Métodos matemáticos para computadora Visión, Robótica y Gráficos

Notas del curso para CS 205A, otoño de 2013

justin salomon
Departamento de Ciencias de la Computación
Universidad Stanford

Contenido

Parte I

Preliminares

capitulo 0

Repaso de Matemáticas

En este capítulo revisaremos las nociones relevantes del álgebra lineal y el cálculo multivariable que figurarán en nuestra discusión de las técnicas computacionales. Tiene la intención de ser una revisión del material de fondo con un sesgo hacia las ideas e interpretaciones comúnmente encontradas en la práctica; el capítulo se puede omitir de forma segura o utilizar como referencia para los estudiantes con una formación más sólida en matemáticas.

0.1 Preliminares: números y conjuntos

En lugar de considerar discusiones algebraicas (ya veces filosóficas) como "¿Qué es un número?", Confiaremos en la intuición y el sentido común matemático para definir algunos conjuntos: • Los números naturales N

= {1, 2, 3 , . . .}

• Los enteros Z = {. . . , −2, −1, 0, 1, 2, . . .}

• Los números racionales Q = {a/b : a, b Z} • Los [1]

números reales R que abarcan Q así como números irracionales como π y $\sqrt{2}$ • Los números

complejos C = {a + bi : a, b R }, donde pensamos que i satisface i = $\sqrt{-1}$.

Vale la pena reconocer que nuestra definición de R está lejos de ser rigurosa. La construcción de los números reales puede ser un tema importante para los practicantes de técnicas criptográficas que utilizan sistemas numéricos alternativos, pero estas complejidades son irrelevantes para la discusión que nos ocupa.

Al igual que con cualquier otro conjunto, N, Z, Q, R y C se pueden manipular mediante operaciones genéricas para generar nuevos conjuntos de números. En particular, recuerde que podemos definir el "producto euclidiano" de dos conjuntos A y B como

$$A \times B = \{(a, b) : a \quad A \text{ y } b \quad B\}.$$

Podemos tomar potencias de conjuntos escribiendo

$$A^n = \underbrace{A \times A \times \cdots \times A}_{n \text{ veces}}.$$

1Esta es la primera de muchas veces que usaremos la notación {A : B}; las llaves deben indicar un conjunto y los dos puntos se pueden leer como "tal que". Por ejemplo, la definición de Q puede leerse como “el conjunto de fracciones a/b tales que a y b son números enteros”. Como segundo ejemplo, podríamos escribir N = {n Z : n > 0}.

Esta construcción produce lo que se convertirá en nuestro conjunto de números favorito en los próximos capítulos:

$$R^{\text{todo}} = \{(a1, a2, \ldots, an) : ai \quad R \text{ para todo } i\}$$

0.2 Espacios vectoriales

Los cursos introductorios de álgebra lineal fácilmente podrían titularse “Introducción a los espacios vectoriales de dimensión finita”. Aunque la definición de un espacio vectorial puede parecer abstracta, encontraremos muchas aplicaciones concretas que satisfacen los aspectos formales y, por lo tanto, pueden beneficiarse de la maquinaria que desarrollaremos.

0.2.1 Definición de espacios vectoriales

Comenzamos definiendo un espacio vectorial y brindando una serie de ejemplos:

Definición 0.1 (Espacio vectorial). Un espacio vectorial es un conjunto V que se cierra bajo la multiplicación y suma escalar.

Para nuestros propósitos, un escalar es un número en R, y las operaciones de suma y multiplicación satisfacen los axiomas habituales (conmutatividad, asociatividad, etc.). Por lo general, es sencillo detectar espacios vectoriales en la naturaleza, incluidos los siguientes ejemplos:

Ejemplo 0.1 (Rn como espacio vectorial). El ejemplo más común de un espacio vectorial es Rn . Aquí, la suma y la multiplicación escalar ocurren componente por componente:

$$(1, 2) + (-3, 4) = (1 - 3, 2 + 4) = (-2, 6)$$

$$10 \cdot (-1, 1) = (10 \cdot -1, 10 \cdot 1) = (-10, 10)$$

Ejemplo 0.2 (Polinomios). Un segundo ejemplo importante de un espacio vectorial es el "anillo" de polinomios con entradas de números reales, denominado R[x]. Un polinomio p R[x] es una función p : R → R de la forma[2]

$$p(x) = \sum_k \quad k \; akx \;\cdot$$

La suma y la multiplicación escalar se llevan a cabo de la forma habitual, por ejemplo, si $p(x) = x^2 + 2x - 1$ y $q(x) = x + 3x^3$, entonces $3p(x) + 5q(x) = 5x^3 + 6x - 3$, que es otro polinomio. Aparte, para ejemplos futuros, tenga en cuenta que 2 (x^3 siendo polinomios incluso funciones como $p(x) = (x - 1)(x + 1) + x$ aunque no están escritos explícitamente en el formulario anterior.

Los elementos v V de un espacio vectorial V se denominan vectores, y una suma ponderada de la forma ∑i aivi , donde ai R y vi V, se conoce como combinación lineal de las vi . En nuestro segundo ejemplo, los “vectores” son funciones, aunque normalmente no usamos este lenguaje para hablar de R[x]. Una forma de vincular estos dos puntos de vista sería identificar el polinomio ∑k akx (a0, a1, a2, · · ·); recuerda que los polinomios tienen un número finito de términos, por lo que la secuencia eventualmente terminará en una cadena de ceros.

2La notación f : A → B significa que f es una función que toma como entrada un elemento del conjunto A y genera un elemento del conjunto B. Por ejemplo, f : R → Z toma como entrada un número real en R y genera un entero Z , como podría ser el caso de f(x) = x, la función de “redondeo hacia abajo”.

0.2.2 Span, independencia lineal y bases

Supongamos que comenzamos con los vectores v1, . . . ,vk V para el espacio vectorial V. Por la Definición 0.1, tenemos dos formas de comenzar con estos vectores y construir nuevos elementos de V: suma y multiplicación escalar. La idea de span es que describe todos los vectores a los que puede llegar a través de estas dos operaciones:

Definición 0.2 (Span). El lapso de un conjunto S V de vectores es el conjunto

$$\text{span } S \equiv \{a_1v_1 + \cdots + a_kv_k : k \geq 0,\ v_i \in V \text{ para todo } i, \text{ y } a_i \in \mathbb{R} \text{ para todo } i\}.$$

Observe que el intervalo S es un subespacio de V, es decir, un subconjunto de V que es en sí mismo un espacio vectorial. Podemos proporcionar algunos ejemplos:

Ejemplo 0.3 (Mixología). El "pozo" típico de una coctelería contiene al menos cuatro ingredientes a disposición del cantinero: vodka, tequila, jugo de naranja y granadina. Suponiendo que tenemos este pozo simple, podemos representar las bebidas como puntos en R4 con una ranura para cada ingrediente. Por ejemplo, un "amanecer de tequila" típico se puede representar usando el punto (0, 1.5, 6, 0.75), que representa cantidades de vodka, tequila, jugo de naranja y granadina (en onzas), resp.

El conjunto de bebidas que se pueden hacer con el pozo típico está contenido en

$$\text{intervalo } \{(1, 0, 0, 0), (0, 1, 0, 0), (0, 0, 1, 0), (0, 0, 0, 1)\},$$

es decir, todas las combinaciones de los cuatro ingredientes básicos. Sin embargo, un cantinero que busca ahorrar tiempo podría notar que muchas bebidas tienen la misma proporción de jugo de naranja a granadina y mezclar las botellas. El nuevo pozo simplificado puede ser más fácil de verter, pero puede hacer fundamentalmente menos bebidas:

$$\text{intervalo } \{(1, 0, 0, 0), (0, 1, 0, 0), (0, 0, 6, 0{,}75)\}$$

Ejemplo 0.4 (Polinomios). Definir el $p_k(x) \equiv x^k$. Entonces, es fácil ver que

$$\mathbb{R}[x] = \text{intervalo } \{p_k : k \geq 0\}.$$

Asegúrese de entender la notación lo suficientemente bien como para ver por qué sucede esto.

Agregar otro elemento a un conjunto de vectores no siempre aumenta el tamaño de su intervalo. Para ejemplo, en R2 es claramente el caso que

$$\text{intervalo } \{(1, 0),(0, 1)\} = \text{intervalo } \{(1, 0),(0, 1),(1, 1)\}.$$

En este caso, decimos que el conjunto {(1, 0),(0, 1),(1, 1)} es linealmente dependiente:

Definición 0.3 (Dependencia lineal). Proporcionamos tres definiciones equivalentes. Un conjunto S V de vectores es linealmente dependiente si:

1. Uno de los elementos de S se puede escribir como una combinación lineal de los otros elementos, o S contiene cero.
2. Existe una combinación lineal no vacía de elementos vk S que da $\sum_{k=1}^{\text{metro}} c_kv_k = 0$ donde $\sum c_k = 0$ para todo k.
3. Existe v S tal que span S = span S\{v}. Es decir, podemos quitar un vector de S sin afectando su amplitud.

Si S no es linealmente dependiente, entonces decimos que es linealmente independiente.

Proporcionar pruebas o evidencia informal de que cada definición es equivalente a sus contrapartes (en forma de "si y solo si") es un ejercicio que vale la pena para los estudiantes menos cómodos con la notación y las matemáticas abstractas.

El concepto de dependencia lineal conduce a una idea de “redundancia” en un conjunto de vectores. En este sentido, es natural preguntarse qué tan grande es el conjunto que podemos elegir antes de agregar otro vector que posiblemente no aumente el lapso. En particular, supongamos que tenemos un conjunto linealmente independiente S V, y ahora elegimos un vector adicional v V. Sumar v a S conduce a uno de dos resultados posibles:

1. El lapso de S {v} es mayor que el lapso de S.
2. Sumar v a S no tiene efecto en el lapso.

La dimensión de V no es más que el número máximo de veces que podemos obtener el resultado 1, sumar v a S y repetir.

Definición 0.4 (Dimensión y base). La dimensión de V es el tamaño máximo |S| de un conjunto linealmente independiente S V tal que genere S = V. Cualquier conjunto S que satisfaga esta propiedad se denomina base de V.

Ejemplo 0.5 (Rn). La base estándar para Rn es el conjunto de vectores de la forma

$$e_k \equiv (0, \ldots, \underbrace{0}_{\text{ranuras } k-1}, \underbrace{1}_{}, \underbrace{0, \ldots, 0}_{\text{tragamonedas } n-k}).$$

Es decir, ek tiene todos ceros excepto uno en la ranura k-ésima. Está claro que estos vectores son linealmente independientes y forman una base; por ejemplo en R3 cualquier vector (a, b, c) se puede escribir como ae1 + be2 + ce3.
Por tanto, la dimensión de Rn es n, como cabría esperar.

Ejemplo 0.6 (Polinomios). Es claro que el conjunto {1, x, x , . . .} es un conjunto linealmente independiente de polinomios que abarca R[x]. Observe que este conjunto es infinitamente grande y, por lo tanto, la dimensión de R[x] es ∞.

0.2.3 Nuestro enfoque: Rn

De particular importancia para nuestros propósitos es el espacio vectorial Rn espacio , el llamado Eu n-dimensional clidiano. Esto no es más que el conjunto de ejes de coordenadas que se encuentran en las clases de matemáticas de la escuela secundaria:

- R1 ≡ R es la recta numérica
- R2 es el plano bidimensional con coordenadas (x, y) • R3 representa el espacio tridimensional con coordenadas (x, y, z)

Casi todos los métodos en este curso tratarán con transformaciones y funciones en Rn .

Por conveniencia, generalmente escribimos vectores en Rn en "forma de columna", como sigue

$$(a_1, \ldots, a_n) \equiv \begin{matrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{matrix}$$

Esta notación incluirá vectores como casos especiales de matrices que se analizan a continuación.

A diferencia de algunos espacios vectoriales, Rn no solo tiene una estructura de espacio vectorial, sino también una construcción adicional que marca la diferencia: el producto escalar.

Definición 0.5 (Producto escalar). El producto escalar de dos vectores a = (a1, . . . , an) yb = (b1, . . . , bn) en Rn viene dado por

$$a \cdot b = \sum_{k=1}^{n} akbk .$$

Ejemplo 0.7 (R2). El producto escalar de (1, 2) y (−2, 6) es 1 · −2 + 2 · 6 = −2 + 12 = 10.

El producto escalar es un ejemplo de métrica, y su existencia le da una noción de geometría a Rn .
Por ejemplo, podemos usar el teorema de Pitágoras para definir la norma o longitud de un vectora como la raíz cuadrada

$$\sqrt{2\,2 + \cdots + un\,a2 \equiv un\,1} = \sqrt{a \cdot a.}$$

Entonces, la distancia entre dos puntosa,b Rn es simplemente b −a2.

Los productos de puntos dan no solo nociones de longitudes y distancias sino también de ángulos. Recuerde la siguiente identidad de trigonometría fora,b R3 :

$$a \cdot b = ab \cos \theta$$

donde θ es el ángulo entre ayb. Sin embargo, para n ≥ 4, la noción de “ángulo” es mucho más difícil de visualizar para Rn . Podríamos definir el ángulo θ entre a yb como el valor θ dado por

$$\theta \equiv \arccos \frac{a \cdot b}{abdominales} .$$

¡Debemos hacer nuestra tarea antes de hacer tal definición! En particular, recuerde que la salida de coseno pone valores en el intervalo [−1, 1], por lo que debemos comprobar que la entrada al arcocoseno (también anotado como cos−1) está en este intervalo; afortunadamente, la conocida desigualdad de Cauchy-Schwarz a ·b ≤ ab garantiza exactamente esta propiedad.

Cuando a = cb para algún c R, tenemos θ = arccos 1 = 0, como era de esperar: el ángulo entre vectores paralelos es cero. ¿Qué significa que los vectores sean perpendiculares? Sustituyamos θ = 90◦ . Entonces nosotros tenemos

$$0 = \cos 90^\circ$$

$$= \frac{a \cdot b}{abdominales}$$

Multiplicar ambos lados por ab motiva la definición:

Definición 0.6 (Ortogonalidad). Dos vectores son perpendiculares u ortogonales cuando a ·b = 0.

Esta definición es algo sorprendente desde un punto de vista geométrico. En particular, hemos logrado definir lo que significa ser perpendicular sin ningún uso explícito de ángulos. Esta construcción facilitará la resolución de ciertos problemas para los cuales la no linealidad del seno y el coseno podría haber causado dolor de cabeza en configuraciones más simples.

Aparte 0.1. Hay muchas preguntas teóricas para reflexionar aquí, algunas de las cuales abordaremos en capítulos futuros cuando estén más motivados:

- ¿Todos los espacios vectoriales admiten productos punto o estructuras similares?
- ¿Todos los espacios vectoriales de dimensión finita admiten productos punto?
- ¿Cuál podría ser un producto escalar razonable entre elementos de R[x]?

Los estudiantes intrigados pueden consultar textos sobre análisis real y funcional.

0.3 Linealidad

Una función entre espacios vectoriales que conserva la estructura se conoce como función lineal:

Definición 0.7 (Linealidad). Supongamos que V y V son espacios vectoriales. Entonces, L : V → V es lineal si satisface los siguientes dos criterios para todo v,v1,v2 V y c R:

- L conserva las sumas: L[v1 +v2] = L[v1] + L[v2]
- L conserva los productos escalares: L[cv] = cL[v]

Es fácil generar mapas lineales entre espacios vectoriales, como podemos ver en los siguientes ejemplos:

Ejemplo 0.8 (Linealidad en Rn). El siguiente mapa f : R2 → R3 es lineal:

$$f(x, y) = (3x, 2x + y, -y)$$

Podemos comprobar la linealidad de la siguiente manera:

- Preservación de la suma:

$$\begin{aligned} f(x1 + x2, y1 + y2) &= (3(x1 + x2), 2(x1 + x2) + (y1 + y2), -(y1 + y2)) \\ &= (3x1, 2x1 + y1, -y1) + (3x2, 2x2 + y2, \\ &\quad -y2) = f(x1, y1) + f(x2, y2) \end{aligned}$$

- Conservación escalar de productos:

$$f(cx, cy) = (3cx, 2cx + cy, -cy) = c(3x, 2x + y, -y) = c\, f(x, y)$$

Por el contrario, g(x, y) ≡ xy2 no es lineal. Por ejemplo, g(1, 1) = 1 pero g(2, 2) = 8 = 2 · g(1, 1), por lo que esta forma no conserva los productos escalares.

Ejemplo 0.9 (Integración). La siguiente L "funcional" de R[x] a R es lineal:

$$L[p(x)] \equiv \int_0^1 p(x)\, dx.$$

Este ejemplo algo más abstracto mapea polinomios p(x) a números reales L[p(x)]. Por ejemplo, podemos escribir

$$L[3x^2 + x - 1] = \int_0^1 (3x^2 1 + x - 1)\, dx = 2 \ -\cdot$$

La linealidad proviene de los siguientes hechos bien conocidos del cálculo:

$$\int_0^1 c \cdot f(x)\, dx = c \int_0^1 f(x)\, dx$$

$$\int_0^1 [f(x) + g(x)]\, dx = \int_0^1 f(x)\, dx + \int_0^1 g(x)\, dx$$

Podemos escribir una forma particularmente agradable para mapas lineales en Rn . Recuerde que el vector a = (a1, . . . , an) es igual a la suma ∑k akek , donde ek es el k-ésimo vector base estándar. Entonces, si L es lineal sabemos:

$$\begin{aligned} L[a] &= L \sum_k a_k e_k \text{ para la base estándar} e_k \\ &= \sum_k L[a_k e_k] \text{ por preservación de la suma} \\ &= \sum_k a_k L[e_k] \text{ por conservación del producto escalar} \end{aligned}$$

Esta derivación muestra el siguiente hecho importante:

L está completamente determinada por su acción sobre la base estándar vectoresek .

Es decir, para cualquier vectora , podemos usar la suma anterior para determinar L[a] combinando linealmente Rn L[e1], . . . ,L[es].

Ejemplo 0.10 (Expandiendo un mapa lineal). Recuerda el mapa del Ejemplo 0.8 dado por f(x, y) = (3x, 2x + y, −y). Tenemos f(e1) = f(1, 0) = (3, 2, 0) y f(e2) = f(0, 1) = (0, 1, −1). Por lo tanto, la fórmula anterior muestra:

$$f(x, y) = x\, f(e_1) + y\, f(e_2) = x \begin{matrix} 3 \\ 2 \\ 0 \end{matrix} + y \begin{matrix} 0 \\ 1 \\ -1 \end{matrix}$$

0.3.1 Matrices

La expansión de mapas lineales anterior sugiere uno de los muchos contextos en los que es útil almacenar múltiples vectores en la misma estructura. Más generalmente, digamos que tenemos n vectores v1, . . . ,vn Rm. Podemos escribir cada uno como un vector columna:

$$v_1 = \begin{matrix} v_{11} \\ v_{21} \\ \vdots \\ v_{m1} \end{matrix} \ ,v_2 = \begin{matrix} v_{12} \\ v_{22} \\ \vdots \\ v_{m2} \end{matrix} \ , \cdots ,v_n = \begin{matrix} v_{1n} \\ v_{2n} \\ \vdots \\ v_{mn} \end{matrix}$$

Transportarlos por separado puede ser engorroso desde el punto de vista de la notación, por lo que para simplificar las cosas simplemente los combinamos en una única matriz de m × n:

$$\begin{bmatrix} | & | & & | \\ v1 & v2 & \cdots & vn \\ | & | & & | \end{bmatrix} = \begin{matrix} v11 & v12 & \cdots & v1n \\ v21 & v22 & \cdots & v2n \\ \vdots & \vdots & \vdots & \vdots \\ vm1 & vm2 & \cdots & vmn \end{matrix}$$

Llamaremos al espacio de tales matrices $R^{m\times n}$.

Ejemplo 0.11 (Matriz de identidad). Podemos almacenar la base estándar para Rn en la "matriz de identidad" n × n En×n dado por:

$$\text{en} \times \text{n} \equiv \begin{bmatrix} | & | & & | \\ e1 & e2 & \cdots & es \\ | & | & & | \end{bmatrix} = \begin{matrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{matrix}$$

Dado que construimos matrices como formas convenientes de almacenar conjuntos de vectores, podemos usar la multiplicación para expresar cómo se pueden combinar linealmente. En particular, una matriz en Rm×n se puede multiplicar por un vector columna en Rn de la siguiente manera:

$$\begin{bmatrix} | & | & & | \\ v1 & v2 & \cdots & vn \\ | & | & & | \end{bmatrix} \begin{matrix} c1 \\ c2 \\ \vdots \\ cn \end{matrix} \equiv c1v1 + c2v2 + \cdots + cnvn$$

La expansión de esta suma produce la siguiente fórmula explícita para los productos matriz-vector:

$$\begin{matrix} v11 & v12 & \cdots & v1n \\ v21 & v22 & \cdots & v2n \\ \vdots & \vdots & \vdots & \vdots \\ vm1 & vm2 & \cdots & vmn \end{matrix} \begin{matrix} c1 \\ c2 \\ \vdots \\ cn \end{matrix} = \begin{matrix} c1v11 + c2v12 + \cdots + cnv1n \\ c1v21 + c2v22 + \cdots + cnv2n \\ \vdots \\ c1vm1 + c2vm2 + \cdots + cnvmn \end{matrix}$$

Ejemplo 0.12 (Multiplicación de matrices de identidad). Es claramente cierto que para cualquier x , podemos escribir Rn x = In×nx, donde In×n es la matriz identidad del ejemplo 0.11.

Ejemplo 0.13 (Mapa lineal). Volvemos una vez más a la expresión del Ejemplo 0.8 para mostrar una forma alternativa más:

$$f(x, y) = \begin{matrix} 3 & 0 \\ 2 & 1 \\ 0 & -1 \end{matrix} \begin{matrix} x \\ y \end{matrix}$$

De manera similar, definimos un producto entre una matriz en M Rm×n y otra matriz en Rn×p mediante la concatenación de productos matriz-vector individuales:

$$\text{METRO} \begin{bmatrix} | & | & & | \\ c1 & c2 & \cdots & cn \\ | & | & & | \end{bmatrix} \equiv \begin{bmatrix} | & | & & | \\ Mc1 & Mc2 & \cdots & Mcn \\ | & | & & | \end{bmatrix}$$

Ejemplo 0.14 (Mixología). Continuando con el Ejemplo 0.3, supongamos que hacemos un tequila sunrise y un segundo brebaje con partes iguales de los dos licores en nuestro pozo simplificado. Para averiguar cuánto de los ingredientes básicos contiene cada pedido, podríamos combinar las recetas para cada columna y usar la matriz multiplicación:

	Pozo 1	Pozo 2	Pozo 3		Beber 1	Beber 2			Beber 1	Beber 2	
Vodka	1	0	0		0	0.75			0 0,75	1,5 6	Vodka
Tequila	0	1	0	.	1.5	0.75	=			0.75	Tequila
DO	0	0	6		1	2				12	DO
Granadina	0	0	0.75						0.75	1.5	Granadina

En general, usaremos letras mayúsculas para representar matrices, como A Rm×n . Usaremos el notación Aij R para denotar el elemento de A en la fila i y la columna j.

0.3.2 Escalares, vectores y matrices

No sorprende que podamos escribir un escalar como un vector 1 × 1 c R1×1 . Similar, como ya sugerido en §0.2.3, si escribimos vectores en Rn en forma de columna, pueden considerarse matrices n × 1 v Rn×1 . Observe que los productos matriz-vector se pueden interpretar fácilmente en este contexto; Por ejemplo, si A Rm×n , x Rn , yb Rm, entonces podemos escribir expresiones como

$$\underset{m\times n}{A}\ \underset{n\times 1}{x} = \underset{m\times 1}{segundo}$$

Introduciremos un operador adicional en matrices que es útil en este contexto:

Definición 0.8 (Transposición). La transpuesta de una matriz A Rm×n es una matriz A Rn×m con elementos (A)ij = Aji.

Ejemplo 0.15 (Transposición). La transpuesta de la matriz

$$un = \begin{matrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{matrix}$$

es dado por

$$un = \begin{matrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{matrix} .$$

Geométricamente, podemos pensar en la transposición como voltear una matriz sobre su diagonal.

Este tratamiento unificado de escalares, vectores y matrices, combinado con operaciones como la transposición y la multiplicación, puede conducir a derivaciones ingeniosas de identidades bien conocidas. Por ejemplo,

podemos calcular los productos punto de los vectoresa,b Rn realizando la siguiente serie de pasos:

$$a \cdot b = \sum_{k=1}^{norte} akbk$$

$$= a1\ a2 \cdots un \begin{matrix} b1 \\ b2 \\ \vdots \\ mil\ millones \end{matrix}$$

$$= ab$$

Se pueden derivar muchas identidades importantes del álgebra lineal encadenando estas operaciones con unas pocas reglas:

$$(A) = A$$

$$(A + B) = A + B$$

$$(AB) = BA$$

Ejemplo 0.16 (Norma residual). Supongamos que tenemos una matriz A y dos vectores x y b. Si deseamos saber qué tan bien se aproxima Ax a b, podemos definir un residualr ≡ b − Ax; este residual es cero exactamente cuando Ax = b. De lo contrario, podríamos usar la norma r2 como proxy de la relación entre Ax yb.
Podemos usar las identidades anteriores para simplificar:

$$r \begin{matrix} 2 \\ 2 \end{matrix} = b - Ax \begin{matrix} 2 \\ 2 \end{matrix}$$

$$= (b - Ax) \cdot (b - Ax) \text{ como se explica en §0.2.3}$$

$$= (b - Ax)\ (b - Ax) \text{ por nuestra expresión para el producto escalar}$$

$$\text{anterior} = (b - x\ A\ )(b - Ax) \text{ por propiedades de}$$

$$\text{transposición} = bb - b\ Ax - x\ A\ b + x\ A\ Ax \text{ después de la multiplicación}$$

Los cuatro términos en el lado derecho son escalares, o equivalentemente matrices 1 × 1. Los escalares considerados como matrices disfrutan trivialmente de una buena propiedad adicional c = c, ¡ya que no hay nada que transponer! Así, podemos escribir

$$x\ A\ b = (x\ A\ b) = b\ Ax$$

Esto nos permite simplificar aún más nuestra expresión:

$$r \begin{matrix} 2 \\ 2 \end{matrix} = bb - 2b\ Ax + x\ A\ Ax$$

$$= \text{hacha} \begin{matrix} 2 \\ 2 \end{matrix} - 2b\ Ax + b \begin{matrix} 2 \\ 2 \end{matrix}$$

Podríamos haber derivado esta expresión usando identidades de productos escalares, pero los pasos intermedios anteriores resultarán útiles en nuestra discusión posterior.

0.3.3 Problema modelo: Ax =b

En la clase de introducción al álgebra, los estudiantes pasan un tiempo considerable resolviendo sistemas lineales como los siguientes para tripletes (x, y, z):

$$\begin{aligned} 3x + 2y + 5z &= 0 \\ -4x + 9y - 3z &= -7\ 2x \\ -3y - 3z &= 1 \end{aligned}$$

Nuestras construcciones en §0.3.1 nos permiten codificar tales sistemas de una manera más limpia:

$$\begin{bmatrix} 3 & 2 & 5 \\ -4 & 9 & -3 \\ 2 & -3 & -3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ -7 \\ 1 \end{bmatrix}$$

Más generalmente, podemos escribir sistemas lineales de ecuaciones en la forma Ax = b siguiendo el mismo patrón anterior; aquí, el vector x es desconocido mientras que A yb son conocidos. No siempre se garantiza que dicho sistema de ecuaciones tenga una solución. Por ejemplo, si A contiene solo ceros, entonces claramente ninguna x satisfará Ax =b siempre que b =0. Aplazaremos una consideración general de cuándo existe una solución a nuestra discusión de solucionadores lineales en capítulos futuros.

Una interpretación clave del sistema Ax =b es que aborda la tarea:

Escriba b como una combinación lineal de las columnas de A.

¿Por qué? Recuerde de §0.3.1 que el producto Ax codifica una combinación lineal de las columnas de A con pesos contenidos en elementos de x. Entonces, la ecuación Ax =b pide que la combinación lineal Ax sea igual al vector b dado. Dada esta interpretación, definimos el espacio columna de A como el espacio de lados derechosb para el cual el sistema tiene solución:

Definición 0.9 (Espacio columna). El espacio columna de una matriz A $\in$ $\mathbb{R}^{m\times n}$ es el lapso de las columnas de A. Podemos escribir como

$$\operatorname{col} A \equiv \{Ax : X \in \mathbb{R}^n\}.$$

Un caso importante es algo más fácil de considerar. Supongamos que A es cuadrado, entonces podemos escribir A $\in$ $\mathbb{R}^{n\times n}$. Además, suponga que el sistema Ax = b tiene una solución para todas las opciones de b. la única condición para que b sea miembro de $\mathbb{R}^n$ se , Entonces, según nuestra interpretación anterior de Ax = b, podemos concluye que las columnas de A generan $\mathbb{R}^n$.

