Métodos Computacionales

Clase X: Gradiente Descendiente y Método de Newton

Introducción

Objetivo de la Optimización

- El objetivo de un problema de optimización es simple:

$$\min f\left(x_1, x_2, \ldots, x_n\right)$$

donde $\mathbf{x} \in \boldsymbol{\mathcal{X}}$, un subconjunto de $\mathbb{R}^{\mathrm{n}}$

Objetivo de la Optimización

- El objetivo de un problema de optimización es simple:

$$\min f\left(x_1, x_2, \ldots, x_n\right)$$

donde $\mathbf{x} \in \boldsymbol{X}$, un subconjunto de $\mathbb{R}^{\mathrm{n}}$

- El punto $\mathbf{x}$ que minimiza la función es:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \chi}{\operatorname{argmin}} f(\boldsymbol{x})$$

El Mapa de la Optimización

- Existen una gran cantidad de métodos de optimización, que aplican a diferentes contextos y problemas.
- Podemos categorizarlos de muchas maneras:
  - Si permiten o no la incorporación de restricciones
  - Si utilizan variables reales o enteras
  - Si son exactos o aproximados
  - Si son deterministas o estocásticos
  - …

Métodos de Optimización

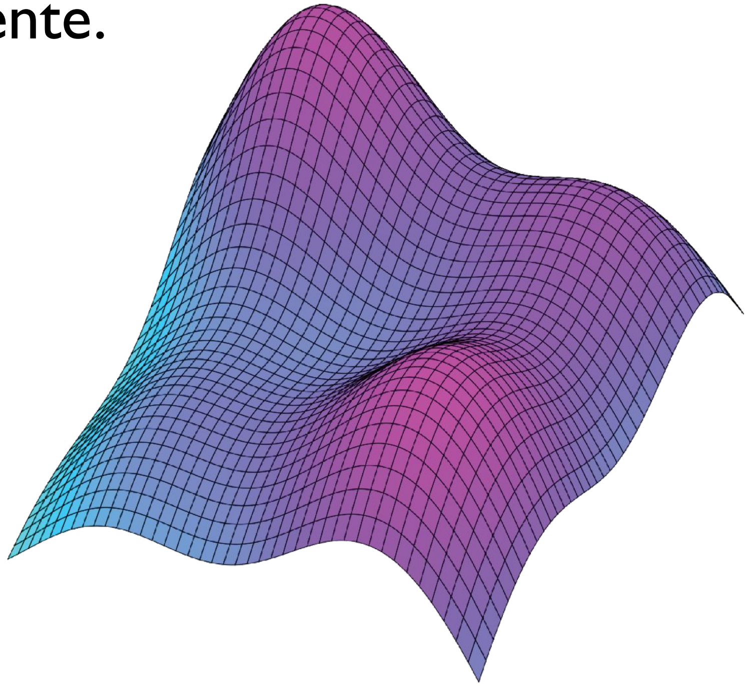
- Optimización con restricciones
  - **Programación Lineal:** Método Simplex, Simplex Dual, Método de Barrera, Método de Punto Interior
  - **Programación Lineal Entera:**
    - *Exactos:* Ramificación y Acotamiento (Branch and Bound), Método de Plano de Corte
    - *Heurísticos:* Recocido Simulado, Búsqueda Tabú, Optimización por Colonia de Hormigas

Métodos de Optimización

- Optimización sin Restricciones
  - **Métodos sin Derivada**: Búsqueda Aleatoria, Algoritmos Genéticos, Recocido Simulado
  - **Métodos de Primer Orden:** Descenso de Gradiente
  - **Métodos de Segundo Orden:** Método de Newton

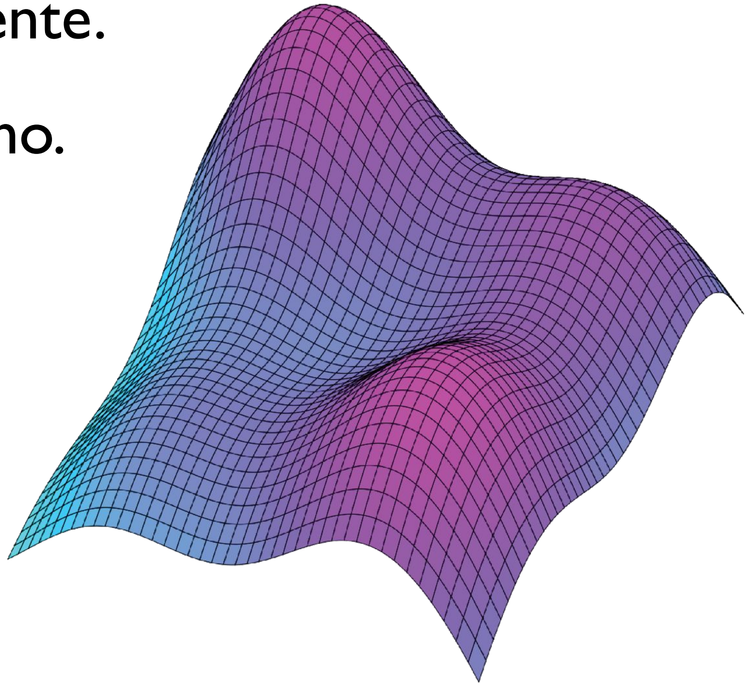
Idea fundamental de la optimización computacional

- Recorrer $f(\mathbf{x})$ de una manera inteligente.



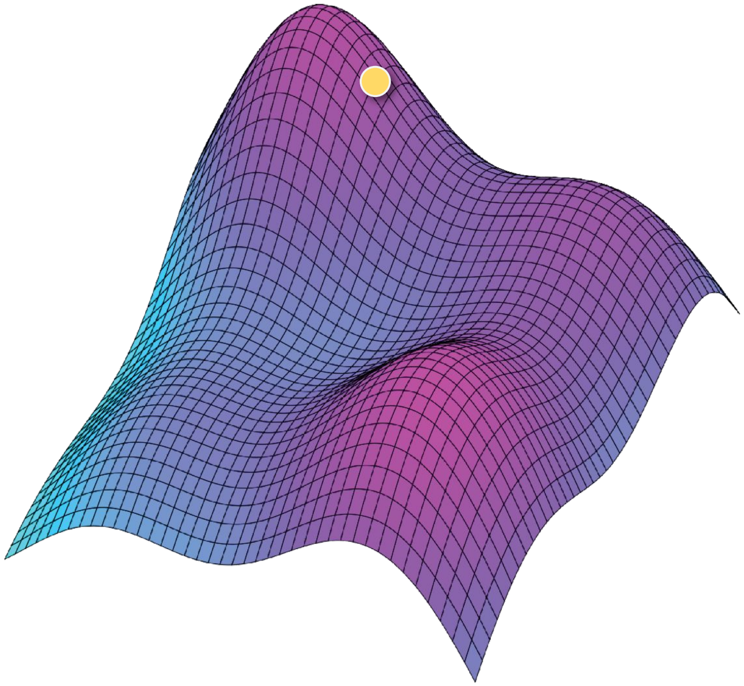
Idea fundamental de la optimización computacional

- Recorrer $f(\mathbf{x})$ de una manera inteligente.
- Evaluar $f(\mathbf{x})$ hasta encontrar el mínimo.



