Optimización de la función $f(x, y) = x^4 - 4x^3 + 4x + y^2$

Nombre: Ruben Martinez Rojas
Grupo: C311

December 5, 2025

1 Modelos a analizar

La función a analizar es:

$$f(x, y) = x^4 - 4x^3 + 4x + y^2$$

2 Análisis de los modelos

2.1 Existencia de solución

Para estudiar la existencia de puntos óptimos, se calcularán los ceros del vector gradiente de la función, dado que todo mínimo (o máximo) local diferenciable debe satisfacer

$$\nabla f(x, y) = 0$$

. El gradiente está dado por el vector de derivadas parciales:

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} 4x^3 - 12x^2 + 4 \\ 2y \end{bmatrix}$$

Para encontrarlos, igualamos cada componente del gradiente a cero:

$$\frac{\partial f}{\partial x} = 4x^3 - 12x^2 + 4 = 0, \qquad \frac{\partial f}{\partial y} = 2y = 0.$$

De la ecuación en y se obtiene directamente:

$$y = 0.$$

Para la ecuación en x, simplificamos dividiendo entre 4:

$$x^3 - 3x^2 + 1 = 0.$$

Esta ecuación cúbica puede tener una, dos o tres raíces reales.

2.2 Convexidad

La matriz Hessiana de la función

$$f(x, y) = x^4 - 4x^3 + 4x + y^2$$

es:

$$H(x, y) = \begin{bmatrix} 12x^2 - 24x & 0 \\ 0 & 2 \end{bmatrix}.$$

Para que la función sea convexa, $H(x, y)$ debe ser definida positiva. Vemos que no lo es para todos los valores de x, por lo tanto la función no es convexa globalmente. Sin embargo, se puede analizar localmente para regiones donde $12x^2 - 24x > 0$, es decir, para $x > 2$ o $x < 0$. Pues en estas regiones donde la Hessiana es definida positiva, la función tiene curvatura positiva en todas las direcciones, lo cual implica que alrededor del punto es similar a un tazón. En estas regiones la curvatura de la funcion no cambia de signo y las direcciones de descenso son estables. Por ello, los métodos de Gradiente, Newton y Quasi-Newton tienden a converger rápidamente hacia un mínimo local.

En cambio, cuando la Hessiana posee valores negativos (como ocurre en la región $0 < x < 2$), la función presenta curvatura negativa en alguna dirección, generando zonas tipo silla de montar. Allí los métodos de optimización tienen comportamientos menos predecibles: el método de Gradiente puede oscilar o avanzar muy lentamente debido a cambios abruptos en la pendiente, mientras que el método de Newton puede producir pasos en direcciones incorrectas, dirigirse hacia un máximo local o incluso divergir por completo. Esto se debe a que la inversión del Hessiano introduce direcciones inestables cuando existen eigenvalores negativos.

Por este motivo, en regiones no convexas es más probable que los métodos queden atrapados en mínimos locales o se comporten de manera errática, y se vuelve necesario usar técnicas globales como Multi-start para explorar distintas zonas del dominio.

La estrategia Multi-start consiste en ejecutar un algoritmo de busqueda de minimo local desde muchos puntos iniciales distintos. Cada ejecución puede converger a un mínimo local diferente, y al comparar los valores finales se elige la mejor solución como candidato al mínimo global. Este enfoque es especialmente útil para funciones no convexas como la presente, ya que ayuda a evitar quedar atrapado en mínimos locales subóptimos. La efectividad del método depende tanto de la distribución de los puntos iniciales como de la capacidad del algoritmo local para converger rápidamente hacia un mínimo en la región donde se inicia el proceso.

2.3 Algoritmos de Optimización

Dado que la función

$$f(x, y) = x^4 - 4x^3 + 4x + y^2$$

no es convexa globalmente, es importante elegir algoritmos adecuados y considerar estrategias de múltiples puntos iniciales para buscar un mínimo global.

1. Método de Descenso por Gradiente (explicación detallada)

Descripción corta. El método de Gradiente es un método iterativo que actualiza la variable en la dirección de máximo descenso local:

$$x_{k+1} = x_k - \alpha \nabla f(x_k),$$

donde $\alpha > 0$ es el *learning rate* (tasa de paso) y $\nabla f(x_k)$ es el gradiente en x_k.

