Optimización unidimensional

Lección 06

Dr. Pablo Alvarado Moya

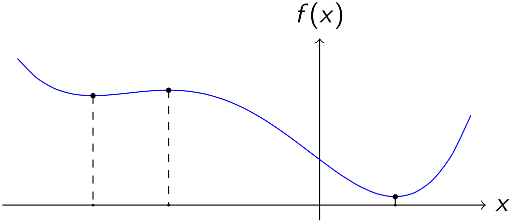
CE3102 Análisis Numérico para Ingeniería
Área de Ingeniería en Computadores
Tecnológico de Costa Rica

II Semestre 2017

Contenido

Introducción

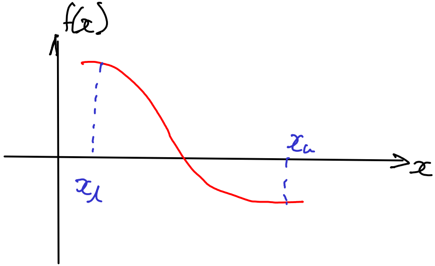
- Optimización es el proceso de encontrar extremos de una función.



- Se distinguen extremos **locales** y **globales**.
- Extremos son mínimos o máximos.
- Optimizadores unidimensionales son cerrados o abiertos.

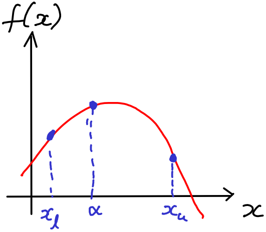
Encerrando una raíz

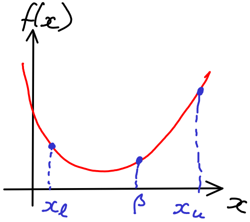
- Buscando raíces: raíz entre puntos con signos distintos



Encerrando un extremo

- Encerrar extremo: requiere tres puntos





- Si funciones son suaves, extremo encerrado en $[x_l, x_u]$ si
  - Máximo: $f(\alpha) > f(x_l)$ y $f(\alpha) > f(x_u)$
  - Mínimo: $f(\beta) < f(x_l)$ y $f(\beta) < f(x_u)$

Rutina de delimitación

Bracketing

- Siguientes algoritmos necesitan un estado inicial (tres puntos)
- Algoritmos de delimitación usualmente toman dos valores x
- Con ellos calculan un tercer valor tal que los "bordes" $[x_l, x_u]$ encierran al extremo
- El tercer punto x cumple que la $f(x)$ es mayor (o menor) que $f(x_l)$ y $f(x_u)$.

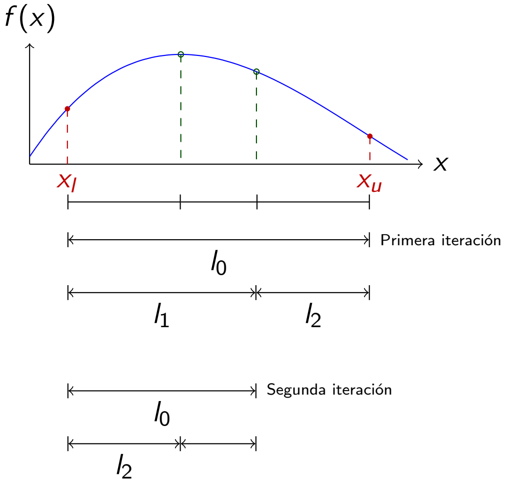
Métodos

- Método cerrado: sección dorada
- Método cerrado: interpolación parabólica
- Método abierto: método de Newton
- Método híbrido: método de Brent

Sección dorada

- Similar a método de bisección
- x_l: límite inferior
- x_u: límite superior
- $[x_l, x_u]$ encierra extremo
- Se necesitan otros dos puntos dentro de intervalo
- Algoritmo selecciona si extremo esta en primeros o segundos 3 puntos
- Elección de puntos intermedios es crítica para eficiencia: Meta es minimizar número de evaluaciones de función

Búsqueda de secciones doradas



Condiciones:

$$l_0 = l_1 + l_2$$

$$\frac{l_1}{l_0} = \frac{l_2}{l_1}$$

y combinando

$$\frac{l_1}{l_1 + l_2} = \frac{l_2}{l_1}$$

Razón áurea

Con $R = l_2/l_1$

$$\begin{aligned}
\frac{l_1}{l_1 + l_2} &= \frac{l_2}{l_1} \\
\frac{1}{1 + \frac{l_2}{l_1}} &= \frac{l_2}{l_1} \\
\frac{1}{1 + R} &= R \\
R^2 + R - 1 &= 0
\end{aligned}$$

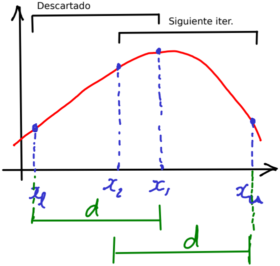
que tiene raíz positiva

$$R = \frac{-1 + \sqrt{1 - 4(-1)}}{2} = \frac{\sqrt{5} - 1}{2} \approx 0,61803$$

Método numérico

- Ejemplo aquí para máximo (similar para mínimo)
- Tome intervalo inicial $[x_l, x_u]$
- Seleccione dos puntos internos con razón dorada:

$$d = R(x_u - x_l)$$
$$x_1 = x_l + d$$
$$x_2 = x_u - d$$



- Si $f(x_1) > f(x_2)$ entonces máximo en $[x_2, x_u]$
- Si $f(x_2) > f(x_1)$ entonces máximo en $[x_l, x_1]$
- Nótese que por razón dorada $f(x_1)$ o $f(x_2)$ ¡se reutiliza en siguiente iteración!

