

DESCENSO GRADIENTE

ALAN REYES-FIGUEROA
MODELACIÓN Y SIMULACIÓN

(AULA 10) 12.AGOSTO.2025

Algoritmos para Optimización

Algoritmos para minimización sin restricciones:

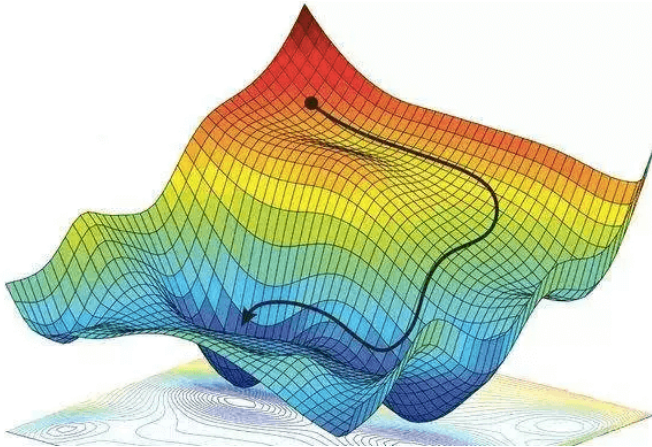
Los algoritmos para minimización sin restricciones son métodos iterativos que encuentran una solución aproximada.

Todos los algoritmos para minimización sin restricciones requieren que el usuario proporcione un punto de partida $\mathbf{x}_0 \in \mathbb{R}^n$. El usuario con conocimiento sobre la función o el conjunto de datos *input* puede estar en una buena posición para elegir $\mathbf{x}_0$ como una estimación razonable de la solución.

De lo contrario, el punto inicial $\mathbf{x}_0$ debe ser elegido por el algoritmo, ya sea mediante un enfoque sistemático o de alguna manera arbitraria (aleatorio dentro de cierto dominio).

- A partir de $\mathbf{x}_0$, se genera una secuencia $\{\mathbf{x}_k\}_{k \geq 0}$ de aproximaciones.
- Para pasar de una iteración $\mathbf{x}_k$ a la siguiente, los algoritmos usan información sobre la función f en $\mathbf{x}_k$, y posiblemente también información de iteraciones anteriores.
- Con esta información, se espera hallar una nueva iteración $\mathbf{x}_{k+1}$, usualmente con la propiedad $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$.

Descenso Gradiente



Descenso Gradiente

- Sin embargo, existen algoritmos no monótonos en los que f no disminuye en cada paso, pero f debería disminuir después de algún número m de iteraciones es decir, $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_{k-j})$ para algún $j \in \{0, 1, \dots, m\}$.

Framework general:

- Elegir $\mathbf{x}_0$,
- Definir cómo actualizar $\mathbf{x}_k$,
- Establecer un criterio de paro.

Descenso Gradiente

¿Cómo actualizar $\mathbf{x}_k$?:

La idea es elegir una dirección $\mathbf{d}_k$ y buscar a lo largo del semirrayo en esta dirección, $\mathbf{x}_{k+1} = \mathbf{x}_k + t\mathbf{d}_k$, para una nueva iteración $\mathbf{x}_{k+1}$ donde la función reduzca su valor.

Definición

Dada $f : \mathbb{R}^n \rightarrow \mathbb{R}$ diferenciable, y un punto $\mathbf{x}_k \in \mathbb{R}^n$, una **dirección de descenso** para f en $\mathbf{x}_k$ es cualquier vector $\mathbf{d} \in \mathbb{R}^n$, tal que

$$f(\mathbf{x}_k + t\mathbf{d}) < f(\mathbf{x}_k), \quad \text{para todo } t \in (0, T). \quad (1)$$

En el contexto de optimización, una dirección de descenso en $\mathbf{x}_k$ mueve el punto $\mathbf{x}_k$ un poco más cerca de un mínimo local.

Estrategia para definir una dirección de descenso:

$\mathbf{d} \in \mathbb{R}^n$ es una dirección de descenso para f en $\mathbf{x}_k$, si y sólo si, $\nabla f(\mathbf{x}_k)^T \mathbf{d} < 0$.

Descenso Gradiente

Algoritmo: (Descenso gradiente, versión naïve)

Inputs: $f : \mathbb{R}^n \rightarrow \mathbb{R}$ función de clase C^1 , $\mathbf{x}_0 \in \mathbb{R}^n$, $\alpha > 0$ tamaño de paso.

Outputs: $\mathbf{x}$ punto crítico de f .

For $k = 0, 1, 2, \dots$ hasta que se cumpla un criterio de paro:

 Compute $\mathbf{d}_k$ a descent direction

 (for example, any $\mathbf{d}_k$ such that $\angle(-\nabla f(\mathbf{x}_k), \mathbf{d}_k) < |\frac{\pi}{2}|$).

 Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha \mathbf{d}_k$.

Return $\mathbf{x}_{k+1}$.

En el caso en que $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$, tenemos

Algoritmo: (Steepest descent, versión naïve)

Inputs: $f : \mathbb{R}^n \rightarrow \mathbb{R}$ función de clase C^1 , $\mathbf{x}_0 \in \mathbb{R}^n$, $\alpha > 0$ tamaño de paso.