Pseudocódigo:

```
Elegir x0, tolerancia epsilon, paso alpha
Para k = 0,1,2,...:
    Calcular gradiente gk = grad(f)(xk)
    Si ||gk|| < epsilon: terminar
    Actualizar x_{k+1} = xk - alpha * gk
Fin
```

Funcionamiento paso a paso.

1. **Inicialización:** Se elige un punto inicial x_0, una tolerancia ε y un número máximo de iteraciones $K_{\max}$. También se selecciona un valor inicial del learning rate α, pero antes de fijarlo se realizan pruebas con varios valores para determinar cuál es el más adecuado en cada región del dominio.

2. **Cálculo del gradiente:** En cada iteración k se calcula el gradiente $g_k = \nabla f(x_k)$. El gradiente señala la dirección de *máximo incremento* de la función. Por tanto, su opuesto $-g_k$ indica la dirección de *máximo descenso local*, que es la dirección óptima para reducir el valor de la función.

3. **Actualización:** El punto se desplaza en la dirección de descenso mediante

$$x_{k+1} = x_k - \alpha\, g_k.$$

   Este paso ajusta la posición según la magnitud del gradiente y el valor elegido de α, controlando qué tan grande es el avance en la iteración.

4. **Criterio de parada:** Se detiene el algoritmo cuando se cumple alguna de las siguientes condiciones:

   - **Cambio pequeño entre iteraciones:**

$$\|x_{k+1} - x_k\| < \varepsilon.$$

     Esto indica que el algoritmo ha entrado en una zona donde la mejora por iteración es insignificante (estancamiento). Este criterio es útil incluso cuando el gradiente no disminuye estrictamente, pero las actualizaciones se vuelven muy pequeñas, señalando convergencia práctica.

   - **Tope de iteraciones $K_{\max}$:** Garantiza seguridad computacional, evitando ciclos infinitos en casos de divergencia o estancamiento. Además, permite comparar el costo computacional entre configuraciones distintas del algoritmo.

5. **Iterar:** El proceso se repite hasta cumplir alguno de los criterios de parada anteriores.

Influencia de cada parámetro:

- **Learning rate α:** controla el tamaño del paso en cada iteración.
  - α **muy pequeño:** garantiza estabilidad y convergencia segura, pero requiere muchas iteraciones.
  - α **intermedio:** proporciona un equilibrio entre velocidad y estabilidad.
  - α **grande:** puede producir oscilaciones, saltarse el mínimo o incluso divergir, especialmente en regiones con alta curvatura.
- **Tolerancia ε:** define el nivel de precisión. Valores más pequeños exigen mayores iteraciones.
- **Máximo de iteraciones $K_{\max}$:** evita ciclos indefinidos y permite controlar el costo computacional.
- **Punto inicial x_0:** en funciones no convexas determina a qué mínimo local converge el algoritmo; por ello se evaluan puntos iniciales en distintas regiones.

2. Descenso por Gradiente con Búsqueda en Línea (Backtracking Armijo)

Pseudocódigo.

```
Gradient Descent con Backtracking Armijo

Entradas:
    x0        -> punto inicial
    epsilon   -> tolerancia
    Kmax      -> máximo número de iteraciones
    alpha0    -> paso inicial (típicamente 1)
    beta      -> factor de reducción (0 < beta < 1)
    c         -> constante de Armijo (típicamente 1e-4)

x = x0
Guardar x en historial

Para k = 0,1,...,Kmax:

    1. Calcular gradiente g = grad_f(x)

    2. Si norm(g) < epsilon:
           terminar

    3. Backtracking Armijo:
            alpha = alpha0
            fx = f(x)
            g_norm_sq = dot(g, g)

            Mientras f(x - alpha*g) > fx - c*alpha*g_norm_sq:
                    alpha = beta * alpha

    4. Actualizar:
            x_new = x - alpha*g

    5. Guardar x_new en historial

    6. Si norm(x_new - x) < epsilon:
           terminar

    7. x = x_new

Fin Para

Salida:
    x final y el historial de iteraciones
```