En este caso, dado que el sistema lineal siempre tiene solución, supongamos que reemplazamos la base estándar e1, . . . ,en para producir vectores x1, . . . ,xn satisfaciendo Axk = ek para cada k. Luego, podemos “apilar” estas expresiones para mostrar:

$$A \begin{bmatrix} | & | & & | \\ x1 & x2 & \cdots & xn \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ Ax1 & Ax2 & \cdots & Axón \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ e1 & e2 & \cdots & es \\ | & | & & | \end{bmatrix} = \text{pulg} \times n,$$

donde $I_{n\times n}$ es la matriz identidad del ejemplo 0.11. Llamaremos a la matriz con columnas xk la inversa A^{-1} , que satisface

$$AA^{-1} = A^{-1}A = I_{n\times n}.$$

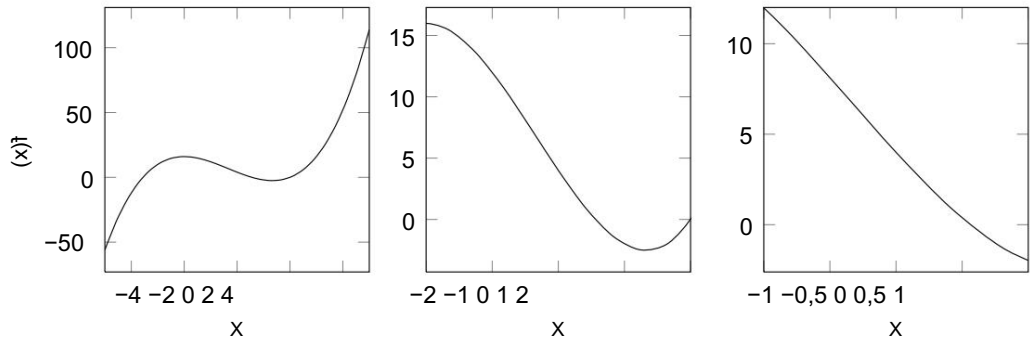


Figura 1: Cuanto más nos acercamos a f(x) = x ^{3} $^{2 + x}$ − 8x + 4, más se parece a una línea.

También es fácil comprobar que (A = A. $^{-1}$) $^{-1}$ Cuando existe tal inversa, es fácil resolver el sistema Ax =b. En particular, encontramos:

$$x = \ln \times nx = (A - 1A)x = A \quad {}^{-1} \quad (Ax) = A - 1b$$

0.4 No linealidad: cálculo diferencial

Si bien la belleza y la aplicabilidad del álgebra lineal lo convierten en un objetivo clave de estudio, las no linealidades abundan en la naturaleza y, a menudo, debemos diseñar sistemas computacionales que puedan lidiar con este hecho de la vida. Después de todo, en el nivel más básico, el cuadrado en la famosa relación E = mc2 hace que sea menos susceptible de análisis lineal.

0.4.1 Diferenciación

Si bien muchas funciones son globalmente no lineales, localmente exhiben un comportamiento lineal. Esta idea de "linealidad local" es uno de los principales motivadores detrás del cálculo diferencial. Por ejemplo, la Figura 1 muestra que si se acerca lo suficiente a una función suave, eventualmente se verá como una línea. La derivada f (x) de una función f(x) : R → R no es más que la pendiente de la línea de aproximación, calculada al encontrar la pendiente de las líneas a través de puntos cada vez más cercanos a x:

$$= \lim_{y \to x} \frac{f(y) - f(x)\ f\ (x)}{y - x}$$

Podemos expresar la linealidad local escribiendo f(x + ∆x) = f(x) + ∆x · f (x) + O(∆x ^{2}).

Si la función f toma múltiples entradas, entonces se puede escribir f(x) : Rn → R para x Rn ; en otras palabras, a cada punto x = (x1, . . . , xn) en el espacio n-dimensional f le asigna un solo número f(x1, . . . , xn). Nuestra idea de linealidad local falla un poco aquí, porque las líneas son objetos unidimensionales. Sin embargo, fijar todas las variables menos una se reduce al caso del cálculo de una sola variable. Por ejemplo, podríamos escribir g(t) = f(t, x2, . . . , xn), donde simplemente fijamos las constantes x2, . . . , xn. Entonces, g(t) es una función diferenciable de una sola variable. Por supuesto, podríamos haber puesto t en cualquiera de los espacios de entrada para f, por lo que en general hacemos la siguiente definición de la derivada parcial de f:

f Definición 0.10 (Derivada parcial). La k-ésima derivada parcial de f, anotada como $\frac{\partial f}{\partial x_k}$, está dada por diferentes f en su k-ésima variable de entrada:

$$\frac{\partial f}{\partial x_k}(x_1, \ldots, x_n) \equiv \frac{d}{dt} f(x_1, \ldots, x_{k-1}, t, x_{k+1}, \ldots, x_n)|_{t=x_k}$$

La notación "$|_{t=x_k}$" debe leerse como "evaluado en $t = x_k$".

Ejemplo 0.17 (Relatividad). La relación $E = mc^2$ se puede considerar como una función de m y c a E. Por lo tanto, podríamos escribir $E(m, c) = mc^2$, dando las derivadas

$$\frac{\partial E}{\partial m} = c^2$$
$$\frac{\partial E}{\partial c} = 2mc$$

Usando el cálculo de una sola variable, podemos escribir:

$$f(x + \Delta x) = f(x_1 + \Delta x_1, x_2 + \Delta x_2, \ldots, x_n + \Delta x_n)$$
$$f = f(x_1, x_2 + \Delta x_2, \ldots, x_n + \Delta x_n) + \frac{\partial}{\partial x_1}\Delta x_1 + O(\Delta x \partial x_1^2) \text{ por cálculo de una sola variable}$$
$$= f(x_1, \ldots, x_n) + \sum_{k=1}^{n} \frac{\partial f}{\partial x_k}\Delta x_k + O(\Delta x \partial x_k^2) \text{ repitiendo esto n veces}$$
$$= f(x) + \nabla f(x) \cdot \Delta x + O(x^2)$$

donde definimos el gradiente de f como

$$\nabla f \equiv \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right] \in \mathbb{R}^n$$

A partir de esta relación, es fácil ver que f puede derivarse en cualquier dirección v; podemos evaluar esta derivada $D_v f$ de la siguiente manera:

$$D_v f(x) \equiv \frac{d}{dt} f(x + tv)|_{t=0} = \nabla f(x) \cdot v$$

Ejemplo 0.18 ($\mathbb{R}^2$). Tome $f(x, y) = x^2 y^3$. Entonces,

$$\frac{\partial f}{\partial x} = 2xy^3$$
$$\frac{\partial f}{\partial y} = 3x^2 y^2$$

Así, podemos escribir $\nabla f(x, y) = (2xy^3, 3x^2 y^2)$. La derivada de f en (1, 2) en la dirección (−1, 4) viene dada por $(-1, 4) \cdot \nabla f(1, 2) = (-1, 4) \cdot (16, 12) = 32$.

Ejemplo 0.19 (Funciones lineales). Es obvio pero vale la pena señalar que el gradiente de $f(x) \equiv a \cdot x + c = (a_1 x_1 + c_1, \ldots, a_n x_n + c_n)$ es a.

Ejemplo 0.20 (Formas cuadráticas). Tome cualquier matriz A Rn×n que , y define f(x) ≡ x Ax. En expansión muestra esta función elemento por elemento

$$f(x) = \sum_{yo} Aijxixj ;$$

Vale la pena expandir f y verificar esta relación explícitamente. Tome algo de k {1, . . . , n}. Entonces, podemos separar todos los términos que contienen xk :

$$f(x) = Akkx_k^2 + x\,k \sum_{yo=k} Aikxi + \sum_{j=k} Akjxj + \sum_{i,j=k} Aijxixj$$

Con esta factorización, es fácil ver

$$\frac{\partial f}{\partial xk} = 2Akkxk + \sum_{yo=k} Aikxi + \sum_{j=k} akjxj$$

$$= \sum_{yo=1}^{\infty} (Aik + Aki)xi$$

¡Esta suma no es más que la definición de la multiplicación matriz-vector! Así, podemos escribir

$$f(x) = Ax + Ax.$$

Hemos generalizado de f : R → R a f : Rn → R. Para alcanzar la generalidad completa, nos gustaría considerar f : Rn → Rm. En otras palabras, f toma n números y genera m números. Afortunadamente, esta extensión es sencilla, porque podemos pensar en f como una colección de funciones de un solo valor f1, . . . , fm : Rn → R unidos en un solo vector. Es decir, escribimos:

$$f(x) = \begin{matrix} f1(x) \\ f2(x) \\ \vdots \\ fm(x) \end{matrix}$$

Cada fk se puede diferenciar como antes, por lo que al final obtenemos una matriz de derivadas parciales llamada jacobiana de f :

Definición 0.11 (jacobiano). El jacobiano de f : Rn → Rm es la matriz D f Rm×n con entradas

$$(Df)ij \equiv \frac{\partial fi}{\partial xj}.$$

Ejemplo 0.21 (Función simple). Supongamos que f(x, y) = (3x, −xy2 , x + y). Entonces,

$$D\,f(x, y) = \begin{matrix} 3 \\ -y^2 \quad 0 \\ 1 \qquad -2xy \\ 1 \end{matrix}$$

Asegúrese de que puede derivar este cálculo a mano.

Ejemplo 0.22 (Multiplicación de matrices). Como era de esperar, el jacobiano de f(x) = Ax para la matriz A está dado por D f(x) = A.

Aquí nos encontramos con un punto común de confusión. Supongamos que una función tiene entrada vectorial y salida escalar, es decir, f : Rn → R. Definimos el gradiente de f como un vector columna, por lo que para alinear esta definición con la del jacobiano debemos escribir

$$\text{re } F = \quad F .$$

0.4.2 Optimización

Recordemos los mínimos y máximos de f del cálculo de una sola variable: R → R debe ocurrir en los puntos x que satisfacen f (x) = 0. Por supuesto, esta condición es necesaria más que suficiente: pueden existir puntos x con f (x) = 0 que no son máximos ni mínimos. Dicho esto, encontrar esos puntos críticos de f puede ser un paso de un algoritmo de minimización de funciones, siempre que el siguiente paso asegure que la x resultante sea realmente un mínimo/máximo.

Si f : Rn → R se minimiza o maximiza en x, debemos asegurarnos de que no existe una sola dirección ∆x desde x en la que f disminuye o aumenta, resp. Por la discusión en §0.4.1, esto significa que debemos encontrar puntos para los cuales f = 0.

Ejemplo 0.23 (Función simple). Supongamos que f(x, y) = x 2x + ^{2} + 2xy + 4y ^{2}2y y ∂x ∂y $\frac{\partial F}{\partial F}$. Entonces, = 2x + ___ = 8y. Así, los puntos críticos de f satisfacen:

$$2x + 2y = 0 \; 2x$$
$$+ 8y = 0$$

Claramente este sistema se resuelve en (x, y) = (0, 0). De hecho, este es el mínimo de f , como se puede ver más claramente al escribir f(x, y) = (x + ^{2}y) 3 años ^{2}.

Ejemplo 0.24 (Funciones cuadráticas). Supongamos que f(x) = xAx +bx + c. Entonces, de los ejemplos de la sección anterior podemos escribir f(x) = (A + A)x +b. Así, los puntos críticos x de f satisfacen (A + A)x +b = 0.

A diferencia del cálculo de una sola variable, cuando hacemos cálculos en Rn podemos agregar restricciones a nuestra optimización. La forma más general de tal problema se ve así:

$$\text{minimizar } f(x) \text{ tal}$$
$$\text{que } g(x) = 0$$

Ejemplo 0.25 (Áreas de rectángulo). Supongamos que un rectángulo tiene un ancho w y una altura h. Un problema clásico de geometría es maximizar el área con un perímetro fijo 1:

$$\text{maximizar qué}$$
$$\text{tal que } 2w + 2h - 1 = 0$$

Cuando agregamos esta restricción, ya no podemos esperar que los puntos críticos satisfagan f(x) = 0, ya que estos puntos podrían no satisfacer g(x) = 0.

Por ahora, suponga g : Rn → R. Considere el conjunto de puntos S0 ≡ {x : g(x) = 0}. Obviamente, dos x,y S0 cualesquiera satisfacen la relación g(y) − g(x) = 0 − 0 = 0. Supongamos que y = x + ∆x para ∆x pequeño. Entonces, g(y) − g(x) = g(x) · ∆x + O(∆x). En otras palabras, si comenzamos en x satisfaciendo g(x) = 0, entonces si nos desplazamos en la dirección ∆x g(x) · ∆x ≈ 0 para continuar satisfaciendo esta relación.

Ahora, recuerda que la derivada de f en la dirección v en x está dada por f ·v. Si x es un mínimo del problema de optimización con restricciones anterior, entonces cualquier pequeño desplazamiento de x a x + v debería causar un aumento de f (x) a f (x + v). Dado que solo nos importan los desplazamientos v conservando la restricción g(x + v) = c, de nuestro argumento anterior queremos f · v = 0 para todo v que satisfaga g(x)·v = 0. En otras palabras, f y g deben ser paralelas, una condición que podemos escribir como f = λ g para algún λ R.

Definir

$$\Lambda(x, \lambda) = f(x) - \lambda g(x).$$

Entonces, los puntos críticos de Λ sin restricciones satisfacen:

$$0 = \frac{\partial \Lambda}{\partial \lambda} = -g(x)$$
$$0 = \quad x\Lambda = \quad f(x) - \lambda \quad g(x)$$

En otras palabras, los puntos críticos de Λ satisfacen g(x) = 0 y f(x) = λ g(x), ¡exactamente las condiciones de optimización que derivamos!

La extensión a restricciones multivariadas produce lo siguiente:

Teorema 0.1 (Método de los multiplicadores de Lagrange). Los puntos críticos del problema de optimización con restricciones anterior son puntos críticos sin restricciones de la función multiplicadora de Lagrange

$$\Lambda(x,\lambda) \equiv f(x) - \lambda \cdot g(x),$$

con respecto a x y λ.

Ejemplo 0.26 (Maximizar área). Continuando con el Ejemplo 0.25, definimos la función multiplicadora de Lagrange Λ(w, h, λ) = wh − λ(2w + 2h − 1). Diferenciando, encontramos:

$$\frac{\partial \Lambda}{\partial w} = h - 2\lambda \; 0 =$$
$$w - 2\lambda = =$$
$$\frac{\partial \Lambda}{\partial h}$$
$$\frac{\partial \Lambda}{\partial \lambda} = 1 - 2w - 2h \; 0 =$$

Entonces, los puntos críticos del sistema satisfacen

$$\begin{matrix} 0 & 1 & -2 \\ 1 & 0 & -2 \\ 2 & 2 & 0 \end{matrix} \begin{matrix} w \\ h \\ \lambda \end{matrix} = \begin{matrix} 0 \\ 0 \\ 1 \end{matrix}$$

Resolviendo el sistema se muestra w = h = 1/4 y λ = 1/8. En otras palabras, para una cantidad fija de perímetro, el rectángulo con área máxima es un cuadrado.

Ejemplo 0.27 (Problemas propios). Suponga que A es una matriz definida positiva simétrica, lo que significa que A = A (simetría) y x Ax > 0 para todo x Rn\{0} (definida positiva). A menudo deseamos minimizar x Ax sujeto a x $\frac{2}{2}$ 1 para una matriz dada A Rn×n ; observe que sin la restricción, el mínimo trivialmente tiene lugar en x = 0. Definimos la función multiplicadora de Lagrange

$$\Lambda(x, \lambda) = x\,Ax - \lambda(x \quad {}^{2}_{2} - 1)$$
$$= x\,Ax - \lambda(x\,x - 1).$$

Derivando con respecto a x, encontramos

$$0 = \quad x\Lambda = 2Ax - 2\lambda x$$

En otras palabras, x es un vector propio de la matriz A:

$$Ax = \lambda x.$$

0.5 Problemas

Problema 0.1. Tómese C1 (R) como el conjunto de funciones f : R → R que admite una primera derivada f (x). ¿Por qué es ^{1} C(R) un espacio vectorial? Demostrar que C1 (R) tiene dimensión ∞.

Problema 0.2. Suponga que las filas de A Rm×n están dadas por las transpuestas de r1, . . . ,rm Rn y las columnas de A Rm×n están dadas por c1, . . . ,cn Rm. Eso es,

$$un = \begin{bmatrix} - & r_1 & - \\ - & r_2 & - \\ & \vdots & \\ - & r_{metro} & - \end{bmatrix} = \begin{bmatrix} | & | & & | \\ c1 & c2 & \cdots & cn \\ | & | & & | \end{bmatrix}.$$

Dé expresiones para los elementos de AA y AA en términos de estos vectores.

Problema 0.3. Dé un sistema lineal de ecuaciones satisfecho por mínimos de la energía f(x) = Ax −b $^{2}_{2}$ con respecto a x, para x Rn, y b Rm. Este sistema se denomina "ecuaciones normales" y aparecerá A Rm×n en otras partes de estas notas; aun así, vale la pena trabajar y comprender completamente la derivación.

Problema 0.4. Suponga que A, B Rn×n . Formule una condición para que los vectores x Rn sean puntos críticos Hacha $^{2}_{2}$ de sujeto a Bx = $^{2}_{2}$. Además, dé una forma alternativa para los valores óptimos de Ax $^{2}_{2}$.

Problema 0.5. Fija algún vectora Rn\{0} y define f(x) =a · x. Dé una expresión para el máximo de f(x) sujeto a x = 1.

Capítulo 1

Análisis numérico y de errores

Al estudiar el análisis numérico, pasamos de tratar con enteros y largos a flotantes y dobles.
Esta transición aparentemente inocente comprende un gran cambio en la forma en que debemos pensar sobre el diseño y la implementación de micrófonos algorítmicos. A diferencia de los conceptos básicos de los algoritmos discretos, ya no podemos esperar nuestros algoritmos para producir soluciones exactas en todos los casos. “Big O” y el conteo de operaciones no siempre reinar; en cambio, incluso en la comprensión de las técnicas más básicas nos vemos obligados a estudiar el compromiso entre tiempo, error de aproximación, etc.

1.1 Almacenamiento de números con partes fraccionarias

Recuerde que las computadoras generalmente almacenan datos en formato binario. En particular, cada dígito de un positivo entero corresponde a una potencia diferente de dos. Por ejemplo, podríamos convertir 463 a binario usando la siguiente tabla:

1	1	1	0	0	1			1	1	1
2^8	2^7	2^6		2^5		2^4	2^3	2^2	2^1	2^0

En otras palabras, esta notación codifica el hecho de que 463 se puede descomponer en potencias de dos únicamente como:

$$463 = 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$$
$$= 256 + 128 + 64 + 8 + 4 + 2 + 1$$

Dejando a un lado los problemas de desbordamiento, todos los números enteros positivos se pueden escribir de esta forma usando un número finito de dígitos Los números negativos también se pueden representar de esta manera, ya sea introduciendo un signo inicial bit o usando el truco del “complemento a dos”.

Tal descomposición inspira una extensión simple a los números que incluyen fracciones: simplemente incluyen potencias negativas de dos. Por ejemplo, descomponer 463.25 es tan simple como sumar dos tragamonedas:

1	1	1	0	0	1	3	2		1	1	1.	0	0	2	1
2^8	2^7	2^6		2^5		2^4			2^2	2^1		2^{-1}			2^{-2}

Sin embargo, al igual que en el sistema decimal, representar partes fraccionarias de números de esta manera es no se comporta tan bien como representar números enteros. Por ejemplo, escribir la fracción 1/3 en binario da la expresión:

$$\frac{1}{3} = 0.0101010101\ldots$$

Tales ejemplos muestran que existen números en todas las escalas que no se pueden representar usando un cadena binaria finita. De hecho, números como π = 11.00100100001 . . .2 tienen expansiones infinitamente largas independientemente de la base (entera) que utilice.

Por esta razón, al diseñar sistemas computacionales que hacen operaciones matemáticas en R en lugar de Z, se ven obligados a hacer aproximaciones para casi cualquier representación numérica razonablemente eficiente. Esto puede llevar a muchos puntos de confusión durante la codificación. Por ejemplo, considere el siguiente C++ retazo:

```
doble x = 1,0;
doble y = x / 3.0;
if ( x == y *3.0) cout <<" ¡Son iguales! " ;
de lo contrario cout << " NO son iguales. " ;
```

Contrariamente a la intuición, este programa imprime "NO son iguales". ¿Por qué? La definición de y hace una aproximación a 1/3 ya que no se puede escribir como una cadena binaria de terminación, redondeando a un número cercano que pueda representar. Por lo tanto, y*3.0 ya no multiplica 3 por 1/3. Una forma de arreglar este problema se encuentra a continuación:

```
doble x = 1,0;
doble y = x / 3.0;
if ( fabs (x - y *3.0) < límites_numéricos < double >:: epsilon ) cout << else cout << NO son iguales . " Son iguales ! " ;
;
```

Aquí, verificamos que x e y*3.0 estén dentro de cierta tolerancia entre sí en lugar de verificar igualdad exacta. Este es un ejemplo de un punto muy importante:

Rara vez, si acaso, se debe usar el operador == y sus equivalentes en valores fraccionarios.

En cambio, se debe usar cierta tolerancia para verificar si los números son iguales.

Por supuesto, aquí hay una compensación: el tamaño de la tolerancia define una línea entre la igualdad y "cercano pero no igual", que debe elegirse con cuidado para una aplicación determinada.

Consideramos algunas opciones para representar números en una computadora a continuación.

1.1.1 Representaciones de puntos fijos

La opción más sencilla para almacenar fracciones es agregar un punto decimal fijo. Es decir, como en En el ejemplo anterior, representamos valores almacenando coeficientes 0/1 frente a potencias de dos que van de 2−k a 2 para algún k, Z. Por ejemplo, representando todos los valores no negativos entre 0 y 127,75 en incrementos de 1/4 es tan fácil como tomar k = 2 y = 7; en esta situación, representamos estos valores utilizan 9 dígitos binarios, de los cuales dos aparecen después del punto decimal.

La principal ventaja de esta representación es que casi todas las operaciones aritméticas se pueden se lleva a cabo utilizando los mismos algoritmos que con los números enteros. Por ejemplo, es fácil ver que

$$a + b = (a \cdot 2^{k} + segundo \cdot 2^{k}) \cdot 2^{-k}.$$

Multiplicar nuestra representación fija por 2k garantiza que el resultado sea integral, por lo que esta observación esencialmente muestra que la suma puede llevarse a cabo usando la suma de enteros esencialmente “ignorando” el punto decimal. Por lo tanto, en lugar de utilizar hardware especializado, el número entero preexistente La unidad lógica aritmética (ALU) realiza matemáticas de punto fijo rápidamente.

La aritmética de punto fijo puede ser rápida, pero puede sufrir serios problemas de precisión. En particular, a menudo ocurre que la salida de una operación binaria como la multiplicación o la división puede requerir más bits que los operandos. Por ejemplo, supongamos que incluimos un punto decimal de precisión y

desea realizar el producto 1/2 · 1/2 = 1/4. Escribimos 0,12 × 0,12 = 0,012, que se trunca a 0. En este sistema, es bastante sencillo combinar números de punto fijo de forma razonable y obtener un resultado irrazonable.

Debido a estos inconvenientes, la mayoría de los principales lenguajes de programación no incluyen por defecto un tipo de datos decimal de punto fijo. Sin embargo, la velocidad y la regularidad de la aritmética de punto fijo pueden ser una ventaja considerable para los sistemas que favorecen la sincronización sobre la precisión. De hecho, algunas unidades de procesamiento de gráficos (GPU) de gama baja implementan solo estas operaciones, ya que unos pocos puntos decimales de precisión son suficientes para muchas aplicaciones gráficas.

1.1.2 Representaciones de punto flotante

Uno de los muchos desafíos numéricos al escribir aplicaciones científicas es la variedad de escalas que pueden aparecer. Solo los químicos manejan valores entre 9,11 × 10−31 y 6,022 × 1023. Una operación tan inocente como un cambio de unidades puede causar una transición repentina entre estos regímenes: la misma observación escrita en kilogramos por año luz se verá considerablemente diferente en megatones. por segundo. Como analistas numéricos, nuestro trabajo es escribir software que pueda hacer la transición entre estas escalas sin imponer al cliente restricciones antinaturales en sus técnicas.

Algunas nociones y observaciones del arte de la medición científica son relevantes para tal discusión. Primero, obviamente una de las siguientes representaciones es más compacta que la otra:

$$6.022 \times 1023 = 602,200,000,000,000,000,000,000$$

Además, en ausencia de equipo científico excepcional, la diferencia entre 6,022 × 1023 y 6,022 × 1023 + 9,11 × 10−31 es insignificante. Una forma de llegar a esta conclusión es decir que 6,022 × 1023 tiene solo tres dígitos de precisión y probablemente representa algún rango de medidas posibles $[6{,}022 \times 1023 - \varepsilon, 6{,}011 \times 1023 + \varepsilon]$ para algunos $\varepsilon \quad 0{,}001 \times 1023$.

Nuestra primera observación fue capaz de compactar nuestra representación de 6.022 × 1023 escribiéndola en notación científica. Este sistema numérico separa los dígitos "interesantes" de un número de su orden de magnitud escribiéndolo en la forma a × 10b para algunos a 1 y b Z. Llamamos a este formato la forma de punto flotante de un número, porque a diferencia de la configuración de punto fijo en §1.1.1, aquí el punto decimal "flota" hacia arriba. Podemos describir los sistemas de punto flotante usando algunos parámetros (CITE):

- La base β N; para la notación científica explicada anteriormente, la base es 10
- La precisión p N que representa el número de dígitos en la expansión decimal
- El rango de exponentes [L, U] que representan los posibles valores de b

Tal expansión se parece a:

$$\underbrace{\pm}_{\text{firmar}} \underbrace{(d0 + d1 \cdot \beta^{-1} + d2 \cdot \beta^{-2} + \cdots + dp-1 \cdot \beta^{1-p})}_{\text{mantisa}} \times \underbrace{\beta^{b}}_{\text{exponente}}$$

donde cada dígito dk está en el rango [0, β − 1] y b [L, U].

Las representaciones de coma flotante tienen una propiedad curiosa que puede afectar al software de formas inesperadas: su espaciado es desigual. Por ejemplo, el número de valores representables entre β y β aunque normalmente es β − β, para β es β − β. Para comprender y la precisión posible con un sistema numérico dado, definiremos la precisión de la máquina εm como la

el más pequeño εm > 0 tal que 1 + εm es representable. Entonces, números como β + εm no se pueden expresar en el sistema numérico porque εm es demasiado pequeño.

Con mucho, el estándar más común para almacenar números de punto flotante es el estándar IEEE 754. Este estándar especifica varias clases de números de coma flotante. Por ejemplo, un número de coma flotante de precisión doble se escribe en base β = 2 (como la mayoría de los números en la computadora), con un solo bit de signo ± , 52 dígitos para d y un rango de exponentes entre −1022 y 1023. El estándar también especifica cómo almacenar ±∞ y valores como NaN, o "no es un número", reservados para los resultados de cálculos como 10/0. Se puede obtener un poco más de precisión escribiendo valores de punto flotante normalizados y asumiendo que el dígito más significativo d0 es 1 y no escribiéndolo.

El estándar IEEE también incluye opciones acordadas para manejar el número finito de valores que se pueden representar dado un número finito de bits. Por ejemplo, una estrategia imparcial común para los cálculos de redondeo es redondear al más cercano, lazos a pares, lo que rompe los lazos equidistantes al redondear al valor de punto flotante más cercano con un bit menos significativo (más a la derecha). Tenga en cuenta que hay muchas estrategias igualmente legítimas para redondear; elegir uno solo garantiza que el software científico funcionará de manera idéntica en todas las máquinas cliente que implementen el mismo estándar.

1.1.3 Opciones más exóticas

En el futuro, supondremos que los valores decimales se almacenan en formato de punto flotante a menos que se indique lo contrario. Esto, sin embargo, no quiere decir que no existan otros sistemas numéricos, y para aplicaciones específicas podría ser necesaria una elección alternativa. Reconocemos algunas de esas situaciones aquí.

El dolor de cabeza de agregar tolerancias para tener en cuenta los errores de redondeo puede ser inaceptable para algunas aplicaciones. Esta situación aparece en aplicaciones de geometría computacional, por ejemplo, cuando la diferencia entre líneas casi paralelas y completamente paralelas puede ser una distinción difícil de hacer. Una solución podría ser usar aritmética de precisión arbitraria, es decir, implementar la aritmética sin redondeo ni error de ningún tipo.

La aritmética de precisión arbitraria requiere una implementación especializada y una cuidadosa consideración de los tipos de valores que necesita representar. Por ejemplo, puede darse el caso de que los números racionales Q sean suficientes para una aplicación dada, lo que puede escribirse como razones a/b para a, b Z.
Las operaciones aritméticas básicas se pueden realizar en Q sin pérdida de precisión. Por ejemplo, es fácil ver

$$\frac{a}{b} \times \frac{c}{d} = \frac{c.a}{bd} \qquad \frac{a}{b} \div \frac{c}{d} = \frac{\text{anuncio}}{\text{antes de Cristo}}.$$

La aritmética en los racionales excluye la existencia de un operador de raíz cuadrada, ya que valores como $\sqrt{2}$ son irracionales. Además, esta representación no es única, ya que, por ejemplo, a/b = 5a/5b.