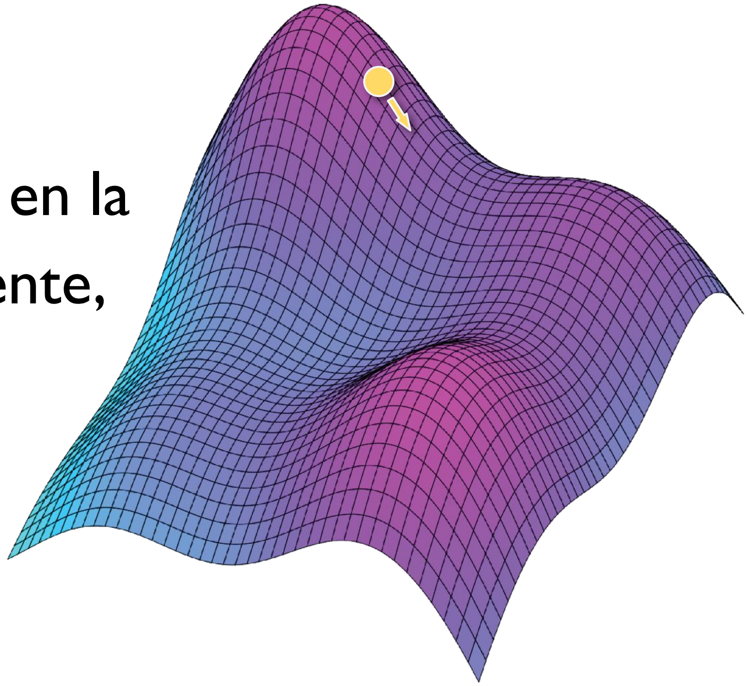
Idea fundamental de la optimización computacional

- Algoritmo de minimización básico:
  - Partimos de punto inicial $f(\mathbf{x}_0)$,



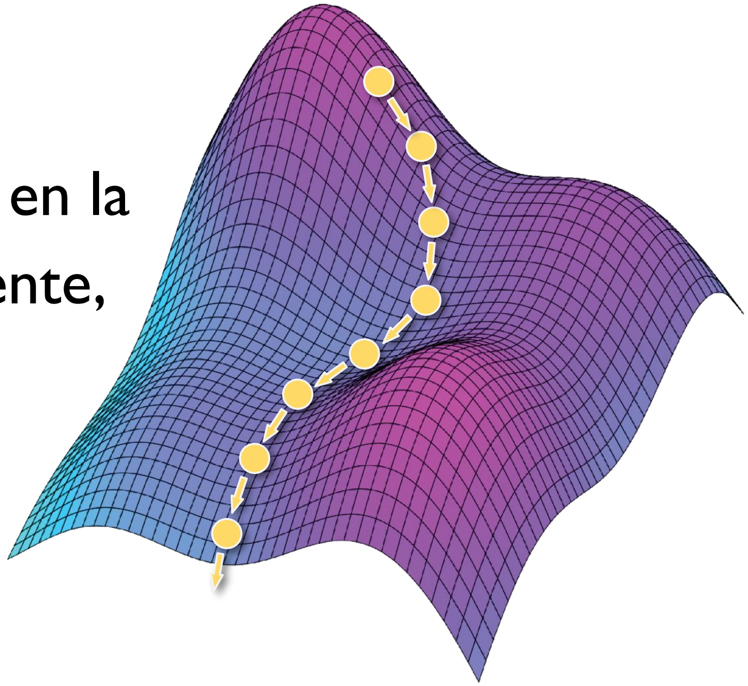
Idea fundamental de la optimización computacional

- Algoritmo de minimización básico:
  - Partimos de punto inicial $f(\mathbf{x}_0)$,
  - Nos movemos $\Delta\mathbf{x}$ en la dirección en la que $f(\mathbf{x})$ disminuye más rápidamente, hasta un punto $f(\mathbf{x}_1)$.



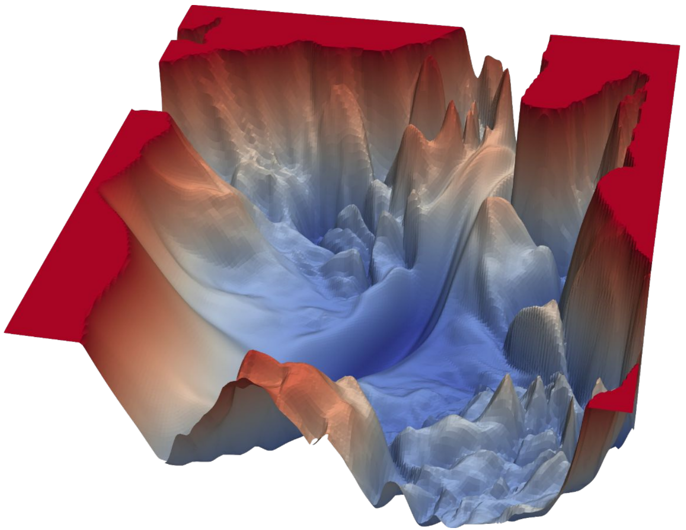
Idea fundamental de la optimización computacional

- Algoritmo de minimización básico:
  - Partimos de punto inicial $f(\mathbf{x}_0)$,
  - Nos movemos $\Delta\mathbf{x}$ en la dirección en la que $f(\mathbf{x})$ disminuye más rápidamente, hasta un punto $f(\mathbf{x}_1)$.
  - Repetimos hasta que no haya una dirección hacia la cual disminuya la función $f(\mathbf{x})$.



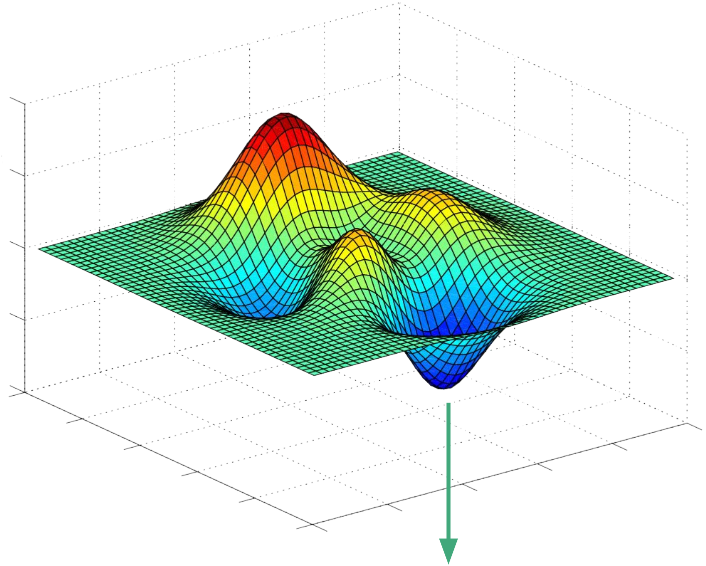
Idea fundamental de la optimización computacional

- Hay muchas posibilidades de **no encontrar el mínimo!!** (no hay convergencia, no encuentro un mínimo global, etc.)
- Existen métodos de optimización que intentan lidiar con diferentes problemas de este tipo.



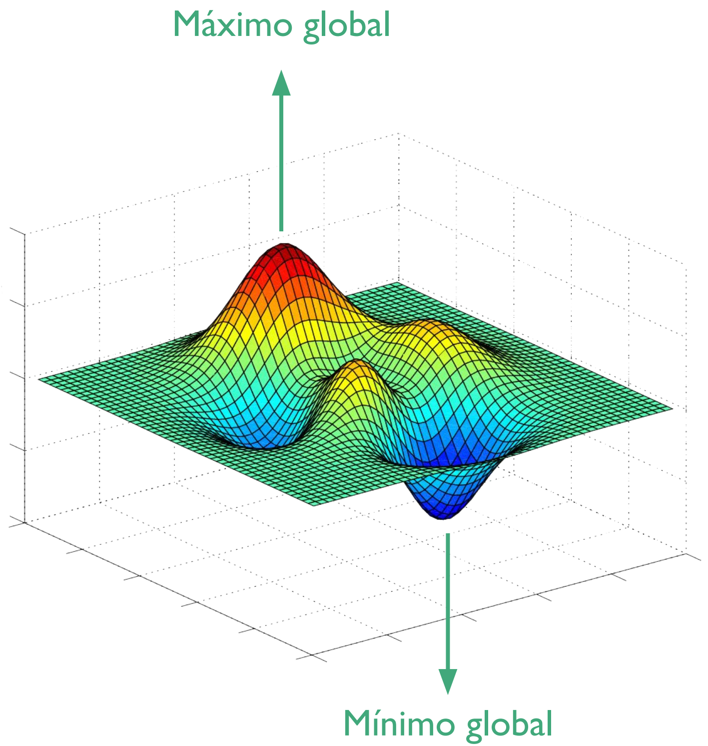
Óptimos locales y óptimos globales

- El punto donde se obtiene el **menor valor** de $f(\mathbf{x})$ se denomina mínimo global.