Explicación Detallada de la Condición de Armijo

La condición de Armijo (también llamada *regla de suficiente descenso*) es una desigualdad que verifica si el paso elegido produce una reducción "adecuada" en el valor de la función objetivo. Matemáticamente, para un punto x_k, dirección de descenso $d_k = -\nabla f(x_k)$ y paso $\alpha > 0$, la condición es:

$$f(x_k + \alpha d_k) \leq f(x_k) + c \cdot \alpha \nabla f(x_k)^T d_k$$

Para descenso por gradiente donde $d_k = -g_k$ (con $g_k = \nabla f(x_k)$), esta desigualdad se convierte en:

$$f(x_k - \alpha g_k) \leq f(x_k) - c\alpha \|g_k\|^2$$

donde:

- $0 < c < 1$ (típicamente $c \approx 10^{-4}$)
- α es el tamaño de paso que estamos probando
- $\|g_k\|^2 = \nabla f(x_k)^T g_k$ (producto punto del gradiente consigo mismo)

Cuando se cumple la condición de Armijo, garantizamos:

1. **Descenso suficiente**
   La reducción en f es al menos una fracción c de lo que predice la aproximación lineal:
   $$f(x_k) - f(x_{k+1}) \geq c\alpha \|g_k\|^2 > 0$$
   Como $\|g_k\|^2 > 0$ (si no estamos en un punto estacionario), la función *decrece estrictamente*.

2. **Monotonicidad**
   La secuencia $\{f(x_k)\}$ es estrictamente decreciente.

3. **Acotación del paso**
   El α aceptado no es excesivamente grande, evitando oscilaciones.

4. **Convergencia global (bajo condiciones)**
   Para funciones con gradiente Lipschitz continuo, el backtracking de Armijo garantiza que eventualmente se encontrará un α que satisfaga la condición, y el método convergerá a un punto estacionario.

Explicación paso a paso del algoritmo.

1. **Inicialización.** Se parte de un punto inicial x_0. También se establece una tolerancia ε y un máximo de iteraciones $K_{\max}$.

   El historial guarda las posiciones del algoritmo para poder estudiar la trayectoria posterior.

2. **Cálculo del gradiente.** En cada iteración se evalúa:

   $$g_k = \nabla f(x_k).$$

   El gradiente indica la dirección de mayor incremento local de la función; por lo tanto, el descenso debe hacerse en la dirección opuesta.

3. **Criterio de parada por gradiente pequeño.** Si

$$\|g_k\| < \varepsilon,$$

entonces la función es prácticamente plana alrededor del punto actual, lo cual indica que se alcanzó un punto cercano a un mínimo.

4. **Cálculo dinámico del paso α mediante Backtracking Armijo.** En lugar de usar un valor fijo de *learning rate*, se escoge un paso que garantice una disminución suficiente en la función.

   - Se comienza con $\alpha_0 = 1$.
   - Se comprueba la condición de Armijo:

$$f(x - \alpha g_k) \leq f(x) - c\,\alpha\,\|g_k\|^2.$$

   - Si la condición no se cumple, el paso se reduce multiplicando por β:

$$\alpha \leftarrow \beta\alpha, \qquad 0 < \beta < 1.$$

   - Este proceso continúa hasta que el paso sea seguro y produzca una reducción adecuada.

   La búsqueda en línea garantiza que:

   - nunca se toma un paso demasiado grande que cause divergencia,
   - tampoco se necesita un α excesivamente pequeño que ralentice el algoritmo.

5. **Actualización del punto.** Una vez determinado el paso adecuado se realiza:

$$x_{k+1} = x_k - \alpha g_k.$$

La magnitud del desplazamiento se ajusta dinámicamente según la curvatura local detectada por Armijo.

6. **Criterio de parada por cambio pequeño entre iteraciones.** Si

$$\|x_{k+1} - x_k\| < \varepsilon,$$

entonces se concluye que el algoritmo está suficientemente estabilizado y no vale la pena continuar.

7. **Iteración.** Se repite el proceso utilizando x_{k+1} como nuevo punto de partida.

3 Descripción del Proceso Experimental

Para analizar el comportamiento del algoritmo de Descenso por Gradiente (Gradient Descent, GD) sobre la función objetivo, se diseñó un experimento sistemático basado en la variación de puntos iniciales y tasas de aprendizaje (*learning rates*). El procedimiento seguido fue el siguiente:

1. **Generación de puntos iniciales.** Se definieron múltiples puntos iniciales distribuidos en tres regiones del plano, con el fin de estudiar cómo la posición inicial influye en la convergencia, la estabilidad y el valor final alcanzado por el algoritmo.