Otras veces puede ser útil poner el error entre paréntesis representando los valores junto con las estimaciones del error como un par a,ε R; pensamos en el par (a, ε) como el rango a ± ε. Luego, las operaciones aritméticas también actualizan no solo el valor sino también la estimación del error, como en

$$(x \pm \varepsilon 1) + (y \pm \varepsilon 2) = (x + y) \pm (\varepsilon 1 + \varepsilon 2 + \text{error}(x + y)),$$

donde el término final representa una estimación del error inducido al sumar x e y.

1.2 Comprender el error

Con la excepción de los sistemas de precisión arbitraria descritos en §1.1.3, casi todas las representaciones computarizadas de números reales con partes fraccionarias se ven obligadas a emplear redondeo y otros esquemas de aproximación. Este esquema representa una de las muchas fuentes de aproximaciones que se encuentran típicamente en los sistemas numéricos:

- El error de truncamiento proviene del hecho de que solo podemos representar un subconjunto finito de todos los posibles conjuntos de valores en R; por ejemplo, debemos truncar secuencias largas o infinitas más allá del punto decimal hasta el número de bits que estamos dispuestos a almacenar.
- El error de discretización proviene de nuestras adaptaciones computarizadas de cálculo, física y otros aspectos de las matemáticas continuas. Por ejemplo, hacemos una aproximación

$$\frac{dy}{dx} \approx \frac{y(x + \varepsilon) - y(x)}{\varepsilon}.$$

  Aprenderemos que esta aproximación es legítima y útil, pero dependiendo de la elección de ε puede no ser completamente correcta.
- El error de modelado proviene de descripciones incompletas o inexactas de los problemas que queremos resolver. Por ejemplo, una simulación para predecir el clima en Alemania puede optar por ignorar el aleteo colectivo de las mariposas en Malasia, aunque el desplazamiento del aire por estas mariposas puede ser suficiente para perturbar un poco los patrones climáticos en otros lugares.
- El error constante empírico proviene de representaciones deficientes de constantes físicas o matemáticas. Por ejemplo, podemos calcular π utilizando una secuencia de Taylor que terminamos antes de tiempo, e incluso los científicos pueden no conocer la velocidad de la luz a más de una cierta cantidad de dígitos.
- El error de entrada puede provenir de aproximaciones de parámetros de un sistema dado generadas por el usuario (¡y de errores tipográficos!). Las técnicas numéricas y de simulación se pueden usar para responder preguntas del tipo "qué pasaría si", en las que se eligen opciones exploratorias de configuraciones de entrada solo para tener una idea de cómo se comporta un sistema.

Ejemplo 1.1 (Física computacional). Supongamos que estamos diseñando un sistema para simular planetas mientras giran alrededor de la tierra. El sistema esencialmente resuelve la ecuación de Newton F = ma integrando fuerzas hacia delante en el tiempo. Los ejemplos de fuentes de error en este sistema pueden incluir:

- Error de truncamiento: uso de punto flotante IEEE para representar parámetros y salida del sistema y truncamiento al calcular el producto ma
- Error de discretización: Reemplazar la aceleración a por una diferencia dividida
- Error de modelado: Descuidar simular los efectos de la luna sobre el movimiento de la tierra dentro del planetario sistema
- Error empírico: Solo ingresando la masa de Júpiter a cuatro dígitos
- Error de entrada: el usuario puede desear evaluar el costo de enviar basura al espacio en lugar de arriesgarse a una acumulación al estilo Wall-E en la Tierra, pero solo puede estimar la cantidad de basura que el gobierno está dispuesto a desechar de esta manera.

1.2.1 Error de clasificación

Dada nuestra discusión anterior, se podría considerar que los siguientes dos números tienen la misma cantidad de error potencial:

$$1 \pm 0{,}01$$

$$105 \pm 0{,}01$$

Aunque tiene el tamaño del rango [1 − 0.01, 1 + 0.01], el rango [105 − 0.01, 105 + 0.01] parece codificar una medida más confiable porque el error 0.01 es mucho más pequeño en relación con 105 que con 1.

La distinción entre estas dos clases de error se describe diferenciando entre ab
error de soluto y error relativo:

Definición 1.1 (Error absoluto). El error absoluto de una medida viene dado por la diferencia entre el valor aproximado y su valor verdadero subyacente.

Definición 1.2 (Error relativo). El error relativo de una medida viene dado por el error absoluto dividido por el valor real.

Una forma de distinguir entre estas dos especies de error es el uso de unidades versus porcentajes.

Ejemplo 1.2 (Error absoluto y relativo). Aquí hay dos declaraciones equivalentes en formas contrastantes:

Absoluto: 2 pulgadas ± 0,02 pulgadas

Relativo: 2 en ± 1%

En la mayoría de las aplicaciones se desconoce el valor real; después de todo, si este no fuera el caso, el uso de una aproximación en lugar del valor verdadero puede ser una proposición dudosa. Hay dos formas populares de resolver este problema. El primero es simplemente ser conservador al realizar los cálculos: en cada paso, tome la estimación de error más grande posible y propague estas estimaciones según sea necesario. Tales estimaciones conservadoras son poderosas porque cuando son pequeñas podemos estar muy seguros de que nuestra solución es útil.

Una resolución alternativa tiene que ver con lo que puedes medir. Por ejemplo, supongamos que deseamos resolver la ecuación $f(x) = 0$ para x dada una función $f : \mathbb{R} \to \mathbb{R}$. Sabemos que en algún lugar existe una raíz x_0 que satisface $f(x_0) = 0$ exactamente, pero si supiéramos esto root nuestro algoritmo no sería necesario en primer lugar. En la práctica, nuestro sistema computacional puede producir alguna x_{est} que satisfaga $f(x_{est}) = \varepsilon$ para alguna ε con $|\varepsilon|$ 1. Es posible que no podamos evaluar la diferencia $x_0 - x_{est}$ ya que x_0 es desconocido. Por otro lado, simplemente evaluando f podemos calcular $f(x_{est}) - f(x_0) \equiv f(x_{est})$ ya que sabemos que $f(x_0) = 0$ por definición. Este valor da una idea de error para nuestro cálculo.

Este ejemplo ilustra la distinción entre error hacia adelante y hacia atrás. El error de avance realizado por una aproximación muy probablemente define nuestra intuición para el análisis de errores como la diferencia entre la solución aproximada y la real, pero como hemos discutido, no siempre es posible calcular. El error inverso, sin embargo, tiene la particularidad de ser calculable pero no nuestro objetivo exacto al resolver un problema dado. Podemos ajustar nuestra definición e interpretación del error hacia atrás a medida que abordamos diferentes problemas, pero una definición adecuada si es vaga es la siguiente:

Definición 1.3 (Error hacia atrás). El error hacia atrás está dado por la cantidad que tendría que cambiar el enunciado de un problema para realizar una aproximación dada de su solución.

Esta definición es algo obtusa, por lo que ilustramos su uso en algunos ejemplos.

Ejemplo 1.3 (Sistemas lineales). Supongamos que deseamos resolver el sistema lineal n × n Ax = b. Llame a la solución verdadera x0 ≡ A −1b. En realidad, debido al error de truncamiento y otros problemas, nuestro sistema produce una solución cercana a xest. El error directo de esta aproximación obviamente se mide usando la diferencia xest −x0; en la práctica este valor es imposible de calcular ya que no conocemos x0. En realidad, xest es la solución exacta a un sistema modificado Ax = mejor para mejor ≡ Axest; por lo tanto, podríamos medir el error hacia atrás en términos de la diferenciab −mejor. A diferencia del error hacia adelante, este error es fácilmente computable sin invertir A, y es fácil ver que xest es una solución al problema exactamente cuando el error hacia atrás (o hacia adelante) es cero.

Ejemplo 1.4 (Resolución de ecuaciones, CITE). Supongamos que escribimos una función para encontrar raíces cuadradas de números positivos que da como resultado $\sqrt{2} \approx 1.4$. El error directo es $|1.4 - 1.41421\cdots| \approx 0.0142$. Observe que $1{,}42 = 1{,}96$, por lo que el error hacia atrás es $|1{,}96 - 2| = 0{,}04$.

Los dos ejemplos anteriores demuestran un patrón más amplio de que el error hacia atrás puede ser mucho más fácil de calcular que el error hacia adelante. Por ejemplo, evaluar el error hacia adelante en el ejemplo 1.3 requería invertir una matriz A mientras que evaluar el error hacia atrás solo requería la multiplicación por A. De manera similar, en el ejemplo 1.4, la transición del error hacia adelante al error hacia atrás reemplazó el cálculo de la raíz cuadrada con la multiplicación.

1.2.2 Acondicionamiento, Estabilidad y Precisión

En casi cualquier problema numérico, cero error hacia atrás implica cero error hacia adelante y viceversa.
Por lo tanto, una pieza de software diseñada para resolver tal problema seguramente puede terminar si encuentra que una solución candidata tiene cero error hacia atrás. Pero, ¿qué pasa si el error hacia atrás es distinto de cero pero pequeño?
¿Implica esto necesariamente un pequeño error de reenvío? Estas preguntas motivan el análisis de la mayoría de las técnicas numéricas cuyo objetivo es minimizar el error hacia adelante pero en la práctica solo pueden medir el error hacia atrás.

Deseamos analizar los cambios en el error hacia atrás en relación con el error hacia adelante para que nuestros algoritmos puedan decir con confianza utilizando solo el error hacia atrás que han producido soluciones aceptables.
Esta relación puede ser diferente para cada problema que queramos resolver, por lo que al final hacemos la siguiente clasificación aproximada:

- Un problema es insensible o bien condicionado cuando pequeñas cantidades de error hacia atrás implican pequeñas cantidades de error hacia adelante. En otras palabras, una pequeña perturbación del enunciado de un problema bien condicionado produce solo una pequeña perturbación de la solución verdadera.

- Un problema es sensible o está mal acondicionado cuando no es así.

Ejemplo 1.5 (ax = b). Supongamos como un ejemplo de juguete que queremos encontrar la solución x0 ≡ b/a de la ecuación lineal ax = b para a, x, b R. El error hacia adelante de una solución potencial x viene dado por x − x0 mientras que el error hacia atrás es dado por b − ax = a(x − x0). Entonces, cuando |a| 1, el problema está bien condicionado ya que los valores pequeños del error hacia atrás a(x − x0) implican valores aún más pequeños de x − x0; por el contrario, cuando |a| 1 el problema está mal condicionado, ya que incluso si a(x − x0) es pequeño, el error directo x − x0 ≡ 1/a · a(x − x0) puede ser grande dado el factor 1/a .

Definimos el número de condición como una medida de la sensibilidad de un problema:

Definición 1.4 (Número de condición). El número de condición de un problema es la relación entre cuánto cambia su solución y cuánto cambia su enunciado bajo pequeñas perturbaciones. Alternativamente, es la relación entre el error hacia adelante y hacia atrás para pequeños cambios en el enunciado del problema.

Ejemplo 1.6 (ax = b, segunda parte). Continuando con el Ejemplo 1.5, podemos calcular el número de condición exactamente:

$$c = \frac{\text{error hacia adelante}}{\text{error hacia atrás}} = \frac{x - x0}{a(x - x0)} \equiv \frac{1}{a}$$

En general, calcular los números de condición es casi tan difícil como calcular el error de reenvío y, por lo tanto, es probable que su cálculo exacto sea imposible. Aun así, muchas veces es posible encontrar límites o aproximaciones para los números de condición para ayudar a evaluar cuánto se puede confiar en una solución.

Ejemplo 1.7 (Búsqueda de raíces). Supongamos que tenemos una función suave $f : R \to R$ y queremos encontrar valores de x con $f(x) = 0$. Observa que $f(x + \Delta) \approx f(x) + \Delta f(x)$. Por lo tanto, una aproximación del número de condición para encontrar x podría ser

$$\frac{\text{cambio en el error hacia adelante}}{\text{cambio en el error hacia atrás}} = \frac{(x + \Delta) - x}{f(x + \Delta) - f(x)} \approx \frac{\Delta}{\Delta f(x)} = \frac{1}{f(x)}$$

Observe que esta aproximación se alinea con la del ejemplo 1.6. Por supuesto, si no conocemos x no podemos evaluar f (x), pero si podemos mirar la forma de f y acotar | f | cerca de x, tenemos una idea de la peor situación posible.

El error hacia adelante y hacia atrás son medidas de la precisión de una solución. En aras de la repetibilidad científica, también deseamos derivar algoritmos estables que produzcan soluciones autoconsistentes para una clase de problemas. Por ejemplo, es posible que no valga la pena implementar un algoritmo que genera soluciones muy precisas solo una quinta parte del tiempo, incluso si podemos volver a usar las técnicas anteriores para verificar si la solución candidata es buena.

1.3 Aspectos prácticos

La infinitud y la densidad de los números reales R pueden causar errores perniciosos al implementar algoritmos numéricos. Si bien la teoría del análisis de errores presentada en §1.2 eventualmente nos ayudará a garantizar la calidad de las técnicas numéricas presentadas en capítulos futuros, vale la pena señalar antes de continuar una serie de errores comunes y "trampas" que impregnan las implementaciones de métodos numéricos.

Introdujimos a propósito el mayor infractor al principio de §1.1, que repetimos en una fuente más grande para enfatizar

bien merecido: Rara vez, si es que alguna vez , se debe usar el operador == y sus equivalentes en valores fraccionarios.

Encontrar un reemplazo adecuado para == y las condiciones correspondientes para terminar un método numérico depende de la técnica en consideración. El ejemplo 1.3 muestra que un método para resolver Ax = b puede terminar cuando el residuob − Ax es cero; dado que no queremos verificar si A*x==b explícitamente, en la práctica las implementaciones verificarán norm(A*xb)<epsilon. Tenga en cuenta que este ejemplo demuestra dos técnicas:

- El uso del error hacia atrásb − Ax en lugar del error hacia adelante para determinar cuándo terminar, y
- Verificar si el error hacia atrás es menor que épsilon para evitar el predicado ==0 prohibido.

El parámetro épsilon depende de cuán precisa debe ser la solución deseada, así como de la resolución del sistema numérico en uso.

Un programador que utilice estos tipos de datos y operaciones debe estar atento a la hora de detectar y prevenir operaciones numéricas deficientes. Por ejemplo, considere el siguiente fragmento de código para calcular la norma x2 para un vector x Rn representado como una matriz 1D x[]:

```
doble norma al cuadrado = 0; for
( int i = 0; i < n ; i ++) normSquared += x
                [ i ]* x [ i ];
volver sqrt (normSquared);
```

Es fácil ver que en teoría mini |xi | ≤ x2/ √ n ≤ maxi |xi |, es decir, la norma de x es del orden de los valores de los elementos contenidos en x. Sin embargo, oculta en el cálculo de x2 está la expresión x[i]*x[i]. Si existe i tal que x[i] es del orden de DOUBLE MAX, el producto x[i]*x[i] se desbordará aunque x2 todavía esté dentro del rango de los dobles. Tal desbordamiento se puede prevenir fácilmente dividiendo x por su valor máximo, calculando la norma y multiplicando:

```
maxElement doble = épsilon; // ¡no quiero dividir por cero! para ( int i = 0; i < n ; i ++)

                maxElement = max ( maxElement for ( int    ,    fab ( x [ i ]));
i = 0; i < n ; i ++) {
                doble escala = x [ i ] / maxElement ; normSquared +=
                escalado * escalado ;
}
return sqrt (normSquared) * maxElement;
```

El factor de escala elimina el problema de desbordamiento al asegurarse de que los elementos que se suman no sean mayores que 1.

Este pequeño ejemplo muestra una de las muchas circunstancias en las que un solo carácter de código puede conducir a un problema numérico no obvio. Si bien nuestra intuición de las matemáticas continuas es suficiente para generar muchos métodos numéricos, siempre debemos verificar dos veces que las operaciones que empleamos sean válidas desde un punto de vista discreto.

1.3.1 Ejemplo a mayor escala: sumatoria

Ahora proporcionamos un ejemplo de un problema numérico causado por la aritmética de precisión finita que se puede resolver usando un truco algorítmico menos que obvio.

Supongamos que deseamos sumar una lista de valores de punto flotante, fácilmente una tarea requerida por los sistemas de contabilidad, aprendizaje automático, gráficos y casi cualquier otro campo. Un fragmento de código para realizar esta tarea que sin duda aparece en innumerables aplicaciones tiene el siguiente aspecto:

```
doble suma = 0; for ( int
i = 0; i < n ; i ++) sum += x [ i ];
```

Antes de continuar, vale la pena señalar que para la gran mayoría de las aplicaciones, esta es una técnica perfectamente estable y ciertamente matemáticamente válida.

Pero, ¿qué puede salir mal? Considere el caso donde n es grande y la mayoría de los valores x[i] son pequeños y positivos. En este caso, cuando i es lo suficientemente grande, la variable suma será grande en relación con x[i]. Eventualmente, sum puede ser tan grande que x[i] afecta solo a los bits de sum de orden más bajo y, en el caso extremo, sum puede ser lo suficientemente grande como para que sumar x[i] no tenga ningún efecto. Si bien un solo error de este tipo podría no ser un gran problema, el efecto acumulado de cometer este error repetidamente podría abrumar la cantidad en la que podemos confiar.

Para comprender matemáticamente este efecto, suponga que calcular una suma a + b puede estar errado tanto como ε > 0. Entonces, el método anterior claramente puede inducir un error del orden de nε, que crece linealmente con n. De hecho, si la mayoría de los elementos x[i] son del orden de ε, ¡entonces no se puede confiar en la suma! Este es un resultado decepcionante: el error puede ser tan grande como la suma misma.

Afortunadamente, hay muchas maneras de hacerlo mejor. Por ejemplo, agregar primero los valores más pequeños podría ayudar a explicar su efecto acumulado. Los métodos por pares que agregan recursivamente pares de valores de x[] y construyen una suma también son más estables, pero pueden ser difíciles de implementar de manera tan eficiente como el bucle for anterior. Afortunadamente, un algoritmo de Kahan (CITE) proporciona un método de "suma compensada" de fácil implementación que es casi igual de rápido.

La observación útil aquí es que en realidad podemos realizar un seguimiento de una aproximación del error en suma durante una iteración dada. En particular, considere la expresión

$$((a + b) - a) - b.$$

Obviamente esta expresión algebraicamente es cero. Numéricamente, sin embargo, este puede no ser el caso. En particular, la suma (a + b) puede redondear el resultado para mantenerlo dentro del ámbito de los valores de punto flotante. Restar a y b uno a la vez produce una aproximación del error inducido por esta operación; observe que las operaciones de resta probablemente estén mejor condicionadas ya que pasar de números grandes a pequeños agrega dígitos de precisión debido a la cancelación.

Así, la técnica de Kahan procede de la siguiente manera:

```
doble suma = 0;
compensación doble = 0; // una aproximación del error

for (int i = 0; i < n; i ++) { // intenta volver a sumar
                tanto x [ i ] como la parte que falta double nextTerm = x [ i ] + compensación ;


                // calcula el resultado de la suma de esta iteración double nextSum = sum +
                nextTerm ;

                // calcula la compensación como la diferencia entre el término que deseas // agregar y el resultado real compensación =
                nextTerm - ( nextSum - sum );


                suma = sumasiguiente;
}
```

En lugar de simplemente mantener la suma, ahora realizamos un seguimiento de la suma y una compensación aproximada de la diferencia entre la suma y el valor deseado. Durante cada iteración, intentamos volver a agregar

esta compensación además del elemento actual de x[], y luego volvemos a calcular la compensación para dar cuenta del último error.

Analizar el algoritmo de Kahan requiere una contabilidad más cuidadosa que analizar la técnica incremental más simple. Verá una derivación de una expresión de error al final de este capítulo; el resultado matemático final será que el error mejora de nε a O(ε + nε mejora considerable cuando 0 < ε ^{2}), a 1.

La implementación de la suma de Kahan es sencilla, pero duplica con creces el número de operaciones del programa resultante. De esta forma, existe un equilibrio implícito entre velocidad y precisión que los ingenieros de software deben realizar al decidir qué técnica es la más adecuada.

En términos más generales, el algoritmo de Kahan es uno de varios métodos que eluden la acumulación de errores numéricos durante el curso de un cálculo que consta de más de una operación. Otros ejemplos incluyen el algoritmo de Bresenham para rasterizar líneas (CITE), que usa solo aritmética de números enteros para dibujar líneas, incluso cuando intersecan filas y columnas de píxeles en ubicaciones no integrales, y Fast Fourier Transform (CITE), que usa efectivamente la partición binaria truco de suma descrito anteriormente.

1.4 Problemas

Problema 1.1. Aquí hay un problema.

Parte II

Álgebra lineal

Capitulo 2

Sistemas Lineales y la LU Descomposición

En el Capítulo 0, analizamos una variedad de situaciones en las que los sistemas lineales de ecuaciones Ax = b aparecen en la teoría matemática y en la práctica. En este capítulo, abordamos el problema básico de frente y exploramos métodos numéricos para resolver tales sistemas.

2.1 Solubilidad de Sistemas Lineales

Como se introdujo en §0.3.3, los sistemas de ecuaciones lineales como

$$3x + 2y = 6$$
$$-4x + y = 7$$

puede escribirse en forma matricial como en

$$\begin{bmatrix} 3 & 2 \\ -4 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 6 \\ 7 \end{bmatrix}.$$

Más generalmente, podemos escribir sistemas de la forma Ax =b para $A \in \mathbb{R}^{m\times n}$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$.

La solucionabilidad del sistema debe caer en uno de tres casos:

1. El sistema puede no admitir soluciones, como en:

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

   Este sistema pide que x = −1 yx = 1 simultáneamente, obviamente dos condiciones incompatibles.

2. El sistema podrá admitir una única solución; por ejemplo, el sistema al comienzo de esta sección se resuelve mediante (x, y) = (−8/11, 45/11).

3. El sistema puede admitir infinitas soluciones, por ejemplo, 0x =0. Nótese que si un sistema Ax =b admite dos soluciones x0 y x1, automáticamente tiene infinitas soluciones de la forma cx0 + (1 − c)x1 para c R, ya que

$$A(cx0 + (1 - c)x1) = cAx0 + (1 - c)Ax1 = cb + (1 - c)b = b.$$

Este sistema lineal se etiquetaría como subdeterminado.

En general, la solucionabilidad de un sistema depende tanto de A como de onb. Por ejemplo, si modificamos el sistema irresoluble anterior para que sea

$$\begin{matrix} 1 & 0 \\ 1 & 0 \end{matrix} \begin{matrix} x \\ y \end{matrix} = \begin{matrix} 1 \\ 1 \end{matrix},$$

entonces el sistema pasa de no tener soluciones a infinitas de la forma (1, y). De hecho, toda matriz A admite un lado derecho b tal que Ax = b es resoluble, ya que Ax = 0 siempre se puede resolver por x ≡ 0 independientemente de A. Recuerde de §0.3.1 que la multiplicación matriz-vector se puede ver como combinando linealmente las columnas de A con pesos de x. Por lo tanto, como se mencionó en §0.3.3, podemos esperar que Ax =b se pueda resolver exactamente cuando b está en el espacio columna de A.

En un sentido amplio, la “forma” de la matriz A Rm×n tiene una influencia considerable en la resolución de Ax =b. Recuerde que las columnas de A son vectores m-dimensionales. Primero, considere el caso cuando A es "ancho", es decir, cuando tiene más columnas que filas (n > m). Cada columna es un vector en Rm, por lo que como máximo el espacio de la columna puede tener una dimensión m. Como n > m, las n columnas de A deben ser linealmente dependientes; esto implica que existe un x0 =0 tal que Ax0 =0. Entonces, si podemos resolver Ax = b para x, entonces A(x + αx0) = Ax + αAx0 = b +0 = b, mostrando que en realidad hay infinitas soluciones x para Ax = b. En otras palabras, hemos demostrado que ningún sistema matricial ancho admite una solución única.

Cuando A es "alto", es decir, cuando tiene más filas que columnas (m > n), entonces las n columnas no pueden abarcar Rm. Entonces, existe algún vectorb0 Rm\col A. Por definición, este b0 no puede satisfacer Ax =b0 para cualquier x. En otras palabras, toda matriz alta A admite sistemas Ax =b0 que no son solucionables.

Las dos situaciones anteriores están lejos de ser favorables para diseñar algoritmos numéricos. Por ejemplo, si un sistema lineal admite muchas soluciones, primero debemos definir qué solución desea el usuario: después de todo, la solución x + 1031x0 podría no ser tan significativa como x − 0.1x0. Por otro lado, en el caso alto, incluso si Ax = b es solucionable para un b particular, cualquier pequeña perturbación Ax =b + εb0 ya no es solucionable; esta situación puede aparecer simplemente porque los procedimientos de redondeo discutidos en el último capítulo solo pueden aproximar A yb en primer lugar.

Dadas estas complicaciones, en este capítulo haremos algunas suposiciones simplificadoras: •

Consideraremos solo el cuadrado A Rn×n .

• Supondremos que A es no singular, es decir, que Ax =b es solucionable para cualquier b.

Recuerde de §0.3.3 que la condición de no singularidad es equivalente a pedir que las columnas de A abarcan Rn e implica la existencia de una matriz A^{-1} satisfaciendo $A^{-1}A = AA^{-1} = I_{n\times n}$.

Una observación engañosa es pensar que resolver Ax =b es equivalente a calcular la matriz explícitamente y A^{-1} luego multiplicar para encontrar x ≡ A −1b. Si bien esta estrategia de solución ciertamente es válida, puede representar una cantidad considerable de exageraciones: después de todo, solo estamos interesados en los valores n 2 en x en lugar de en n . Además, incluso cuando A se porta bien, puede darse el caso de que escribir A^{-1} produce dificultades numéricas que se pueden eludir.

2.2 Estrategias de solución ad-hoc

En la introducción al álgebra, a menudo abordamos el problema de resolver un sistema lineal de ecuaciones como una forma de arte. La estrategia es "aislar" las variables, escribiendo iterativamente formas alternativas del sistema lineal hasta que cada línea tenga la forma x = const.

Al formular algoritmos sistemáticos para resolver sistemas lineales, es instructivo llevar a cabo un ejemplo de este proceso de solución. Considere el siguiente sistema:

$$\begin{aligned} y - z &= -1 \\ 3x - y + z &= 4 \\ x + y - 2z &= -3 \end{aligned}$$

Paralelamente, podemos mantener una versión matricial de este sistema. En lugar de escribir Ax = b explícitamente, podemos ahorrar un poco de espacio escribiendo la siguiente matriz "aumentada":

$$\left[\begin{array}{cccc} 0 & 1 & -1 & -1 \\ 3 & -1 & 1 & 4 \\ 1 & 1 & -2 & -3 \end{array}\right]$$

Siempre podemos escribir sistemas lineales de esta manera siempre que estemos de acuerdo en que las variables permanezcan en el lado izquierdo de las ecuaciones y las constantes en el lado derecho.

Tal vez deseemos tratar primero con la variable x. Por conveniencia, podemos permutar las filas del sistema para que la tercera ecuación aparezca primero:

$$\begin{aligned} x + y - 2z &= -3 \\ y - z &= -1 \\ 3x - y + z &= 4 \end{aligned} \qquad \left[\begin{array}{cccc} 1 & 1 & -2 & -3 \\ 0 & 1 & -1 & -1 \\ 3 & -1 & 1 & 4 \end{array}\right]$$

Entonces podemos sustituir la primera ecuación en la tercera para eliminar el término 3x. Esto es lo mismo que escalar la relación x + y − 2z = −3 por −3 y sumar el resultado a la tercera ecuación:

$$\begin{aligned} x + y - 2z &= -3 \\ y - z &= -1 \\ -4y + 7z &= 13 \end{aligned} \qquad \left[\begin{array}{cccc} 1 & 1 & -2 & -3 \\ 0 & 1 & -1 & -1 \\ 0 & -4 & 7 & 13 \end{array}\right]$$

Del mismo modo, para eliminar y de la tercera ecuación podemos multiplicar la segunda ecuación por 4 y sumar el resultado a la tercera:

$$\begin{aligned} x + y - 2z &= -3 \\ y - z &= -1 \\ 3z &= 9 \end{aligned} \qquad \left[\begin{array}{cccc} 1 & 1 & -2 & -3 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 3 & 9 \end{array}\right]$$

Ahora hemos aislado z! Por lo tanto, podemos escalar la tercera fila en 1/3 para generar una expresión para z:

$$\begin{aligned} x + y - 2z &= -3 \\ y - z &= -1 \\ z &= 3 \end{aligned} \qquad \left[\begin{array}{cccc} 1 & 1 & -2 & -3 \\ 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & 3 \end{array}\right]$$

Ahora, podemos sustituir z = 3 en las otras dos ecuaciones para eliminar z de todas las filas menos de la última:

$$\begin{array}{l} x + y = 3 \\ y = 2 \\ z = 3 \end{array} \qquad \left|\begin{array}{cccc} 1 & 1 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array}\right.$$

Finalmente hacemos una sustitución similar para y para completar la solución:

$$\begin{array}{l} x = 1 \\ y = 2 \\ z = 3 \end{array} \qquad \left|\begin{array}{cccc} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{array}\right.$$

Este ejemplo puede ser algo pedante, pero mirar hacia atrás en nuestra estrategia arroja algunas observaciones importantes sobre cómo podemos resolver sistemas lineales:

- Escribimos sistemas sucesivos $A_i x = b_i$ que pueden verse como simplificaciones del original $Ax = b$.
- Resolvimos el sistema sin escribir A^{-1}.
- Repetidamente usamos algunas operaciones simples: escalar, agregar y permutar filas del sistema.
- Se aplicaron las mismas operaciones a A yb. Si escalamos la fila k-ésima de A, también escalamos la k-ésima fila de b. Si sumamos las filas k y de A, sumamos las filas k y ofb.
- Menos obviamente, los pasos de la solución no dependían de b. Es decir, todas nuestras decisiones sobre cómo resolver fueron motivadas por la eliminación de los valores distintos de cero en A en lugar de examinar los valores enb;b simplemente apareció por el camino.
- Terminamos cuando redujimos el sistema a $I_n \times n x = b$.

