USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
FACULTADE DE MATEMÁTICAS

Traballo Fin de Grao

Resolución numérica del problema no lineal de mínimos cuadrados. Aplicaciones a la estimación de parámetros de modelos matemáticos.

Dídac Blanco Morros

Curso Académico

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

GRAO DE MATEMÁTICAS

Traballo Fin de Grao

Resolución numérica del problema no lineal de mínimos cuadrados. Aplicaciones a la estimación de parámetros de modelos matemáticos.

Dídac Blanco Morros

Febrero, 2022

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Trabajo propuesto

Área de Coñecemento:
Título:
Breve descrición do contido
Recomendacións
Outras observacións

Índice

Resumen

Abstract

Introducción

Capítulo 1

Fundamentos de la optimización sin restricciones

Un problema de optimización sin restricciones tiene la forma

$$\min_x f(x) \tag{1.1}$$

donde $x \in \mathbb{R}^n$ y $f : \mathbb{R}^n \to \mathbb{R}$ es continuamente diferenciable, la llamamos **función objetivo**. La dificultad de un problema como este viene de no conocer el comportamiento global de f, normalmente solo disponemos de la evaluación de f en algunos puntos, y a lo mejor de algunas de sus derivadas. El trabajo de los algoritmos de optimización es identificar la solución sin usar demasiado tiempo ni almacenamiento computacional.

Notar que podemos usar la formulación (1.1) para referirnos tanto a los problemas de minimización como de maximización, basta sustituir f por $-f$.

Tenemos dos tipos de solución. Un punto x^* se dice **mínimo global** si $f(x^*) \leq f(x)$ para todo $x \in \mathbb{R}^n$. Como no se suele tener un conocimiento a gran escala de f debido a su coste, la mayoría de algoritmos solo encuentran mínimos locales, lo cual es suficiente para muchos casos prácticos. Un punto x^* se dice **mínimo local** si existe una vecinidad $\mathcal{V}$ de x^* tal que $f(x^*) \leq f(x)$ para todo $x \in \mathcal{V}$.

Aún así, los algoritmos para encontrar mínimos globales se suelen construír a partir de una secuencia de otros algoritmos de optimización local. También podemos aprovechar características fáciles de detectar en la función objetivo, como la convexidad, que nos asegura que un mínimo local será también global.

Todo algoritmo de optimización sin restricciones comienza con un punto de partida, denotado normalmente como x_0. Aunque generalmente el usuario introduce una estimación razonable, el

punto puede ser elegido por el algoritmo, tanto de forma sistemática como aleatoria. El algoritmo itera sobre x_0, creando una sucesión $\{x_k\}_{k=0}^{n}$ la cual termina cuando no pueda continuar o cuando ya se haya acercado razonablemente a la solución. Para decidir como se avanza de un x_k al siguiente, los algoritmos utilizan información sobre $f(x_k)$ o incluso en los puntos anteriores $x_0, x_1, \ldots, x_{k-1}$ con el objetivo de que $f(x_{k+1}) < f(x_k)$. Hablaremos de las dos estrategias fundamentales que se utilizan para avanzar de x_k a x_{k+1}, *búsqueda de línea* y *región de confianza*.

1.1. Búsqueda de línea

En este caso el algoritmo tiene dos tareas a partir de cada iteración, primero elige una *dirección* d_k y tomando el punto de partida busca en esa dirección el nuevo valor. Es decir, dado x_k

$$x_{k+1} = x_k + \alpha_k d_k \tag{1.2}$$

para un d_k elegido previamente, y un *paso* α_k obtenido solucionando otro problema de minimización más simple por ser unidimensional:

$$\min_{\alpha_k > 0} f\left(x_k + \alpha_k d_k\right). \tag{1.3}$$

Si se toma el α_k óptimo se le llama búsqueda de línea *exacta* u *óptima*. Para evitar el gran coste computacional que puede llegar a tomar, lo más común es tomar un α_k que aporte un descenso aceptable, en cuyo caso se le llama búsqueda de línea *inexacta* o *aproximada*. Desde el nuevo punto se busca otra dirección y paso para repetir el proceso. Veamos brevemente cómo se eligen d_k y α_k.

