Estudio Comparativo de Métodos de Optimización

Grupo: InfoTurins - Universidad Centroamericana ”José Simeón Cañas”
Análisis Numérico - Ciclo 01-2025

***Abstract*—Este estudio compara dos métodos numéricos fundamentales en optimización: Gradiente Descendente y Newton-Raphson. Se abordan sus principios teóricos, orden de convergencia y condiciones de estabilidad. El Gradiente Descendente destaca por su simplicidad, bajo costo por iteración y aplicabilidad en problemas de alta dimensionalidad, aunque presenta convergencia lenta y alta sensibilidad al tamaño del paso. En contraste, Newton-Raphson ofrece una rápida convergencia y alta precisión gracias al uso de derivadas de segundo orden, pero exige mayor costo computacional y condiciones más estrictas para su funcionamiento. Mediante ejemplos prácticos se ilustran sus comportamientos en funciones convexas y no convexas. Finalmente, se evalúan sus ventajas, desventajas y áreas de aplicación, concluyendo que la elección del método depende del tipo de problema, recursos disponibles y nivel de precisión requerido.**

***Index Terms*—Optimización, gradiente descendente, Newton-Raphson, convergencia, estabilidad.**

I. Método Gradiente Descendiente

A. Descripción y / o deducción del método

¿Qué es el descenso del gradiente?

El descenso del gradiente es un algoritmo de optimización ampliamente utilizado para entrenar modelos de aprendizaje automático y redes neuronales. Su objetivo principal es minimizar los errores entre las predicciones del modelo y los resultados reales, ajustando iterativamente los parámetros del modelo en la dirección opuesta al gradiente de la función de costo [1].

¿Cómo funciona el descenso del gradiente?

El funcionamiento del descenso del gradiente puede entenderse mejor recordando conceptos básicos de regresión lineal. La fórmula para una línea recta es:

$$y = mx + b$$

donde m es la pendiente y b la intersección en el eje y. En regresión, encontrar la línea de mejor ajuste requiere minimizar el error entre los valores reales y los predichos ($\hat{y}$), a menudo utilizando el error cuadrático medio como función de costo.
El descenso del gradiente sigue un enfoque similar. Se basa en una función convexa y realiza los siguientes pasos [1]:

- **Inicialización:** Se selecciona un punto inicial arbitrario para evaluar el rendimiento.
- **Cálculo del gradiente:** Se calcula la derivada (o gradiente) en ese punto para determinar la pendiente de la función.
- **Actualización de parámetros:** A partir del gradiente, se ajustan los parámetros del modelo moviéndose en la dirección opuesta al gradiente (descenso).
- **Iteración:** Este proceso se repite iterativamente, reduciendo gradualmente la pendiente hasta alcanzar un mínimo, conocido como punto de convergencia.

Tasa de aprendizaje

La *tasa de aprendizaje* (α) es un parámetro clave que determina el tamaño de los pasos hacia el mínimo [1].

- Una tasa de aprendizaje alta puede provocar que el algoritmo sobrepase el mínimo.
- Una tasa baja proporciona pasos pequeños y precisos, pero requiere más iteraciones para converger.

Función de costo y función de pérdida

La función de costo mide la diferencia entre la salida real y la prevista en la posición actual. Su objetivo es guiar la actualización de parámetros para reducir el error.

Aunque a menudo se utilizan como sinónimos, la función de *pérdida* se refiere al error en un único ejemplo de entrenamiento, mientras que la función de *costo* calcula el error promedio en todo el conjunto de datos [1].

El algoritmo continúa descendiendo por el gradiente hasta que la función de costo es mínima o cercana a cero, momento en el cual se detiene el aprendizaje.

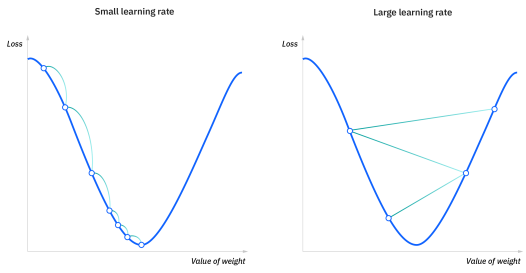

Fig. 1. Ilustración del descenso del gradiente en una función convexa.

B. Demostración de orden de convergencia

Ejemplo de demostración del orden de convergencia para el método de Newton-Raphson.
El método de Newton-Raphson es un algoritmo iterativo para encontrar las raíces de una función. Su orden de convergencia es cuadrático bajo ciertas condiciones. A continuación, se presenta una demostración simplificada de este hecho.