Usaremos todas estas observaciones generales sobre la resolución de sistemas lineales para nuestro beneficio.

2.3 Codificación de operaciones de fila

Volviendo al ejemplo de §2.2, vemos que resolver el sistema lineal en realidad solo implicó aplicar tres operaciones: permutación, escalado de filas y sumar la escala de una fila a otra.
De hecho, podemos resolver cualquier sistema lineal de esta manera, por lo que vale la pena explorar estas operaciones con más detalle.

2.3.1 Permutación

Nuestro primer paso en §2.2 fue intercambiar dos de las filas en el sistema de ecuaciones. De manera más general, podríamos indexar las filas de matrices usando los números 1, . . . , m. Entonces, una permutación de esas filas se puede escribir como una función σ tal que la lista σ(1), . . . , σ(m) cubre el mismo conjunto de índices.

Si ek es la k-ésima función de base estándar, entonces es fácil ver que el producto e $_k$ A produce la fila k-ésima de la matriz A. Por lo tanto, podemos "apilar" o concatenar estos vectores de fila verticalmente para producir una matriz que permuta las filas de acuerdo con σ :

$$P\sigma \equiv \begin{matrix} - & e_{\sigma(1)} & - \\ - & e_{\sigma(2)} & - \\ & \vdots & \\ - & e_{\sigma(m)} & - \end{matrix}$$

Es decir, el producto PσA es exactamente la matriz A con filas permutadas según σ.

Ejemplo 2.1 (Matrices de permutación). Supongamos que deseamos permutar filas de una matriz en R3×3 con σ(1) = 2, σ(2) = 3 y σ(3) = 1. Según nuestra fórmula tendríamos

$$P\sigma = \begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{matrix}$$

Del Ejemplo 2.1, podemos ver que Pσ tiene unos en posiciones de la forma (k, σ(k)) y ceros en cualquier otro lugar. El par (k, σ(k)) representa la afirmación: “Nos gustaría que la fila k de la matriz de salida fuera la fila σ(k) de la matriz de entrada”. Con base en esta descripción de una matriz de permutación, es fácil ver que la inversa de Pσ es la transpuesta P, ya que esto simplemente intercambia los roles de las filas y las columnas; ahora tomamos la fila σ(k) de la entrada y la colocamos en fila k de la salida. En otras palabras, p.
$_\sigma$ Pσ = Im×m.

2.3.2 Escalado de filas

Supongamos que escribimos una lista de constantes a1, . . . , am y busca escalar la fila k-ésima de alguna matriz A por ak . Obviamente, esto se logra aplicando la matriz de escala Sa dada por:

$$S_a \equiv \begin{matrix} a_1 & 0 & 0 & \cdots & 0 \\ 0 & a_2 & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & & a_m \end{matrix}$$

Suponiendo que todas las ak satisfacen ak = 0, es fácil invertir Sa “reduciendo la escala:”

$$S_a^{-1} = S_{1/a} \equiv \begin{matrix} 1/a_1 & 0 & 0 & \cdots & 1/a_2 & 0 & \cdots \\ 0 & & & & & & \\ \vdots & \vdots & \vdots & \ddots & \vdots & & \\ 0 & 0 & \cdots & 1/a_m & & & \end{matrix}$$

2.3.3 Eliminación

Finalmente, supongamos que deseamos escalar la fila k por una constante c y agregar el resultado a la fila . Esta operación puede parecer menos natural que las dos anteriores pero en realidad es bastante práctica: es la única que

¡Necesitas combinar ecuaciones de diferentes filas del sistema lineal! Realizaremos esta operación utilizando una “matriz de eliminación” M tal que el producto MA aplique esta operación a la matriz A.

Recuerde que el producto $e_k A$ elige la k-ésima fila de A. Luego, premultiplicar por e produce un matriz k A, que es cero excepto que la -ésima fila es igual a la k-ésima de A.

Ejemplo 2.2 (Construcción de matriz de eliminación). Llevar

$$A = \begin{matrix} 1\,2\,3 \\ 4\,5\,6 \\ 7\,8\,9 \end{matrix}$$

Supongamos que deseamos aislar la tercera fila de $A \in R^{3\times 3}$ y moverla a la fila dos. Como se discutió anteriormente, esta operación se logra escribiendo:

$$e_2e_3 A = \begin{matrix} 0 & & 1\,2\,3\,4 \\ 1 & 0\,0\,1 & 5\,6 \\ 0 & & 7\,8\,9 \end{matrix}$$

$$= \begin{matrix} 0 & \\ 1 & 7\,8\,9 \\ 0 & \end{matrix}$$

$$= \begin{matrix} 0\,0\,0 \\ 7\,8\,9 \\ 0\,0\,0 \end{matrix}$$

Por supuesto, multiplicamos arriba de derecha a izquierda, pero con la misma facilidad podríamos haber agrupado el producto como $(e_2e_3)A$. La estructura de este producto es fácil de ver:

$$e_2e_3 = \begin{matrix} 0 & & 0\,0\,0\,0 \\ 1 & 0\,0\,1 = & 0\,1 \\ 0 & & 0\,0\,0 \end{matrix}$$

Hemos logrado aislar la fila k y moverla a la fila . Nuestra operación de eliminación original quería sumar c veces el renglón k al renglón $A = k(I_{n\times n} + ce$, que ahora podemos lograr como la suma $A + ce_k)A$.

Ejemplo 2.3 (Resolución de un sistema). Ahora podemos codificar cada una de nuestras operaciones de la Sección 2.2 utilizando las matrices que hemos construido anteriormente:

1. Permuta las filas para mover la tercera ecuación a la primera fila:

$$PAG = \begin{matrix} 0\,0\,1 \\ 1\,0\,0 \\ 0\,1\,0 \end{matrix}$$

2. Escale la fila uno por -3 y agregue el resultado a la fila tres: $E_1 = I_{3\times 3} - 3e_{3e_1}$

3. Escale la fila dos por 4 y agregue el resultado a la fila tres: $E_2 = I_{3\times 3} + 4e_{3e_2}$

4. Escale la fila tres en 1/3: S = diag(1, 1, 1/3)

5. Escale la fila tres por 2 y agréguela a la fila uno: $E_3 = I_{3\times 3} + 2e_{1e_{13}}$

6. Agregue la fila tres a la fila dos: $E_4 = I_{3\times 3} + e_{2e_{23}}$

7. Escale la fila tres por -1 y agregue el resultado a la fila uno: $E_5 = I_{3\times 3} - e_{1e_{13}}$

Por lo tanto, la inversa de A en la Sección 2.2 satisface

$$A^{-1} = E_5E_4E_3SE_2E_1P.$$

¡Asegúrate de entender por qué estas matrices aparecen en orden inverso!

2.4 Eliminación Gaussiana

La secuencia de pasos elegida en la Sección 2.2 no fue única: hay muchos caminos diferentes que pueden conducir a la solución de Ax =b. Nuestros pasos, sin embargo, siguieron la estrategia de eliminación gaussiana, un famoso algoritmo para resolver sistemas de ecuaciones lineales.

En términos más generales, digamos que nuestro sistema tiene la siguiente "forma":

$$\text{un segundo} = \left|\begin{matrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{matrix}\right.$$

El algoritmo procede en las fases que se describen a continuación.

2.4.1 Sustitución hacia adelante

Considere el elemento superior izquierdo de nuestra matriz:

$$\text{un segundo} = \left|\begin{matrix} \textcircled{\times} & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{matrix}\right.$$

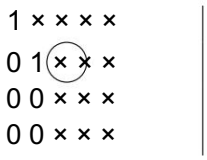
Llamaremos a este elemento nuestro primer pivote y supondremos que es distinto de cero; si es cero podemos permutar filas para que no sea así. Primero aplicamos una matriz de escala para que el pivote sea igual a uno:

$$\left|\begin{matrix} \textcircled{1} & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{matrix}\right.$$

Ahora, usamos la fila que contiene el pivote para eliminar todos los demás valores debajo de la misma columna:

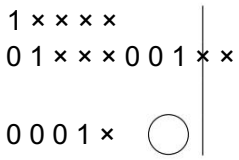
$$\left|\begin{matrix} \textcircled{1} & \times & \times & \times & 0 & \times & \times & \times \\ \times & 0 & \times & \times & \times & 0 & \times & \times \\ \times & \times & & & & & & \end{matrix}\right.$$

Ahora movemos nuestro pivote a la siguiente fila y repetimos una serie similar de operaciones:



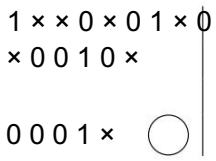
Note que algo bueno sucede aquí. Después de eliminar el primer pivote de todas las demás filas, la primera columna es cero debajo de la fila 1. Esto significa que podemos agregar con seguridad múltiplos de la fila dos a las filas de abajo sin afectar los ceros en la columna uno.

Repetimos este proceso hasta que la matriz se vuelve triangular superior:

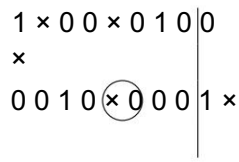


2.4.2 Sustitución hacia atrás

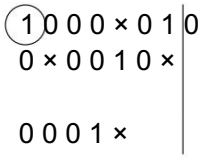
La eliminación de las × restantes del sistema ahora es un proceso sencillo. Ahora, procedemos en orden inverso de filas y eliminamos hacia atrás. Por ejemplo, después de la primera serie de pasos de sustitución hacia atrás, nos queda la siguiente forma:



De manera similar, la segunda iteración produce:



Después de nuestro paso final de eliminación, nos quedamos con nuestra forma deseada:



El lado derecho ahora es la solución al sistema lineal Ax =b.

2.4.3 Análisis de Eliminación Gaussiana

Cada operación de fila en la eliminación gaussiana (escalado, eliminación e intercambio de dos filas) obviamente toma O (n) tiempo para completarse, ya que debe iterar sobre los n elementos de una fila (o

dos) de A. Una vez que elegimos un pivote, tenemos que hacer n sustituciones hacia adelante o hacia atrás en por debajo o por encima de ese pivote, respectivamente; esto significa que el trabajo para un solo pivote en total es $O(n^2$ las filas). En total, elegimos un pivote por fila, agregando un factor final de n. Por lo tanto, es bastante fácil ver que Gaussian carreras de eliminación en $O(n^3)$ tiempo.

Una decisión que tiene lugar durante la eliminación gaussiana que no hemos discutido es la elección de los pivotes. Recuerde que podemos permutar filas del sistema lineal como mejor nos parezca antes de realizar una sustitución hacia atrás o hacia adelante. Esta operación es necesaria para poder tratar con todas las matrices A posibles. Por ejemplo, considere lo que sucedería si no usáramos el pivote en las siguientes matriz:

$$A = \begin{pmatrix} \textcircled{0} & 1 & 1 \\ & & 0 \end{pmatrix}$$

Observe que el elemento encerrado en un círculo es exactamente cero, por lo que no podemos esperar dividir la fila uno por ningún número para reemplazar ese 0 con un 1. Esto no significa que el sistema no se pueda resolver, solo significa que debemos pivotar, lo que se logra intercambiando el filas primera y segunda, para poner un valor distinto de cero en esa ranura.

De manera más general, supongamos que nuestra matriz se ve así:

$$A = \begin{pmatrix} \textcircled{\varepsilon} & 1 \\ 1 & 0 \end{pmatrix},$$

donde $0 < \varepsilon$ 1. Si no pivotamos, entonces la primera iteración de la eliminación gaussiana produce:

$$\tilde{A} = \begin{pmatrix} \textcircled{1} & 1/\varepsilon \\ & -1/\varepsilon \end{pmatrix} 0,$$

Hemos transformado una matriz A que parece casi una matriz de permutación (¡de hecho, una $A^{-1} \approx$, a forma muy fácil de resolver el sistema!) en un sistema con valores potencialmente enormes de $1/\varepsilon$.

Este ejemplo muestra que hay casos en los que podemos desear pivotar incluso cuando hacerlo estrictamente hablando no es necesario. Dado que estamos escalando por el recíproco del valor del pivote, claramente la opción más estable numéricamente es tener un pivote grande: los pivotes pequeños tienen recíprocos grandes, escalando números a valores grandes en regímenes que probablemente pierdan precisión. Hay dos estrategias pivotantes bien conocidas:

1. La rotación parcial mira a través de la columna actual y permuta las filas de la matriz para que el valor absoluto más grande aparece en la diagonal.

2. El pivote completo itera sobre toda la matriz y permuta filas y columnas para obtener el mayor valor posible en la diagonal. Nótese que permutar columnas de una matriz es una operación válida: corresponde a cambiar el etiquetado de las variables en el sistema, o post-multiplicar A por una permutación.

Ejemplo 2.4 (Pivote). Supongamos que después de la primera iteración de la eliminación gaussiana nos queda la siguiente matriz:

$$\begin{pmatrix} 1 & 10 & -10 \\ 0 & 0.1 & 9 \\ 0 & 4 & 6.2 \end{pmatrix}$$

Si implementamos un pivote parcial, buscaremos solo en la segunda columna e intercambiaremos la segunda y la tercera fila; fíjate que dejamos el 10 en la primera fila ya que ese ya ha sido visitado por el algoritmo:

1 10 −10
0 4 6.2
0 0.1 9

Si implementamos el pivoteo completo, moveremos el 9:

1 −10 10
0 9 0.1
0 6.2 4

Obviamente, el pivoteo completo produce los mejores valores numéricos posibles, pero el costo es una búsqueda más costosa de elementos grandes en la matriz.

2.5 Factorización LU

Hay muchas ocasiones en las que deseamos resolver una secuencia de problemas Ax1 =b1, Ax2 =b2, ... Como ya hemos discutido, los pasos de la eliminación gaussiana para resolver Ax = bk dependen principalmente de la estructura de A más que de los valores en un bk particular . Dado que A se mantiene constante aquí, es posible que deseemos "recordar" los pasos que tomamos para resolver el sistema para que cada vez que se nos presente una nueva b no tengamos que comenzar desde cero.

Solidificando esta sospecha de que podemos mover algunos de los $O(n^3)$ para la eliminación gaussiana en el tiempo de precálculo, recuerde el sistema triangular superior resultante después de la etapa de sustitución hacia adelante:

$$\left[\begin{array}{cccc|c} 1 & \times & \times & \times & \times \\ 0 & 1 & \times & \times & \times \\ 0 & 0 & 1 & \times & \times \\ 0 & 0 & 0 & 1 & \times \end{array}\right]$$

De hecho, resolver este sistema por sustitución hacia atrás solo toma $O(n^2)$ tiempo! ¿Por qué? Sustitución hacia atrás en este caso es mucho más fácil gracias a la estructura de los ceros en el sistema. Por ejemplo, en la primera serie de sustituciones hacia atrás obtenemos la siguiente matriz:

$$\left[\begin{array}{cccc|c} 1 & \times & 0 & \times & \times \\ 0 & \times & 1 & \times & \times \\ 0 & 0 & 0 & 1 & \times \end{array}\right]$$

Como sabemos que los valores (encerrados en un círculo) a la izquierda del pivote son cero por construcción, no necesitamos copiarlos explícitamente. Por lo tanto, este paso solo tomó un tiempo O (n) en lugar de $O(n^2)$ tomado por sustitución hacia adelante.

Ahora, nuestro próximo pivote hace una sustitución similar:

$$\left[\begin{array}{cccc|c} 1 & \times & 0 & 0 & \times \\ 0 & 1 & 0 & 0 & \times \\ 0 & 0 & 1 & 0 & \times \\ 0 & 0 & 0 & 1 & \times \end{array}\right]$$

Una vez más, los ceros a ambos lados del 1 no necesitan copiarse explícitamente.

Así, hemos encontrado:

Observación. Mientras que la eliminación gaussiana toma $O(n^3)$ tiempo, resolver sistemas triangulares toma $O(n^2)$ tiempo.

2.5.1 Construcción de la factorización

Recuerde de §2.3 que todas las operaciones en la eliminación gaussiana se pueden considerar como una multiplicación previa de Ax = b por diferentes matrices M para obtener un sistema más fácil (MA)x = Mb. Como demostramos en el ejemplo 2.3, desde este punto de vista, cada paso de la eliminación gaussiana representa un sistema (Mk · · · M2M1A)x = Mk · · · M2M1 b . Por supuesto, almacenar explícitamente estas matrices Mk como n × n objetos es una exageración, pero tener en cuenta esta interpretación desde una perspectiva teórica simplifica muchos de nuestros cálculos.

Después de la fase de sustitución directa de la eliminación gaussiana, nos queda una matriz triangular superior, que podemos llamar U Rn×n . Desde la perspectiva de la multiplicación de matrices, podemos escribir:

$$M_k \cdots M_1 A = U,$$

o equivalente,

$$\begin{aligned} A &= (M_k \cdots M_1)^{-1} U \\ &= (M_1^{-1} M_2^{-1} \cdots M_k^{-1}) U \\ &\equiv LU, \text{ si hacemos la definición } L \equiv M_1^{-1} M_2^{-1} \cdots M_k^{-1} \end{aligned}$$

Todavía no sabemos nada sobre la estructura de L, pero sabemos que los sistemas de la forma Uy = d son más fáciles de resolver ya que U es triangular superior. Si L es igualmente agradable, podríamos resolver Ax =b en dos pasos, escribiendo (LU)x =b, o x = U−1L −1b: 1. Resuelva Ly =b para y, obteniendo y

= L −1b.

2. Ahora que tenemos y, resuelva Ux = y, obteniendo x = U−1y = U−1 (L −1b) = (LU) −1b = A −1b.
Ya sabemos que este paso solo toma O(n ^{2}) tiempo.

Nuestra tarea restante es asegurarnos de que L tenga una buena estructura que haga que resolver Ly = b sea más fácil que resolver Ax = b. Afortunadamente, y como era de esperar, encontraremos que L es triangular inferior y, por lo tanto, se puede resolver usando O (n ^{2}) sustitución hacia adelante.

Para ver esto, supongamos por ahora que no implementamos el pivoteo. Entonces, cada una de nuestras matrices Mk es una matriz de escala o tiene la estructura

$$M_k = I_{n\times n} + c e_{\_} e_k^T,$$

donde > k ya que solo hemos realizado sustitución hacia adelante. Recuerde que esta matriz tiene un propósito específico: escalar la fila k por c y sumar el resultado a la fila. Obviamente, esta operación es fácil de deshacer: Escale la fila k por c y reste el resultado de la fila. Podemos comprobar esto formalmente:

$$\begin{aligned} (I_{n\times n} + c e_{\_} e_k^T)(I_{n\times n} - c e_{\_} e_k^T) &= I_{n\times n} + (-c e_{\_} e_k^T + c e_{\_} e_k^T) - c^2 e_{\_} (e_k^T e_{\_}) e_k^T \\ &= I_{n\times n} - c^2 e_{\_} (0) e_k^T \\ &= I_{n\times n} \text{ desde } e_k^T \cdot e_{\_} = 0, \text{ y } k = \end{aligned}$$

Entonces, la matriz L es el producto de matrices de escala y matrices de la forma $M_{-1} = I_{n\times n} + cee_k$ son triangular inferior cuando > k. Las matrices de escala son diagonales y la matriz M es triangular inferior. Mostrará en el ejercicio 2.1 que el producto de matrices triangulares inferiores es triangular inferior, demostrando a su vez que L es triangular inferior según sea necesario.

Hemos demostrado que si es posible llevar a cabo la eliminación gaussiana de A sin usar pivote, puede factorizar A = LU en el producto de matrices triangulares inferior y superior. Cada sustitución hacia adelante y hacia atrás toma un tiempo $O(n^2)$, por lo que si esta factorización se puede calcular con anticipación, la solución lineal se puede llevar a cabo más rápido que la $O(n^3)$ Eliminación gaussiana. Mostrarás en completa . Ejercicio 2.2 ¿Qué sucede cuando realizamos LU con pivote; sin cambios importantes? son necesarios.

2.5.2 Implementación de LU

Una implementación simple de la eliminación gaussiana para resolver Ax = b es bastante sencilla de formular. En particular, como hemos discutido anteriormente, podemos formar la matriz aumentada (A |b) y aplicar operaciones de fila una a la vez a este bloque n × (n + 1) hasta que se vea como $(I_{n\times n}A^{-1}|b)$.
Este proceso, sin embargo, es destructivo, es decir, al final solo nos importa la última columna de la matriz aumentada y no hemos guardado evidencia de nuestra ruta de solución. Tal comportamiento claramente no es aceptable para la factorización LU.

Examinemos lo que sucede cuando multiplicamos dos matrices de eliminación:

$$(I_{n\times n} - cee_k)(I_{n\times n} - cpepe_k) = I_{n\times n} - cee_k - cpepe_k$$

Como en nuestra construcción de la inversa de una matriz de eliminación, el producto de los dos términos ei se anula ya que la base estándar es ortogonal. Esta fórmula muestra que después de escalar el pivote a 1, el producto de las matrices de eliminación utilizadas para sustituir hacia adelante ese pivote tiene la forma:

$$METRO = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \bigcirc & 0 \\ 0 & \times & 1 & 0 \\ 0 & \times & 0 & 1 \end{matrix},$$

donde los valores × son los valores utilizados para eliminar el resto de la columna. La multiplicación de matrices de esta forma juntas muestra que los elementos debajo de la diagonal de L solo provienen de los coeficientes utilizados para lograr la sustitución.

Podemos tomar una decisión final para mantener los elementos a lo largo de la diagonal de L en la factorización LU igual a 1. Esta decisión es legítima, ya que siempre podemos posmultiplicar una L por una matriz de escala S llevando estos elementos a 1 y escriba LU = (LS)(S −1U) sin afectar el patrón triangular de L o U. Con esta decisión en su lugar, podemos comprimir nuestro almacenamiento de L y U en una sola matriz n × n cuyo triángulo superior es U y que es igual a L debajo de la diagonal; los elementos diagonales faltantes de L son todos 1.

Ahora estamos listos para escribir pseudocódigo para la estrategia de factorización LU más simple en la que no permutamos filas o columnas para obtener pivotes:

```
// Toma como entrada una matriz n - por - n A [i , j ]
// Edita A en su lugar para obtener la factorización LU compacta descrita anteriormente

para pivote de 1 a n {
        pivotValue = A [pivote, pivote]; // ¡Mala suposición de que esto es distinto de cero!
```

```
        para eliminaciónRow from ( pivote +1) a n { // Eliminar valores debajo del pivote
                // Cuánto escalar la fila pivote para eliminar el valor en el actual
                // fila ; notamos que dividimos por porque estoy escalando la fila pivote a 1 ,
                // valor pivote
                escala = A [filaEliminación, pivote]/ valorPivote;

                // Dado que / restamos / escalamos veces la fila pivote de la fila actual
                // durante la eliminación gaussiana // , nosotros/añadimos/lo en la operación inversa
                almacenado en L
                A [filaeliminación, pivote] = escala;

                // Ahora, cómo en la eliminación gaussiana, realiza la operación de fila en el resto
                // de la fila: esto se convertirá en U
                para eliminaciónCol de (pivote +1) a n
                        A [fila de eliminación , eliminaciónCol ] -= A [ pivote , eliminaciónCol ]*escala ;
        }
}
```

2.6 Problemas

Problema 2.1. El producto de cosas triangulares inferiores es triangular inferior; el producto de matrices pivote se ve bien

Problema 2.2. Implementar LU con pivote

Problema 2.3. LU no cuadrada

Capítulo 3

Diseño y análisis lineal Sistemas

Ahora que tenemos algunos métodos para resolver sistemas lineales de ecuaciones, podemos usarlos para resolver una variedad de problemas. En este capítulo, exploraremos algunas de esas aplicaciones y las técnicas analíticas que las acompañan para caracterizar los tipos de soluciones que podemos esperar.

3.1 Solución de Sistemas Cuadrados

Al comienzo del capítulo anterior hicimos varias suposiciones sobre los tipos de sistemas lineales que íbamos a resolver. Si bien esta restricción no era trivial, de hecho, muchas, si no la mayoría, de las aplicaciones de los solucionadores lineales se pueden plantear en términos de sistemas lineales cuadrados e invertibles. Exploramos algunas aplicaciones contrastantes a continuación.

3.1.1 Regresión

Comenzaremos con una aplicación simple que aparece en el análisis de datos conocida como regresión. Supongamos que llevamos a cabo un experimento científico y deseamos comprender la estructura de nuestros resultados experimentales. Una forma de modelar tal relación podría ser escribir las variables independientes del experimento en algún vector x Rn y considerar la variable dependiente como una función f(x) : Rn → R. Nuestro objetivo es predecir la salida de f sin realizar el experimento completo.

Ejemplo 3.1 (Experimento biológico). Supongamos que deseamos medir el efecto de los fertilizantes, la luz solar y el agua en el crecimiento de las plantas. Podríamos hacer una serie de experimentos aplicando diferentes cantidades de fertilizante (en cm3), luz solar (en vatios) y agua (en ml) y midiendo la altura de la planta después de unos días. Podríamos modelar nuestras observaciones como una función f : R3 → R tomando los tres parámetros que deseamos probar y dando como resultado la altura de la planta.

En la regresión paramétrica, hacemos una suposición simplificadora sobre la estructura de f . Por ejemplo, supongamos que f es lineal:

$$f(x) = a_1x_1 + a_2x_2 + \cdots + a_nx_n.$$

Entonces, nuestro objetivo se vuelve más concreto: estimar los coeficientes ak .

Supongamos que hacemos una serie de experimentos que muestran la forma $x^{(k)} \rightarrow y^{(k)} \equiv f(x^{(k)})$. Conectándose a nuestro $x \rightarrow y$ para f, obtenemos una serie de declaraciones:

$$\begin{aligned} y^{(1)} &= f(x^{(1)}) = a_1 x_1^{(1)} + a_2 x_2^{(1)} + \cdots + a_n x_n^{(1)} \\ y^{(2)} &= f(x^{(2)}) = a_1 x_1^{(2)} + a_2 x_2^{(2)} + \cdots + a_n x_n^{(2)} \\ &\vdots \end{aligned}$$

Note que contrario a nuestra notación $Ax = b$, las incógnitas aquí son las variables a_k, no las x. Si hacemos n observaciones, podemos escribir:

$$\begin{bmatrix} - & x^{(1)} & - \\ & x^{(2)} & - \\ & \vdots & \\ - & x^{(n)} & - \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

En otras palabras, si realizamos n intentos de nuestro experimento y los escribimos en las columnas de una matriz $X \in \mathbb{R}^{n \times n}$ y escribimos las variables dependientes en un vector $y \in \mathbb{R}^n$ los coeficientes a se pueden recuperar resolviendo $X a = y$.

De hecho, podemos generalizar nuestro enfoque a otras formas no lineales más interesantes para la función f. Lo que importa aquí es que f es una combinación lineal de funciones. En particular, suponga que $f(x)$ toma la siguiente forma:

$$f(x) = a_1 f_1(x) + a_2 f_2(x) + \cdots + a_m f_m(x),$$

donde $f_k : \mathbb{R}^n \rightarrow \mathbb{R}$ y deseamos estimar los parámetros a_k. Entonces, por una derivación paralela dadas m observaciones de la forma $x^{(k)} \rightarrow y^{(k)}$ podemos encontrar los parámetros resolviendo:

$$\begin{bmatrix} f_1(x^{(1)}) & f_2(x^{(1)}) & \cdots & f_m(x^{(1)}) \\ f_1(x^{(2)}) & f_2(x^{(2)}) & \cdots & f_m(x^{(2)}) \\ \vdots & \vdots & \cdots & \vdots \\ f_1(x^{(m)}) & f_2(x^{(m)}) & \cdots & f_m(x^{(m)}) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

Es decir, incluso si las f son no lineales, podemos aprender pesos a_k usando técnicas puramente lineales.

Ejemplo 3.2 (Regresión lineal). El sistema $Xa = y$ puede recuperarse de la formulación general tomando $f_k(x) \equiv x_k$.

Ejemplo 3.3 (Regresión polinomial). Supongamos que observamos una función de una sola variable $f(x)$ y deseamos escribirla como un polinomio de grado n

$$f(x) \equiv a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n.$$

Dados n pares $x^{(k)} \rightarrow y^{(k)}$, podemos resolver los parámetros a través del sistema

$$\begin{bmatrix} 1 & x^{(1)} & (x^{(1)})^2 & \cdots & (x^{(1)})^n \\ 1 & x^{(2)} & (x^{(2)})^2 & \cdots & (x^{(2)})^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & x^{(n)} & (x^{(n)})^2 & \cdots & (x^{(n)})^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

En otras palabras, tomamos $f_k(x) = x^k$ en nuestra forma general anterior. Por cierto, la matriz del lado izquierdo de esta relación se conoce como matriz de Vandermonde, que tiene muchas propiedades especiales específicas de su estructura.