Outputs: $\mathbf{x}$ punto crítico de f .

For $k = 0, 1, 2, \dots$ hasta que se cumpla un criterio de paro:

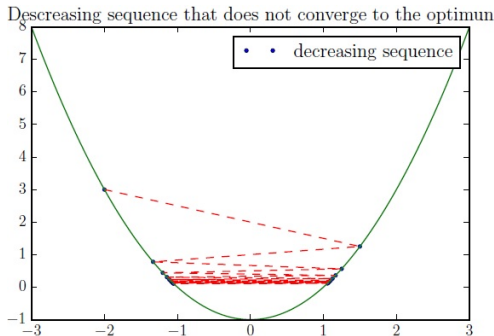
 Set $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \nabla f(\mathbf{x}_k)$.

Return $\mathbf{x}_{k+1}$.

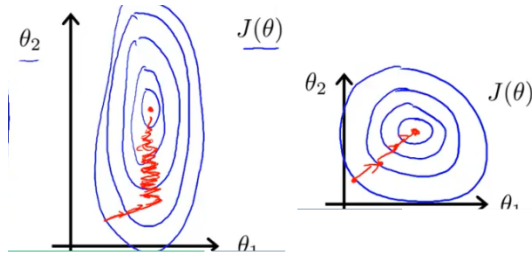
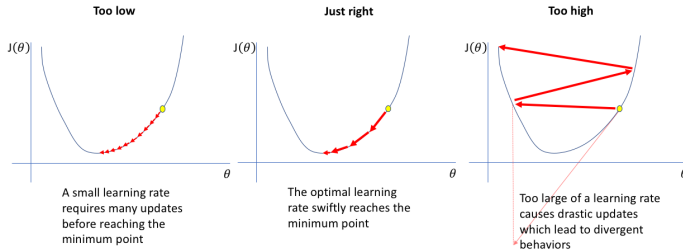
Descenso Gradiente

A la constante $\alpha_k > 0$ se le llama el **tamaño de paso**. Usualmente este tamaño de paso α_k cambia en cada iteración, y se elige en función de la iteración y del punto, α_k . El caso más simple se da al elegir $\alpha_k = \alpha$ constante, como en los algoritmos naïve anteriores.

Elegir el tamaño de paso adecuado es crucial! Si α_k es demasiado grande, es posible que el algoritmo no detecte las regiones donde se encuentra el mínimo local.



Descenso Gradiente



Descenso Gradiente

Ejemplo: Considere la función $f : \mathbb{R} \rightarrow \mathbb{R}$, $f(\mathbf{x}) = \mathbf{x}^2$. f es diferenciable y $\nabla f(\mathbf{x}) = 2\mathbf{x}$.

- Tomando $\alpha = 1$, obtenemos la iteración de descenso máximo

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \nabla f(\mathbf{x}) = \mathbf{x}_k - 2\mathbf{x}_k = -\mathbf{x}_k,$$

la cual es una secuencia alternante $\mathbf{x}_0, -\mathbf{x}_0, \mathbf{x}_0, -\mathbf{x}_0, \dots$, no convergente.

- Tomando $\alpha = \frac{1}{4}$, obtenemos la iteración de descenso máximo

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \frac{1}{4}\nabla f(\mathbf{x}) = \mathbf{x}_k - \frac{2}{4}\mathbf{x}_k = \frac{1}{2}\mathbf{x}_k.$$

Esta es una secuencia geométrica convergente $\mathbf{x}_0, \frac{1}{2}\mathbf{x}_0, \frac{1}{4}\mathbf{x}_0, \frac{1}{8}\mathbf{x}_0, \dots$

Una estrategia empírica muy simple, pero bastante útil, para elegir α es comenzar con un valor pequeño (e.g. $\alpha = 0.1$). Si con este valor de α no se observa convergencia del método de descenso gradiente, se prueban valores usando una escala potencial:

- $\alpha = 0.01, \alpha = 0.001; \alpha = 0.0001, \dots$
- $\alpha = \rho^1 \alpha_0, \alpha = \rho^2 \alpha_0, \alpha = \rho^3 \alpha_0, \dots$, donde $0 < \rho < 1$ (por ejemplo: $\rho = \frac{1}{2}, \frac{1}{4}$ ó $\rho = \frac{1}{10}$)

Descenso Gradiente

Criterios de paro: Existen muchos criterios de paro que pueden usarse para detener los algoritmos de optimización numérica.