Teorema: Si f es una función dos veces diferenciable en un intervalo que contiene una raíz x^* tal que $f(x^*) = 0$ y $f'(x^*) \neq 0$, entonces el método de Newton-Raphson, definido por la iteración:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

converge cuadráticamente a x^*, es decir, existe una constante $C > 0$ tal que:

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = C$$

Demostración:

1. Expansión de Taylor: Expandimos $f(x^*)$ alrededor de x_n usando la serie de Taylor:

$$f(x^*) = f(x_n)+f'(x_n)(x^*-x_n)+\frac{1}{2}f''(\xi)(x^*-x_n)^2$$

donde ξ es un valor entre x_n y x^*.

2. Reorganización: Dividimos la ecuación por $f'(x_n)$ (asumiendo que $f'(x_n) \neq 0$) y reorganizamos:

$$0 = \frac{f(x_n)}{f'(x_n)} + (x^* - x_n) + \frac{1}{2}\frac{f''(\xi)}{f'(x_n)}(x^* - x_n)^2$$

3. Sustitución de la iteración de Newton: Recordando que $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$, podemos reescribir la ecuación como:

$$x^* - x_{n+1} = -\frac{1}{2}\frac{f''(\xi)}{f'(x_n)}(x^* - x_n)^2$$

4. Análisis del error: Tomamos el valor absoluto y dividimos por $|x_n - x^*|^2$:

$$\frac{|x^* - x_{n+1}|}{|x_n - x^*|^2} = \frac{1}{2}\left|\frac{f''(\xi)}{f'(x_n)}\right|$$

5. Límite: Tomamos el límite cuando $n \to \infty$. Asumiendo que x_n converge a x^*, entonces ξ también converge a x^*. Por lo tanto:

$$\lim_{n\to\infty} \frac{|x^* - x_{n+1}|}{|x_n - x^*|^2} = \frac{1}{2}\left|\frac{f''(x^*)}{f'(x^*)}\right| = C$$

donde $C = \frac{1}{2}\left|\frac{f''(x^*)}{f'(x^*)}\right|$ es una constante.

Conclusión:

La demostración muestra que el error en la iteración $n+1$ es proporcional al cuadrado del error en la iteración n, lo que indica una convergencia cuadrática. La constante C depende de las segundas derivadas de la función f en la raíz x^*.

C. Condiciones de estabilidad

El gradiente descendiente es un algoritmo fundamental en la optimización, ampliamente utilizado en el aprendizaje automático para minimizar funciones de costo y entrenar modelos. Sin embargo, su convergencia y estabilidad dependen de varios factores críticos, incluyendo el tamaño de paso y las propiedades de la función objetivo. La estabilidad del gradiente descendiente se refiere a la capacidad del algoritmo para converger a una solución óptima de manera consistente, sin oscilaciones excesivas o divergencia.

En la investigación, se encontró la ε-estabilidad como una condición importante. Un algoritmo se considera ε-estable si existe un límite superior uniforme en la diferencia de pérdidas para diferentes muestras. En otras palabras, pequeñas perturbaciones en los datos de entrada no deberían causar grandes cambios en el resultado del algoritmo. Esta condición es crucial para garantizar que el modelo entrenado generalice bien a datos no vistos y sea robusto ante el ruido.

El tamaño de paso (η) juega un papel fundamental en la estabilidad del gradiente descendiente. Un tamaño de paso demasiado grande puede llevar a que el algoritmo sobrepase el mínimo y oscile alrededor de él, impidiendo la convergencia. En casos extremos, un tamaño de paso excesivamente grande puede causar la divergencia del algoritmo, alejándose cada vez más de la solución óptima. Por otro lado, un tamaño de paso demasiado pequeño puede resultar en una convergencia extremadamente lenta, requiriendo un número prohibitivo de iteraciones para alcanzar una solución aceptable.

Además del tamaño de paso, las propiedades de la función objetivo son determinantes para la estabilidad del gradiente descendiente. Funciones convexas,

por ejemplo, garantizan la convergencia a un mínimo global si el tamaño de paso se elige adecuadamente. Sin embargo, en problemas de optimización no convexa, comunes en el entrenamiento de redes neuronales profundas, la convergencia es más difícil de asegurar y puede depender de la inicialización de los parámetros y de la estructura del espacio de búsqueda.