Ejemplo 3.4 (Oscilación). Supongamos que deseamos encontrar a y φ para una función $f(x) = a\cos(x + \varphi)$. Recuerde de la trigonometría que podemos escribir $\cos(x + \varphi) = \cos x \cos \varphi - \operatorname{sen} x \operatorname{sen} \varphi$. Por lo tanto, dados dos puntos de muestra, podemos usar la técnica anterior para encontrar $f(x) = a_1 \cos x + a_2 \operatorname{sen} x$, y aplicando esta identidad podemos escribir

$$2 \text{ un} = \text{un } 1 \quad + \text{un } \frac{2}{2}$$

$$a2\ \varphi = -\arctan \frac{}{a1}$$

Esta construcción se puede extender para encontrar $f(x) = \sum_k a_k \cos(x + \varphi_k)$, dando una forma de motivar la transformada discreta de Fourier de f .

3.1.2 Mínimos cuadrados

Las técnicas de §3.1.1 proporcionan métodos valiosos para encontrar una f continua que coincida exactamente con un conjunto de pares de datos $x_k \to y_k$. Hay dos inconvenientes relacionados con este enfoque:

- Puede haber algún error al medir los valores x_k e y_k . En este caso, una relación aproximada $f(x_k) \approx y_k$ puede ser aceptable o incluso preferible a una $f(x_k) = y_k$ exacta .
- Observe que si hubiera m funciones f_k total, entonces necesitaríamos exactamente m observaciones $x_k \to y_k$. Tendrían que descartarse observaciones adicionales, o tendríamos que cambiar la forma de f .

Ambos problemas están relacionados con el problema mayor del sobreajuste: ajustar una función con n grados de libertad a n puntos de datos no deja margen para el error de medición.

Más generalmente, supongamos que deseamos resolver el sistema lineal $Ax = b$ para x. Si denotamos la fila k de A $a_{sr\ k}$,entonces nuestro sistema parece

$$\begin{array}{ccccc} b1 & - r\ 1 & - & x1 & r1 \cdot x \\ b2 & - r\ 2 & - & x2 & r2 \cdot x \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ & - r \text{ norte} & - & xn & rn \cdot x \end{array}$$

por definición de multiplicación de matrices.

Así, cada fila del sistema corresponde a una observación de la forma$r_k \cdot x = b_k$. Es decir, otra forma más de interpretar el sistema lineal $Ax = b$ es como n declaraciones de la forma, "El producto escalar de x conr_k es b_k ".

Desde este punto de vista, un sistema alto $Ax = b$ con $A \in \mathbb{R}^{m \times n}$ y $m > n$ simplemente codifica más de n de estas observaciones de producto escalar. Sin embargo, cuando hacemos más de n observaciones, pueden ser incompatibles; como se explica en §2.1, es probable que los sistemas altos no admitan una solución. En nuestra configuración “experimental” explicada anteriormente, esta situación podría corresponder a errores en la medición de los pares $x_k \to y_k$.

Cuando no podemos resolver exactamente $Ax = b$, podemos relajar un poco el problema para aproximar $Ax \approx b$. En particular, podemos pedir que el residuo $b - Ax$ sea lo más pequeño posible minimizando el

norma b − Ax. Observe que si existe una solución exacta para el sistema lineal, entonces esta norma se minimiza en cero, ya que en este caso tenemos b − Ax = b −b = 0. Minimizar b − Ax es lo mismo que minimizar b − Ax ^{2}, que expandimos en el Ejemplo 0.16 a:

$$\text{b - hacha}^{\,2} = x\,A\,Ax - 2b\,Ax + b^{\,2}.1$$

El gradiente de esta expresión con respecto a x debe ser cero en su mínimo, dando como resultado el siguiente sistema:

$$0 = 2A\,Ax - 2A\,b$$

O equivalentemente: A Ax = A b.

Esta famosa relación es digna de un teorema:

Teorema 3.1 (Ecuaciones normales). Los mínimos del residuo b − Ax para A Rm×n (sin restricción en mo n) satisfacen AAx = A b.

Si al menos n filas de A son linealmente independientes, entonces la matriz AA Rn×n es invertible. En este caso, el residuo mínimo ocurre (únicamente) en (A A) −1A b, o de manera equivalente, resolver el problema de mínimos cuadrados es tan fácil como resolver el sistema lineal cuadrado A Ax = A b del Teorema 3.1.
Por lo tanto, hemos ampliado nuestro conjunto de estrategias de solución a A Rm×n con m ≥ n aplicando solo técnicas para matrices cuadradas.

El caso indeterminado m < n es considerablemente más difícil de tratar. En particular, perdemos la posibilidad de una solución única para Ax = b. En este caso, debemos hacer una suposición adicional sobre x para obtener una solución única, por ejemplo, que tiene una norma pequeña o que contiene muchos ceros. Cada supuesto de regularización conduce a una estrategia de solución diferente; exploraremos algunos en los ejercicios que acompañan a este capítulo.

3.1.3 Ejemplos adicionales

Una habilidad importante es poder identificar sistemas lineales "en la naturaleza". Aquí enumeramos rápidamente algunos ejemplos más.

Alineación

Supongamos que tomamos dos fotografías de la misma escena desde diferentes posiciones. Una tarea común en la visión por computadora y los gráficos es unirlos. Para ello, el usuario (o un sistema automático) puede marcar una serie de puntos xk ,yk R2 tal que xk en la imagen uno corresponde a yk en la imagen dos. Por supuesto, probablemente se cometieron errores al hacer coincidir estos puntos, por lo que deseamos encontrar una transformación estable entre las dos imágenes sobremuestreando el número de pares necesarios (x, y).

Suponiendo que nuestra cámara tiene una lente estándar, las proyecciones de la cámara son lineales, por lo que es razonable se supone que existe algún A R2×2 y un vector de traslaciónb R2 tal que

$$yk \approx Axk + b.$$

[1]Puede ser valioso volver a los preliminares del Capítulo 0 en este punto para su revisión.

Nuestras variables desconocidas aquí son A yb en lugar de xk e yk .

En este caso, podemos encontrar la transformación resolviendo:

$$\min_{A,b} \sum_{k=1}^{pag} (Axk + b) - yk \quad ^2.$$

Esta expresión es una vez más una suma de expresiones lineales al cuadrado en nuestras incógnitas A yb, y por una derivación similar a nuestra discusión del problema de los mínimos cuadrados, puede resolverse linealmente.

Desconvolución

Muchas veces tomamos fotografías accidentalmente que están algo desenfocadas. Si bien una foto que está completamente borrosa puede ser una causa perdida, si hay un desenfoque localizado o de pequeña escala, es posible que podamos recuperar una imagen más nítida utilizando técnicas computacionales. Una estrategia simple es la desconvolución, que se explica a continuación.

Podemos pensar en una fotografía como un punto en Rp . , donde p es el número de píxeles; por supuesto, si el La foto está en color. Es posible que necesitemos tres valores (RGB) por píxel, lo que produce una técnica similar en R3p. . Independientemente, muchos desenfoques de imagen simples son lineales, por ejemplo, convolución gaussiana u operaciones que promedian píxeles con sus vecinos en la imagen. En el procesamiento de imágenes, estas operaciones lineales a menudo tienen otras propiedades especiales como la invariancia de desplazamiento, pero para nuestros propósitos podemos pensar en la borrosidad como un operador lineal x → G x.

Supongamos que tomamos una foto borrosa x0 Rp . Entonces, podríamos intentar recuperar la imagen nítida. x Rp resolviendo el problema de mínimos cuadrados

$$\min_{x \ Rp} x0 - GRAMO \quad x \quad ^2.$$

Es decir, le pedimos que cuando difumine x con G, obtenga la foto observada x0. Por supuesto, muchas imágenes nítidas pueden producir el mismo resultado borroso bajo G, por lo que a menudo agregamos términos adicionales a la minimización anterior para pedir que x0 no varíe mucho.

3.2 Propiedades especiales de los sistemas lineales

Nuestro análisis de la eliminación gaussiana y la factorización LU condujo a un método completamente genérico para resolver sistemas de ecuaciones lineales. Si bien esta estrategia siempre funciona, a veces podemos ganar velocidad o ventajas numéricas al examinar el sistema particular que estamos resolviendo. Aquí discutimos algunos ejemplos comunes en los que saber más sobre el sistema lineal puede simplificar las estrategias de solución.

3.2.1 Matrices definidas positivas y factorización de Cholesky

Como se muestra en el Teorema 3.1, al resolver un problema de mínimos cuadrados Ax ≈ b se obtiene una solución x que satisface el sistema lineal cuadrado (A A)x = A b. Independientemente de A, la matriz AA tiene algunas propiedades especiales que hacen que este sistema sea especial.

Primero, es fácil ver que AA es simétrica, ya que

$$(A A) = A (A) = A A.$$

Aquí, simplemente usamos las identidades (AB) = BA y (A) = A. Podemos expresar esta simetría en términos de índice escribiendo (A A)ij = (A A)ji para todos los índices i, j. Esta propiedad implica que es suficiente almacenar solo los valores de AA sobre o sobre la diagonal, ya que el resto de los elementos se pueden obtener por simetría.

Además, AA es una matriz semidefinida positiva, como se define a continuación:

Definición 3.1 (Positivo (Semi-)Definido). Una matriz B Rn×n es semidefinida positiva si para todo x Rn , x Bx ≥ 0. B es definida positiva si x Bx > 0 siempre que x =0.

Es fácil demostrar que AA es semidefinido positivo, ya que:

$$x\,A\,Ax = (Ax)\,(Ax) = (Ax)\cdot(Ax) = \|Ax\|_2^2 \geq 0.$$

De hecho, si las columnas de A son linealmente independientes, entonces AA es definida positiva.

Más generalmente, supongamos que deseamos resolver un sistema definido positivo simétrico Cx = d. Como ya hemos explorado, podríamos factorizar en LU la matriz C, pero de hecho podemos hacerlo algo mejor. Escribimos C Rn×n como una matriz de bloques:

$$C = \begin{matrix} c11 & v^\top \\ v & \tilde{C} \end{matrix}$$

donde v Rn−1 y C˜ R(n−1)×(n−1) . Gracias a la estructura especial de C, podemos hacer la siguiente observación:

$$Ce1 = \begin{matrix} 1 & 0 & \cdots & 0 \end{matrix} \begin{matrix} c11 & v^\top \\ v & \tilde{C} \end{matrix} \begin{matrix} 1 \\ 0 \\ \vdots \\ 0 \end{matrix}$$

$$= \begin{matrix} 1 & 0 & \cdots & 0 \end{matrix} \begin{matrix} c11 \\ v \end{matrix}$$

$$= c11$$

> 0 ya que C es definido positivo ye1 =0.

Esto muestra que, ignorando los problemas numéricos, no tenemos que usar el pivote para asegurar que c11 = 0 para la eliminación gaussiana del primer paso.

Continuando con la eliminación gaussiana, podemos aplicar una matriz de sustitución directa E, que genéricamente tiene la forma

$$mi = \begin{matrix} 1/\sqrt{c11} & 0 \\ r & Yo_{(n-1)\times(n-1)} \end{matrix}.$$

Aquí, el vectorr Rn−1 contiene los múltiplos de la fila 1 para cancelar el resto de la primera columna de C. ¡También escalamos la fila 1 en $1/\sqrt{c11}$ por razones que se harán evidentes en breve!

Por diseño, después de la sustitución hacia adelante conocemos el producto

$$CE = \begin{matrix} \sqrt{c11} & v^\top/\sqrt{c11} \\ 0 & D \end{matrix}$$

para algún D R(n−1)×(n−1).

Aquí es donde divergimos de la eliminación gaussiana: en lugar de pasar a la segunda fila, podemos posmultiplicar por E para obtener un producto ECE:

$$\text{CEPE} = (\text{CE})\text{E} = \begin{bmatrix} c_{11} & v \\ 0 & D \end{bmatrix} \begin{bmatrix} 1/\sqrt{c_{11}} & r \\ 0 & Y_{o(n-1)\times(n-1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{D} \end{bmatrix}$$

Es decir, hemos eliminado la primera fila y la primera columna de C! Además, es fácil comprobar que la matriz $\tilde{D}$ también es definida positiva.

Podemos repetir este proceso para eliminar todas las filas y columnas de C simétricamente. Aviso que usamos tanto simetría como definición positiva para derivar la factorización, ya que

- la simetría nos permitió aplicar la misma E a ambos lados, y
- la definición positiva garantiza que $c_{11} > 0$, lo que implica que $\sqrt{c_{11}}$ existe.

Al final, similar a la factorización LU, ahora obtenemos una factorización C = LL para una matriz triangular inferior L. Esto se conoce como la factorización de Cholesky de C. Si tomar las raíces cuadradas a lo largo de la diagonal causa problemas numéricos, una factorización LDL relacionada , donde D es una matriz diagonal, evita este problema y es fácil de derivar de la discusión anterior.

La factorización de Cholesky es importante por varias razones. Lo más destacado es que se necesita la mitad de la memoria para almacenar L que la factorización LU de C o incluso C mismo, ya que los elementos sobre la diagonal son cero y, como en LU, resolver Cx = d es tan fácil como sustituir hacia adelante y hacia atrás . Explorará otras propiedades de la factorización en los ejercicios.

Al final, el código para la factorización de Cholesky puede ser muy breve. Para derivar un particular com forma de pacto, supongamos que elegimos una fila arbitraria k y escribimos L en forma de bloque aislando esa fila:

$$L = \begin{bmatrix} L_{11} & 0 & 0 \\ k & k & 0 \\ L_{31} & k & L_{33} \end{bmatrix}$$

Aquí, L11 y L33 son ambas matrices cuadradas triangulares inferiores. Entonces, realizando un producto se obtiene:

$$LL = \begin{bmatrix} L_{11} & 0 & 0 \\ k & k & 0 \\ L_{31} & k & L_{33} \end{bmatrix} \begin{bmatrix} L_{11} & 0 & 0 \\ k\,L_{kk} & k & 0 \\ L_{31} & k & L_{33} \end{bmatrix} = \begin{bmatrix} \times & \times & \times \\ L_{11}\,L_{kk} + k \times & 2k & \times \\ \times & \times & \times \end{bmatrix}$$

Omitimos valores del producto que no son necesarios para nuestra derivación.

Al final, sabemos que podemos escribir C = LL. El elemento central del producto muestra:

$$k = \sqrt{c_{kk} - k^2_2}$$

donde k Rk−1 contiene los elementos de la k-ésima fila de L a la izquierda de la diagonal. Además, el elemento central izquierdo del producto muestra

$$L_{11}k = c_k$$

donde ck contiene los elementos de C en la misma posición ask . Dado que L11 es triangular inferior, este ¡El sistema se puede resolver por sustitución directa!

Observe que nuestra discusión anterior produce un algoritmo para calcular la factorización de Cholesky de arriba a abajo, ya que L11 ya estará calculado cuando lleguemos a la fila k. Proporcionamos pseudocódigo a continuación, adaptado de CITE:

```
// Toma como entrada una matriz n - por - n A [i , j ]
// Edita A en su lugar para obtener la factorización de Cholesky en su triángulo inferior

para k de 1 a n {
        // Atrás - sustituir para encontrar l_k
        para i de 1 a k -1 { // elemento i de l_k
                suma = 0;
                para j de 1 a i -1
                        suma += A [i, j]* A [k, j];
                un , yo ] = ( A [k , i ] - suma )/ A [i , i ];
        }

        // Aplicar la fórmula para l_kk
        norma al cuadrado = 0
        para j de 1 a i -1
                normSquared += A [k, j]^2;
        un , k ] = raíz cuadrada ( A [k , k ] - norma al cuadrado );
}
```

Al igual que con la factorización LU, este algoritmo claramente se ejecuta en $O(n^3)$ tiempo.

3.2.2 Escasez

Muchos sistemas lineales de ecuaciones naturalmente disfrutan de propiedades de escasez, lo que significa que la mayoría de los las entradas de A en el sistema Ax = b son exactamente cero. La escasez puede reflejar una estructura particular en un problema dado, incluidos los siguientes casos de uso:

- En el procesamiento de imágenes, muchos sistemas de edición y comprensión de fotografías expresan relaciones entre los valores de los píxeles y los de sus vecinos en la cuadrícula de la imagen. Una imagen puede ser un punto en Rp para p píxeles, pero al resolver Ax =b para una imagen de nuevo tamaño-p, A Rp×p puede tener solo O (p) en lugar de O (p^2) distintos de cero ya que cada fila solo involucra un solo píxel y sus vecinos arriba/abajo/izquierda/derecha.

- En aprendizaje automático, un modelo gráfico utiliza una estructura gráfica G ≡ (V, E) para expresar distribuciones de probabilidad sobre varias variables. Cada variable se representa mediante un nodo v V de la gráfica, y la arista e E representa una dependencia probabilística. Sistemas lineales que surgen en este contexto a menudo tiene una fila por vértice v V con ceros solo en columnas que involucran v y sus vecinos.

- En geometría computacional, las formas a menudo se expresan usando conjuntos de triángulos vinculados juntos en una malla. Las ecuaciones para el suavizado de superficies y otras tareas vinculan una vez más las posiciones y otros valores en un vértice dado con los de sus vecinos en la malla.

Ejemplo 3.5 (Parametrización de armónicos). Supongamos que deseamos usar una imagen para texturizar una malla triangular. Una malla se puede representar como una colección de vértices V R3 unidos entre sí por aristas E V × V para formar triángulos. Dado que los vértices de la geometría están en R3 , debemos encontrar una manera de asignarlos al plano de la imagen para almacenar la textura como una imagen. Por lo tanto, debemos asignar coordenadas de textura t(v) R2 en el plano de la imagen a cada v V. Consulte la Figura NÚMERO para ver una ilustración.

Una estrategia para hacer este mapa implica una única solución lineal. Supongamos que la malla tiene topología de disco, es decir, se puede mapear al interior de un círculo en el plano. Para cada vértice vb en el límite de la malla, podemos especificar la posición de vb colocándolo en un círculo. En el interior, podemos pedir que la posición del mapa de texturas sea el promedio de sus posiciones vecinas:

$$1\ t(v) = \left|\frac{}{n(v)}\right| \sum_{w\ n(v)} t(w)$$

Aquí, n(v) V es el conjunto de vecinos de v V en la malla. Así, cada v V está asociado con una ecuación lineal, ya sea fijándolo en el límite o pidiendo que su posición sea igual al promedio de sus posiciones vecinas. Este |V| × |V| sistema de ecuaciones conduce a una estrategia de parametrización estable conocida como parametrización armónica; la matriz del sistema solo tiene O(|V|) distintos de cero en las ranuras correspondientes a los vértices y sus vecinos.

Por supuesto, si A Rn×n es escasa hasta el punto de que contiene valores O(n) en lugar de O(n , no hay ^{2}) distinto de cero razón para almacenar A como una matriz n × n. En cambio, las técnicas de almacenamiento de matrices dispersas solo almacenan los O(n) distintos de cero en una estructura de datos más razonable, por ejemplo, una lista de tripletas de fila/columna/valor. , o iterando sobre filas o columnas individuales.

Desafortunadamente, es fácil ver que la factorización LU de un A disperso puede no resultar en matrices L y U dispersos; esta pérdida de estructura limita severamente la aplicabilidad del uso de estos métodos para resolver Ax =b cuando A es grande pero escaso. Afortunadamente, hay muchos solucionadores de dispersión directos que adaptan LU a matrices dispersas que pueden producir una factorización similar a LU sin inducir mucho relleno o ceros adicionales; la discusión de estas técnicas está fuera del alcance de este texto. Alternativamente, se han utilizado técnicas iterativas para obtener soluciones aproximadas a sistemas lineales; aplazaremos la discusión de estos métodos para capítulos futuros.

Ciertas matrices no solo son dispersas sino también estructuradas. Por ejemplo, un sistema tridiagonal de ecuaciones lineales tiene el siguiente patrón de valores distintos de cero:

```
× ×
× × ×
  × × ×
    × × ×
      × ×
```

En los ejercicios que siguen a este capítulo, derivará una versión especial de la eliminación gaussiana para tratar con esta estructura simple con bandas.

En otros casos, las matrices pueden no ser escasas, pero pueden admitir una representación escasa. Para examen Por ejemplo, considere la matriz cíclica:

abcdabccdabbcda

Obviamente, esta matriz se puede almacenar usando solo los valores a, b, c, d. Las técnicas especializadas para esta y otras clases de matrices están bien estudiadas y, a menudo, son más eficientes que la eliminación genérica de Gauss.

3.3 Análisis de sensibilidad

Como hemos visto, es importante examinar la matriz de un sistema lineal para averiguar si tiene propiedades especiales que puedan simplificar el proceso de solución. La escasez, la definición positiva, la simetría, etc., pueden proporcionar pistas sobre el solucionador adecuado para un problema en particular.

Sin embargo, incluso si una estrategia de solución dada podría funcionar en teoría, es igualmente importante comprender qué tan bien podemos confiar en la respuesta a un sistema lineal dada por un solucionador en particular. Por ejemplo, debido al redondeo y otros efectos discretos, podría darse el caso de que una implementación de la eliminación gaussiana para resolver Ax = b arroje una solución x0 tal que 0 < Ax0 −b 1; en otras palabras, x0 solo resuelve el sistema aproximadamente.

Una forma de comprender la probabilidad de estos efectos de aproximación es a través del análisis de sensibilidad. En este enfoque, nos preguntamos qué le sucedería a x si en lugar de resolver Ax = b en realidad resolvemos un sistema perturbado de ecuaciones (A + δA)x = b + δb. Hay dos formas de ver las conclusiones de este tipo de análisis:

1. Es probable que cometamos errores al representar A yb gracias al redondeo y otros efectos. Luego, este análisis muestra la mejor precisión posible que podemos esperar para x dados los errores cometidos al representar el problema.

2. Si nuestro solucionador genera una aproximación x0 a la solución de Ax =b, es una solución exacta al sistema Ax0 = b0 si definimosb0 ≡ Ax0 (¡asegúrese de entender por qué esta oración no es una tautología!). Comprender cómo los cambios en x0 afectan los cambios en b0 muestra qué tan sensible es el sistema a las respuestas ligeramente incorrectas.

Note que nuestra discusión aquí es similar y de hecho está motivada por nuestras definiciones de error hacia adelante y hacia atrás en capítulos anteriores.

3.3.1 Normas de matrices y vectores

Antes de que podamos analizar la sensibilidad de un sistema lineal, debemos tener cierto cuidado al definir qué significa que un cambio δx sea “pequeño”. Generalmente, deseamos medir la longitud, o norma, de un vector x. Ya hemos encontrado la norma de dos de un vector:

$$\|x\|_2 \equiv \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

para $x \in \mathbb{R}^n$. Esta norma es popular gracias a su conexión con la geometría euclidiana, pero de ninguna manera es la única norma sobre $\mathbb{R}^n$. En general, definimos una norma de la siguiente manera:

Definición 3.2 (Norma vectorial). Una norma vectorial es una función $\|\cdot\| : \mathbb{R}^n \to [0, \infty)$ satisfaciendo lo siguiente condiciones:

- $\|x\| = 0$ si y sólo si $x = 0$.

• cx = |c|x para todos los escalares c R y vectores x Rn .

• x +y ≤ x + y para todo x,y Rn .

Si bien usamos el subíndice dos · 2 para denotar la norma de dos de un vector, a menos que indiquemos lo contrario, usaremos la notación x para denotar la norma de dos de x. Aparte de esta norma, hay muchos otros ejemplos: • La p-norma xp, para p ≥ 1, dada por:

$$xp \equiv (|x1|^{pag} + |x2|^{pag} + \cdots + |xn|^{pag})^{1/p}$$

De particular importancia es la norma 1 o norma “taxicab”, dada por

$$x1 \equiv \sum_{k=1}^{norte} |xk|.$$

Esta norma recibe su apodo porque representa la distancia que recorre un taxi entre dos puntos de una ciudad donde las carreteras solo van de norte a sur y de este a oeste.

• La ∞-norma x∞ dada por:

$$x\infty \equiv \max(|x1|, |x2|, \cdots, |xn|).$$

En cierto sentido, muchas normas sobre Rn son las mismas. En particular, supongamos que decimos que dos normas son equivalentes cuando satisfacen la siguiente propiedad: Definición

3.3 (Normas equivalentes). Dos normas · y · son equivalentes si existen constantes clow y chigh tales que

$$clownx \leq x \leq chighx$$

para todo x Rn·

Esta condición garantiza que, hasta algunos factores constantes, todas las normas concuerdan en qué vectores son "pequeños" y "grandes". De hecho, enunciaremos sin demostración un famoso teorema del análisis: Teorema 3.2

(Equivalencia de normas sobre Rn). Todas las normas sobre Rn son equivalentes.

Este resultado un tanto sorprendente implica que todas las normas vectoriales tienen el mismo comportamiento aproximado, pero la elección de una norma para analizar o plantear un problema en particular puede marcar una gran diferencia.
Por ejemplo, en R3 , la norma ∞ considera que el vector (1000, 1000, 1000) tiene la misma norma que (1000, 0, 0) , mientras que la norma 2 ciertamente se ve afectada por los valores adicionales distintos de cero.

Dado que perturbamos no solo vectores sino también matrices, también debemos poder tomar la norma de una matriz. Por supuesto, la definición básica de una norma no cambia en Rn×m. Por esta razón, podemos “desenrollar” cualquier matriz en Rm×n a un vector en Rnm para adoptar cualquier norma vectorial a las matrices. Una de esas normas es la norma de Frobenius, dada por

$$AFro \equiv \sqrt{\sum_{yo,\,j} a\,ij^{2}}.$$

Tales adaptaciones de normas vectoriales, sin embargo, no siempre son muy significativas. En particular, la prioridad para comprender la estructura de una matriz A suele ser su acción sobre los vectores, es decir, los resultados probables cuando A se multiplica por una x arbitraria. Con esta motivación, podemos definir la norma inducida por una norma vectorial de la siguiente manera:

Definición 3.4 (Norma inducida). La norma sobre Rm×n inducida por una norma · sobre Rn viene dada por

$$A \equiv \max\{Ax : x = 1\}.$$

Es decir, la norma inducida es la longitud máxima de la imagen de un vector unitario multiplicada por A.

Dado que las normas vectoriales satisfacen cx = |c|x, es fácil ver que esta definición es equivalente a requerir

$$A \equiv \max_{x \quad Rn\setminus\{0\}} \frac{\text{Hacha}}{X}.$$

Desde este punto de vista, la norma de A inducida por · es la mayor proporción alcanzable de la norma de Ax relativa a la de la entrada x.

Esta definición general hace que sea algo difícil calcular la norma A dada una matriz A y una elección de · . Afortunadamente, las normas matriciales inducidas por muchas normas vectoriales populares se pueden simplificar. Enunciamos algunas de tales expresiones sin demostración:

- La norma inducida de A es la suma máxima de cualquier columna de A:

$$A1 = \max_{1\le j\le n} \sum_{i=1}^{\text{metro}} |aij|$$

- La ∞-norma inducida de A es la suma máxima de cualquier fila de A:

$$A\infty = \max_{1\le i\le m} \sum_{j=1}^{\text{norte}} |aij|$$

- La norma doble inducida, o norma espectral, de A Rn×n es la raíz cuadrada del valor propio más grande de A A. Es decir,

$$A \;{}_2^2 = \max\{\lambda : \text{existe } x \quad R^{\text{norte}} \text{ con } A\,Ax = \lambda x\}$$

Al menos las dos primeras normas son relativamente fáciles de calcular; volveremos a la tercera mientras discutimos los problemas de valores propios.

3.3.2 Números de condición

Ahora que tenemos herramientas para medir la acción de una matriz, podemos definir el número de condición de un sistema lineal adaptando nuestra definición genérica de números de condición del Capítulo 1. Seguimos el desarrollo presentado en CITE.