2. **Ejecución del GD para distintos learning rates.** Para cada punto inicial se ejecutó el algoritmo de GD utilizando un conjunto de tasas de aprendizaje predefinidas:

$$lr \in \{0.001,\ 0.005,\ 0.01,\ 0.05,\ 0.1,\ 0.2,\ 0.3,\ 0.5\}.$$

   En cada ejecución, el método avanzó mediante la regla

$$x_{k+1} = x_k - lr\,\nabla f(x_k),$$

   manteniendo un límite máximo de iteraciones y criterios de paro basados en la norma del gradiente y el desplazamiento entre iteraciones.

3. **Registro de estadísticas por ejecución.** Cada corrida del algoritmo almacenó información relevante, incluyendo:

   - valor inicial y final de la función $f(x)$,
   - número total de iteraciones realizadas,
   - norma del gradiente a lo largo del proceso,
   - causas de terminación (convergencia, máximo de iteraciones o divergencia por *overflow*).

   Los resultados se clasificaron en tres categorías: *convergencia*, *no convergencia* y *divergencia*.

4. **Construcción de tablas y consolidación de resultados.** Los datos obtenidos se organizaron en tablas que resumen, para cada combinación de región y learning rate:

   - el número de ejecuciones que convergieron,
   - las que no convergieron por alcanzar el máximo de iteraciones,
   - las que divergieron por errores numéricos.

5. **Análisis estadístico y comparación.** A partir de las estadísticas agregadas se evaluó:

   - la estabilidad del método según el learning rate,
   - la velocidad de convergencia medida en iteraciones promedio,
   - la calidad del punto final alcanzado (mínimo local o global),
   - el comportamiento particular de cada región.

   Estos análisis permitieron identificar intervalos óptimos de aprendizaje y zonas de inestabilidad.

En conjunto, este proceso permitió caracterizar de manera completa cómo afecta la tasa de aprendizaje, la posición inicial y la geometría de la función al desempeño del Descenso por Gradiente, proporcionando una base estadística sólida para la selección de parámetros adecuados.

4 Conclusiones del analisis del algoritmo de Gradiente Descendiente

1. Los learning rates pequeños convergen pero requieren muchas iteraciones

Los casos con learning rate = 0.001 y algunos con 0.005 muestran convergencias extremadamente lentas: con iteraciones promedio entre 1000 y 2400. En gran parte de las regiones, al utilizar estos learning rates la razón de no convergencia es que precisaban mas de 2400 iteraciones para poder alcanzar un resultado válido.

Esto indica que el GD con tasas de aprendizaje demasiado pequeñas no avanza lo suficiente en cada paso y puede estancarse antes de llegar al mínimo.

2. Existe una zona óptima de aprendizaje: 0.05 – 0.1

Al analizar las estadísticas de convergencia para lr = 0.05, el método converge de forma estable con aproximadamente 120–130 iteraciones, y para lr = 0.1, la convergencia es incluso más rápida: entre 60 y 70 iteraciones en promedio. Alcanzando con ambos casos resultados finales cercanos al mínimo global (aprox. $f(x) = -15.23$).

3. A partir de lr ≥ 0.2 el método diverge sistemáticamente

Los datos muestran que el 100% de los casos con lr = 0.2, 0.3 y 0.5 presentan divergencia por overflow. Se registraron 1401 divergencias de 2400 experimentos realizados, equivalente al 58.375% del total de ejecuciones, siendo la causa es siempre overflow numérico, no oscilación o crecimiento lento. Además, el lr mas frecuente en divergencias fue 0.2.

4. Influencia de la región

Las estadísticas permiten observar que la Región 1 es la que presenta más divergencias y la Región 3 es la que mejor alcanza valores cercanos al mínimo global (f_final alrededor de -15.23).

4.1 5. La convergencia lleva consistentemente al mismo mínimo

En todos los casos donde el algoritmo converge:

$$f_{\text{final}} \in [-15.2344, -1.4456]$$

- Los valores cercanos a -1.4456 corresponden a un mínimo local.
- Los valores alrededor de -15.23 corresponden al mínimo global.

5 Conclusión Final

Los datos muestran de manera consistente que:

El intervalo óptimo para un comportamiento estable y eficiente es $lr \in [0.05, 0.1]$.

Fuera de esta región, el algoritmo se vuelve ineficiente (si lr es muy pequeño) o completamente inestable (si lr es grande). El análisis también evidencia que las regiones muestran comportamientos distintos, siendo la Región 3 la más estable y cercana al mínimo global.

