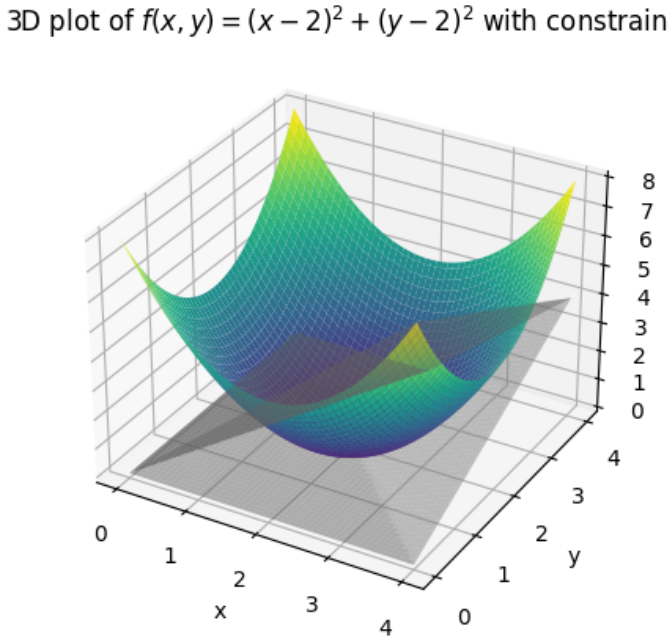

Figura 5: Función objetivo y planos de restricción en 3D

2.1. Método de Penalización Interior

Se aplica en problemas de la forma:

$$\begin{aligned} & \text{mín } f(x) \\ \text{s.a.} \quad & g(x) \leq 0_m \end{aligned} \tag{6}$$

donde $f : \mathbb{R}^n \to \mathbb{R}$ es un campo escalar, las $g : \mathbb{R}^n \to \mathbb{R}^m$ representan las restricciones de desigualdad, $0_m \in \mathbb{R}^m$ es el vector nulo m-dimensional y $S = \{x \in \mathbb{R}^n : g(x) \leq 0_m\}$ es la región factible [1].

La idea de la penalización interior es añadir las restricciones $g(x)$ como una parte de la función objetivo, tal que se forme una barrera que tienda a infinito (en un problema de minimización) a lo largo de la frontera de la región factible. Luego puede aplicarse un método de optimización sin restricciones,

[1] Recordemos que $S = \text{int}(S) \cup \bar{S}$, con $\text{int}(S) = \{x \in \mathbb{R}^n : g(x) < 0_m\}$, y $\bar{S} = \{x \in \mathbb{R}^n : g(x) = 0_m\}$

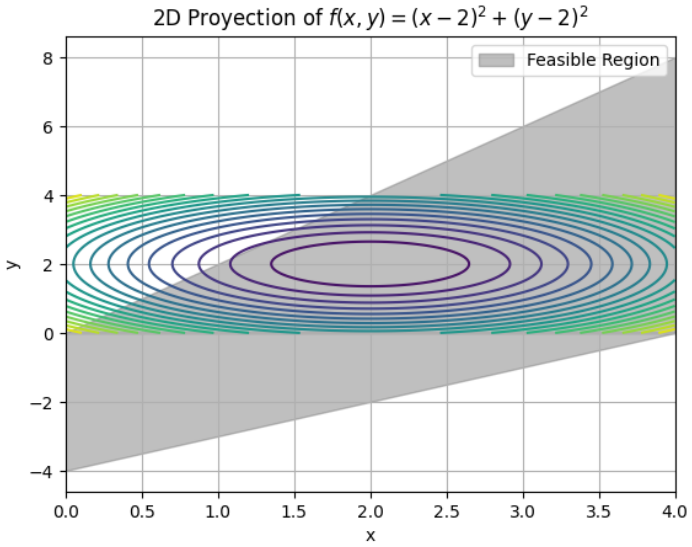

Figura 6: Función objetivo y planos de restricción en 2D

porque todo punto en $\mathbb{R} - \bar{S}$ ha sido penalizado.

Así, definamos entonces la *función de penalización interior* $P : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ como,

$$P(x; \mu) = f(x) + \mu\phi(g(x)) \tag{7}$$

donde μ es el parámetro de penalización, y $\phi : \mathbb{R}^m_+ = \{p \in \mathbb{R}^m : p > 0_m\} \to \mathbb{R}$ que satisface:

$$\phi(g(x)) > 0, \qquad \forall x \in \text{int}(S) \tag{8}$$

$$\lim_{x \to \bar{S}} \phi(g(x)) = \infty \tag{9}$$

Con el fin de resolver el PPNL 5, usaremos la función de penalización de barrera logarítmica, definida como:

$$\phi(g(x)) = -\sum_{j=1}^{m} \log(-g_j(x)), \qquad 0 < -g_j(x) \leq 1 \tag{10}$$

para todo $j = 1, 2, \ldots, m$. En resumen, el problema 5 se puede escribir de la forma:

$$\begin{aligned} \text{mín } & f(x) = (x-2)^2 + (y-2)^2 \\ \text{s.a.} \quad & g_1(x) = -x - y + 4 \leq 0 \\ & g_2(x) = -2x + y \leq 0 \end{aligned} \tag{11}$$

para transformarse en un problema irrestricto como se sigue:

$$\begin{aligned} \text{mín } & f(x) = (x-2)^2 + (y-2)^2 - \mu(\log(x + y - 4) + \log(2x - y)) \\ \text{s.a.} \quad & x, y \in \mathbb{R}. \end{aligned} \tag{12}$$