La mayor parte de algoritmos de este tipo necesitan que d_k sea una dirección descendente, esto es, $d_k^T \nabla f_k < 0$, lo cual asegura que en esa dirección se podrá reducir el valor de f. Esta suele tener la forma

$$d_k = -B_k^{-1} \nabla f_k \tag{1.4}$$

con B_k una aproximación de la matriz Hessiana $\nabla^2 f(x_k)$ simétrica y no singular. Según lo que acabamos de decir, necesitamos que B_k sea definida positiva. En las tres corrientes principales se elige un B_k distinto, en el *método del descenso máximo* o *descenso del gradiente*, se usa la matriz identidad I. En el *método de Newton* se usa la matriz exacta, mientras que en los *métodos Quasi-Newton* la matriz Hessiana es aproximada para cada x_k.

En el caso de la elección de α_k, el caso ideal sería encontrar el óptimo en 1.3, pero esto es en general demasiado costoso. Debido a ese coste, se suelen utilizar búsquedas inexactas probando una serie de puntos hasta que alguno cumpla unas condiciones preestablecidas con las que se acepta el paso dado. Estas condiciones son por ejemplo las condiciones *Wolfe* o las condiciones

Goldstein. Esta elección se hace en dos fases, primero un proceso elige un intervalo conteniendo los pasos deseables y una segunda fase donde se va reduciendo el intervalo por técnicas de interpolación o bisección.

1.2. Región de confianza

Esta estrategia enfoca el problema de otro modo, primero se fija una distancia máxima Δ_k para definir la región, que generalmente es de la forma

$$\Omega_k = \{x : \|x - x_k\| \leq \Delta_k\} \tag{1.5}$$

y luego ya se busca la dirección y paso. A partir de la información conocida de f, para cada x_k se modela una función m_k que se comporte de manera similar a f cerca de este punto. Generalmente se utiliza el modelo cuadrático de la forma

$$m_k := q^{(k)}(p) = f(x_k) + g_k^T p + \frac{1}{2} p^T G_k p, \tag{1.6}$$

donde $g_k = \nabla f(x_k)$ y $G_k = \nabla^2 f(x_k)$. Este modelo cuadrático es el utilizado en los métodos de búsqueda de línea para determinar la dirección de búsqueda, mientras que en este caso lo usamos para tener una representación adecuada de la función objetivo y así elegir el mínimo dentro de esta región. Este método nos evita el problema de que la Hessiana no sea definida positiva. En cada iteración, una vez elegido Δ_k se resuelve el siguiente problema:

$$\begin{aligned} \min_p \quad & q^{(k)}(p) = f(x_k) + g_k^T p + \frac{1}{2} p^T B_k p \\ \text{s.a.} \quad & \|p\| \leq \Delta_k. \end{aligned} \tag{1.7}$$

Notamos que en el modelo se escribe B_k en lugar de G_k, pues no siempre se usa esta última. Debido al coste computacional, como vimos en la elección de la dirección de búsqueda, a veces se prefiere aproximar de alguna manera más o menos eficiente, e incluso puede ser aceptable tomar la matriz 0.

También se puede elegir qué norma define la región de confianza, cambiando así la forma de esta y ofreciendo distintos resultados, aunque generalmente se utiliza la bola definida por $\|p\|_2 \leq \Delta_k$.

La efectividad de cada iteración depende de la elección del radio Δ_k, es por ello que puede que la primera elección de este no sea la definitiva. Es decir, se toma un radio a raíz de la información que se tenga, esta puede incluír la de pasos anteriores, y luego se decide si este radio nos da un resultado aceptable. Un radio demasiado pequeño nos puede hacer perder la oportunidad de ser mucho más rápidos, pero un paso demasiado grande, el mínimo de la función modelo m_k puede

estar lejos del mínimo de la función objetivo. Este último caso es el que se comprueba y se decide si reducir la región de confianza.