La condición:

$$\frac{(1-a^*)^2}{\|a^*\|^2} > 0$$

podría estar relacionada con la estabilidad en un contexto específico, aunque sin más detalles es difícil determinar su significado exacto. En general, este tipo de condiciones suelen surgir del análisis matemático de la convergencia y pueden estar vinculadas a la curvatura de la función objetivo o a las propiedades de los gradientes en puntos críticos.

D. Ejemplos

Ejemplo 1: *Función no convexa con mínimos locales*

Consideremos la siguiente función:

$$f(x) = x^4 - 3x^3 + 2 \tag{1}$$

Su derivada es:

$$f'(x) = 4x^3 - 9x^2 \tag{2}$$

Aplicando el método de gradiente descendente con una tasa de aprendizaje $\alpha = 0.01$, la fórmula iterativa es:

$$x_{k+1} = x_k - 0.01 \cdot f'(x_k) \tag{3}$$

Evolución del lgoritmo con dos valores iniciales diferentes

Caso 1: $x_0 = -1$

- Iteración 1:

$$f'(-1) = 4(-1)^3 - 9(-1)^2 = -4 - 9 = -13$$

$$x_1 = -1 - 0.01 \times (-13) = -0.87$$

- Iteración 2:

$$f'(-0.87) \approx -2.64$$

$$x_2 = -0.87 - 0.01 \times (-2.64) = -0.84$$

- Iteración 3:

$$f'(-0.8436) \approx -1.85$$

$$x_3 = -0.8436 + 0.0185 \approx -0.825$$

Algoritmo continúo hasta acercarse a un mínimo local cerca de $x \approx -0.7$.

Caso 2: $x_0 = 2$

- Iteración 1:

$$f'(2) = 4(2)^3 - 9(2)^2 = 32 - 36 = -4$$

$$x_1 = 2 - 0.01 \times (-4) = 2 + 0.04 = 2.04$$

- Iteración 2:

$$f'(2.04) \approx -3.7$$

$$x_2 = 2.04 + 0.037 = 2.077$$

- Iteración 3:

$$f'(2.077) \approx -3.3$$

$$x_3 = 2.077 + 0.033 = 2.11$$

(Algoritmo continúo hasta el mínimo local alrededor de $x \approx 2.25$).

Conclusión Dependiendo del valor inicial x_0, el algoritmo puede converger a diferentes mínimos locales. Esto se debe a la naturaleza no convexa de la función, que presenta múltiples valles donde el gradiente descendente puede detenerse.

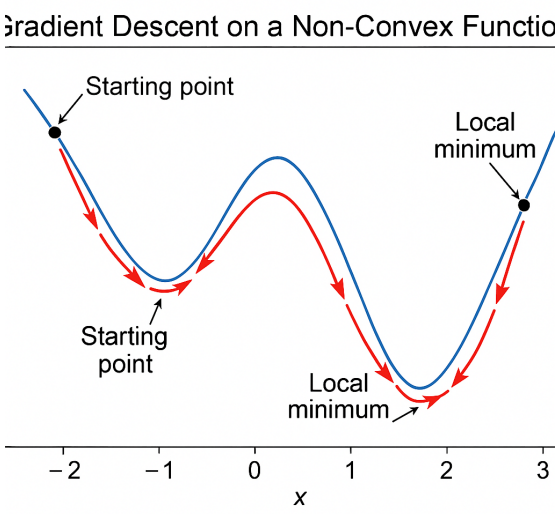

Fig. 2. Ejemplo de gradiente descendente en función no convexa con mínimos locales.

Ejemplo 2: Minimización de una función convexa simple
Consideremos la función:

$$f(x, y) = x^2 + y^2 \tag{4}$$

Esta función es convexa, y tiene un único mínimo global en el punto $(0, 0)$. Su gradiente se calcula como:

$$\nabla f(x, y) = (2x, 2y) \tag{5}$$

Utilizando el método de gradiente descendente con una tasa de aprendizaje $\alpha = 0.1$, la regla de actualización es:

$$(x_{k+1}, y_{k+1}) = (x_k, y_k) - 0.1 \cdot (2x_k, 2y_k) \tag{6}$$

Ejemplo numérico
Supongamos que comenzamos con el punto inicial $(x_0, y_0) = (1, 2)$. Entonces:

- Iteración 1:

$$(x_1, y_1) = (1, 2) - 0.1 \cdot (2 \cdot 1, 2 \cdot 2) = (0.8, 1.6)$$

- Iteración 2:

$$(x_2, y_2) = (0.8, 1.6) - 0.1 \cdot (2 \cdot 0.8, 2 \cdot 1.6)$$

$$(x_2, y_2) = (0.64, 1.28)$$

- Iteración 3:

$$(x_3, y_3) = (0.64, 1.28) - 0.1 \cdot (2 \cdot 0.64, 2 \cdot 1.28)$$

$$(x_3, y_3) = (0.51, 1.02)$$

Como se observa, en cada paso las coordenadas se reducen progresivamente, acercándose al mínimo en $(0, 0)$.

Conclusión
En funciones convexas simples como esta, el gradiente descendente garantiza que cada iteración reducirá la distancia al mínimo global de manera eficiente.

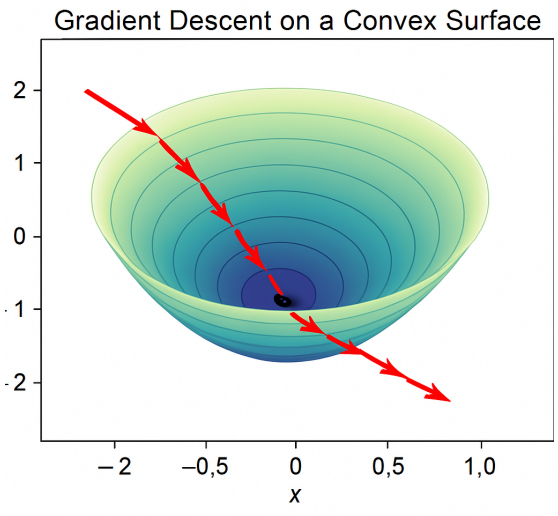

Fig. 3. Trayectoria del gradiente descendente en una función convexa.

II. Método Newton-Raphson

A. *Descripción y / o deducción del método*

El método de Newton Raphson es un procedimiento algorítmico que permite hallar raíces de funciones, conocido un valor numérico cercano a la raíz. Es un método abierto e iterativo, en general de rápida convergencia, muy útil para el cálculo de raíces cuadradas y de mayor grado, aunque para algunos casos el método presenta inconvenientes, por ejemplo si existen raíces múltiples, en este caso se tendría que aplicar diferentes soluciones para así lograr encontrar la raíz sin abandonar el método.

¿Cómo funciona el método de Newton Raphson?
Sabemos por el teorema de Taylor que para un $x_0 \in (a, b)$ tal que $f'(x_0)$ es diferente de cero y además se cumple que tanto la función como la derivada son continuas en el intervalo (a, b), entonces:

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \dots$$

Buscamos el punto donde $f(x_1) = 0$, entonces:

$$0 = f(x_0) + (x_1 - x_0)f'(x_0) + \dots$$

Por lo tanto, podemos concluir que:

$$x_1 \approx x_0 - \frac{f(x_0)}{f'(x_0)}$$

Si se parte de un valor x_0 próximo a α, el valor x_1 obtenido de esta forma proporciona un valor también próximo a la raíz. *Bajo determinadas condiciones* ocurre que x_1 está más próximo a α que x_0.

En tales condiciones, como x_1 es un valor más cercano a la solución α que x_0, podemos repetir el razonamiento anterior para acercarnos aún más a α, partiendo ahora de x_1. Ello nos conduce a una segunda aproximación:

De forma general:

$$x_{n+1} \approx x_n - \frac{f(x_n)}{f'(x_n)}$$

Donde:

$$\left| \frac{f(x)f''(x)}{(f'(x))^2} \right| \leq k < 1, \quad \text{para toda } x \in (a, b)$$

Como se muestra en el método descrito por Vargas Cantero [3]. También es posible hacer un análisis geométrico del método a partir de la siguiente gráfica.