Ahora, del valor de μ depende mucho la convergencia del algoritmo, por lo que para hallar el valor de este parámetro definiremos una sucesión $\{\mu_n\}_{n=0}^{\infty}$ tal que cuando $n \to \infty$, $\mu_n \to 0$, de modo que cada vez se penalice menos la restricción, pero se inicie cada vez más cerca del óptimo para cada μ_i. La forma de unir esta sucesión de μs y el problema irrestricto 12 se presenta en el siguiente algoritmo.

- **Paso 1.** Se selecciona un punto inicial $x_0 \in \text{int}(S)$, un parámetro de penalización inicial $\mu_0 > 0$ y se hace $n = 0$. Sea $\varepsilon > 0$ el parámetro de tolerancia y $\eta \in (0, 1)$ la constante de recalibración de μ.
- **Paso 2.** Se resuelve el problema,

$$\text{mín } P(x; \mu_n) = f(x) + \mu_n \phi(g(x)) \tag{13}$$
$$\text{s.a.} \quad x \in \text{int}(S) \tag{14}$$

  mediante un método de descenso o el método de Newton.
- **Paso 3.** Si $|x_n - x_{n-1}| < \varepsilon$ o $n > J$, entonces se detiene el proceso.
- **Paso 4.** Se hace $\mu_{n+1} := \eta\mu_n$, $n := n + 1$ y se regresa al paso 2.

donde si en algún momento $|\nabla f(x_n)| < \epsilon$ para un $\epsilon > 0$ fijo, entonces x_n es óptimo (o muy cercano a este).

Ahora, en la implementación, se consideró $\mu_0 = 100$, $\eta = 1/10$, $x_0 = (0.5, 0.5)$, $\epsilon = 10^{-15}$. Los resultados de las últimas iteraciones se exponen en la figura 7.

	Iteration	x	y	f(x)	Gradient Norm
0	0	3.000000	2.000000	1.000000e+00	2.000000e+00
1	1	2.000000	2.000000	2.154995e-09	6.464992e-01
2	2	2.000000	2.000000	2.085680e-09	3.232496e-01
3	3	2.000000	2.000000	2.016365e-09	1.616248e-01
4	4	2.000000	2.000000	1.947051e-09	8.081240e-02
5	5	2.000000	2.000000	1.877736e-09	4.040620e-02
6	6	2.000000	2.000000	1.808421e-09	2.020309e-02
7	7	2.000000	2.000000	1.739107e-09	1.010154e-02
8	8	2.000000	2.000000	1.669792e-09	5.050748e-03
9	9	2.000000	2.000000	1.600480e-09	2.525334e-03
10	10	2.000000	2.000000	1.531173e-09	1.262588e-03
11	11	2.000000	2.000000	1.461889e-09	6.311357e-04
12	12	2.000000	2.000000	1.392700e-09	3.152514e-04
13	13	2.000000	2.000000	1.323884e-09	1.569951e-04
14	14	2.000001	2.000001	1.256538e-09	7.725384e-05
15	15	2.000002	2.000002	1.194656e-09	3.627230e-05
16	16	2.000003	2.000003	1.149510e-09	1.430195e-05
17	17	2.000004	2.000004	1.133181e-09	3.226826e-06
18	18	2.000005	2.000005	1.131982e-09	1.828806e-07
19	19	2.000005	2.000005	1.131978e-09	5.912204e-10
20	20	2.000005	2.000005	1.131978e-09	6.032073e-15
21	21	2.000005	2.000005	1.131978e-09	5.146827e-16

Figura 7: Iteraciones del método de Newton con penalización interior para hallar x^*.

Conclusiones

- La función objetivo es cuadrática y convexa, por ende tiene un único mínimo local (y absoluto) en $\mathbb{R}^2$. Como el óptimo $x^* = (2, 2)$ queda en el interior de la región factible, en realidad emplear la secuencia de μs es desproporcionado para el problema que se debe resolver. Empleando $\mu = 0$, el método de Newton converge inmediatamente (porque la función es cuadrática) sin necesidad de considerar tan siquiera la penalización.
- Naturalmente, en caso de no conocer el mínimo teórico de la función debe usarse el método de penalización para cerciorarnos de que $x^* \in S$. Por lo tanto, aunque este resulte un ejemplo

sencillo que no requiere de la robustez del método de penalización, en el marco teórico presentado se aprecia la astucia del argumento detrás de la penalización interna.

- Aunque el método no converge exactamente a $x^* = (2, 2)$, se cumple el criterio del ϵ. Las máquinas como esta que ejecutan el algoritmo no procesan infinitos bits, y siempre la solución será muy cercana a $(2, 2)$ por fenómenos de underflow.