Siguiente iteración

- Siguiente iteración debe reasignar puntos:
  - Si se sigue con triada izquierda $(f(x_2) > f(x_1))$

$$
\begin{aligned}
x_u &\leftarrow x_1 & \qquad & x_l \text{ se mantiene} \\
x_1 &\leftarrow x_2 & & \text{¡se reutiliza } x_2\text{!} \\
x_2 &\leftarrow x_u - R(x_u - x_l)
\end{aligned}
$$

  - Si se sigue con triada derecha $(f(x_1) > f(x_2))$

$$
\begin{aligned}
x_l &\leftarrow x_2 & \qquad & x_u \text{ se mantiene} \\
x_2 &\leftarrow x_1 & & \text{¡se reutiliza } x_1\text{!} \\
x_1 &\leftarrow x_l + R(x_u - x_l)
\end{aligned}
$$

- En cada iteración intervalo con extremo se reduce al $61,8\,\%$
- En cada iteración de bisección hay reducción al $50\,\%$

Estimación del error

- Peor caso en una iteración: máximo en sección de extremos $x_{l,u}$:

$$\Delta x = (1 - R)(x_u - x_l)$$

- Error se aproxima con tamaño de intervalo, normalizado por el óptimo estimado hasta el momento:

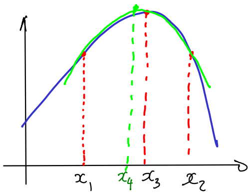
$$\epsilon_a = (1 - R)\left|\frac{x_u - x_l}{x_{opt}}\right| 100\,\%$$

Reducción de evaluaciones de función

- Evaluación dorada es parte de otros procesos, por lo que función original será evaluada considerable número de veces.
- Evaluación de funciones a optimizar toma mucho tiempo (resultados de simulaciones, soluciones de sistemas, etc.)

Interpolación parabólica

- Idea: con los 3 puntos disponibles interpolar parábola
- Usar máximo de esa parábola como siguiente aproximación



- Resolviendo resulta en (x_1, x_2, x_3 triada inicial):

$$x_4 = \frac{1}{2} \cdot \frac{f(x_1)(x_2^2 - x_3^2) + f(x_2)(x_3^2 - x_1^2) + f(x_2)(x_1^2 - x_2^2)}{f(x_1)(x_2 - x_3) + f(x_2)(x_3 - x_1) + f(x_3)(x_1 - x_2)}$$

- Dos estrategias:
  - Usar los puntos en orden secuencial (los últimos 3 siempre)
  - Usar los dos más cercanos al nuevo punto

Método de Newton

- Aproximación de derivada de función con series de Taylor:

$$f'(x_{i+1}) = f'(x_i) + (x_{i+1} - x_i)f''(x_i) + \ldots$$

- Si x_{i+1} es extremo, entonces $f'(x_{i+1}) = 0$ y por tanto

$$\begin{aligned} f'(x_i) + (x_{i+1} - x_i)f''(x_i) &= 0 \\ (x_{i+1} - x_i) &= -\frac{f'(x_i)}{f''(x_i)} \\ x_{i+1} &= x_i - \frac{f'(x_i)}{f''(x_i)} \end{aligned}$$

- Idéntico a Newton-Raphson, pero aplicado a $f'(x)$
- Puede diverger.
- Debe verificarse signo de segunda derivada para asegurar extremo correcto

Método de Brent

- Idea: combinar sección dorada (segura pero lenta) con interpolación parabólica (rápida pero inestable)
- Si función se comporta mal: use sección dorada
- Si función es suave: use interpolación parabólica
- Combinación de métodos debe:
  1. Evitar evaluaciones innecesarias cuando se cambia de método
  2. Debe tener cuidado en fase final, con error estimado (cerca de precisión numérica)
  3. Debe tener detección robusta de tipo de función para seleccionar el método apropiado

Método de Brent (2) (1)

- Método necesita
  - Intervalo inicial $[x_l, x_u]$.
  - x mejor extremo hasta el momento
  - w segundo mejor extremo hasta el momento
  - v valor anterior de w
  - u último valor evaluado de la función
- Se intenta interpolación parabólica con $\{x, v, w\}$
- Paso se acepta si
  1. extremo cae en intervalo $[x_l, x_u]$
  2. paso desde mejor extremo actual x es menos de $\frac{1}{2}$ de paso trasanterior (No se usa paso anterior para no castigar errores aislados)
- Método usa un rango de tolerancia, bajo el cual el algoritmo para o ajusta valores de x para no re-evaluar la función

Resumen

Este documento ha sido elaborado con software libre incluyendo LATEX, Beamer, GNUPlot, GNU/Octave, XFig, Inkscape, LTI-Lib-2, GNU-Make y Subversion en GNU/Linux