Supongamos que perturbamos δA de una matriz A y una perturbación correspondiente δb. Para cada ε ≥ 0, ignorando los tecnicismos de la invertibilidad podemos escribir un vector x(ε) como la solución a

$$(A + \varepsilon \cdot \delta A)x(\varepsilon) = b + \varepsilon \cdot \delta b.$$

Si diferenciamos ambos lados con respecto a ε y aplicamos la regla del producto, obtenemos el siguiente resultado:

$$\delta A \cdot x + (A + \varepsilon \cdot \delta A)\frac{dx}{d\varepsilon} = \delta b$$

En particular, cuando ε = 0 encontramos

$$\delta A \cdot x(0) + A \frac{dx}{d\varepsilon}_{\varepsilon=0} = \delta b$$

o equivalente,

$$\frac{dx}{d\varepsilon}_{\varepsilon=0} = A^{-1}(\delta b - \delta A \cdot x(0)).$$

Usando la expansión de Taylor, podemos escribir

$$x(\varepsilon) = x + \varepsilon x(0) + O(\varepsilon^2),$$

donde definimos x (0) = sistema $\frac{dx}{d\varepsilon}_{\varepsilon=0}$. Por lo tanto, podemos expandir el error relativo cometido resolviendo el perturbado:

$$\begin{aligned}
\frac{x(\varepsilon) - x(0)}{x(0)} &= \frac{\varepsilon x(0) + O(\varepsilon^2)}{(0)} \text{ por la expansión de Taylor} \\
&= \frac{\varepsilon A^{-1}(\delta b - \delta A \cdot x(0)) + O(\varepsilon x(0)^2)}{} \text{ por la derivada que calculamos} \\
&\leq \frac{|\varepsilon|}{x(0)}(A^{-1}\delta b + A^{-1}\delta A \cdot x(0))) + O(\varepsilon^2) \\
&\quad \text{por la desigualdad triangular } A + B \leq A + B \\
&\leq |\varepsilon| A^{-1} \frac{\delta b}{x(0)} + \delta A + O(\varepsilon^2) \text{ por la identidad } AB \leq AB \\
&= |\varepsilon| A^{-1} A \frac{\delta b}{\text{hacha}(0)} + \frac{\delta A}{A} + O(\varepsilon^2) \\
&\leq |\varepsilon| A^{-1} A \frac{\delta b}{\text{hacha}(0)} + \frac{\delta A}{A} + O(\varepsilon^2) \text{ ya que } Ax(0) \leq Ax(0) \\
&= |\varepsilon| A^{-1} A \frac{\delta b}{b} + \frac{\delta A}{A} + O(\varepsilon^2) \text{ ya que } Ax(0) \leq Ax(0) \\
&\quad \text{ya que por definición, } Ax(0) = b
\end{aligned}$$

Aquí hemos aplicado algunas propiedades de la norma matricial que se derivan de las propiedades correspondientes de los vectores. Observe que la suma D ≡ δb/b + δA/A codifica las perturbaciones relativas de A yb. Desde este punto de vista, a primer orden hemos acotado el error relativo de perturbar el sistema por ε usando un factor κ ≡ AA −1:

$$\frac{x(\varepsilon) - x(0)}{x(0)} \leq \varepsilon \cdot D \cdot \kappa + O(\varepsilon^2)$$

De esta forma, la cantidad κ acota el condicionamiento de los sistemas lineales en los que interviene A. Por ello, hacemos la siguiente definición:

Definición 3.5 (Número de condición de matriz). El número de condición de $A \in \mathbb{R}^{n\times n}$ para una norma matricial dada $\cdot$ es

$$\text{cond. } A \equiv \|A\|\,\|A^{-1}\|.$$

Si A no es invertible, tomamos cond $A \equiv \infty$.

Es fácil ver que cond $A \geq 1$ para todo A, que escalar A no tiene efecto en su número de condición y que el número de condición de la matriz identidad es 1. Estas propiedades contrastan con el determinante, que puede escalar hacia arriba y hacia abajo a medida que escala A.

Si $\cdot$ es inducida por una norma vectorial y A es invertible, entonces tenemos

$$\begin{aligned}
\|A^{-1}\| &= \max_{x=0} \frac{\|A^{-1}x\|}{\|x\|} && \text{por definición}\\
&= \max_{y=0} \frac{\|y\|}{\|Ay\|} && \text{sustituyendo } y = A^{-1}x\\
&= \min_{y=0} \frac{\|Ay\|}{\|y\|}^{-1} && \text{tomando el recíproco}
\end{aligned}$$

En este caso, el número de condición de A está dado por:

$$\text{cond. } A = \max_{x=0} \frac{\|Ax\|}{\|x\|} \left(\min_{y=0} \frac{\|Ay\|}{\|y\|}\right)^{-1}$$

En otras palabras, cond A mide el estiramiento máximo a mínimo posible de un vector x bajo A.

Más generalmente, una propiedad de estabilidad deseable de un sistema Ax =b es que si se perturba un orbe, la solución x no cambia considerablemente. Nuestra motivación para la condición A muestra que cuando el número de condición es pequeño, el cambio en x es pequeño en relación con el cambio en el orbe A, como se ilustra en la Figura NÚMERO. De lo contrario, un pequeño cambio en los parámetros del sistema lineal puede causar grandes desviaciones en x; esta inestabilidad puede hacer que los solucionadores lineales cometan grandes errores en x debido al redondeo y otras aproximaciones durante el proceso de solución.

la norma $\|A^{-1}\|$ puede ser tan difícil como calcular el inverso completo A^{-1} límite. Una forma de bajar el número de condición es aplicar la identidad $\|A^{-1}x\| \leq \|A^{-1}\|\|x\|$. Así, para cualquier $x = 0 \geq \|A^{-1}x\|/\|x\|$. De este modo, podemos escribir $\|A^{-1}\|$

$$\text{cond. } A = \|A\|\,\|A^{-1}\| \geq \frac{\|A\|\,\|x\|}{\|Ax\|}.$$

Entonces, podemos acotar el número de condición resolviendo $A^{-1}x$ para algunos vectores x; por supuesto, la necesidad de un solucionador lineal para encontrar $A^{-1}x$ crea una dependencia circular en el número de condición para evaluar la calidad de la estimación. Cuando $\cdot$ es inducida por la norma de dos, en capítulos futuros proporcionaremos estimaciones más confiables.

3.4 Problemas

Algo como:

- Regresión Kernel como ejemplo de §3.1.1
- Solución de norma mínima para Ax =b, la matriz de mínimos cuadrados es invertible de lo contrario
- Versiones variacionales de la regularización de Tikhonov/regresión de “cresta” (no es el enfoque habitual a esto, pero lo que sea); completando la historia indeterminada de esta manera
- 1 • L enfoques de regularización para el contraste: dibuje una imagen de por qué esperar escasez, dibuje círculos unitarios, muestra que la norma p no es una norma para $p < 1$, toma el límite cuando $p \to 0$
- Mini-Riesz: matriz derivada del producto interno, se usa para mostrar cómo rotar el espacio
- solución tridiagonal
- propiedades del número de condición

Capítulo 4

Espacios de columna y QR

Una forma de interpretar el problema lineal Ax = b para x es escribir b como una combinación lineal de las columnas de A con pesos dados en x. Esta perspectiva no cambia cuando permitimos que A Rm×n no sea cuadrado, pero la solución puede no existir o ser única dependiendo de la estructura del espacio de columnas. Por estas razones, algunas técnicas para la factorización de matrices y el análisis de sistemas lineales buscan representaciones más simples del espacio de la columna para eliminar la ambigüedad de la resolución y abarcan más explícitamente que las factorizaciones basadas en filas como LU.

4.1 La estructura de las ecuaciones normales

Como hemos mostrado, una condición necesaria y suficiente para que x sea una solución del problema de mínimos cuadrados Ax ≈ b es que x satisfaga las ecuaciones normales (A A)x = A b. Este teorema sugiere que resolver mínimos cuadrados es una extensión bastante simple de las técnicas lineales. Métodos como la factorización de Cholesky también muestran que la estructura especial de los problemas de mínimos cuadrados se puede utilizar en beneficio del solucionador.

Sin embargo, existe un gran problema que limita el uso de este enfoque. Por ahora, suponga que A es cuadrada; entonces podemos escribir:

$$\begin{aligned}\text{cond } AA &= A\,A(A\,A)^{-1}\\ &\approx A\,A\,A^{-1}A^{-1}\\ &= \text{un}^{2}\,A^{-1}{}^{2}\\ &= (\text{condición } A)^{2}\end{aligned}$$

(Uno)$^{-1}$ dependiendo de la elección de ·

Es decir, ¡el número de condición de AA es aproximadamente el cuadrado del número de condición de A! Por lo tanto, mientras que las estrategias lineales genéricas pueden funcionar en AA cuando el problema de los mínimos cuadrados es "fácil", cuando las columnas de A son casi linealmente dependientes, es probable que estas estrategias generen un error considerable ya que no tratan con A directamente.

Intuitivamente, una razón principal por la que la cond A puede ser grande es que las columnas de A pueden parecer "similares". Piense en cada columna de A como un vector en Rm. Si dos columnas ai y aj satisfacen ai ≈ entonces la longitud residual aj , mínimos cuadrados b − Ax probablemente no sufriría mucho si reemplazamos múltiplos de ai con múltiplos de aj o viceversa. Esta amplia gama de soluciones equivalentes casi, pero no completamente, produce un acondicionamiento deficiente. Mientras que el vector x resultante es inestable, sin embargo, el producto

Ax permanece casi sin cambios, por diseño de nuestra sustitución. Por lo tanto, si deseamos resolver Ax ≈b simplemente escribiendo b en el espacio columna de A, cualquier solución sería suficiente.

Para resolver estos problemas mal condicionados, emplearemos una estrategia alternativa con mayor atención al espacio columna de A en lugar de emplear operaciones de fila como en la eliminación gaussiana. De esta forma, podemos identificar esas casi dependencias explícitamente y tratarlas de una manera numéricamente estable.

4.2 Ortogonalidad

Hemos determinado cuándo es difícil el problema de los mínimos cuadrados, pero también podemos preguntarnos cuándo es más sencillo. Si podemos reducir un sistema al caso sencillo sin inducir problemas de condicionamiento en el camino, habremos encontrado una forma más estable de solucionar los problemas explicados en §4.1.

Obviamente, el sistema lineal más fácil de resolver es In×nx = b: ¡La solución simplemente es x ≡ b! Es poco probable que ingresemos explícitamente este sistema lineal en particular en nuestro solucionador, pero podemos hacerlo accidentalmente mientras resolvemos mínimos cuadrados. En particular, aun cuando A = In×n —de hecho, A no necesita ser una matriz cuadrada—, en circunstancias particularmente afortunadas podemos encontrar que la matriz normal AA satisface AA = In×n. Para evitar confusiones con el caso general, usaremos la letra Q para representar dicha matriz.

Rezar simplemente para que QQ = In×n probablemente no produzca una estrategia de solución deseable, pero podemos examinar este caso para ver cómo se vuelve tan favorable. Escriba las columnas de Q como vectores q1, · · · ,qn Rm. Entonces, es fácil verificar que el producto QQ tiene la siguiente estructura:

$$Q^\top Q = \begin{bmatrix} - & q_1 & - \\ - & q_2 & - \\ & \vdots & \\ - & q_n & - \end{bmatrix} \begin{bmatrix} | & | & & | \\ q_1 & q_2 & \cdots & q_n \\ | & | & & | \end{bmatrix} = \begin{bmatrix} q_1 \cdot q_1 & q_1 \cdot q_2 & \cdots & q_1 \cdot q_n \\ q_2 \cdot q_2 & q_2 \cdot q_2 & \cdots & q_2 \cdot q_n \\ \vdots & \vdots & \ddots & \vdots \\ q_n \cdot q_1 & q_n \cdot q_2 & \cdots & q_n \cdot q_n \end{bmatrix}$$

Establecer la expresión de la derecha igual a In × n produce la siguiente relación:

$$q_i \cdot q_j = \begin{cases} 1 & \text{cuando } i = j \\ 0 & \text{cuando } i = j \end{cases}$$

En otras palabras, las columnas de Q son de longitud unitaria y ortogonales entre sí. Decimos que forman una base ortonormal para el espacio columna de Q:

Definición 4.1 (Ortonormal; matriz ortogonal). Un conjunto de vectores {v1, · · · ,vk} es ortonormal si vi = 1 para todo i y vi ·vj = 0 para todo i = j. Una matriz cuadrada cuyas columnas son ortonormales se llama matriz ortogonal.

Motivamos nuestra discusión preguntando cuándo podemos esperar QQ = In×n. Ahora es fácil ver que esto ocurre cuando las columnas de Q son ortonormales. Además, si Q es cuadrada e invertible con QQ = In×n, simplemente multiplicando ambos lados de esta expresión por Q−1 encontramos Q−1 = Q. Por lo tanto, resolver Qx = b en este caso es tan fácil como multiplicar ambos lados por la transpuesta Q.

La ortonormalidad también tiene una fuerte interpretación geométrica. Recuerde del Capítulo 0 que podemos considerar dos vectores ortogonales ayb como perpendiculares. Entonces, un conjunto ortonormal de vectores

simplemente es un conjunto de vectores perpendiculares de longitud unitaria en · Si Q es ortogonal, entonces su acción no Rn que no afecta la longitud de los vectores:

$$Qx^{2} = x\,Q\,Qx = x\,In{\times}nx = x \cdot x = x^{2}$$

De manera similar, Q no puede afectar el ángulo entre dos vectores, ya que:

$$(Qx) \cdot (Qy) = x\,Q\,Qy = x\,In{\times}ny = x \cdot y$$ Desde este

punto de vista, si Q es ortogonal, entonces Q representa una isometría de Rn , es decir, conserva longitudes y ángulos. En otras palabras, puede rotar o reflejar vectores pero no puede escalarlos o cortarlos.
Desde un nivel alto, el álgebra lineal de matrices ortogonales es más fácil porque su acción no afecta la geometría del espacio subyacente de ninguna manera trivial.

4.2.1 Estrategia para matrices no ortogonales

Excepto en circunstancias especiales, la mayoría de nuestras matrices A al resolver Ax = bo el correspondiente problema de mínimos cuadrados no serán ortogonales, por lo que la maquinaria de §4.2 no se aplica directamente. Por esta razón, debemos hacer algunos cálculos adicionales para conectar el caso general con el ortogonal.

Tome una matriz A Rm×n, y denote su espacio de columna como col A; recuerde que col A representa el lapso de las columnas de A. Ahora, suponga que una matriz B Rn×n es invertible. Podemos hacer una simple observación sobre el espacio columna de AB relativo al de A: Lema 4.1

(Invarianza del espacio columna). Para cualquier A Rm×n e invertible B Rn×n ,

columna A = columna AB.

Prueba. Supongamosb col A. Entonces, por definición de multiplicación por A existe x con Ax = b. Entonces, (AB) · (B −1x) = Ax =b, sob col AB. Por el contrario, toma c col AB, entonces existe y con (AB)y = c. Entonces, A · (By) = c, mostrando que c está en la columna A. □

Recuerde la descripción de la "matriz de eliminación" de la eliminación gaussiana: comenzamos con una matriz A y aplicamos matrices de operación por filas Ei tales que la secuencia A, E1A, E2E1A, . . . sistemas lineales secuencialmente más fáciles representados. El lema anterior sugiere una estrategia alternativa para situaciones en las que nos preocupamos por el espacio de la columna: Aplicar operaciones de columna a A por post-multiplicación hasta que las columnas sean ortonormales. Es decir, obtenemos un producto Q = AE1E2 · · · Ek tal que Q es ortonormal. Siempre que los Ei sean invertibles, el lema muestra que col Q = col A. La inversión de estas operaciones produce una factorización A = QR para R = $E_k^{-1} E_{k-1}^{-1} \cdots E_1^{-1}$.

E Como en la factorización LU, si diseñamos R con cuidado, la solución de mínimos Los problemas de cuadrados Ax ≈b pueden simplificar. En particular, cuando A = QR, podemos escribir la solución de A Ax = A b de la siguiente manera:

$$\begin{aligned} x &= (A\,A)^{-1} A\,b \\ &= (RQ\,QR)^{-1} R\,Q\,b \text{ ya que } A = QR \\ &= (R\,R)^{-1} R\,Q\,b \text{ ya que } Q \text{ es ortogonal} \\ &= R^{-1} (R\,)^{-1} R\,Q\,b \text{ ya que } (AB)^{-1} = B^{-1} A^{-1} \\ &= R^{-1} Q b \_ \end{aligned}$$

O de manera equivalente, Rx = Q b

Por lo tanto, si diseñamos R para que sea una matriz triangular, entonces resolver el sistema lineal Rx = Qb es tan simple como sustituir hacia atrás.

Nuestra tarea para el resto del capítulo es diseñar estrategias para tal factorización.

4.3 Ortogonalización de Gram-Schmidt

Nuestro primer enfoque para encontrar factorizaciones QR es el más simple de describir e implementar, pero puede tener problemas numéricos. Lo usamos aquí como una estrategia inicial y luego lo mejoraremos con mejores operaciones.

4.3.1 Proyecciones

Supongamos que tenemos dos vectores a yb. Entonces, podríamos preguntar fácilmente "¿Qué múltiplo de a es el más cercano ab?" Matemáticamente, esta tarea es equivalente a minimizar ca −b sobre todos los posibles c R. Si pensamos en a y b como matrices n × 1 y c como una matriz 1 × 1, entonces esto no es más que un problema de mínimos cuadrados no convencional. · c ≈b. En este caso, las ecuaciones normales muestrana a · c = ab, o

$$c = \frac{a \cdot b}{un \cdot un} = \frac{a \cdot b}{a^2}.$$

Denotamos esta proyección de b sobre a como:

$$\text{proyecto}_a \cdot bb = ca = \frac{a}{a \cdot a} a = \frac{a \cdot b}{a^{un\,2}}$$

Obviamente cálculo$_a$ b es paralelo a a. ¿Qué pasa con el resto b − proj $_a$ ¿b? Podemos hacer un simple de proyecto para averiguar:

$$\begin{aligned} a \cdot (b - \text{proyecto}_a\, b) &= a \cdot b - a \cdot \frac{a \cdot b}{a^{un\,2}} \\ &= a \cdot b - \frac{a \cdot b}{a^2}(a \cdot a) \\ &= a \cdot b - a \cdot b \\ &= 0 \end{aligned}$$

Así, hemos descompuesto b en una componente paralela a a y otra ortogonal toa.

Ahora, suponga que aˆ1, aˆ2, · · · , aˆk son ortonormales; pondremos sombreros sobre los vectores con longitud unitaria. Entonces, para cualquier i podemos ver:

$$\text{proja\^{}i}\ b = (a\hat{}i \cdot b)a\hat{}i$$

El término norma no aparece porque aˆi = 1 por definición. Podríamos proyectar b sobre el tramo {aˆ1, · · · , aˆk} minimizando la siguiente energía sobre c1, . . . , ck R:

$$\left\| c1\hat{a}1 + c2\hat{a}2 + \cdots + ck\,\hat{a}k - b \right\|^2 = \sum_{yo=1}^{k}\sum_{j=1}^{k} cicj(\hat{a}i \cdot \hat{a}j) - 2b \cdot \sum_{yo=1}^{k} cia\hat{i} + b \cdot b$$

aplicando $\|v\|^2 = v \cdot v$

$$= \sum_{yo=1}^{k} c_{yo}^2 - 2ci\, b \cdot \hat{a}i + \|b\|^2$$ ya que los aˆi son ortonormales

Tenga en cuenta que el segundo paso aquí solo es válido debido a la ortonormalidad. Como mínimo, la derivada con respecto a ci es cero para cada ci , dando como resultado:

$$ci = \hat{a}i \cdot b$$

Así, hemos demostrado que cuando aˆ1, · · · , aˆk son ortonormales, se cumple la siguiente relación:

$$\text{projspan}\{\hat{a}1,\cdots,\hat{a}k\}b = (\hat{a}1 \cdot b)\hat{a}1 + \cdots + (\hat{a}k \cdot b)\hat{a}k$$

Esto es simplemente una extensión de nuestra fórmula de proyección, y por una prueba similar es fácil ver que

$$\hat{a}i \cdot (b - \text{projspan}\{\hat{a}1,\cdots,\hat{a}k\}b) = 0.$$

Es decir, hemos separado b en una componente paralela al lapso de los aˆi y un residuo perpendicular.

4.3.2 Ortogonalización de Gram-Schmidt

Nuestras observaciones anteriores conducen a un algoritmo simple para la ortogonalización, o encontrar una base ortogonal {aˆ1, · · · , aˆk} cuyo lapso es el mismo que el de un conjunto de vectores de entrada linealmente independientes {v1, · · · ,vk}:

1. conjunto

$$\hat{a}1 \equiv \frac{v1}{\|v1\|}.$$

Es decir, tomamos aˆ1 como un vector unitario paralelo a v1.

2. Para i de 2 a k,

(a) Calcule la proyección

$$pi \equiv \text{projspan}\{\hat{a}1,\cdots,\hat{a}i-1\}\,vi\,.$$

Por definición, aˆ1, · · · , aˆi−1 son ortonormales, por lo que se aplica nuestra fórmula anterior.

(b) Definir

$$\hat{a}i \equiv \frac{vi - pi}{\|vi - pi\|}.$$

Esta técnica, conocida como "ortogonalización de Gram-Schmidt", es una aplicación directa de nuestra discusión anterior. La clave para la demostración de esta técnica es notar que span {v1, · · · ,vi} = span {aˆ1, · · · , aˆi} para cada i {1, · · · , k}. El paso 1 claramente hace que este sea el caso para i = 1, y para i > 1, la definición de aˆi en el paso 2b simplemente elimina la proyección sobre los vectores que ya hemos visto.

Si comenzamos con una matriz A cuyas columnas son v1, · · · ,vk , entonces podemos implementar Gram Schmidt como una serie de operaciones de columna en A. Dividir la columna i de A por su norma es equivalente a posmultiplicar A por ak matriz diagonal × k. De manera similar, restar la proyección de una columna sobre las columnas ortonormales a su izquierda como en el paso 2 es equivalente a posmultiplicar por una matriz triangular superior: ¡Asegúrese de entender por qué es así! Por lo tanto, se aplica nuestra discusión en §4.2.1 y podemos usar Gram-Schmidt para obtener una factorización A = QR.

Desafortunadamente, el algoritmo de Gram-Schmidt puede introducir serias inestabilidades numéricas debido al paso de resta. Por ejemplo, supongamos que proporcionamos los vectores v1 = (1, 1) yv2 = (1 + ε, 1) como entrada a Gram-Schmidt para algún 0 < ε 1. Observe que una base obvia para el intervalo {v1,v2} es { (1, 0), (0, 1)}. Pero, si aplicamos Gram-Schmidt, obtenemos:

$$aˆ1 = \frac{v1}{v1} = \frac{1}{\sqrt{2}} \begin{matrix}1\\1\end{matrix}$$

$$p2 = \frac{2+\varepsilon}{2} \begin{matrix}1\\1\end{matrix}$$

$$v2 - p2 = \begin{matrix}1+\varepsilon\\1\end{matrix} - \frac{2+\varepsilon}{2} \begin{matrix}1\\1\end{matrix}$$

$$= \frac{1}{2} \begin{matrix}\varepsilon\\-\varepsilon\end{matrix}$$

Observe que v2 − p2 = ($\sqrt{2}$/2) · ε, por lo que calcular aˆ2 requerirá la división por un escalar del orden de ε. La división por números pequeños es una operación numérica inestable que debemos evitar.

4.4 Transformaciones de cabeza de familia

En §4.2.1, motivamos la construcción de la factorización QR mediante operaciones de post-multiplicación y columna. Esta construcción es razonable en el contexto del análisis de espacios de columnas, pero como vimos en nuestra derivación del algoritmo de Gram-Schmidt, las técnicas numéricas resultantes pueden ser inestables.

En lugar de comenzar con A y luego multiplicar por operaciones de columna para obtener Q = AE1 · · · Ek , sin embargo, podemos preservar nuestra estrategia de alto nivel de la eliminación gaussiana. Es decir, podemos partir de A y premultiplicar por matrices ortogonales Qi para obtener Qk · · · Q1A = R; estas Q actuarán como operaciones de fila, eliminando elementos de A hasta que el producto resultante R sea triangular superior. Entonces, gracias a la ortogonalidad de las Q podemos escribir A = QR, obteniendo Q = Q1 · · · Qk el factor QR ización ya que el producto de matrices ortogonales es ortogonal.

Las matrices de operación por filas que usamos en la eliminación gaussiana y LU no serán suficientes para la factorización QR ya que no son ortogonales. Se han sugerido varias alternativas; presentaremos una estrategia común presentada en 1958 por Alston Scott Householder.

El espacio de las matrices ortogonales n × n es muy grande, por lo que debemos encontrar un espacio más pequeño de Qi con el que sea más fácil trabajar. De nuestras discusiones geométricas en §4.2, sabemos que las matrices ortogonales deben preservar ángulos y longitudes, por lo que intuitivamente solo pueden rotar y reflejar vectores.

Afortunadamente, las reflexiones pueden ser fáciles de escribir en términos de proyecciones, como se ilustra en la Figura NÚMERO. Supongamos que tenemos un vector b que deseamos reflejar sobre un vector v. Hemos demostrado que el residuo r ≡ b − proj 2proj $_v$ b es perpendicular a v. Como en la figura NÚMERO, la diferencia $_v$ b −b refleja b sobre v.

Podemos expandir nuestra fórmula de reflexión de la siguiente manera:

$$_{2\text{proyecto}\,v}\cdot bb - b = 2\,\frac{v}{v\cdot v}\,v - b \text{ por definición de proyección}$$

$$= 2v\cdot\frac{vb}{v.v.} - b \text{ usando notación matricial}$$

$$= \frac{2vv}{v.v.} - \text{Pulgadas} \times n\ b$$

$$\equiv -Hv\ b \text{ donde se introduce el negativo para alinearlo con otros tratamientos}$$

Por lo tanto, podemos pensar en reflejar b sobre v como aplicar un operador lineal −Hv a b. Por supuesto, Hv sin el negativo sigue siendo ortogonal, así que lo usaremos de ahora en adelante.

Supongamos que estamos haciendo el primer paso de sustitución directa durante la eliminación gaussiana. Entonces, deseamos premultiplicar A por una matriz que lleve la primera columna de A, que denotaremos como, a algún múltiplo del primer vector identidade1. En otras palabras, queremos para algún c R:

$$ce1 = Hva$$

$$= \text{Pulgadas} \times n - \frac{2vv}{v.v.}\ a$$

$$= a - 2v\ \frac{\text{Virginia}}{v.v.}$$

Mover términos alrededor de espectáculos

$$v = (a - ce1)\cdot\frac{v.v.}{2va}$$

En otras palabras, v debe ser paralela a la diferencia a − ce1. De hecho, escalar v no afecta la fórmula de Hv , por lo que podemos elegir v =a − ce1. Entonces, para que nuestra relación se sostenga, debemos tener

$$1 = \frac{v.v.}{2va}$$

$$= \frac{a^{\,2} - 2ce1\cdot a + c^{\,2}}{2(a\cdot a - ce1\cdot a)}$$

$$\text{O, } 0 = un^{\,2\,2-c} = c = \pm a$$

Con esta elección de c, hemos demostrado:

$$HvA = \begin{matrix} c \times \times \times \\ 0 \times \times \times \\ \vdots\ \ \vdots\ \ \vdots\ \ \vdots \\ 0 \times \times \times \end{matrix}$$

¡Acabamos de lograr un paso similar a la eliminación directa usando solo matrices ortogonales!

Procediendo, en la notación de CITE durante el k-ésimo paso de triangularización tenemos un vector a que podemos dividir en dos componentes:

$$un = \begin{matrix} a1 \\ a2 \end{matrix}$$

Aquí, a1 Rk y a2 Rm−k . Deseamos encontrar v tal que

$$Hva = \begin{matrix} a1 \\ 0 \\ \vdots \\ 0 \end{matrix}$$

Siguiendo una derivación paralela a la anterior, es fácil demostrar que

$$v = \begin{matrix} 0 \\ a2 \end{matrix} - cek$$

realiza exactamente esta transformación cuando c = ±a2; normalmente elegimos el signo de c para evitar la cancelación haciendo que tenga signo opuesto al del k-ésimo valor ina.

El algoritmo para Householder QR es, por lo tanto, bastante sencillo. Para cada columna de A, calculamos v eliminando los elementos inferiores de la columna y aplicamos Hv a A. El resultado final es una matriz triangular superior R = Hvn · · · Hv1 A. La matriz ortogonal Q está dada por el producto H que se puede almacenar implícitamente como una lista de vectores v, que encaja en el triángulo inferior como v1 que se muestra arriba.

4.5 Factorización QR reducida

Concluimos nuestra discusión volviendo al caso más general Ax ≈ b cuando A Rm×n no es cuadrado. Note que ambos algoritmos que hemos discutido en este capítulo pueden factorizar matrices no cuadradas A en productos QR, pero el resultado es algo diferente:

- Al aplicar Gram-Schmidt, hacemos operaciones de columna en A para obtener Q por ización ortogonal. Por esta razón, la dimensión de A es la de Q, resultando Q Rm×n y R Rn×n .
- Cuando usamos reflexiones de jefe de hogar, obtenemos Q como el producto de un número de m × m matrices de reflexión, dejando R Rm×n .

Supongamos que estamos en el caso típico de los mínimos cuadrados, para los cuales m n. Seguimos prefiriendo usar el método del amo de casa debido a su estabilidad numérica, ¡pero ahora la matriz Q de m × m podría ser demasiado grande para almacenarla! Afortunadamente, sabemos que R es triangular superior. Por ejemplo, considere la estructura de una matriz R de 5 × 3:

$$R = \begin{matrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

Es fácil ver que cualquier cosa por debajo del cuadrado superior n × n de R debe ser cero, lo que produce una simplificación
catión:

$$A = QR = Q1\ Q2 \begin{matrix} R1 \\ 0 \end{matrix} = Q1R1$$

Aquí, Q1 Rm×n y R1 Rn×n todavía contienen el triángulo superior de R. Esto se llama el triángulo "reducido" QR factorización de A, ya que las columnas de Q1 contienen una base para el espacio de columnas de A en lugar de para todo Rm; ocupa mucho menos espacio. Tenga en cuenta que la discusión en §4.2.1 aún se aplica, por lo que la factorización QR reducida se puede usar para mínimos cuadrados de manera similar.