6 Procesamiento de Datos con el Método de Armijo

En esta sección se describe el procedimiento utilizado para ejecutar el algoritmo de Descenso por Gradiente con búsqueda en línea mediante la condición de Armijo, así como el procesamiento y registro estadístico de los resultados obtenidos. El objetivo fue evaluar el desempeño del método bajo diferentes configuraciones de parámetros y obtener métricas cuantitativas que permitan comparar su estabilidad y eficiencia.

6.1 Generación y carga de los puntos iniciales

El análisis se realizó utilizando el mismo conjunto de 300 puntos iniciales previamente generados para utilizar con el algoritmo de Grdiente Decsendiente

6.2 Ejecución del algoritmo de Armijo

Para cada punto inicial se aplicó el método de Descenso por Gradiente, empleando una búsqueda en línea basada en la condición de Armijo. Dada una posición x_k y su gradiente $\nabla f(x_k)$, el procedimiento determina un paso α tal que:

$$f(x_k - \alpha \nabla f(x_k)) \leq f(x_k) - c\,\alpha\,\|\nabla f(x_k)\|^2.$$

El valor inicial del paso α_0 se reduce sucesivamente mediante el factor β hasta satisfacer la desigualdad anterior. Se exploraron diversas combinaciones de parámetros:

$$\alpha_0 \in \{0.5,\, 0.8,\, 1.0\}, \qquad \beta \in \{0.5,\, 0.7\}, \qquad c \in \{10^{-4},\, 10^{-3}\}.$$

Esto generó múltiples ejecuciones independientes del algoritmo, una por cada punto inicial y cada combinación de parámetros.

6.3 Criterios de parada

El algoritmo se detuvo siguiendo dos condiciones clásicas:

- **Norma del gradiente pequeña:** $\|\nabla f(x_k)\| < \text{tol}$, indicando proximidad a un punto crítico.
- **Cambio pequeño entre iteraciones:** $\|x_{k+1} - x_k\| < \text{tol}$.

Se estableció un máximo de iteraciones (200 iteraciones por ejecución) para evitar bucles prolongados, aunque en la práctica ninguna ejecución alcanzó este límite debido a la eficacia del método.

6.4 Registro de resultados

Para cada ejecución se almacenaron las siguientes métricas:

- número total de iteraciones realizadas,
- valor final de la función objetivo,
- coordenadas finales alcanzadas,
- distancia euclidiana del punto final al origen,
- indicador de convergencia o no convergencia,
- parámetros α_0, β y c utilizados en la ejecución.

Todos estos datos fueron consolidados en un único archivo CSV para facilitar su posterior análisis estadístico.

6.5 Procesamiento estadístico

A partir del archivo consolidado se calcularon estadísticas globales para cada combinación de parámetros, incluyendo:

- valor promedio, mínimo y máximo del número de iteraciones;
- valor promedio, mínimo y máximo de la función objetivo final;
- distancia promedio a la solución final;
- tasa de convergencia expresada en porcentaje;
- número total de ejecuciones exitosas y fallidas.

Estas estadísticas permiten identificar las configuraciones más eficientes y analizar cómo influyen los parámetros de Armijo en la estabilidad, rapidez y calidad de convergencia del algoritmo.

7 Análisis Estadístico de los Resultados del Método de Armijo

7.1 Robustez del Método

En todas las configuraciones evaluadas, el método de Armijo logró convergencia en el *100%* de las ejecuciones. Esto implica que, para este problema en particular, la búsqueda en línea garantiza estabilidad incluso cuando se modifican ampliamente los parámetros de control. A diferencia del descenso por gradiente con tasa de aprendizaje fija, no se registraron casos de divergencia numérica ni estancamiento prematuro.

7.2 Influencia de los Parámetros sobre el Número de Iteraciones

El número promedio de iteraciones presenta variaciones significativas según los valores de α_0 y β. Los resultados indican que:

- El desempeño más eficiente se obtuvo con $\alpha_0 = 0.8$ y $\beta = 0.5$, alcanzando un promedio de aproximadamente 83 iteraciones.
- El peor desempeño se observó con $\alpha_0 = 0.8$ y $\beta = 0.7$, con promedios entre 108 y 111 iteraciones.
- El parámetro c no produjo variaciones relevantes en el comportamiento del algoritmo, dado que las diferencias entre $c = 10^{-4}$ y $c = 10^{-3}$ fueron menores al 1%.