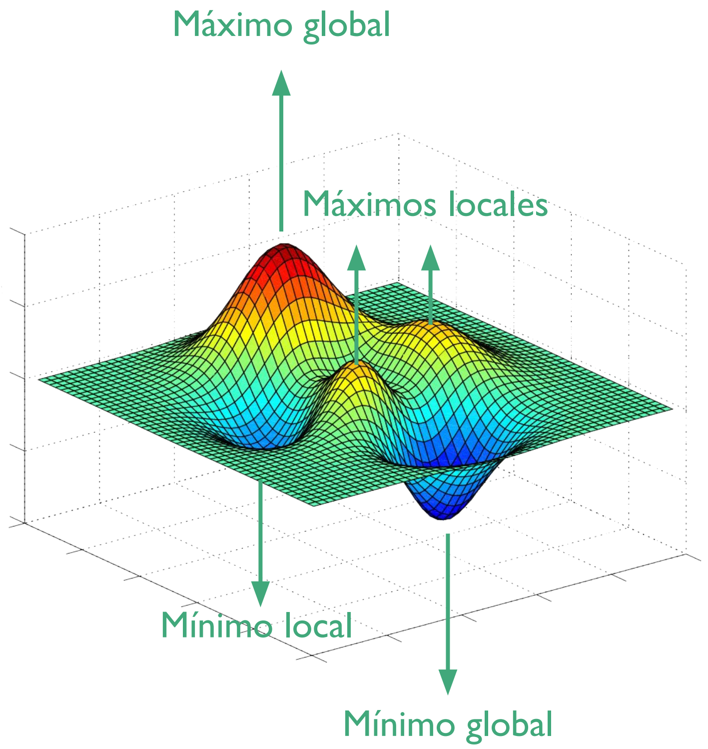
Óptimos locales y óptimos globales

- El punto donde se obtiene el **menor valor** de $f(\mathbf{x})$ se denomina mínimo global.
- El punto donde se obtiene el **mayor valor** de $f(\mathbf{x})$ se denomina máximo global.



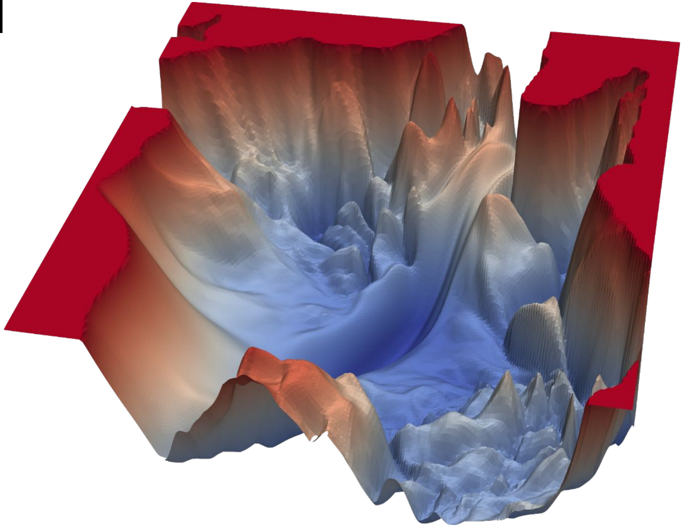
Óptimos locales y óptimos globales

- El punto donde se obtiene el **menor valor** de $f(\mathbf{x})$ se denomina mínimo global.
- El punto donde se obtiene el **mayor valor** de $f(\mathbf{x})$ se denomina máximo global.
- Una función puede tener múltiples **máximos y mínimos locales**.



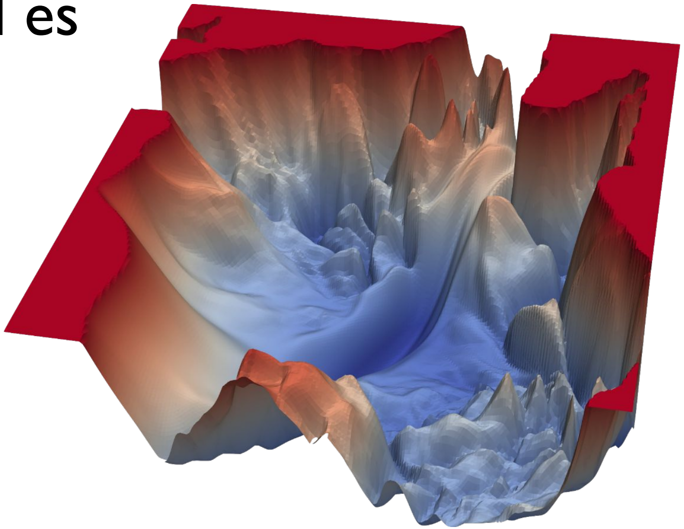
Óptimos locales y óptimos globales

- En general, encontrar un óptimo global es computacionalmente imposible.



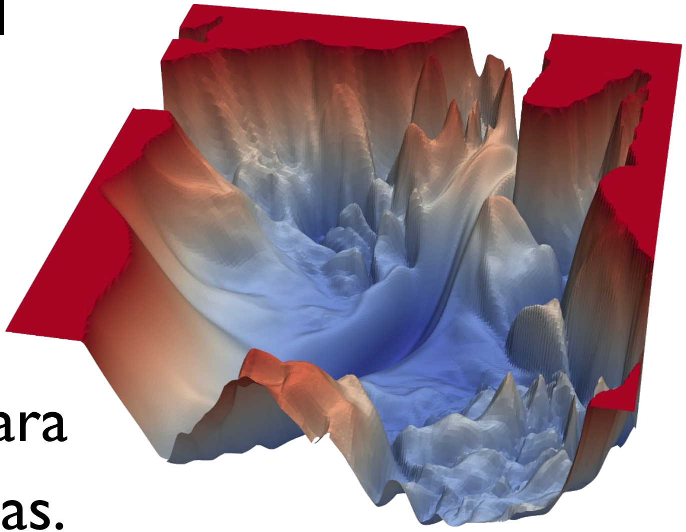
Óptimos locales y óptimos globales

- En general, encontrar un óptimo global es computacionalmente imposible.
- La mayoría de las veces sólo podemos encontrar óptimos locales.



Óptimos locales y óptimos globales

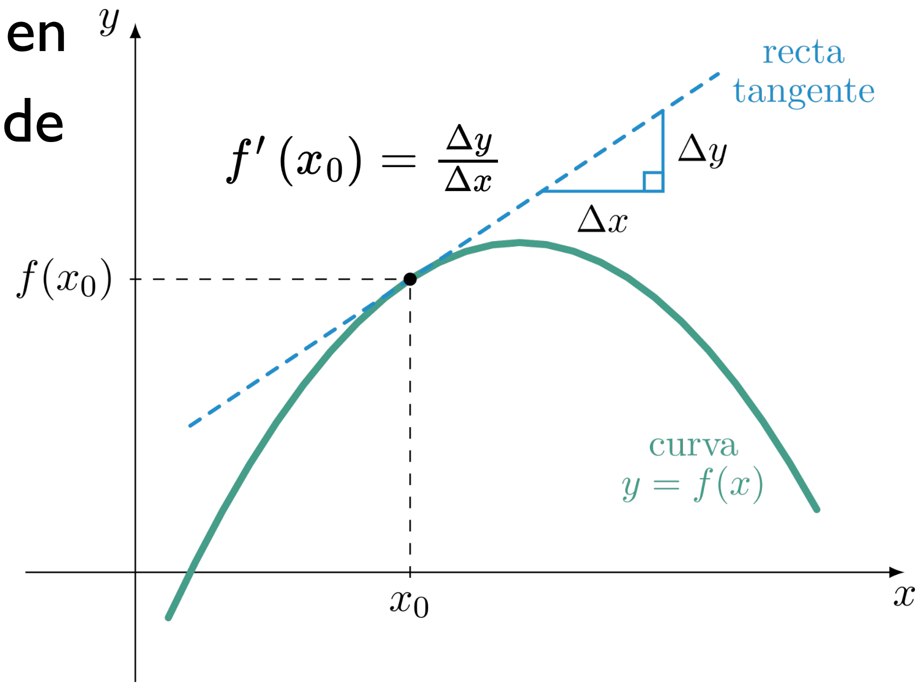
- En general, encontrar un óptimo global es computacionalmente imposible.
- La mayoría de las veces sólo podemos encontrar óptimos locales.
- Aunque decepcionante, es suficiente para resolver una gran cantidad de problemas.