4.6 Problemas

- tridiagonalización con Jefe de Hogar
- Dados
- QR indeterminado

Capítulo 5

vectores propios

Dirigimos nuestra atención ahora a un problema no lineal sobre matrices: encontrar sus valores propios y vectores propios. Los vectores propios x y sus correspondientes valores propios λ de una matriz cuadrada A están determinados por la ecuación Ax = λx. Hay muchas maneras de ver que este problema es no lineal.
Por ejemplo, hay un producto de incógnitas λ yx, y para evitar la solución trivial x = 0 restringimos x = 1; esta restricción es circular en lugar de lineal. Gracias a esta estructura, nuestros métodos para encontrar espacios propios serán considerablemente diferentes de las técnicas para resolver y analizar sistemas de ecuaciones lineales.

5.1 Motivación

A pesar de la apariencia arbitraria de la ecuación Ax = λx, el problema de encontrar vectores propios y valores propios surge naturalmente en muchas circunstancias. Motivamos nuestra discusión con algunos ejemplos a continuación.

5.1.1 Estadísticas

Supongamos que tenemos una maquinaria para recolectar varias observaciones estadísticas sobre una colección de elementos. Por ejemplo, en un estudio médico podemos recopilar la edad, el peso, la presión arterial y la frecuencia cardíaca de 100 pacientes. Entonces, cada paciente i puede ser representado por un punto xi en R4 almacenando estos cuatro valores.

Por supuesto, tales estadísticas pueden exhibir una fuerte correlación. Por ejemplo, es probable que los pacientes con presión arterial más alta tengan un peso o una frecuencia cardíaca más altos. Por esta razón, aunque recopilamos nuestros datos en R4 , en realidad pueden, hasta cierto punto aproximado, vivir en un espacio dimensional más bajo capturando mejor las relaciones entre las diferentes variables.

Por ahora, supongamos que, de hecho, existe un espacio unidimensional que se aproxima a nuestro conjunto de datos. Entonces, esperamos que todos los puntos de datos sean casi paralelos a algún vector v, de modo que cada uno pueda escribirse como xi ≈ civ para diferentes ci R. Desde antes, sabemos que la mejor aproximación de xi paralelo a v es proj $_v$ xi :

$$\text{proyecto}_v = \frac{xi \cdot v\, xi}{v \cdot v} v \text{ por definición}$$

$$= (xi \cdot \hat{v})\hat{v} \text{ ya que } v \cdot v = v^2$$

Aquí, definimos $\hat{v} \equiv v/v$. Por supuesto, la magnitud de v no importa para el problema en cuestión, por lo que es razonable buscar en el espacio de los vectores unitarios $\hat{v}$.

Siguiendo el patrón de mínimos cuadrados, tenemos un nuevo problema de optimización:

$$\begin{aligned} &\text{minimizar} \sum_i \left\| x_i - \text{proj}_{\hat{v}} x_i \right\|^2 \\ &\text{tal que } \hat{v} = 1 \end{aligned}$$

Podemos simplificar un poco nuestro objetivo de optimización:

$$\begin{aligned} \sum_i \left\| x_i - \text{proy}_{\hat{v}} x_i \right\|^2 &= \sum_i \left\| x_i - (x_i \cdot \hat{v})\hat{v} \right\|^2 \text{ por definición de proyección} \\ &= \sum_i \|x_i\|^2 - (x_i \cdot \hat{v})^2 \text{ ya que } \hat{v} = 1 \text{ y } \|w\|^2 = w \cdot w \\ &= \text{constante} - \sum_i (x_i \cdot \hat{v})^2 \end{aligned}$$

Esta derivación muestra que podemos resolver un problema de optimización equivalente:

$$\begin{aligned} &\text{maximizar } \|X\hat{v}\|^2 \\ &\text{tal que } \|\hat{v}\|^2 = 1, \end{aligned}$$

donde las columnas de X son los vectores x_i. Observe que $\|X\hat{v}\|^2 = \hat{v}^\top X^\top X \hat{v}$, por lo que en el ejemplo 0.27 el vector $\hat{v}$ corresponde al vector propio de $X^\top X$ con el valor propio más alto. El vector $\hat{v}$ se conoce como el primer componente principal del conjunto de datos.

5.1.2 Ecuaciones diferenciales

Muchas fuerzas físicas se pueden escribir como funciones de posición. Por ejemplo, la fuerza entre dos partículas en las posiciones x e y en $\mathbb{R}^3$ ejercida por un resorte se puede escribir como $k(x - y)$ por la ley de Hooke; tales fuerzas de resorte se utilizan para aproximar las fuerzas que mantienen unida la tela en muchos sistemas de simulación. Aunque estas fuerzas no tienen necesariamente una posición lineal, a menudo las aproximamos de forma lineal. En particular, en un sistema físico con n partículas se codifican las posiciones de todas las partículas simultáneamente en un vector $X \in \mathbb{R}^{3n}$. Entonces, si asumimos tal aproximación, podemos escribir que las fuerzas en el sistema son aproximadamente $F \approx AX$ para alguna matriz A.

Recuerde la segunda ley de movimiento de Newton $F = ma$, o la fuerza es igual a la masa por la aceleración. En nuestro contexto, podemos escribir una matriz de masa diagonal $M \in \mathbb{R}^{3n \times 3n}$ que contiene la masa de cada partícula en el sistema. Entonces, sabemos $F = MX''$ donde primo denota diferenciación en el tiempo. Por supuesto, $X' = (X')'$, por lo que al final tenemos un sistema de ecuaciones de primer orden:

$$\frac{d}{dt}\begin{bmatrix} X \\ V \end{bmatrix} = \begin{bmatrix} 0 & I_{3n\times 3n} \\ M^{-1}A & 0 \end{bmatrix}\begin{bmatrix} X \\ V \end{bmatrix}$$

Aquí, calculamos simultáneamente ambas posiciones en $X \in \mathbb{R}^{3n}$ y las velocidades $V \in \mathbb{R}^{3n}$ de todas las n partículas como funciones del tiempo.

De manera más general, las ecuaciones diferenciales de la forma $x' = Ax$ aparecen en muchos contextos, incluida la simulación de telas, manantiales, calor, olas y otros fenómenos. Supongamos que conocemos los vectores propios

x1, . . . ,xk de A, tal que Axi = λixi . Si escribimos la condición inicial de la ecuación diferencial en términos de los vectores propios, como

$$x(0) = c1x1 + \cdots + ckxk ,$$

entonces la solución de la ecuación se puede escribir en forma cerrada:

$$x(t) = c1e^{\lambda 1t} x1 + \cdots + cke^{\lambda k t} xk ,$$

Esta solución es fácil de comprobar a mano. Es decir, si escribimos las condiciones iniciales de esta ecuación diferencial en términos de los vectores propios de A, entonces conocemos su solución para todos los tiempos t ≥ 0 de forma gratuita. Por supuesto, esta fórmula no es el final de la historia de la simulación: encontrar el conjunto completo de vectores propios de A es costoso y A puede cambiar con el tiempo.

5.2 Incrustación espectral

Supongamos que tenemos una colección de n elementos en un conjunto de datos y una medida wij ≥ 0 de qué tan similares son cada par de elementos i y j; supondremos wij = wji. Por ejemplo, tal vez nos den una colección de fotografías y usemos wij para comparar la similitud de sus distribuciones de color. Es posible que deseemos clasificar las fotografías en función de su similitud para simplificar la visualización y exploración de la colección.

Un modelo para ordenar la colección podría ser asignar un número xi a cada elemento i, pidiendo que a los objetos similares se les asignen números similares. Podemos medir qué tan bien una tarea agrupa objetos similares usando la energía

$$E(x) = \sum_{yo} wij\,(xi - xj)^2 .$$

Es decir, E(x) solicita que los elementos iyj con puntajes de similitud altos se asignen a valores cercanos.

Por supuesto, minimizar E(x) sin restricciones da un mínimo obvio: xi = const. por todo yo ¡Agregar una restricción x = 1 no elimina esta solución constante! En particular, tomando xi = 1/ √ n para todo i da x = 1 y E(x) = 0 de una manera poco interesante. Por lo tanto, debemos eliminar este caso también:

$$\begin{aligned} &\text{minimizar } E(x) \\ &\text{tal que } x^2 = 1 \\ &\qquad 1 \cdot x = 0 \end{aligned}$$

Observe que nuestra segunda restricción pide que la suma de x sea cero.

Una vez más podemos simplificar la energía:

$$\begin{aligned} E(x) &= \sum_{yo} wij\,(xi - xj)^2 \\ &= \sum_{yo} wij\,(x_i^2 - 2xixj + xj^2) \\ &= \sum_i aix_{yo}^2 - 2\sum_{yo} wijxixj + \sum_j bjx_j^2 \\ &\qquad \text{para } ai \equiv \sum_j wij \text{ y } bj \equiv \sum_i wij \\ &= x\,(A - 2W + B)x \text{ donde diag}(A) = a \text{ y diag}(B) = b = x\,(2A - 2W)x \text{ por} \\ &\quad \text{simetría de } W \end{aligned}$$

Es fácil comprobar que 1 es un vector propio de 2A − 2W con valor propio 0. Más interesante aún, el vector propio correspondiente al segundo valor propio más pequeño corresponde a la solución de nuestro objetivo de minimización anterior. (TODO: Agregar prueba KKT de la conferencia)

5.3 Propiedades de los vectores propios

Hemos establecido una variedad de aplicaciones que necesitan cálculo de espacio propio. Sin embargo, antes de que podamos explorar algoritmos para este propósito, examinaremos más de cerca la estructura del problema de valores propios.

Podemos comenzar con algunas definiciones que probablemente sean evidentes en este punto:

Definición 5.1 (Valor propio y vector propio). Un vector propio x = 0 de una matriz A Rn×n es cualquier vector que satisface Ax = λx para algún λ R; el λ correspondiente se conoce como valor propio. Los valores propios complejos y los vectores propios satisfacen las mismas relaciones con λ C y x Cn .

Definición 5.2 (Espectro y radio espectral). El espectro de A es el conjunto de valores propios de A. El radio espectral ρ(A) es el valor propio λ que maximiza |λ|.

La escala de un vector propio no es importante. En particular, al escalar un vector propio x por c se obtiene A(cx) = cAx = cλx = λ(cx), por lo que cx es un vector propio con el mismo valor propio. A menudo restringimos nuestra búsqueda agregando una restricción x = 1. Incluso esta restricción no elimina por completo la ambigüedad, ya que ahora ±x son ambos vectores propios con el mismo valor propio.

Las propiedades algebraicas de los vectores propios y los valores propios fácilmente podrían llenar un libro. Limitaremos nuestra discusión a algunos teoremas importantes que afectan el diseño de algoritmos numéricos; seguiremos el desarrollo de CITE AXLER. Primero, debemos verificar que cada matriz tenga al menos un vector propio para que nuestra búsqueda no sea en vano. Nuestra estrategia usual es notar que si λ es un valor propio tal que Ax = λx, entonces (A − λIn×n)x =0; por lo tanto, λ es un valor propio exactamente cuando la matriz A − λIn×n no es de rango completo.

Lema 5.1 (Teorema 2.1 de CITE). Toda matriz A Rn×n tiene al menos un vector propio (complejo).

Prueba. Tome cualquier vector x Rn\{0}. El conjunto {x, Ax, A 2x, · · · A nx} debe ser linealmente dependiente porque contiene n + 1 vectores en n dimensiones. Entonces, existen constantes c0, . . . , cn R con cn = 0 tal que

$$0 = c_0 x + c_1 A x + \cdots + c_n A^n x.$$

Podemos escribir un polinomio

$$f(z) = c_0 + c_1 z + \cdots + c_n z^n.$$

Por el Teorema Fundamental del Álgebra, existen n raíces zi C tales que f(z) =

$$c_n(z - z_1)(z - z_2)\cdots(z - z_n).$$

Entonces nosotros tenemos:

$$\begin{aligned} 0 &= c_0 x + c_1 A x + \cdots + c_n A^n x \\ &= (c_0 I_{n\times n} + c_1 A + \cdots + c_n A^n)x \\ &= c_n(A - z_1 I_{n\times n})\cdots(A - z_n I_{n\times n})x \text{ por nuestra factorización} \end{aligned}$$

Por lo tanto, al menos un A − zi In×n tiene un espacio nulo, lo que demuestra que existe v con Av = ziv, según sea necesario. □

Hay un hecho adicional que vale la pena verificar para motivar nuestra discusión sobre la computación de vectores propios. tación:

Lema 5.2 (CITE Proposición 2.2). Los vectores propios correspondientes a diferentes valores propios deben ser linealmente independientes.

Prueba. Supongamos que este no es el caso. Entonces existen vectores propios x1, · · · ,xk con valores propios distintos λ1, · · · , λk que son linealmente dependientes. Esto implica que hay coeficientes c1, . . . , ck no todo cero con 0 = c1x1 + · · · + ckxk . Si premultiplicamos por la matriz (A − λ2 In×n)· · ·(A − λk In×n), encontramos:

$$0 = (A - \lambda_2 I_{n\times n})\cdot\cdot\cdot(A - \lambda_k I_{n\times n})(c_1x_1 + \cdot\cdot\cdot + c_kx_k)$$
$$= c_1(\lambda_1 - \lambda_2)\cdot\cdot\cdot(\lambda_1 - \lambda_k)x_1 \text{ ya que } Ax_i = \lambda_i x_i$$

Dado que todos los λi son distintos, esto muestra que c1 = 0. Una prueba similar muestra que el resto de los ci tienen que ser cero, lo que contradice la dependencia lineal. □

Este lema muestra que una matriz n × n puede tener como máximo n valores propios distintos, ya que un conjunto de n valores propios produce n vectores linealmente independientes. El número máximo de vectores propios linealmente independientes correspondientes a un solo valor propio λ se conoce como la multiplicidad geométrica de λ.

Sin embargo, no es cierto que una matriz deba tener exactamente n vectores propios linealmente independientes. Este es el caso de muchas matrices, a las que llamaremos no defectuosas:

Definición 5.3 (No defectuoso). Una matriz A Rn×n no es defectuosa o es diagonalizable si sus vectores propios generan Rn .

Llamamos a tal matriz diagonalizable por la siguiente razón: si una matriz es diagonalizable, entonces tiene n vectores propiosx1, . . . ,xn Rn con valores propios correspondientes (posiblemente no únicos) λ1, . . . , n.
Tome las columnas de X como los vectores xi y defina D como la matriz diagonal con valores propios λ1, . . . , λn a lo largo de la diagonal. Entonces, por definición de valores propios tenemos AX = XD; esta es simplemente una versión "apilada" de Axi = λixi . En otras palabras,

$$D = X^{-1}AX,$$

lo que significa que A está diagonalizado por una transformación de similitud A → X −1AX:

Definición 5.4 (Matrices similares). Dos matrices A y B son semejantes si existe T con B = T −1AT.

Matrices similares tienen los mismos valores propios, ya que si Bx = λx, entonces T −1ATx = λx. De manera equivalente, A(Tx) = λ(Tx), mostrando que Tx es un vector propio con valor propio λ.

5.3.1 Matrices definidas positivas y simétricas

Como era de esperar, dada nuestra consideración especial de las matrices normales AA, las matrices simétricas y/o definidas positivas disfrutan de una estructura de vector propio especial. Si podemos verificar cualquiera de estos vínculos de propiedad, se pueden usar algoritmos especializados para extraer sus vectores propios más rápidamente.

Primero, podemos probar una propiedad de las matrices simétricas que elimina la necesidad de matrices complejas. aritmética. Comenzamos haciendo una generalización de matrices simétricas a matrices en Cn×n :

Definición 5.5 (Complejo conjugado). El complejo conjugado de un número z ≡ a + bi C es $\bar{z} \equiv a - bi$.

Definición 5.6 (Transposición conjugada). La transpuesta conjugada de A Cm×n es AH ≡ A¯.

Definición 5.7 (matriz hermitiana). Una matriz A Cn×n es hermitiana si A = A H.

Observe que una matriz simétrica A Rn×n es automáticamente hermítica porque no tiene parte compleja. Con esta ligera generalización en su lugar, podemos probar una propiedad de simetría para valores propios.
Nuestra demostración utilizará el producto escalar de vectores en Cn , dado por

$$x,y = \sum_i xi\bar{y}i ,$$

donde x,y Cn . Nótese que una vez más esta definición coincide con x ·y cuando x,y Rn . En su mayor parte, las propiedades de este producto interno coinciden con las del producto escalar en Rn , con la excepción de , un notable que v, w = w,v.

Lema 5.3. Todos los valores propios de las matrices hermitianas son reales.

Prueba. Supongamos que A Cn×n es hermitiano con Ax = λx. Escalando podemos suponer x 1. ^{2} = x, x = Entonces, tenemos:

λ = λx,x ya que x tiene norma 1
= λx,x por linealidad de ·, ·
= Ax,x ya que Ax = λx
= (Ax) x¯ por definición de = x ·, ·
(A¯x) expandiendo el producto y usando ab = a¯ ¯b = x, A Hx por definición de A ^{H} y ·, ·
H = x, Ax ya que A = A =
λ¯x,x ya que Ax = λx = λ¯ ya que x tiene norma 1

Así λ = λ¯ , lo cual puede suceder solo si λ R, según sea necesario. □

Las matrices simétricas y hermitianas también disfrutan de una propiedad de ortogonalidad especial para su eigenvec tores:

Lema 5.4. Los vectores propios correspondientes a valores propios distintos de matrices hermitianas deben ser ortogonales.

Prueba. Supongamos que A Cn×n es hermitiano, y supongamos que λ = μ con Ax = λx y Ay = μy. Por el lema anterior conocemos λ, μ R. Entonces, Ax,y = λx,y. Pero como A es hermítica también podemos escribir Ax,y = x, A Hy = x, Ay = μx,y. Así, λx,y = μx,y. Como λ = μ, debemos tener x,y = 0.

□

Finalmente, podemos enunciar sin demostración un resultado supremo del álgebra lineal, el teorema espectral. Este teorema establece que ninguna matriz simétrica o hermitiana puede ser defectuosa, lo que significa que una matriz n × n que satisface esta propiedad tiene exactamente n vectores propios ortogonales.

Teorema 5.1 (Teorema espectral). Supongamos que A Cn×n es hermítica (si A Rn×n , supongamos que es simétrica). Entonces, A tiene exactamente n vectores propios ortonormales x1, · · · ,xn con valores propios (posiblemente repetidos) λ1, . . . , n . En otras palabras, existe una matriz ortonormal X de vectores propios y una matriz diagonal D de valores propios tal que D = X AX.

Este teorema implica que cualquier vector y Rn se puede descomponer en una combinación lineal de los vectores propios de una matriz hermitiana A. Muchos cálculos son más fáciles sobre esta base, como se muestra a continuación:

Ejemplo 5.1 (Cálculo usando vectores propios). Toma x1, . . . ,xn Rn sean los vectores propios de longitud unitaria de la matriz simétrica A Rn×n . Supongamos que deseamos resolver Ay =b. Podemos escribir

$$b = c_1x_1 + \cdots + c_nx_n,$$

donde ci =b · xi por ortonormalidad. Es fácil adivinar la siguiente solución:

$$y = \frac{c_1}{\lambda_1}x_1 + \cdots + \frac{c_n}{\lambda_n}x_n.$$

En particular, encontramos:

$$\begin{aligned} Ay &= A\left(\frac{c_1}{\lambda_1}x_1 + \cdots + \frac{c_n}{\lambda_n}x_n\right) \\ &= \frac{c_1}{\lambda_1}Ax_1 + \cdots + \frac{c_n}{\lambda_n}Ax_n \\ &= c_1x_1 + \cdots + c_nx_n \\ &= b, \text{ como se desee.} \end{aligned}$$

El cálculo anterior es tanto un resultado positivo como negativo. Muestra que, dados los vectores propios de A simétrica, las operaciones como la inversión son sencillas. Por otro lado, esto significa que encontrar el conjunto completo de vectores propios de una matriz simétrica A es "al menos" tan difícil como resolver Ax =b.

Volviendo de nuestra incursión en los números complejos, volvemos a los números reales para demostrar un último hecho útil, aunque directo, sobre las matrices definidas positivas:

Lema 5.5. Todos los valores propios de matrices definidas positivas son no negativos.

Prueba. Tome A Rn×n definida positiva y suponga que Ax = λx con $\|x\|_2 = 1$. Por definición positiva, sabemos que $x^\top Ax \geq 0$. Pero, $x^\top Ax = x^\top(\lambda x) = \lambda x^\top x = \lambda$, según sea necesario. □

5.3.2 Propiedades especializadas[1]

Polinomio característico

Recuerde que el determinante de una matriz det A satisface la relación de que det A = 0 si y sólo si A es invertible. Por lo tanto, una forma de encontrar los valores propios de una matriz es encontrar las raíces del polinomio característico

$$p_A(\lambda) = \det(A - \lambda I_{n\times n}).$$

No definiremos determinantes en nuestra discusión aquí, pero la simplificación de pA revela que es un polinomio de grado n-ésimo en λ. Esto proporciona una razón alternativa por la que hay como máximo n valores propios distintos, ya que hay como máximo n raíces de esta función.

A partir de esta construcción, podemos definir la multiplicidad algebraica de un valor propio como su multiplicidad como raíz de pA. Es fácil ver que la multiplicidad algebraica es al menos tan grande como la geométrica

[1]Esta sección se puede omitir si los lectores carecen de suficientes antecedentes, pero se incluye para completar.

multiplicidad. Si la multiplicidad algebraica es 1, la raíz se llama simple, porque corresponde a un solo vector propio que es linealmente dependiente con cualquier otro. Los valores propios para los que las multiplicidades algebraicas y geométricas no son iguales se denominan defectuosos.

En análisis numérico evitamos discutir el determinante de una matriz. Si bien es una construcción teórica conveniente, su uso práctico es limitado. Los determinantes son difíciles de calcular. De hecho, los algoritmos de valores propios no intentan encontrar las raíces de pA , ya que hacerlo requeriría la evaluación de un determinante. Además, el determinante det A no tiene nada que ver con el condicionamiento de A, por lo que un determinante cercano a cero de det(A − λIn×n) podría no mostrar que λ es casi un valor propio de A.

Forma normal de Jordan

Solo podemos diagonalizar una matriz cuando tiene un espacio propio completo. Sin embargo, todas las matrices son similares a una matriz en forma normal de Jordan, que tiene la siguiente forma: • Los valores

distintos de cero están en las entradas diagonales aii y en la “superdiagonal” ai(i+1) .

- Los valores diagonales son valores propios repetidos tantas veces como su multiplicidad; la matriz es bloque diagonal sobre estos grupos.
- Los valores fuera de la diagonal son 1 o 0.

Por lo tanto, la forma se parece a la siguiente

$$\begin{bmatrix} \lambda_1 & 1 & & & & \\ & \lambda_1 & 1 & & & \\ & & \lambda_1 & & & \\ & & & \lambda_2 & 1 & \\ & & & & \lambda_2 & \\ & & & & & \lambda_3 \\ & & & & & & \ddots \end{bmatrix}$$

La forma normal de Jordan es teóricamente atractiva porque siempre existe, pero la estructura 1/0 es discreta e inestable bajo perturbaciones numéricas.

5.4 Cálculo de valores propios

El cálculo y la estimación de los valores propios de una matriz es un problema bien estudiado con muchas soluciones potenciales. Cada solución se adapta a una situación diferente, y lograr el máximo acondicionamiento o velocidad requiere experimentar con varias técnicas. Aquí, cubrimos algunas de las soluciones más populares y directas al problema del valor propio que se encuentra con frecuencia en la práctica.

5.4.1 Iteración de potencia

Por ahora, suponga que A Rn×n es simétrica. Entonces, por el teorema espectral podemos escribir vectores propios x1, . . . ,xn Rn ; los ordenamos de tal manera que sus valores propios correspondientes satisfagan |λ1| ≥ |λ2| ≥ · · · ≥ |λn|.

Supongamos que tomamos un vector arbitrario v. Dado que los vectores propios de A generan R^n, podemos escribir:

$$v = c_1x_1 + \cdots + c_nx_n.$$

Entonces,

$$\begin{aligned} Av &= c_1Ax_1 + \cdots + c_nAx_n \\ &= c_1\lambda_1x_1 + \cdots + c_n\lambda_nx_n \text{ ya que } Ax_i = \lambda_ix_i \\ \lambda_2 \lambda_n \\ \cdots + c_nx_n \lambda_1 \lambda_1 &= \lambda_1 \left(c_1x_1 + c_2x_2 + \cdots\right) \end{aligned}$$

$$A^2v = \lambda_1^2 \left(c_1x_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^2 c_2x_2 + \cdots + \left(\frac{\lambda_n}{\lambda_1}\right)^2 c_nx_n\right)$$

$$\vdots$$

$$A^kv = \lambda_1^k \left(c_1x_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k c_2x_2 + \cdots + \left(\frac{\lambda_n}{\lambda_1}\right)^k c_nx_n\right)$$

Observe que cuando $k \to \infty$, la razón $(\lambda_i/\lambda_1) \to 0$ a menos que $\lambda_i = \lambda_1$, ya que λ_1 tiene la mayor magnitud de cualquier valor propio por definición. Por lo tanto, si x es la proyección de v sobre el espacio de los vectores propios con valores propios λ_1, entonces a medida que $k \to \infty$ la siguiente aproximación es cada vez más exacta:

$$\lambda_1^k A^k v \approx x.$$

Esta observación conduce a un algoritmo extremadamente simple para calcular un vector propio x de A correspondiente al mayor valor propio λ_1:

1. Tome $v_1 \in R^n$ como un vector arbitrario distinto de cero.
2. Iterar hasta convergencia para k creciente:

$$v_k = Av_{k-1}$$

Este algoritmo, conocido como iteración de potencia, producirá vectores v_k cada vez más paralelos al x_1 deseado. Se garantiza que convergerá, incluso cuando A es asimétrica, aunque la prueba de este hecho es más complicada que la derivación anterior. La única vez que esta técnica puede fallar es si accidentalmente elegimos v_1 tal que $c_1 = 0$, pero las probabilidades de que esto ocurra son escasas o nulas.

Por supuesto, si $|\lambda_1| > 1$, entonces $v_k \to \infty$ cuando $k \to \infty$, una propiedad indeseable para la aritmética de coma flotante. Recuerde que solo nos importa la dirección del vector propio en lugar de su magnitud, por lo que la escala no tiene efecto en la calidad de nuestra solución. Por lo tanto, para evitar esta situación de divergencia, simplemente podemos normalizar en cada paso, produciendo el algoritmo de iteración de potencia normalizada:

1. Tome $v_1 \in R^n$ como un vector arbitrario distinto de cero.
2. Iterar hasta convergencia para k creciente:

$$w_k = Av_{k-1}$$

$$v_k = \frac{w_k}{\|w_k\|}$$

Note que no decoramos la norma · con un subíndice particular. Matemáticamente, cualquier norma será suficiente para evitar el problema de la divergencia, ya que hemos demostrado que todas las normas sobre Rn son equivalentes. En la práctica, a menudo usamos la norma infinita · ∞; en este caso es fácil comprobar que w k → |λ1|.

5.4.2 Iteración inversa

Ahora tenemos una estrategia para encontrar el valor propio de mayor magnitud λ1. Supongamos que A es invertible, de modo que podamos evaluar y = A −1v resolviendo Ay = v usando técnicas cubiertas en capítulos anteriores.

Si Ax = λx, entonces x = λA −1x, o de manera equivalente

$$A^{-1}x = \frac{1}{\lambda}x.$$

demostrado que 1/λ es un valor propio de A, entonces |b| ≥ |a| para A^{-1} con vector propio x. Note que si |a| ≥ |b| Así, hemos cualquier a, b R, por lo que el valor propio de menor magnitud de A es el mayor. Esta observación produce una estrategia para vector propio de magnitud de A llamado A^{-1} encontrar λn en lugar de λ1
iteración de potencia inversa:

1. Tome v1 Rn como un vector arbitrario distinto de cero.
2. Iterar hasta convergencia para k creciente:
   (a) Resolver para w k : Aw k = vk−1
   (b) Normalizar: vk = $\frac{w_k}{\|w_k\|}$

Repetidamente estamos resolviendo sistemas de ecuaciones usando la misma matriz A, que es una aplicación perfecta de las técnicas de factorización de los capítulos anteriores. Por ejemplo, si escribimos A = LU, entonces podríamos formular una versión equivalente pero considerablemente más eficiente de iteración de potencia inversa:

1. Factor A = LU
2. Tome v1 Rn como un vector arbitrario distinto de cero.
3. Iterar hasta la convergencia para aumentar k:
   (a) Resolver para yk mediante sustitución hacia adelante: Lyk = vk−1
   (b) Resolver para w k mediante sustitución hacia atrás: Uw k = yk
   (c) Normalizar: vk = $\frac{w_k}{\|w_k\|}$

5.4.3 Cambio

Supongamos que λ2 es el valor propio con la segunda magnitud más grande de A. Dada nuestra derivación original de iteración de potencia, es fácil ver que la iteración de potencia converge más rápidamente cuando |λ2/λ1| es pequeña, ya que en este caso la potencia $(\lambda_2/\lambda_1)^k$ decae rápidamente. Por el contrario, si esta relación es cercana a 1, pueden ser necesarias muchas iteraciones de iteración de potencia antes de que se aísle un solo vector propio.