Una vez tomado el radio, encontrar el mínimo es directo en el caso de que B_k sea definida positiva, basta tomar $p_k^B = -B_k^{-1} g_k$. En caso contrario tampoco supone una tarea muy costosa ya que sólo se necesita una solución aproximada para garantizar la convergencia.

Capítulo 2

Mínimos Cuadrados

El problema de mínimos cuadrados surge de la necesidad de ajustar modelos que nos permitan predecir ciertos comportamientos en una amplia variedad de campos. Dados unos datos $(t_1, y_1), (t_2, y_2), \cdots, (t_m, y_m)$, queremos ajustar una función $\phi(t, x)$ de forma que se minimicen los residuos $r_i(x) = \phi(t_i, x) - y_i$ para $i = 1, \cdots, m$

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} r(x)^T r(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x), \quad m \geq n, \tag{2.1}$$

donde $r(x) = (r_1(x), r_2(x), \ldots, r_m(x))^T$, con $r_i : \mathbb{R}^n \to \mathbb{R}$ funciones continuamente diferenciables.

Veamos las propiedades de este modelo concreto de optimización sin restricciones y cómo se pueden aprovechar para formular algoritmos eficientes y robustos. Sea $J(x)$ la matriz Jacobiana de $r(x)$,

$$J(x) = \begin{bmatrix} \nabla r_1(x)^T \\ \nabla r_2(x)^T \\ \vdots \\ \nabla r_m(x)^T \end{bmatrix} = \begin{bmatrix} \frac{\partial r_1}{\partial x_1}(x) & \frac{\partial r_1}{\partial x_2}(x) & \cdots & \frac{\partial r_1}{\partial x_n}(x) \\ \frac{\partial r_2}{\partial x_1}(x) & \frac{\partial r_2}{\partial x_2}(x) & \cdots & \frac{\partial r_2}{\partial x_n}(x) \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial r_m}{\partial x_1}(x) & \frac{\partial r_m}{\partial x_2}(x) & \cdots & \frac{\partial r_m}{\partial x_n}(x) \end{bmatrix}. \tag{2.2}$$

El gradiente y la Hessiana de f se pueden expresar como sigue:

$$g(x) = \nabla f(x) = \sum_{i=1}^{m} r_i(x) \nabla r_i(x) = J(x)^T r(x) \tag{2.3}$$

$$\begin{aligned} G(x) = \nabla^2 f(x) &= \sum_{i=1}^{m} \nabla r_i(x) \nabla r_i(x)^T + \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x) \\ &= J(x)^T J(x) + S(x). \end{aligned} \tag{2.4}$$

Si nos fijamos en la formulación de la matriz Hessiana, el cálculo del primer termino es directo gracias a que ya obtenemos $J(x)$ para calcular el gradiente (2.3), así que el coste se reduce al

segundo término, que hemos denotado $S(x)$. Veamos la expresión del modelo cuadrático de $f(x)$ utilizando (2.1), (2.3) y (2.4):

$$\begin{aligned} q^{(k)}(x) &= f(x_k) + g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T G_k(x - x_k), \\ &= \frac{1}{2} r(x_k)^T r(x_k) + (J(x_k)^T r(x))^T (x - x_k) + \end{aligned} \tag{2.5}$$

2.1. Problema Lineal

El primer caso más sencillo es si $\phi(t, x)$ es una función lineal, en cuyo caso los residuos $r_i(x)$ también serán lineales. Por ser ϕ lineal, se puede representar como Jx, con J una matriz $m \times n$. Realizaremos un estudio del caso lineal para tener un conocimiento de como se enfocan estos problemas, que nos servirá para entender mejor el caso no lineal. Si escribimos el vector residuo como $r(x) = Jx - y$, la función objetivo nos queda de la forma

$$f(x) = \frac{1}{2}\|Jx - y\|^2. \tag{2.6}$$

En consecuencia, tomando como referencia (2.3) y (2.4) y teniendo en cuenta que en este caso particular $\nabla^2 r_i = 0$, nos queda

$$\nabla f(x) = J^T(Jx - y), \qquad \nabla^2 f(x) = J^T J. \tag{2.7}$$

Por ser f convexa en este caso (2.6) (y por los resultados previos), dado un punto x^* tal que $\nabla f(x^*) = 0$, este será mínimo global. Por tanto, x^* satisface el siguiente sistema lineal:

$$J^T J x^* = J^T y. \tag{2.8}$$

Veamos cómo se enfoca elementalmente este sistema de ecuaciones, conocidas como *ecuaciones normales* de (2.6). Lo más común en este caso para resolver numéricamente es usar distintos tipos de factorización sobre la matriz $J^T J$ o sobre J, para luego resolver con sustituciones triangulares. El primer algoritmo que se plantea es a partir de la **factorización de Cholesky**, comenzando por computar la matriz de coeficientes $J^T J$ y el lado derecho $J^T y$. Después se computa la factorización de Cholesky

$$J^T J = \bar{R}^T \bar{R}. \tag{2.9}$$

Para que esta exista, necesitamos que $m \geq n$ y que J sea de rango n, lo que permite que $J^T J$ sea simétrica y definida positiva. Se termina realizando las dos sustituciones triangulares con los factores de Cholesky para encontrar x^*. La principal desventaja de este método es que el condicionamiento de $J^T J$ es el cuadrado del condicionamiento de J, y esto puede llevar a

errores de aproximación. Además, si J está mal condicionada, ni quiera se puede llevar a cabo la factorización.

Una segunda posibilidad es basarse en la **factorización QR**, que evita el problema de depender del cuadrado del condicionamiento de J, ya que aplicaremos la factorización directamente a J. Se aprovecha que la norma Euclídea no se ve afectada por trasformaciones ortogonales para partir de la igualdad

$$\|Jx - y\| = \|Q^T(Jx - y)\|, \tag{2.10}$$

siendo Q una matriz ortogonal $m \times m$. Factorizando con una matriz pivote Π, la solución es

$$x^* = \Pi R^{-1} Q_1^T y. \tag{2.11}$$

Donde R es una matriz $n \times n$ triangular superior con elementos positivos en la diagonal y Q_1 son las primeras n columnas de Q, ambos producto de la factorización QR.

En este caso, el error relativo es proporcional al condicionamiento de J y no de su cuadrado como antes. Aún así, hay situaciones en las que necesitamos asegurar que la obtención sea de algún modo más robusta, incluso necesitamos más información acerca de la sensibilidad de la solución a perturbaciones en J o en y. Para este caso particular nos basaremos en la **descomposición de valores singulares (DVS)**.

En este caso la resolución se enfoca de forma similar a la anterior. Partimos de la descomposición $J = USV^T$, con U matriz $m \times m$, V matriz $n \times n$, ambas ortogonales y S matriz $n \times n$ de elementos diagonales $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$. Aprovechamos estas propiedades para obtener

$$x^* = VS^{-1}U_1^T y. \tag{2.12}$$

Denotando por $u_i \in \mathbb{R}^m$ y $v_i \in \mathbb{R}^n$ las columnas de U y V respectivamente, escribimos

$$x^* = \sum_{i=1}^{n} \frac{i_i^T y}{\sigma_i} v_i. \tag{2.13}$$

Fórmula de donde obtenemos información útil como la sensibilidad al aproximar x^*.

Las 3 opciones son buenas según las condiciones en las que nos encontremos. La resolución basada en Cholesky es útil cuando $m \gg n$ y es práctico trabajar almacenando $J^T J$ en lugar de J, siempre y cuando J sea de rango completo y bien condicionada. Si esto último no se cumple, la factorización QR es un enfoque más equilibrado, mientras que DVS es el más costoso a cambio de ser el más fiable.

Por último, mencionar que existen métodos para problemas muy grandes, en los que se usan técnicas iterativas como el método de gradientes conjugados para resolver el sistema.

Anexo I

Título del Anexo I

Anexo II

Título del Anexo II

Bibliografía

[1] Nocedal, J., & Wright, S. (2006). *Numerical Optimization* (2nd ed.). Springer.

[2] Sun, W., & Yuan, Y.-X. (2006). *Optimization theory and methods: Nonlinear programming* (2006th ed.). Springer.