Métodos de Primer Orden

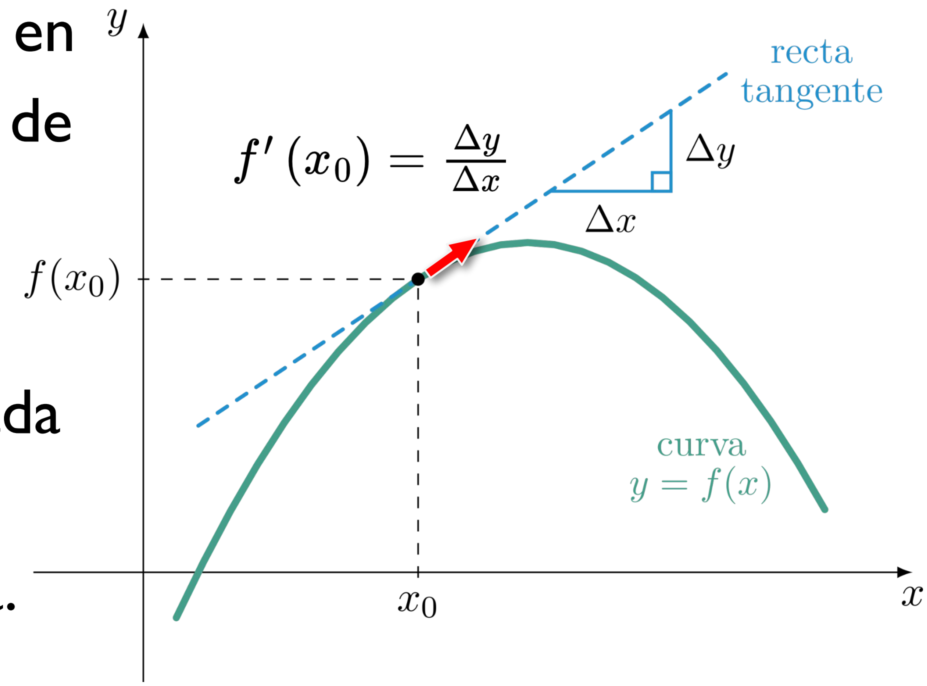
Dirección de descenso (2D)

- Derivada de la función $f(x)$ en un punto x_0 es la pendiente de la recta tangente en $f(x_0)$.



Dirección de descenso (2D)

- Derivada de la función $f(x)$ en un punto x_0 es la pendiente de la recta tangente en $f(x_0)$.
- Nos dice cuánto escalar un pequeño cambio en la entrada para obtener un cambio correspondiente en la salida.



Dirección de descenso (2D)

- Ejercicio:

  a. Dado el punto inicial $x_0 = 9.45$ encontrar la pendiente de la recta tangente en ese punto para $f(x) = x^2$.

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir del punto inicial.

Dirección de descenso (2D)

- Ejercicio:

  a. Dado el punto inicial $x_0 = 9.45$ encontrar la pendiente de la recta tangente en ese punto para $f(x) = x^2$.

$$f(x) = x^2 \longrightarrow f'(x) = 2x$$

Dirección de descenso (2D)

- Ejercicio:

  a. Dado el punto inicial $x_0 = 9.45$ encontrar la pendiente de la recta tangente en ese punto para $f(x) = x^2$.

$$f(x) = x^2 \longrightarrow f'(x) = 2x$$

$$f'(9.45) = 2(9.45) = \underline{18.9}$$

Dirección de descenso (2D)

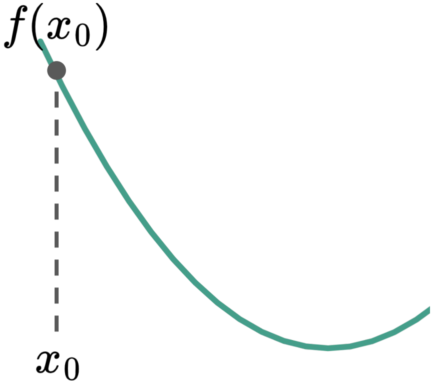
- Ejercicio:

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.

Dirección de descenso (2D)

- Ejercicio:

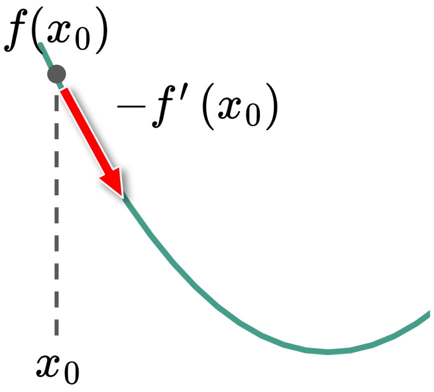
  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

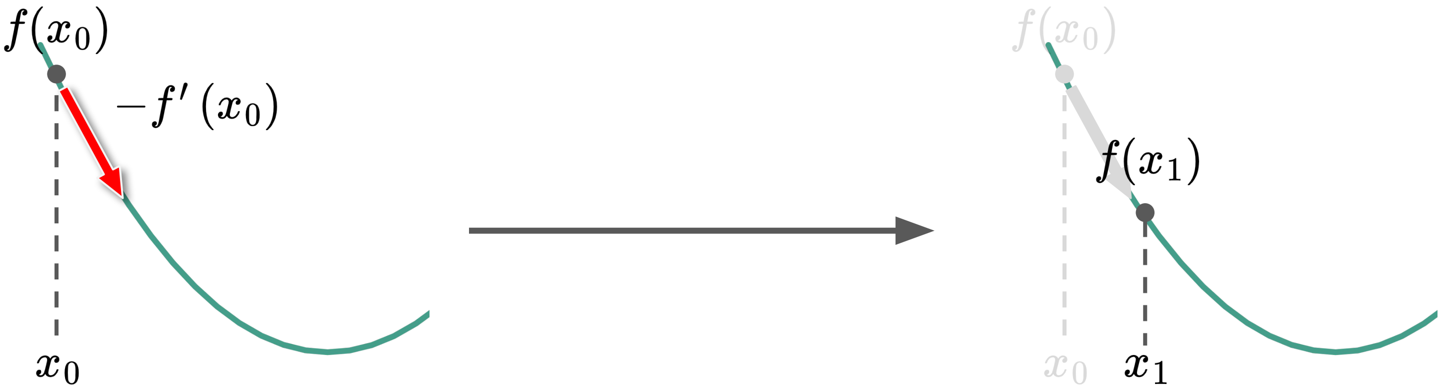
  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

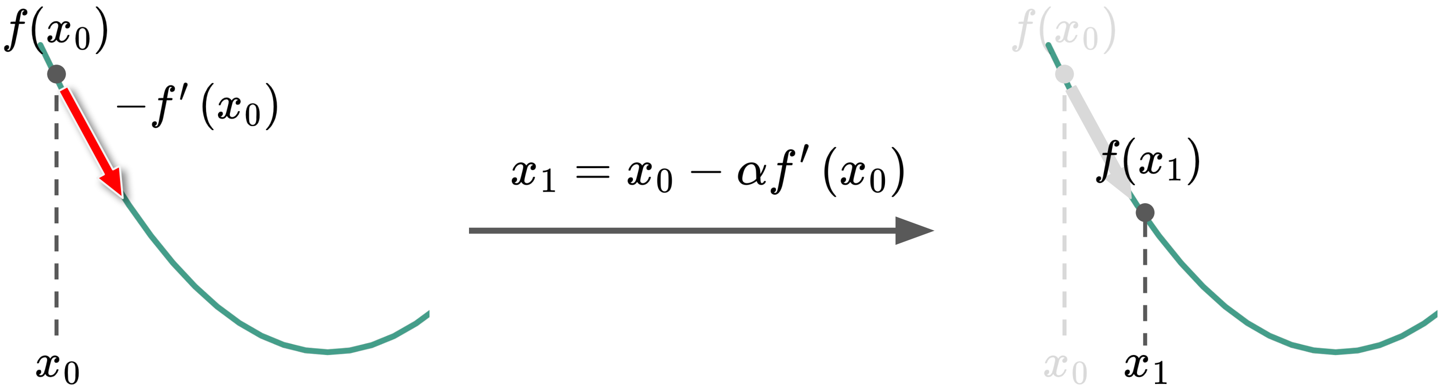
  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