- Error absoluto de iteraciones: Se mide el error absoluto entre dos iteraciones consecutivas

$$||\mathbf{x}_{k+1} - \mathbf{x}_k||_{norm} < tol.$$

- Error relativo de iteraciones: Se compara el error relativo entre dos iteraciones consecutivas $\mathbf{x}_k$ y $\mathbf{x}_{k+1}$

$$\frac{||\mathbf{x}_{k+1} - \mathbf{x}_k||_{norm}}{||\mathbf{x}_k||_{norm}} < tol.$$

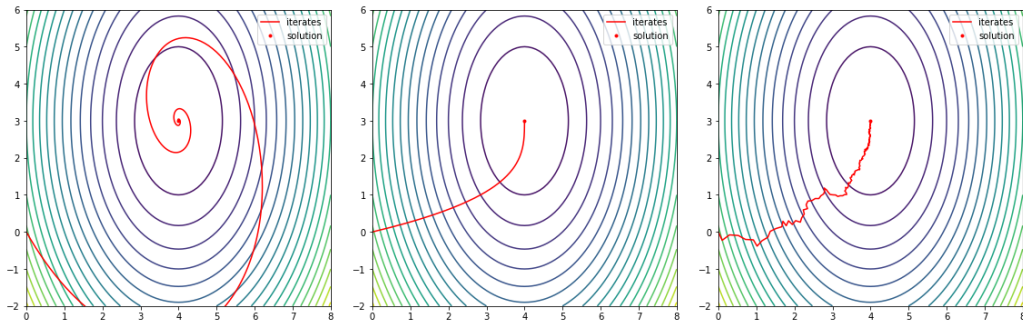
- Error abs/rel del valor de la función: Se mide el error entre dos valores de $f(\mathbf{x}_k)$ en iteraciones consecutivas. Así

$$|f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)| < tol.$$

- Norma del gradiente: En un mínimo local, sabemos que $\nabla f(\mathbf{x}) = \mathbf{0}$. Se busca entonces que las normas del gradiente sean suficientemente pequeñas

$$||\nabla f(\mathbf{x}_k)||_{norm} < tol.$$

Descenso Gradiente



Varios métodos gradiente aplicados a una función cuadrática: (a) Descenso gradiente con dirección de descenso con ángulo constante φ con $\nabla f(\mathbf{x}_k)$; (b) Descenso máximo; (c) Descenso gradiente con dirección de descenso aleatoria.

Descenso de Cauchy

Otra estrategia más adecuada para elegir el tamaño de paso es el llamado **esquema de Cauchy**.

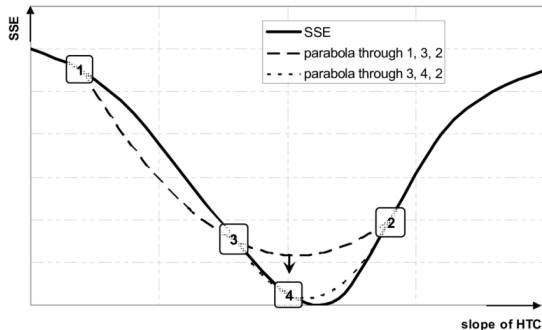
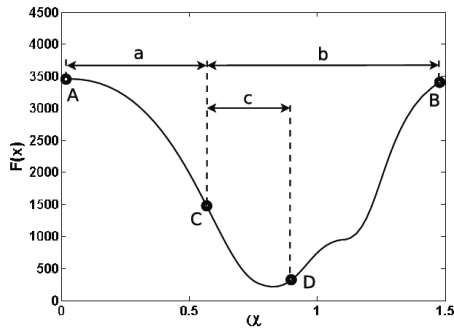
Este consiste en lo siguiente: Dado $\mathbf{x}_k \in \mathbb{R}^n$, luego de elegir la dirección de búsqueda $\mathbf{d}_k$, buscamos cuál es el valor de $\alpha_k > 0$ que minimiza la función f , restringida a la recta $\mathbf{x}_k + t\mathbf{d}_k$, $t > 0$. Esto es, definimos

$$\alpha_k = \operatorname{argmin}_{t \in \mathbb{R}} f(\mathbf{x}_k + t\mathbf{d}_k). \quad (2)$$

Observe que (2) corresponde a un problema de minimización 1-dimensional. Es posible aplicar aquí las técnicas de optimización que aprendieron en Métodos Numéricos I.

- Método de búsqueda de Fibonacci (*Fibonacci search*),
- Método de la razón áurea (*golden ration search*),
- Interpolación parabólica (*quadratic interpolation*),
- Método de Newton,
- ...

Descenso de Cauchy



Optimización 1-dimensional: (a) *Golden-search*, (b) interpolación parabólica.

Ver <https://web2.qatar.cmu.edu/~gdicaro/15382/additional/one-dimensional-search-methods.pdf>

Descenso de Cauchy

Algoritmo: (*Descenso gradiente*, versión esquema de Cauchy)

Inputs: $f : \mathbb{R}^n \rightarrow \mathbb{R}$ función de clase C^1 , $\mathbf{x}_0 \in \mathbb{R}^n$.

Outputs: $\mathbf{x}$ punto crítico de f .

For $k = 0, 1, 2, \dots$ hasta que se cumpla un criterio de paro:

 Define $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$, or any other descent direction.

 Compute α_k such that

$$\alpha_k = \operatorname{argmin}_{t \in \mathbb{R}} f(\mathbf{x}_k + t\mathbf{d}_k),$$

 by any 1-dimensional optimization method,

 Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.

Return $\mathbf{x}_{k+1}$.