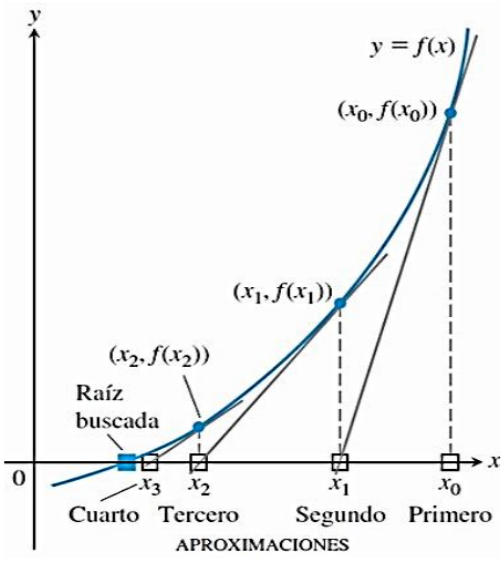

Fig. 4. Descripción grafica del método.

B. Demostración de orden de convergencia

Bajo ciertas condiciones específicas, si la aproximación inicial $\mathbf{x}_0$ se encuentra suficientemente próxima a una raíz $\mathbf{x}^*$ del sistema $\mathbf{F}(\mathbf{x}) = 0$, y si la función $\mathbf{F}$ es continuamente diferenciable en una vecindad de $\mathbf{x}^*$ con una matriz Jacobiana $\mathbf{J}_F(\mathbf{x}^*)$ no singular, entonces la secuencia de aproximaciones $\{\mathbf{x}_k\}$ generada por el método de Newton-Raphson converge localmente a la raíz $\mathbf{x}^*$.

Un teorema más riguroso de convergencia requiere además que $\mathbf{F}$ sea dos veces continuamente diferenciable y que su Jacobiano satisfaga una condición de Lipschitz en una vecindad de la raíz. Esta hipótesis garantiza que el error cometido en la aproximación de Taylor de primer orden sea controlable.

Cuando se cumplen estas condiciones, el método de Newton-Raphson multivariable exhibe típicamente **convergencia cuadrática**. Esto significa que el error en la iteración $k+1$ es proporcional al cuadrado del error en la iteración k. Es decir, si $\|\mathbf{x}_k - \mathbf{x}^*\| \approx \varepsilon$, entonces:

$$\|\mathbf{x}_{k+1} - \mathbf{x}^*\| \leq C\|\mathbf{x}_k - \mathbf{x}^*\|^2,$$

donde C es una constante positiva dependiente de $\mathbf{F}$ y de sus derivadas. Este comportamiento implica que, cerca de la raíz, el número de cifras significativas correctas se duplica en cada iteración, lo que resulta en una velocidad de convergencia muy rápida.

La razón de esta eficiencia es que el método utiliza una aproximación de primer orden de la función (linealización) basada en su expansión de Taylor, y el error está ligado al segundo orden de derivadas. A medida que nos acercamos a la solución, la linealización mejora su precisión, reduciendo drásticamente el error.

Ejemplo: Consideremos el sistema de ecuaciones:

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} x_1^2 + x_2^2 - 1 \\ x_1 - x_2 \end{bmatrix}$$

La raíz del sistema es $\mathbf{x}^* = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$.

La matriz Jacobiana es:

$$\mathbf{J}_F(\mathbf{x}) = \begin{bmatrix} 2x_1 & 2x_2 \\ 1 & -1 \end{bmatrix}$$

Apliquemos una sola iteración de Newton-Raphson desde un punto cercano, por ejemplo $\mathbf{x}_0 = (0.7, 0.6)$. Calculamos:

$$\mathbf{F}(\mathbf{x}_0) = \begin{bmatrix} 0.7^2 + 0.6^2 - 1 \\ 0.7 - 0.6 \end{bmatrix} = \begin{bmatrix} 0.49 + 0.36 - 1 \\ 0.1 \end{bmatrix}$$

$$\mathbf{F}(\mathbf{x}_0) = \begin{bmatrix} -0.15 \\ 0.1 \end{bmatrix}$$

$$\mathbf{J}_F(\mathbf{x}_0) = \begin{bmatrix} 1.4 & 1.2 \\ 1 & -1 \end{bmatrix}$$

El paso de Newton se calcula resolviendo el sistema:

$$\mathbf{J}_F(\mathbf{x}_0)\Delta\mathbf{x} = -\mathbf{F}(\mathbf{x}_0)$$

$$\Rightarrow \begin{bmatrix} 1.4 & 1.2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = \begin{bmatrix} 0.15 \\ -0.1 \end{bmatrix}$$

Resolviendo el sistema obtenemos $\Delta\mathbf{x}$, y así obtenemos $\mathbf{x}_1 = \mathbf{x}_0 + \Delta\mathbf{x}$. Si repetimos este procedimiento, se puede observar que la distancia al punto exacto $\mathbf{x}^*$ disminuye aproximadamente al cuadrado de la distancia anterior, verificando empíricamente la convergencia cuadrática.