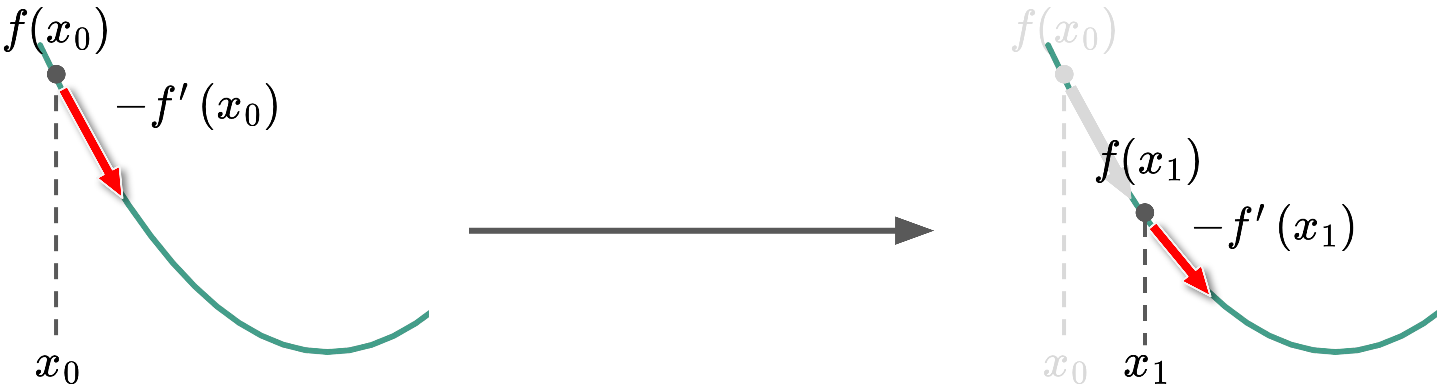
  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

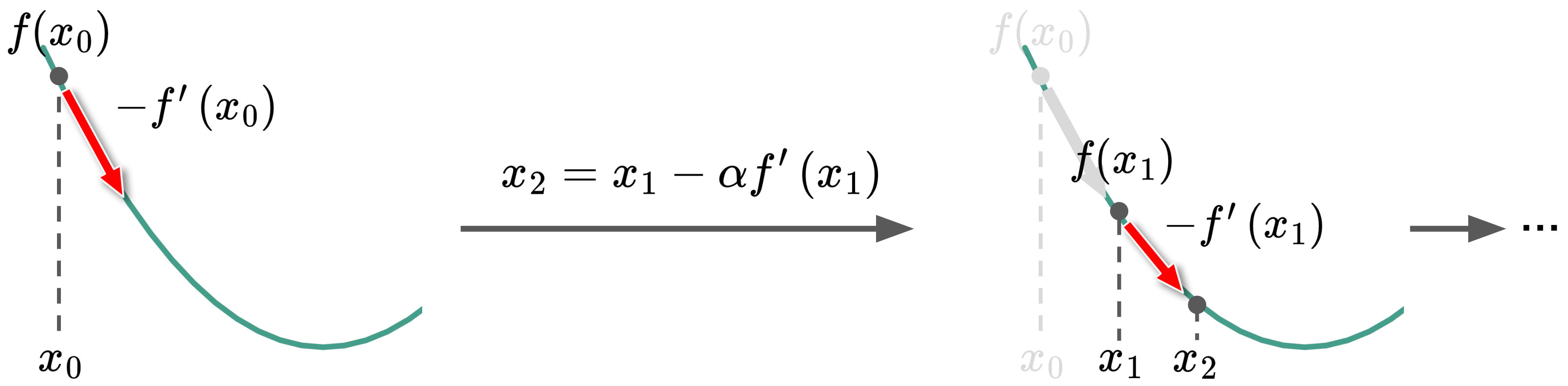
  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.



Dirección de descenso (2D)

- Ejercicio:

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.

$$\left.\begin{aligned} x_1 &= x_0 - \alpha f'(x_0) \\ x_2 &= x_1 - \alpha f'(x_1) \\ x_3 &= x_2 - \alpha f'(x_2) \\ &\cdots \\ x_{t+1} &= x_t - \alpha f'(x_t) \end{aligned}\right\} \text{para t+1 iteraciones}$$

Dirección de descenso (2D)

- Ejercicio:

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.

$$\left.\begin{aligned} x_1 &= x_0 - \alpha f'(x_0) \\ x_2 &= x_1 - \alpha f'(x_1) \\ x_3 &= x_2 - \alpha f'(x_2) \\ &\dots \\ x_{t+1} &= x_t - \alpha f'(x_t) \end{aligned}\right\} \quad |f(x_{t+1}) - f(x_t)| < \epsilon$$

umbral de convergencia

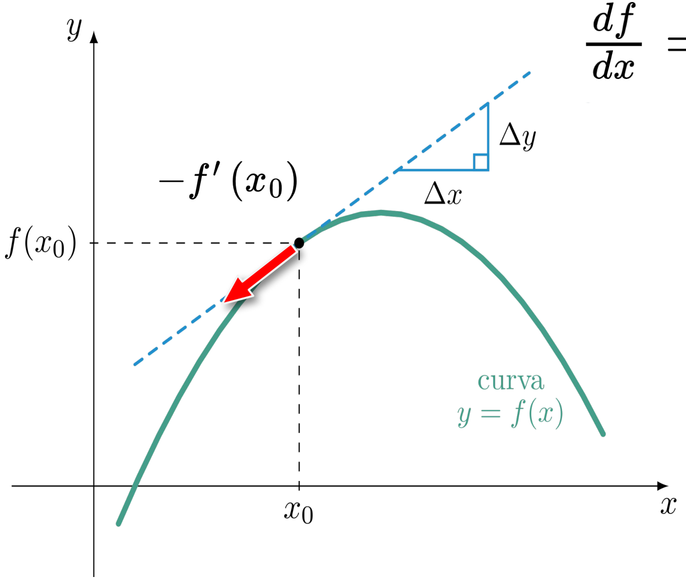
Dirección de descenso (2D)

- Ejercicio:

  b. Escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir de un punto inicial.

  Solución completa: <Notebook>

Dirección de descenso

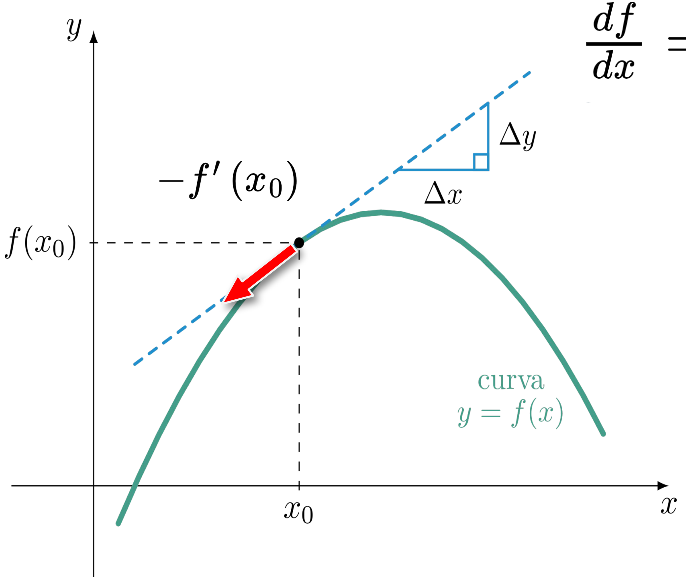


$$\frac{df}{dx} = f'(x)$$

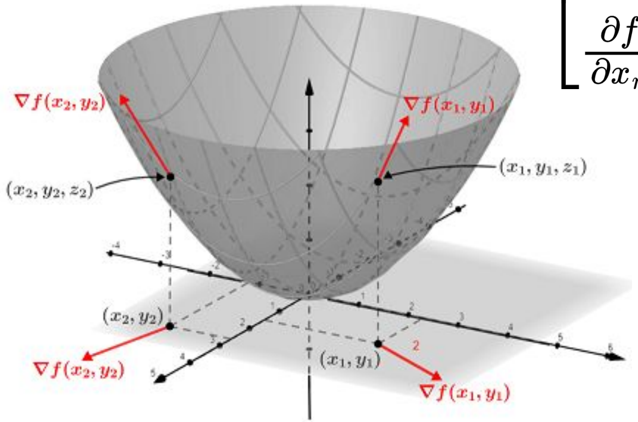
Caso Univariado

Dirección de descenso

$$\frac{df}{dx} = f'(x)$$



$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\mathbf{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\mathbf{x}) \end{bmatrix}$$