Estos resultados confirman que la elección de β afecta directamente la agresividad con la que el paso α se reduce durante la búsqueda en línea. Valores de β más grandes conducen a una reducción más lenta del paso, generando iteraciones adicionales antes de satisfacer la condición de Armijo.

7.3 Análisis del Valor Final de la Función

Los valores finales de la función objetivo se encontraron en el rango comprendido entre aproximadamente -15.23 y -1.45. Todos los experimentos fueron capaces de alcanzar mínimos locales o el mínimo global, dependiendo del punto inicial. Las configuraciones con $\alpha_0 = 0.8$ lograron de forma más consistente valores finales más cercanos al mínimo global.

7.4 Distancia Final al Punto Óptimo

El análisis de la distancia entre el punto final y el origen muestra una tendencia coherente con el comportamiento de la función objetivo:

- Las configuraciones con $\alpha_0 = 0.8$ alcanzaron las menores distancias promedio, confirmando un mejor acercamiento al mínimo global.
- Los valores con $\beta = 0.7$ produjeron distancias ligeramente mayores que aquellos con $\beta = 0.5$, indicando que una reducción más gradual de α puede llevar a zonas menos favorables del paisaje de la función.

7.5 Conclusión General

El análisis estadístico demuestra que el método de Armijo es altamente estable y eficiente para este problema. Todos los experimentos convergieron, sin presentar divergencias numéricas, oscilaciones ni estancamiento en el máximo de iteraciones. La configuración más eficiente fue $\alpha_0 = 0.8$ y $\beta = 0.5$, logrando el menor número promedio de iteraciones y valores finales próximos al mínimo global. Por otro lado, el parámetro c tuvo un impacto marginal en el rendimiento, lo cual coincide con la teoría clásica de métodos de búsqueda en línea.

Estos resultados indican que, cuando se requiere robustez y estabilidad ante cambios en el punto inicial o en la estructura local de la función, la regla de Armijo constituye una alternativa ampliamente superior al uso de un valor fijo de tasa de aprendizaje.

8 Conclusión General

El análisis comparativo entre el descenso por gradiente con tasa de aprendizaje fija y el método con búsqueda en línea mediante la regla de Armijo permite establecer diferencias claras en estabilidad, robustez y eficiencia, utilizando exclusivamente los datos obtenidos durante las ejecuciones experimentales.

En primer lugar, el algoritmo con **tasa de aprendizaje fija** mostró un comportamiento altamente sensible al valor del parámetro. Para valores pequeños de lr el proceso converge, pero lo hace de manera lenta y con un número elevado de iteraciones. A medida que el lr se incrementa, el método comienza a presentar oscilaciones, y a partir de $lr \geq 0.2$ los experimentos mostraron divergencia sistemática debido a incrementos descontrolados en la norma del gradiente y errores numéricos por desbordamiento. Estos resultados confirman la limitada región de estabilidad del método cuando el paso no se ajusta dinámicamente.

En contraste, el método de **búsqueda en línea con Armijo** logró convergencia en todos los casos donde el gradiente no era numéricamente degenerado. Gracias al ajuste adaptativo del paso, el algoritmo evitó las oscilaciones y la divergencia incluso cuando el valor inicial de α era grande. Los datos muestran que Armijo fue capaz de reducir el paso de manera automática cuando la función no satisfacía la condición de descenso suficiente, garantizando estabilidad global y trayectorias de descenso más controladas.

Comparando ambos métodos en conjunto, se observa que Armijo ofrece una **mayor robustez** al no depender críticamente de la elección manual de un lr adecuado. Además, dado que ajusta dinámicamente el tamaño de paso, la trayectoria hacia el mínimo es más eficiente que en el caso de tasas fijas pequeñas, sin incurrir en los riesgos de explosión numérica que aparecen con tasas fijas grandes. En términos de confiabilidad y consistencia, el descenso con Armijo resulta claramente superior.

En conclusión, los resultados obtenidos indican que, para problemas donde la forma de la función objetivo o la escala del gradiente no se conocen a priori, el método con búsqueda en línea tipo Armijo representa una alternativa más estable, segura y eficiente que el descenso con tasa de aprendizaje fija, ofreciendo convergencia confiable sin necesidad de calibración manual del parámetro de paso.