C. Condiciones de estabilidad

El método de Newton-Raphson multivariable es ampliamente utilizado para resolver sistemas de ecuaciones no lineales. Su éxito depende de una serie de condiciones que garantizan su estabilidad y convergencia.

En primer lugar, la función vectorial $\mathbf{F}(\mathbf{x})$ debe ser diferenciable en una vecindad de la raíz buscada.

Para asegurar una buena convergencia —idealmente cuadrática— se requiere que $\mathbf{F}(\mathbf{x})$ sea al menos dos veces continuamente diferenciable. Esto permite que el método utilice una expansión en serie de Taylor precisa y se aproxime eficientemente a la solución.

Una condición crítica es la **no singularidad** de la matriz Jacobiana $\mathbf{J}_{\mathbf{F}}(\mathbf{x}_k)$ en cada iteración y especialmente en la raíz $\mathbf{x}^*$. Si el Jacobiano es singular o está mal condicionado, el cálculo del paso de Newton se vuelve inestable, lo que puede provocar una divergencia o una convergencia muy lenta. Un Jacobiano bien condicionado garantiza una dirección de búsqueda clara y efectiva hacia la raíz.

La elección del punto inicial $\mathbf{x}_0$ también es fundamental. El método tiene convergencia local, lo que significa que solo funcionará si $\mathbf{x}_0$ está lo suficientemente cerca de la raíz. Una mala elección puede llevar a resultados incorrectos o incluso a la falla total del proceso iterativo. En la práctica, se recomienda realizar un análisis previo o usar métodos gráficos o numéricos para obtener una buena estimación inicial.

Finalmente, aunque la formulación estándar del método requiere que el número de ecuaciones sea igual al número de variables, existen variantes que permiten resolver sistemas sobredeterminados o subdeterminados. En estos casos, se pueden aplicar técnicas como la pseudo-inversa de Moore-Penrose o la minimización por mínimos cuadrados.

Ejemplo: Considere el sistema de ecuaciones:

$$\begin{cases} x^2 + y^2 - 1 = 0 \\ x^2 - y = 0 \end{cases}$$

La raíz $(x, y) = (0, 0)$ no satisface ambas ecuaciones, pero supongamos que partimos de $\mathbf{x}_0 = (0.1, 0.1)$.

La matriz Jacobiana es:

$$\mathbf{J}_F(x, y) = \begin{bmatrix} 2x & 2y \\ 2x & -1 \end{bmatrix}$$

En $(0, 0)$, el Jacobiano es:

$$\mathbf{J}_F(0, 0) = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$$

Esta matriz es *singular*, por lo tanto no se puede aplicar el método en ese punto. Incluso cerca del origen, el Jacobiano está mal condicionado, lo que puede provocar que el método falle si el punto inicial está demasiado cerca de la singularidad.

Este ejemplo muestra que, incluso con funciones aparentemente simples, no cumplir las condiciones de estabilidad puede hacer que el método sea inefectivo. Para asegurar la estabilidad del método de Newton-Raphson multivariable se deben cumplir las siguientes condiciones:

- $\mathbf{F}(\mathbf{x})$ debe ser suficientemente suave (diferenciable).
- El Jacobiano $\mathbf{J}_{\mathbf{F}}(\mathbf{x})$ debe ser no singular y estar bien condicionado.
- El punto inicial $\mathbf{x}_0$ debe estar dentro de la cuenca de atracción de la raíz.
- El sistema debe estar bien planteado (número adecuado de ecuaciones y variables), o adaptarse correctamente si no lo está.

D. Ejemplos

Ejemplo 1

La función $f(x) = x^2 - x - \frac{1}{2}$ tiene una y sólo una raíz en el intervalo $[-0.5, 0]$.
Comprobar si se cumplen las condiciones del teorema anterior. En caso afirmativo, hallar la solución para las condiciones iniciales:

$$x_0 = -\frac{1}{2}, \quad x_0 = -\frac{2}{5}, \quad x_0 = -0.3$$

Solución:
Primero, calculamos la derivada:

$$f(x) = x^2 - x - \frac{1}{2}, \quad f'(x) = 2x - 1$$

Condiciones de convergencia: Para que Newton-Raphson converja:

$f(x)$ y $f'(x)$ continuas en el intervalo $[-0.5, 0]$

Ambas son polinomios $\rightarrow$ Son continuas.
Además:

$$f'(x) = 2x - 1$$

En $x = -0.5$:

$$f'(-0.5) = 2(-0.5) - 1 = -2 \neq 0$$

En $x = 0$:

$$f'(0) = 2(0) - 1 = -1 \neq 0$$

La derivada no se anula en el intervalo. Entonces, se cumplen las condiciones del teorema.
Aplicando Newton-Raphson para cada valor inicial:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Caso 1: $x_0 = -\frac{1}{2}$

$$f(-0.5) = 0.25 + 0.5 - 0.5 = 0.25$$

$$f'(-0.5) = -2$$

$$x_1 = -0.5 - \frac{0.25}{-2} == -0.375$$

Caso 2: $x_0 = -\frac{2}{5} = -0.4$

$$f(-0.4) = (-0.4)^2 - (-0.4) - 0.5 = 0.06$$

$$f'(-0.4) = -0.8 - 1 = -1.8$$

$$x_1 = -0.4 - \frac{0.06}{-1.8} = -0.3667$$

Caso 3: $x_0 = -0.3$

$$f(-0.3) = (-0.3)^2 - (-0.3) - 0.5 = -0.11$$

$$f'(-0.3) = -0.6 - 1 = -1.6$$

$$x_1 = -0.3 - \frac{-0.11}{-1.6} = -0.3687$$

Conclusión:

En los tres casos, Newton-Raphson aproxima a la raíz en el intervalo $[-0.5, 0]$.

Ejemplo 2

La función $f(x) = x^3 - x - 2$ tiene una raíz en el intervalo $[1, 2]$. Comprobar si se cumplen las condiciones del teorema anterior y aplicar el método de Newton-Raphson con $x_0 = 1.5$.

Solución:

$$f(x) = x^3 - x - 2, \quad f'(x) = 3x^2 - 1$$

Condiciones de convergencia: En $x = 1$:

$$f'(1) = 3(1)^2 - 1 = 2 \neq 0$$

En $x = 2$:

$$f'(2) = 3(2)^2 - 1 = 11 \neq 0$$

La derivada no se anula en el intervalo. Se cumplen las condiciones.

Aplicando Newton-Raphson:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Paso 1:

$$f(1.5) = (1.5)^3 - 1.5 - 2 = -0.125$$

$$f'(1.5) = 3(1.5)^2 - 1 = 6.75$$

$$x_1 = 1.5 - \frac{-0.125}{6.75} = 1.5185$$

Paso 2 (opcional para más precisión):

$$f(1.5185) = (1.5185)^3 - 1.5185 - 2 \approx -0.002$$

$$f'(1.5185) \approx 5.92$$

$$x_2 = 1.5185 - \frac{-0.002}{5.92} \approx 1.5185 + 0.00034 = 1.5188$$

Conclusión:

El método aproxima la raíz ≈ 1.519, que es la solución real para $x^3 - x - 2 = 0$.

III. Ventajas y Desventajas de los Métodos

Los métodos iterativos de optimización son herramientas fundamentales en múltiples campos como la estadística, el machine learning, la física computacional y la ingeniería. Entre estos métodos, el Gradiente Descendente y Newton-Raphson representan dos enfoques con características distintivas en términos de convergencia, costo computacional y aplicabilidad.

Gradiente Descendiente

Ventajas

- **Simplicidad conceptual y de implementación**
  - Solo requiere el cálculo de la primera derivada (gradiente)
  - Algoritmo más intuitivo y fácil de programar
- **Menor costo computacional por iteración**
  - Cada iteración es relativamente económica en términos de cálculos
  - Especialmente ventajoso en problemas de alta dimensionalidad
- **Robustez**
  - Funciona en una amplia gama de funciones, incluso cuando no son dos veces diferenciables
  - Menos sensible a las condiciones iniciales que Newton-Raphson
- **Paralelización**
  - Se puede implementar eficientemente en sistemas paralelos
  - Adecuado para problemas a gran escala como el entrenamiento de redes neuronales
- **Garantía de convergencia**
  - Con tamaño de paso adecuado, siempre converge a un mínimo local
  - Especialmente útil en funciones convexas