Caso Univariado

Caso Multivariado

Dirección de descenso (formal)

- Una dirección $\mathbf{d}$ es una dirección de descenso si existe un valor lo suficientemente pequeño ($\neq 0$) tal que, si nos movemos en dicha dirección, garantiza que $f(\mathbf{x})$ va a decrecer.

Dirección de descenso (formal)

- Una dirección $\mathbf{d}$ es una dirección de descenso si existe un valor lo suficientemente pequeño ($\neq 0$) tal que, si nos movemos en dicha dirección, garantiza que $f(\mathbf{x})$ va a decrecer.
- El gradiente evaluado en un punto $\mathbf{x_t}$ apunta en la dirección de máximo aumento de f.

Dirección de descenso (formal)

- Una dirección $\mathbf{d}$ es una dirección de descenso si existe un valor lo suficientemente pequeño ($\neq 0$) tal que, si nos movemos en dicha dirección, garantiza que $f(\mathbf{x})$ va a decrecer.
- El gradiente evaluado en un punto $\mathbf{x_t}$ apunta en la dirección de máximo aumento de f.
- El gradiente negativo es una dirección de descenso, $\mathbf{d} = -\nabla f$ llamada **steepest descent**.

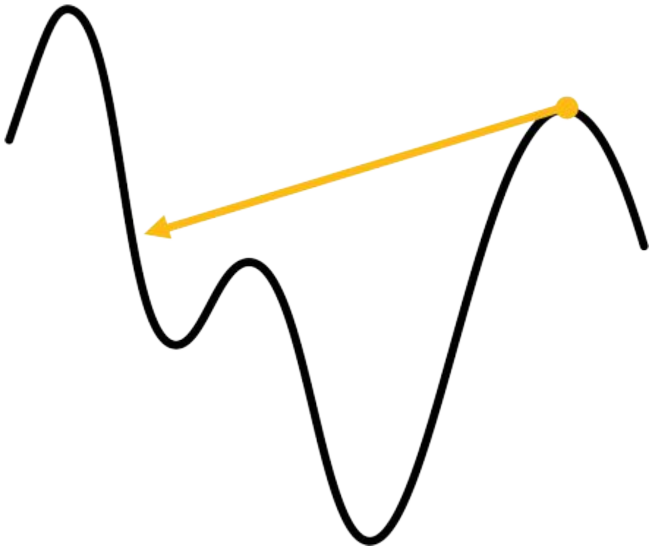
Algoritmo de Gradiente Descendiente

1. Elegir un punto inicial $\mathbf{x}_0$
2. Mientras no haya convergencia (o no se haya alcanzado un límite de iteraciones):
   a. Computar el gradiente en el punto $\nabla f(\mathbf{x}_t)$
   b. Computar el nuevo punto $\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha \nabla f(\mathbf{x}_t)$
   c. Verificar convergencia y actualizar $\mathbf{x}_t = \mathbf{x}_{t+1}$

Tasa de aprendizaje (α)

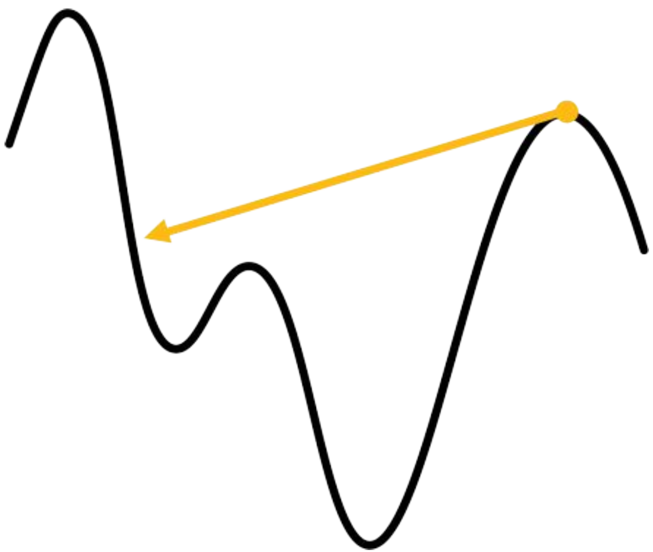
- La tasa de aprendizaje o (learning rate) es el **hiper parámetro** más importante de nuestro algoritmo.
- Afecta directamente cuestiones de performance, pero también de convergencia.
- Una selección incorrecta de este valor puede hacer que el algoritmo fracase completamente en su búsqueda del mínimo.

Tasa de aprendizaje (α)

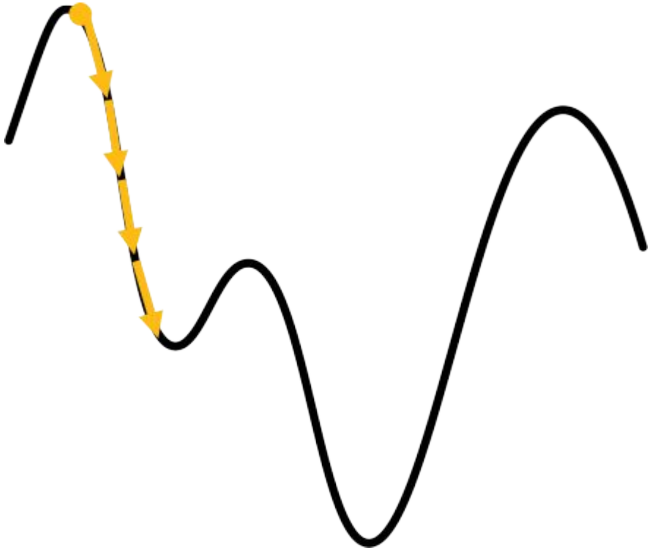


α demasiado grande

Tasa de aprendizaje (α)



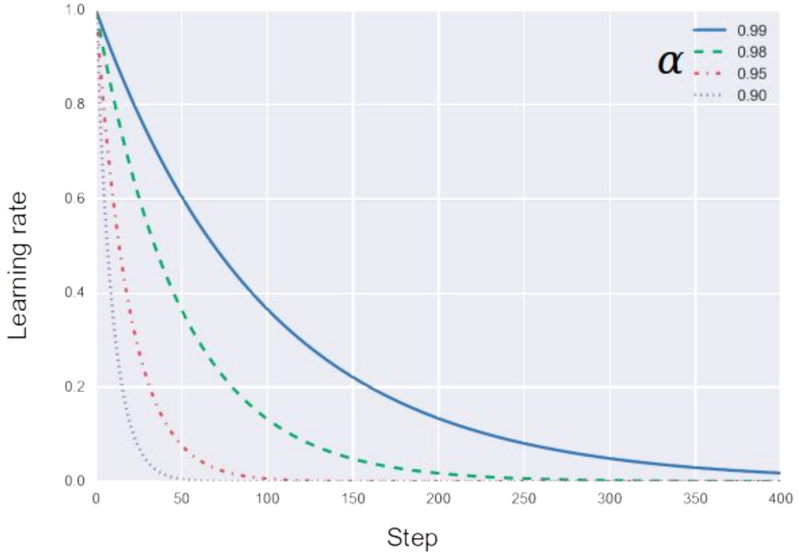
α demasiado grande



α demasiado chico

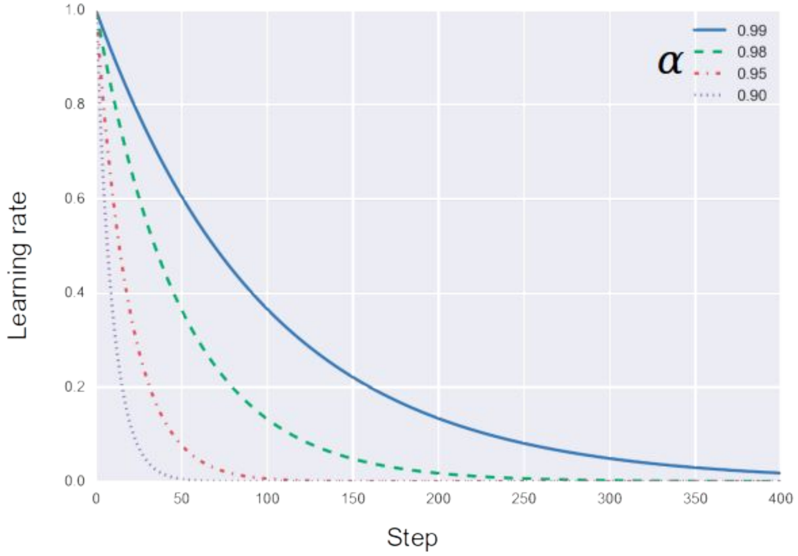
Tasa de aprendizaje (α)

- En la práctica, la mayoría de los algoritmos de optimización implementan tasas de aprendizaje adaptativas.



Tasa de aprendizaje (α)

- En la práctica, la mayoría de los algoritmos de optimización implementan tasas de aprendizaje adaptativas.
- Comienzan con factores grandes y lo reducen exponencialmente.



Algoritmo de Gradiente Descendiente

- Si bien parece ser la mejor decisión, puede resultar un algoritmo muy lento para cierto tipo de funciones.

Algoritmo de Gradiente Descendiente

- Si bien parece ser la mejor decisión, puede resultar un algoritmo muy lento para cierto tipo de funciones.
- Para determinar la dirección de descenso en cada iteración t se utilizan:
  - La derivada o el **gradiente** en el punto $\mathbf{x}_t$ (**métodos de primer orden**).
  - La **curvatura** (o convexidad) en el punto $\mathbf{x}_t$ (**métodos de segundo orden**).

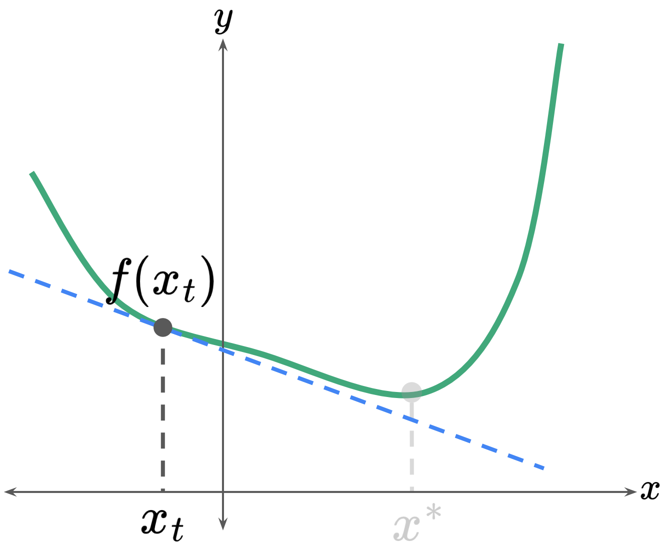
Métodos de Segundo Orden

Motivación

- Los métodos de primer orden son baratos de computar (sólo necesitan evaluar el gradiente en un punto).
- Pueden necesitar muchas iteraciones para alcanzar la convergencia.
- Podemos aliviar este problema si contamos con una **mejor forma de escalar el gradiente** (parámetro α).

Método de Newton

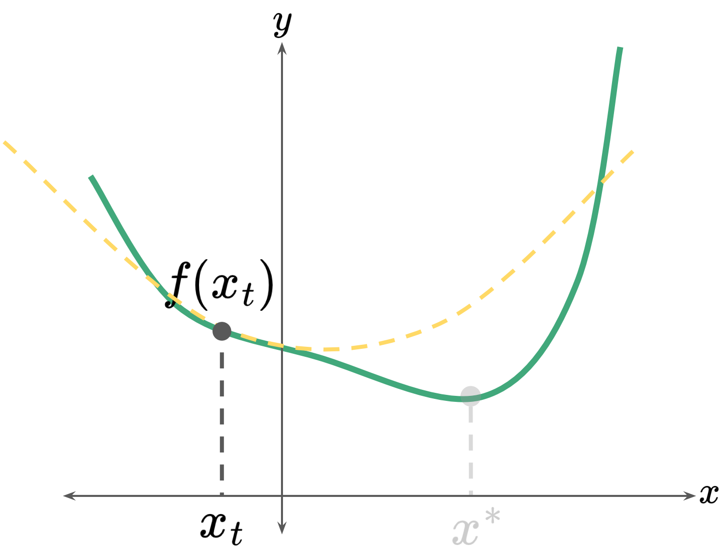
- Gradiente Descendente: aproximación de la función f mediante una recta.



$$f(x_{t+1}) \approx f\left(x_t\right) + f'\left(x_t\right)\left(x_{t+1} - x_t\right)$$

Método de Newton

- Gradiente Descendente: aproximación de la función f mediante una recta.
- **Observación:** usar la segunda derivada en el punto x_t puede aproximar mejor la función.



$$f(x_{t+1}) \approx \underbrace{f\left(x_t\right) + f'\left(x_t\right)\left(x_{t+1} - x_t\right) + f''\left(x_t\right)\frac{\left(x_{t+1}-x_t\right)^2}{2}}_{\text{Aproximación de Taylor}}$$

Aproximación de Taylor (repaso)

- Recordemos la serie de Taylor para una función de una variable:

$$f(x+\Delta x) = f(x) + \frac{df}{dx}(x)(\Delta x) + \frac{1}{2}\frac{d^2 f}{dx^2}(x)(\Delta x)^2 + \frac{1}{3!}\frac{d^3 f}{dx^3}(x)(\Delta x)^3 + \cdots + \frac{1}{n!}\frac{d^n f}{dx^n}(x)(\Delta x)^n$$

Aproximación de Taylor (repaso)

- Recordemos la serie de Taylor para una función de una variable:

$$f(x+\Delta x) = f(x) + \frac{df}{dx}(x)(\Delta x) + \frac{1}{2}\frac{d^2 f}{dx^2}(x)(\Delta x)^2 + \frac{1}{3!}\frac{d^3 f}{dx^3}(x)(\Delta x)^3 + \cdots + \frac{1}{n!}\frac{d^n f}{dx^n}(x)(\Delta x)^n$$

- Como Δx es chico, en general, tomar hasta el término cuadrático alcanza para una buena aproximación (local):

$$f(x+\Delta x) \cong f(x) + \frac{df}{dx}(x)(\Delta x) + \frac{1}{2}\frac{df}{dx}(x)(\Delta x)^2$$

Método de Newton: derivación

- Partiendo de la aproximación de Taylor de segundo orden:

$$f(x_{t+1}) \approx f\left(x_t\right) + f'\left(x_t\right)\left(x_{t+1} - x_t\right) + f''\left(x_t\right)\frac{\left(x_{t+1}-x_t\right)^2}{2}$$

Método de Newton: derivación

- Partiendo de la aproximación de Taylor de segundo orden:

$$f(x_{t+1}) \approx f\left(x_t\right) + f'\left(x_t\right)\left(x_{t+1} - x_t\right) + f''\left(x_t\right)\frac{\left(x_{t+1}-x_t\right)^2}{2}$$