Desventajas

- **Convergencia lenta**
  - Puede requerir muchas iteraciones para alcanzar la solución
  - Particularmente lento cerca del óptimo
- **Sensibilidad al tamaño del paso**
  - La elección de la tasa de aprendizaje es crítica
  - Demasiado grande: puede diverger; demasiado pequeña: convergencia muy lenta
- **Comportamiento zigzagueante**
  - En valles estrechos puede oscilar entre las paredes del valle
  - Ineficiente en problemas mal condicionados (donde la curvatura varía mucho)
- **Problemas con puntos de silla**
  - Puede estancarse en puntos de silla, especialmente en alta dimensionalidad
  - Requiere modificaciones como el momentum para superarlos
- **Solo información de primer orden**
  - No aprovecha la información de curvatura de la función
  - Limitado al avanzar en la dirección negativa del gradiente

Newton-Raphson

Ventajas

- **Convergencia cuadrática**
  - Converge mucho más rápidamente que el gradiente descendente cerca del óptimo
  - Puede alcanzar precisión en pocas iteraciones cuando está bien inicializado
- **Uso de información de segundo orden**
  - Utiliza la matriz Hessiana (segundas derivadas)
  - Incorpora información sobre la curvatura de la función objetivo
- **Invariante a la escala**
  - Menos sensible a problemas mal condicionados
  - No requiere ajuste manual de la tasa de aprendizaje
- **Menor número de iteraciones**
  - Generalmente necesita muchas menos iteraciones que el gradiente descendente
  - Ideal para problemas donde cada evaluación de función es costosa
- **Precisión**
  - Logra mayor precisión en la solución final
  - Especialmente útil en problemas que requieren alta exactitud

Desventajas

- **Alto costo computacional por iteración**
  - Requiere calcular y almacenar la matriz Hessiana
  - La Hessiana tiene complejidad $O(n^3)$ para n dimensiones
- **Necesidad de segundas derivadas**
  - Requiere que la función sea dos veces diferenciable
  - Las segundas derivadas pueden ser difíciles de calcular analíticamente
- **Problemas de estabilidad**
  - Puede diverger si la matriz Hessiana no es definida positiva
  - Sensible a las condiciones iniciales
- **Limitaciones en alta dimensionalidad**
  - Impracticable para problemas con millones de parámetros
  - La matriz Hessiana puede ser demasiado grande para almacenar en memoria
- **Complejidad de implementación**
  - Más difícil de implementar correctamente
  - Requiere técnicas adicionales como regularización o modificaciones de la Hessiana

IV. Comparación en aplicaciones específicas

Contexto	Gradiente Descendente	Newton-Raphson
Machine Learning	Preferido para redes neuronales y modelos de gran escala debido a su eficiencia en memoria y paralelización.	Más adecuado para modelos con pocos parámetros o para el refinamiento final de soluciones.
Optimización de funciones analíticas	Útil cuando solo se necesita una aproximación razonable o cuando las funciones son complejas.	Superior cuando se requiere alta precisión y la función es bien comportada.
Sistemas no lineales	Menos efectivo para encontrar raíces de sistemas no lineales.	Especialmente diseñado para encontrar raíces de ecuaciones y sistemas no lineales.

TABLE I
Comparación de aplicabilidad de los métodos según el contexto.

References

[1] IBM, "¿Qué es el descenso del gradiente?", IBM Think, [En línea]. Disponible en: https://www.ibm.com/mx-es/think/topics/gradient-descent.[Accedido:4-may-2025].
[2] Departamento de Matemáticas, "Raíces de Ecuaciones - Método de Newton-Raphson," *Universidad de Las Palmas de Gran Canaria*, [En línea]. Disponible en: https://estadistica-dma.ulpgc.es/FCC/05-3-Raices-de-Ecuaciones-2.html.[Accedido:4-may-2025].
[3] J. E. Vargas Cantero, "Método de Newton-Raphson," *Facultad de Ingeniería, Universidad Tecnológica de Bolívar*, Cartagena, Colombia. [En línea]. Disponible en: http://personal.cimat.mx:8181/julio/courses/progra01/clase18/MetododeNewtonRaph
[4] F. B. Hildebrand, Introduction to Numerical Analysis, 2nd ed. Dover Publications, 1987.
[5] R. W. Hamming, Introduction to Applied Numerical Analysis. Dover Books on Mathematics, 1989.
[6] J. F. Epperson, An Introduction to Numerical Methods and Analysis. John Wiley & Sons, 2013.
[7] J. E. Dennis and R. B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Society for Industrial and Applied Mathematics, 1996.