- Definimos Δx como $(x_{t+1} - x_t)$, derivamos respecto a Δx e igualamos a 0 para encontrar la expresión del mínimo:

$$0 = \nabla f(x_t) + \nabla^2 f(x_t)\Delta x$$

$$0 = \nabla f(x_t) + \nabla^2 f(x_t)(x_{t+1} - x_t)$$

$$x_{t+1} = x_t - \nabla^2 f(x_t)^{-1}\nabla f(x_t)$$

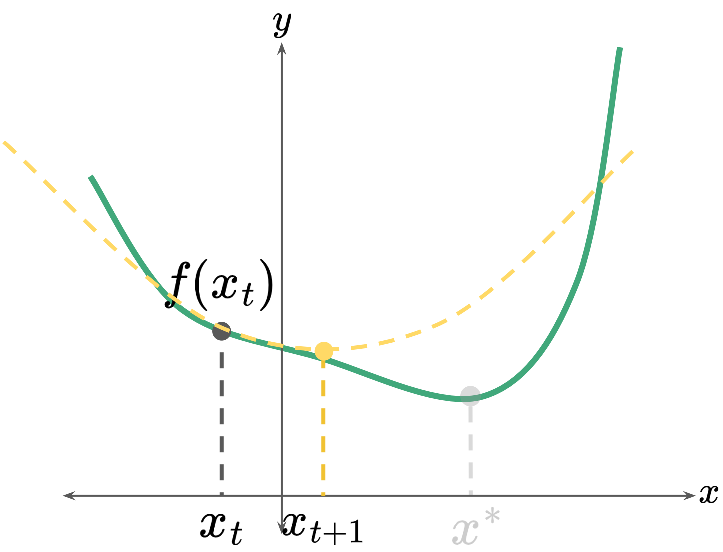
Método de Newton

- En vez de minimizar usando:

$$x_{t+1} = x_t + \alpha f'(x_t)$$

- Vamos a usar:

$$x_{t+1} = x_t - \frac{1}{f''(x_t)} f'(x_t)$$



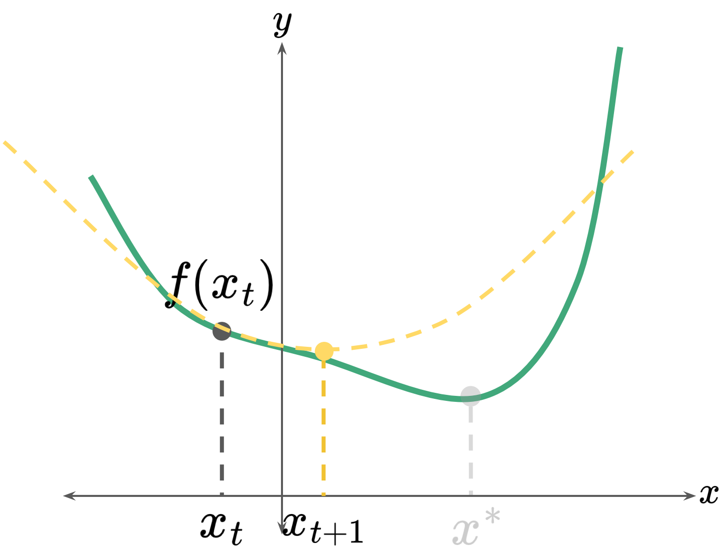
Método de Newton

- En vez de minimizar usando:

$$x_{t+1} = x_t + \alpha f'(x_t)$$

- Vamos a usar (multivariado):

$$x_{t+1} = x_t - \underbrace{\nabla^2 f(x_t)}_{\text{Matriz Hessiana } H}{}^{-1} \nabla f(x_t)$$



Matriz Hessiana

- Una **matriz Hessiana**, es matriz cuadrada de n x n de segundas derivadas parciales de una función f en n variables:

$$H_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Matriz Hessiana - Ejemplo

- Supongamos una función en dos variables x e y:

$$f(x, y) = x^2 + xy + y^2$$

Matriz Hessiana - Ejemplo

- Supongamos una función en dos variables x e y:

$$f(x,y) = x^2 + xy + y^2$$

- Su matriz Hessiana está dada por:

$$H_f = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \longrightarrow H_f = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Algoritmo del Método de Newton

1. Elegir un punto inicial $\mathbf{x}_0$
2. Mientras no haya convergencia (o no se haya alcanzado un límite de iteraciones):
   a. Computar el gradiente en el punto $\nabla f(\mathbf{x}_t)$
   b. Computar la matriz Hessiana en el punto $\mathrm{H}_f(\mathbf{x}_\mathrm{t})$
   c. Resolver $\mathbf{d}_t = \mathrm{H}_f(\mathbf{x}_\mathrm{t})^{-1}\nabla f(\mathbf{x}_t)$ para encontrar la dirección de descenso
   d. Computar el nuevo punto $\mathbf{x}_{t+1} = \mathbf{x}_t - \mathbf{d}_t$
   e. Verificar convergencia y actualizar $\mathbf{x}_t = \mathbf{x}_{t+1}$

Algoritmo del Método de Newton

- Ejercicio: Dado el punto inicial $x_0 = 9.45$ y la función $f(x) = x^2$, escribir un algoritmo en Python capaz de hallar el mínimo de la función $f(x)$ a partir del punto inicial usando el Método de Newton.

  Solución completa: <Notebook>

Consideraciones finales

- La matriz Hessiana debe ser **invertible** para poder ejecutar el algoritmo de Newton.

Consideraciones finales

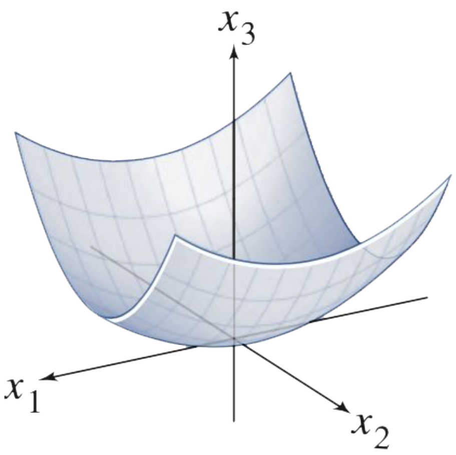
- La matriz Hessiana debe ser **invertible** para poder ejecutar el algoritmo de Newton.
- La función debe ser **dos veces diferenciable** en el punto. Si esto no ocurre, entonces la matriz Hessiana no está definida en ese punto, y por lo tanto tampoco es invertible en ese punto.

Consideraciones finales

- La matriz Hessiana debe ser **invertible** para poder ejecutar el algoritmo de Newton.
- La función debe ser **dos veces diferenciable** en el punto. Si esto no ocurre, entonces la matriz Hessiana no está definida en ese punto, y por lo tanto tampoco es invertible en ese punto.
- Puede no alcanzar la convergencia (ciclando), o detenerse en un "saddle point" en vez de a un mínimo local.

Consideraciones finales

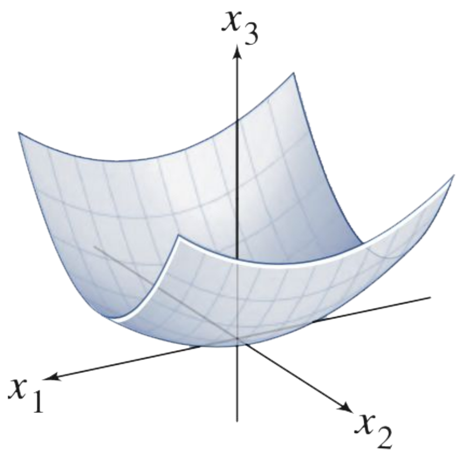
- Las matrices Hessianas son matrices simétricas.
- Son **definidas positivas** sí y sólo si sus autovalores, $\lambda_1, \lambda_2, \ldots, \lambda_n$ son todos mayores a cero.



Definida positiva

Consideraciones finales

- Las matrices Hessianas son matrices simétricas.
- Son **definidas positivas** sí y sólo si sus autovalores, $\lambda_1, \lambda_2, \ldots, \lambda_n$ son todos mayores a cero.
- Si la matriz Hessiana es definida positiva, se garantiza la existencia de un mínimo (función convexa).



Definida positiva