Si los valores propios de A son λ1, . . . , λn, entonces es fácil ver que los valores propios de A − σIn×n son λ1 − σ, . . . , λn − σ. Entonces, una estrategia para hacer que la iteración de potencia converja rápidamente es elegir σ tal que:

$$\frac{\lambda 2 - \sigma}{\lambda 1 - \sigma} < \frac{\lambda 2}{\lambda 1} .$$

Por supuesto, adivinar tal σ puede ser un arte, ya que los valores propios de A obviamente no se conocen inicialmente. De manera similar, si pensamos que σ está cerca de un valor propio de A, entonces A − σIn×n tiene un valor propio cercano a 0 que podemos revelar por iteración inversa.

Una estrategia que hace uso de esta observación se conoce como iteración del cociente de Rayleigh. Si tenemos una conjetura fija en un vector propio x de A, entonces, por NÚMERO, la aproximación de mínimos cuadrados del valor propio σ correspondiente está dada por

$$\sigma \approx \frac{x \text{ hacha}}{X \frac{2}{2}} .$$

Esta fracción se conoce como cociente de Rayleigh. Por lo tanto, podemos intentar aumentar la convergencia iterando de la siguiente manera:

1. Tome v1 Rn como un vector arbitrario distinto de cero o una suposición inicial de un vector propio.

2. Iterar hasta convergencia para k creciente:

   (a) Escriba la estimación actual del valor propio

$$\sigma k = \frac{v\ k-1 A v k-1}{v k-1 \frac{2}{2}}$$

   (b) Resuelva para w k : (A − σk In×n)w k (c) = vk−1

   Normalice: vk = $\frac{w\ k}{w\ k}$

Esta estrategia converge mucho más rápido dada una buena suposición inicial, pero la matriz A − σk In×n es diferente en cada iteración y no se puede prefactorizar usando LU o la mayoría de las otras estrategias. Por lo tanto, se necesitan menos iteraciones, ¡pero cada iteración lleva más tiempo!

5.4.4 Encontrar valores propios múltiples

Hasta ahora, hemos descrito técnicas para encontrar un solo par de valor propio/vector propio: iteración de potencia para encontrar el valor propio más grande, iteración inversa para encontrar el valor más pequeño y cambiar a valores objetivo intermedios. Por supuesto, para muchas aplicaciones un solo valor propio no será suficiente. Gracias, podemos extender nuestras estrategias para manejar este caso también.

Deflación

Recuerde nuestra estrategia de iteración de potencia: elija un v1 arbitrario y multiplíquelo iterativamente por A hasta que solo sobreviva el valor propio más grande λ1 . Tome x1 como el vector propio correspondiente.

Sin embargo, descartamos rápidamente un modo de falla improbable de este algoritmo cuando v1 ·x1 = 0. En este caso, no importa cuántas veces premultipliques por A, nunca recuperarás un vector

paralelo a x1, ya que no puede amplificar un componente cero. La probabilidad de elegir tal v1 es exactamente cero, por lo que en todos los casos, excepto en los más perniciosos, la iteración de energía permanece segura.

Podemos darle la vuelta a este inconveniente para formular una estrategia para encontrar más de un valor propio cuando A es simétrico. Supongamos que encontramos x1 y λ1 a través de la iteración de potencia como antes. Ahora, reiniciamos la iteración de energía, pero antes de comenzar el proyecto x1 fuera de v1. Entonces, dado que los vectores propios de A son ortogonales, la iteración de potencia recuperará el segundo valor propio más grande.

Por cuestiones numéricas, puede darse el caso de que aplicando A a un vector se introduzca una pequeña componente paralela a x1. En la práctica podemos evitar este efecto proyectando en cada iteración. Al final, esta estrategia produce el siguiente algoritmo para calcular los valores propios en orden de magnitud descendente:

- Para cada valor propio deseado = 1, 2, . . .
  1. Tome v1 Rn como un vector arbitrario distinto de cero.
  2. Iterar hasta la convergencia para aumentar k: (a)

     Proyecte los vectores propios que ya hemos calculado:

     $$uk = vk-1 - projspan\{x1,...,x-1\}\ vk-1$$

     (b) Multiplique Auk = w k (c)

     Normalice: vk = $\frac{w\,k}{w\,k}$
  3. Agregue el resultado de la iteración al conjunto de xi

El ciclo interno es equivalente a la iteración de potencia en la matriz AP, donde P proyecta x1, . . . ,x−1 .
Es fácil ver que AP tiene los mismos vectores propios que A; sus valores propios son λ , . . . , λn con los valores propios restantes llevados a cero.

En términos más generales, la estrategia de deflación implica modificar la matriz A para que la iteración de potencia revele un vector propio que aún no ha calculado. Por ejemplo, AP es una modificación de A, de modo que los valores propios grandes que ya hemos calculado se eliminan.

Nuestra estrategia de proyección falla si A es asimétrica, ya que en ese caso sus vectores propios pueden no ser ortogonales. Otras estrategias de deflación menos obvias pueden funcionar en este caso. Por ejemplo, suponga que Ax1 = λ1x1 con x1 = 1. Considere que H es la matriz de jefe de hogar tal que Hx1 = e1, el primer vector de base estándar. Una vez más, las transformadas de similitud no afectan el conjunto de vectores propios, por lo que podríamos intentar conjugar por H. Considere lo que sucede cuando multiplicamos HAH pore1:

HAHe1 = HAHe1 ya que H es simétrica
= HAx1 ya que Hx1 =e1 y H = λ1Hx1 ^{2} = pulg × n
ya que Ax1 = λ1x1 = λ1e1 por definición de H

Así, la primera columna de HAH es λ1e1, mostrando que HAH tiene la siguiente estructura (CITE BREZO):

$$HAH = \begin{matrix} \lambda 1 & b \\ 0 & \text{segundo} \end{matrix} .$$

La matriz B R(n−1)×(n−1) tiene valores propios λ2, . . . , n. Por lo tanto, otra estrategia para la deflación es construir matrices B cada vez más pequeñas con cada valor propio calculado mediante iteración de potencia.

La deflación tiene el inconveniente de que debemos calcular cada vector propio por separado, lo que puede ser lento y acumular errores si los valores propios individuales no son precisos. Nuestras estrategias restantes intentan encontrar más de un vector propio a la vez.

Recuerde que las matrices similares A y B = T −1AT deben tener los mismos valores propios. Por lo tanto, un algoritmo que intente encontrar los valores propios de A puede aplicar libremente transformaciones de similitud a A. Por supuesto, aplicar T en general puede ser una proposición difícil, ya que efectivamente requeriría invertir T, por lo que buscamos matrices T cuyas inversas sean fáciles de calcular. aplicar.

Uno de nuestros motivadores para derivar la factorización QR fue que la matriz Q es ortogonal, lo que satisface Q−1 = Q. Por lo tanto, Q y Q−1 son igualmente sencillos de aplicar, lo que hace que las matrices ortogonales sean buenas opciones para las transformaciones de similitud.

Pero, ¿qué matriz ortogonal Q debemos elegir? Idealmente, Q debería involucrar la estructura de A y ser sencillo de calcular. No está claro cómo aplicar estratégicamente transformaciones simples como matrices de jefe de hogar para revelar múltiples valores propios,2 pero sí sabemos cómo generar uno de esos Q simplemente factorizando A = QR. Entonces, podríamos conjugar A por Q para encontrar: Q −1AQ = Q AQ = Q (QR)Q = (Q Q)RQ = RQ

¡Sorprendentemente, conjugar A = QR por la matriz ortogonal Q es idéntico a escribir el producto RQ!

Con base en este razonamiento, en la década de 1950, múltiples grupos de matemáticos europeos hipotetizaron dimensionó el mismo algoritmo iterativo elegante para encontrar los valores propios de una matriz A:

1. Tome A1 = A.
2. Para k = 1, 2, . . .
   (a) Factorice Ak = QkRk .
   (b) Escriba Ak+1 = RkQk .

Por nuestra derivación anterior, todas las matrices Ak tienen los mismos valores propios que A. Además, suponga que las Ak convergen en alguna A∞. Entonces, podemos factorizar A∞ = Q∞R∞, y por convergencia sabemos que A∞ = Q∞R∞ = R∞Q∞. Por NÚMERO, los valores propios de R∞ son simplemente los valores a lo largo de la diagonal de R∞, y por NÚMERO el producto R∞Q∞ = A∞ a su vez debe tener los mismos valores propios. Finalmente, por construcción, A∞ tiene los mismos valores propios que A. Entonces, hemos demostrado que si la iteración QR converge, revela los valores propios de A de manera directa.

Por supuesto, la derivación anterior asume que existe A∞ con Ak → A∞ cuando k → ∞. De hecho, la iteración QR es un método estable que garantiza la convergencia en muchas situaciones importantes, y la convergencia puede incluso mejorarse cambiando las estrategias. No derivaremos aquí las condiciones exactas, pero en cambio podemos proporcionar alguna intuición de por qué esta estrategia aparentemente arbitraria debería converger. A continuación proporcionamos algunas intuiciones para el caso simétrico A = A, que es más fácil de analizar gracias a la ortogonalidad de los vectores propios en este caso.

Supongamos que las columnas de A están dadas por a1, . . . ,an, y considere la matriz A^k para k grande Nosotros puede escribir:

$$A^k = A^{k-1} \cdot A = \begin{pmatrix} | & | & & | \\ A^{k-1}a_1 & A^{k-1}a_2 & \cdots & A^{k-1}a_n \\ | & | & & | \end{pmatrix}$$

2¡Sin embargo, las técnicas más avanzadas hacen exactamente esto!

Por nuestra derivación de iteración de potencia, la primera columna de A^k en general es paralela al vector propio x1 con la mayor magnitud $|\lambda 1|$ ya que tomamos un vector a1 y lo multiplicamos por A muchas veces.
Aplicando nuestra intuición de la deflación, supongamos que proyectamos a1 fuera de la segunda columna de A^k. Este Un vector debe ser casi paralelo a x2, ya que es el segundo valor propio más dominante. Proceder inductivamente, la factorización de A^k = QR produciría un conjunto de vectores casi propios como las columnas de Q, en orden de magnitud de valor propio decreciente, con los valores propios correspondientes a lo largo de la diagonal de R.

Por supuesto, calcular A para k grande lleva el número de condición de A a la k-ésima potencia, por lo que es probable que falle QR en la matriz resultante; esto es evidente ya que todas las columnas de A^k deberían parecerse a x1 para k grande. Sin embargo, podemos hacer la siguiente observación:

$$
\begin{aligned}
A &= Q_1R_1 \\
A^2 &= (Q_1R_1)(Q_1R_1) \\
&= Q_1(R_1Q_1)R_1 \\
&= Q_1Q_2R_2R_1 \text{ usando la notación de la iteración QR anterior, ya que } A_2 = R_1Q_1 \\
&\vdots \\
A^k &= Q_1Q_2 \cdots Q_kR_kR_{k-1} \cdots R_1
\end{aligned}
$$

Agrupar las variables Qi y las variables Ri por separado proporciona una factorización QR de A. Esperamos que las columnas de $Q_1 \cdots Q_k$ converjan a los vectores propios de A. De este modo,

Por un argumento similar, podemos encontrar

$$
\begin{aligned}
A &= Q_1R_1 \\
A_2 &= R_1Q_1 \text{ por nuestra construcción de iteración QR} \\
&= Q_2R_2 \text{ por definición de la factorización} \\
A_3 &= R_2Q_2 \text{ de la iteración QR} \\
&= Q_2^{-1}A_2Q_2 \text{ ya que } A_2 = Q_2R_2 \\
&= Q_2^{-1}R_1Q_1Q_2 \text{ ya que } A_2 = R_1Q_1 \\
&= Q_2^{-1}Q_1^{-1}AQ_1Q_2 \text{ ya que } A = Q_1R_1 \\
&\vdots \\
A_{k+1} &= Q_k^{-1} \cdots Q_1^{-1}AQ_1 \cdots Q_k \text{ inductivamente} \\
&= (Q_1 \cdots Q_k)^{-1}A(Q_1 \cdots Q_k)
\end{aligned}
$$

donde Ak es la k-ésima matriz de la iteración QR. Así, Ak+1 es simplemente la matriz A conjugada por el producto $Q_k^{-1} \equiv Q_1 \cdots Q_k$. Anteriormente argumentamos que las columnas de Q_k^{-1} convergen en los vectores propios de A. Por lo tanto, dado que la conjugación por la matriz de vectores propios produce una matriz diagonal de valores propios, sabemos que $A_{k+1} = Q_k^{-1}AQ$ tendrá valores propios aproximados de A a lo largo de su diagonal como $k \to \infty$.

Nuestra justificación de la iteración QR implicó analizar las columnas de A de la k como $k \to \infty$ como una extensión iteración de potencia. Más generalmente, para un vector$b$ $\mathbb{R}^n$podemos examinar la llamada matriz de Krylov

$$K_k = \begin{bmatrix} | & | & | & | \\ b & Ab & A^2b & \cdots & A^{k-1}b \\ | & | & | & | \end{bmatrix}.$$

Los métodos que analizan K_k para encontrar vectores propios y valores propios generalmente se conocen como métodos del subespacio de Krylov. Por ejemplo, el algoritmo de iteración de Arnoldi utiliza la ortogonalización de Gram-Schmidt para mantener una base ortogonal $\{q_1, \ldots, q_k\}$ para el espacio columna de K_k :

1. Comience tomando q_1 como un vector de norma unitaria arbitrario
2. Para $k = 2, 3, \ldots$

   (a) Takea$_k$ = Aq_{k-1} (b) Proyecte las q que ya ha calculado:

$$b_k = a_k - \text{proj}_{\text{span}\{q_1,\ldots,q_{k-1}\}} a_k$$

   (c) Vuelva a normalizar para encontrar el siguiente $q_k = b_k/\|b_k\|$.

La matriz Q_k cuyas columnas son los vectores que se encuentran arriba es una matriz ortogonal con el mismo espacio de columnas que K_k , y las estimaciones de valores propios se pueden recuperar de la estructura de $Q_k^* A Q_k$. El uso de Gram-Schmidt hace que esta técnica sea inestable y el tiempo empeora progresivamente a medida que aumenta k, sin embargo, se necesitan muchas extensiones para que sea factible. Por ejemplo, una estrategia consiste en ejecutar algunas iteraciones de Arnoldi, usar la salida para generar una estimación mejor del q_1 inicial y reiniciar. Los métodos de esta clase son adecuados para problemas que requieren múltiples vectores propios en uno de los extremos del espectro sin calcular el conjunto completo.

5.5 Sensibilidad y condicionamiento

Como advertimos, solo hemos esbozado algunas técnicas de valor propio de una literatura rica y de larga data. Se ha experimentado con casi cualquier técnica algorítmica para encontrar espectros, desde métodos iterativos para encontrar raíces en el polinomio característico hasta métodos que dividen matrices en bloques para procesamiento paralelo.

Al igual que en los solucionadores lineales, podemos evaluar el condicionamiento de un problema de valores propios independientemente de la técnica de solución. Este análisis puede ayudar a comprender si un esquema iterativo simplista tendrá éxito para encontrar los vectores propios de una matriz determinada o si se necesitan métodos más complejos; es importante notar que el condicionamiento de un problema de valores propios no es lo mismo que el número de condición de la matriz para resolver sistemas, ya que estos son problemas separados.

Supongamos que una matriz A tiene un vector propio x con valor propio λ. Analizar el condicionamiento del problema de valores propios implica analizar la estabilidad de x y λ ante perturbaciones en A. Con este fin, podríamos perturbar A mediante una pequeña matriz δA, cambiando así el conjunto de vectores propios. En particular, podemos escribir los vectores propios de $A + \delta A$ como perturbaciones de los vectores propios de A resolviendo el problema

$$(A + \delta A)(x + \delta x) = (\lambda + \delta\lambda)(x + \delta x).$$

Expandiendo ambos lados se obtiene:

$$Ax + A\delta x + \delta A \cdot x + \delta A \cdot \delta x = \lambda x + \lambda \delta x + \delta \lambda \cdot x + \delta \lambda \cdot \delta x$$

Suponiendo que δA es pequeño, supondremos3 que δx y δλ también son pequeños. Los productos entre estas variables son despreciables, lo que da la siguiente aproximación:

$$Ax + A\delta x + \delta A \cdot x \approx \lambda x + \lambda \delta x + \delta \lambda \cdot x$$

Como Ax = λx, podemos restar este valor de ambos lados para encontrar:

$$A\delta x + \delta A \cdot x \approx \lambda \delta x + \delta \lambda \cdot x$$

Ahora aplicamos un truco analítico para completar nuestra derivación. Como Ax = λx, sabemos que (A − λIn×n)x =0, por lo que A − λIn×n no es de rango completo. La transpuesta de una matriz es de rango completo solo si la matriz es de rango completo, por lo que sabemos que (A − λIn×n) = A − λIn×n también tiene un vector espacial nulo y. Así A y = λy; podemos llamar a y el vector propio izquierdo correspondiente a x. Podemos multiplicar por la izquierda nuestra estimación de perturbación anterior por y :

$$y\,(A\delta x + \delta A \cdot x) \approx y\,(\lambda \delta x + \delta \lambda \cdot x)$$

Como A y = λy, podemos simplificar:

$$y\,\delta A \cdot x \approx \delta \lambda y\, x$$

Reordenando rendimientos:

$$\approx \quad \frac{y\,(\delta A)x\,\delta \lambda}{yx}$$

Supongamos que x = 1 y y = 1. Entonces, si tomamos normas en ambos lados, encontramos:

$$|\delta \lambda| \quad \frac{\delta A 2\,|}{y \cdot x|}$$

Entonces, en general, el condicionamiento del problema de valores propios depende del tamaño de la perturbación δA, como se esperaba, y del ángulo entre los vectores propios izquierdo y derecho x e y. Podemos usar 1/x·y como un número de condición aproximado. Observe que x = y cuando A es simétrico, lo que produce un número de condición de 1; esto refleja el hecho de que los vectores propios de las matrices simétricas son ortogonales y, por lo tanto, están separados al máximo.

5.6 Problemas

3¡Esta suposición debería ser revisada en un tratamiento más riguroso!

Capítulo 6

Valor singular de descomposición

En el Capítulo 5, derivamos una serie de algoritmos para calcular los valores propios y los vectores propios de las matrices A Rn×n . Habiendo desarrollado esta maquinaria, completamos nuestra discusión inicial de álgebra lineal numérica derivando y haciendo uso de una factorización matricial final que existe para cualquier matriz A Rm×n : la descomposición en valores singulares (SVD).

6.1 Derivación de la SVD

Para A Rm×n , podemos pensar en la función x → Ax como un mapa que lleva puntos en Rn a puntos en Rm. Desde esta perspectiva, podríamos preguntarnos qué sucede con la geometría de Rn en el proceso y, en particular, el efecto que tiene A sobre las longitudes y los ángulos entre vectores.

Aplicando nuestro punto de partida habitual para los problemas de valores propios, podemos preguntarnos el efecto que tiene A sobre las longitudes de los vectores examinando los puntos críticos de la relación

$$R(x) = \frac{\text{Hacha}}{x}$$

sobre varios valores de x. Escalar x no importa, ya que

$$R(\alpha x) = \frac{A \cdot \alpha x}{\alpha x} = \frac{|a|}{|a|} \cdot \frac{\text{Hacha}}{x} = \frac{\text{Hacha}}{x} = R(x).$$

Por lo tanto, podemos restringir nuestra búsqueda a x con x = 1. Además, dado que R(x) ≥ 0, podemos considerar [R(x)]2 = Ax = x A Ax. Sin embargo, como hemos mostrado en capítulos anteriores, los puntos críticos de x A Ax sujetos a x = 1 son exactamente los vectores propios xi que satisfacen A Axi = λixi ; observe λi ≥ 0 y xi · xj = 0 cuando i = j ya que AA es simétrica y semidefinida positiva.

Según nuestro uso de la función R, la base {xi} es razonable para estudiar los efectos geométricos de A. Volviendo a este objetivo original, defina yi ≡ Axi . Podemos hacer una observación adicional sobre yi que revela una estructura de valores propios aún más fuerte:

$$\begin{aligned}
\lambda_i y_i &= \lambda_i \cdot Ax_i && \text{por definición de } y_i \\
&= A(\lambda_i x_i) \\
&= A(A\,Ax_i) && \text{ya que } x_i \text{ es un vector propio de } AA \\
&= (AA)(Ax_i) && \text{por asociatividad} \\
&= (AA)y_i
\end{aligned}$$

Así, tenemos dos casos:

1. Cuando λi = 0, entonces yi = 0. En este caso, xi es un vector propio de AA y yi = Axi es AAxi = un autovector correspondiente de AA con yi = Axi = Axi √ λixi.

2. Cuando λi = 0, yi =0.

Una demostración idéntica muestra que si y es un vector propio de AA, entonces x ≡ A y es cero o un vector propio de AA con el mismo valor propio.

Tome k como el número de valores propios estrictamente positivos λi > 0 discutidos anteriormente. Por nuestra construcción anterior, podemos tomar x1, . . . ,xk Rn sean los vectores propios de AA y los vectores propios correspondientes y1, . . . ,yk Rm de AA tal que

$$A\,Axi = \lambda i xi$$

$$AAyi = \lambda i yi$$

para valores propios λi > 0; aquí normalizamos tal que xi = yi = 1 para todo i. Siguiendo la notación tradicional, podemos definir las matrices V Rn×k y U Rm×k cuyas columnas son xi 's y yi 's, resp.

Podemos examinar el efecto de estas nuevas matrices de base en A. Tome ei como el i-ésimo vector de base estándar. Entonces,

$$\begin{aligned} U^{T} AV e_i &= U^{T} Axi \text{ por definición de } V \\ &= \frac{1}{\sqrt{\lambda i}} U^{T} A(\lambda i xi) \text{ ya que asumimos } \lambda i > 0 \ \lambda i\ 1 \\ &= \frac{1}{\sqrt{\lambda i}} U^{T} A(A\,Axi) \text{ ya que xi es un vector propio de AA } \lambda i\ 1 \\ &= \frac{1}{\sqrt{\lambda i}} U^{T} (AA)Axi \text{ por asociatividad } \lambda i\ 1 \\ &= \frac{\sqrt{\lambda i}}{\sqrt{\lambda i}\sqrt{\lambda i}} U^{T} (AA)yi \text{ ya que reescalamos para que yi = 1 } \sqrt{\lambda i} = \\ &\text{yi ya que AAyi = } \lambda i yi \\ &= \lambda i ei \end{aligned}$$

Tome $\Sigma = \mathrm{diag}(\sqrt{\lambda 1}, \ldots, \sqrt{\lambda k})$. Entonces, la derivación anterior muestra que $U^{T} AV = \Sigma$.

Complete las columnas de U y V hasta U Rm×m y V Rn×n sumando los vectores ortonormales xi y yi con A Axi =0 y AAyi =0, resp. En este caso es fácil mostrar UAVei =0 y/o UAV =0. Así, si tomamos

$$\Sigma ij \equiv \begin{cases} \sqrt{\lambda i}\ i = j \text{ y } i \le k \\ 0 \quad \text{de lo contrario} \end{cases}$$

entonces podemos extender nuestra relación anterior para mostrar UAV = Σ, o de manera equivalente

$$A = U\Sigma V.$$

Esta factorización es exactamente la descomposición en valores singulares (SVD) de A. Las columnas de U abarcan el espacio de columnas de A y se denominan vectores singulares por la izquierda; las columnas de V abarcan su espacio de fila y son los vectores singulares correctos. Los elementos diagonales σi de Σ son los valores singulares de A; por lo general, se ordenan de manera que σ1 ≥ σ2 ≥ · · · ≥ 0. Tanto U como V son matrices ortogonales.

El SVD proporciona una caracterización geométrica completa de la acción de A. Dado que U y V son ortogonales, se pueden considerar como matrices de rotación; como matriz diagonal, Σ simplemente escala las coordenadas individuales. Así, todas las matrices A Rm×n son una composición de una rotación, una escala y una segunda rotación.

6.1.1 Cálculo de la SVD

Recuerde que las columnas de V simplemente son los vectores propios de AA, por lo que pueden calcularse usando las técnicas discutidas en el capítulo anterior. Como A = UΣV sabemos que AV = UΣ. Por lo tanto, las columnas de U correspondientes a valores singulares distintos de cero en Σ simplemente son columnas normalizadas de AV; las columnas restantes satisfacen AAui =0, que se puede resolver mediante la factorización LU.

Esta estrategia no es de ninguna manera el enfoque más eficiente o estable para calcular el SVD, pero funciona razonablemente bien para muchas aplicaciones. Omitiremos enfoques más especializados para encontrar el SVD, pero tenga en cuenta que muchas son simples extensiones de la iteración de potencia y otras estrategias que ya hemos cubierto que operan sin formar AA o AA explícitamente.

6.2 Aplicaciones de la SVD

Dedicamos el resto de este capítulo a presentar muchas aplicaciones de la SVD. El SVD aparece innumerables veces tanto en la teoría como en la práctica del álgebra lineal lineal numérica, y su importancia difícilmente puede exagerarse.

6.2.1 Resolución de Sistemas Lineales y Pseudoinversos

En el caso especial donde A Rn×n es cuadrado e invertible, es importante notar que la SVD puede usarse para resolver el problema lineal Ax =b. En particular, tenemos UΣV x =b, o

$$x = V\Sigma^{-1}U\,b.$$

En este caso Σ es una matriz diagonal cuadrada, entonces Σ^{-1} simplemente es la matriz cuyas entradas diagonales son $1/\sigma_i$.

Calcular la SVD es mucho más costoso que la mayoría de las técnicas de solución lineal que presentamos en el Capítulo 2, por lo que esta observación inicial es principalmente de interés teórico. Más generalmente, supongamos que deseamos encontrar una solución de mínimos cuadrados para Ax ≈ b, donde A Rm×n no es necesariamente cuadrado. De nuestra discusión de las ecuaciones normales, sabemos que x debe satisfacer A Ax = A b. Hasta ahora, en su mayoría hemos descartado el caso cuando A es "corto" o "indeterminado", es decir, cuando A tiene más columnas que filas. En este caso la solución a las ecuaciones normales no es única.

Para cubrir los tres casos, podemos resolver un problema de optimización de la siguiente forma:

$$\begin{aligned} &\text{minimizar } \|x\|^2 \\ &\text{tal que } A^\top Ax = A^\top b \end{aligned}$$

En palabras, esta optimización pide que x satisfaga las ecuaciones normales con la norma mínima posible.
Ahora, escribamos A = UΣV. Entonces,

$$\begin{aligned} AA &= (U\Sigma V)\,(U\Sigma V) \\ &= V\Sigma\, U\, U\Sigma V \text{ ya que } (AB) = BA \\ &= V\Sigma\, \Sigma V \text{ ya que U es ortogonal} \end{aligned}$$

Por lo tanto, pedir que A Ax = A b es lo mismo que preguntar

$$V\Sigma\, \Sigma V\, x = V\Sigma U\, b$$

O de manera equivalente, $\Sigma y = d$

si tomamos $d \equiv Ub$ y $y \equiv V x$. Observe que $y = x$ ya que U es ortogonal, por lo que nuestra optimización se convierte en:

$$\begin{aligned} &\text{minimizar } \|y\|_2 \\ &\text{tal que } \Sigma y = d \end{aligned}$$

Sin embargo, dado que Σ es diagonal, la condición $\Sigma y = d$ simplemente establece $\sigma_i y_i = d_i$; entonces, siempre que $\sigma_i = 0$ debemos tener $y_i = d_i/\sigma_i$. Cuando $\sigma_i = 0$, no hay restricción sobre y_i , así que como estamos minimizando y, también podemos tomar $y_i = 0$. En otras palabras, la solución a esta optimización es $y = \Sigma^+ d$, donde $\Sigma^+ \in \mathbb{R}^{n\times m}$ tiene la siguiente forma:

$$\Sigma^+_{ij} \equiv \begin{cases} 1/\sigma_i & i = j,\ \sigma_i = 0,\ \text{y } i \le k \\ 0 & \text{de lo contrario} \end{cases}$$

Esta forma a su vez produce $x = Vy = V\Sigma^+ d = V\Sigma^+ Ub$.
Con esta motivación, hacemos la siguiente definición:

Definición 6.1 (Pseudoinversa). La pseudoinversa de $A = U\Sigma V \in \mathbb{R}^{m\times n}$ es $A^+ \equiv V\Sigma^+ U \in \mathbb{R}^{n\times m}$.

Nuestra derivación anterior muestra que la pseudoinversa de A disfruta de las siguientes propiedades:

- Cuando A es cuadrada e invertible, $A^+ = A^{-1}$.
- Cuando A está sobredeterminado, $A^+ b$ da la solución de mínimos cuadrados para $Ax \approx b$.
- Cuando A está subdeterminado, $A^+ b$ da la solución de mínimos cuadrados para $Ax \approx b$ con mínimo (Norma euclidiana.

De esta manera, finalmente podemos unificar los casos indeterminado, completamente determinado y sobredeterminado de $Ax \approx b$.

6.2.2 Descomposición en productos externos y aproximaciones de rango bajo

Si desarrollamos el producto $A = U\Sigma V$, es fácil mostrar que esta relación implica:

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T ,$$

donde ≡ min{m, n}, y ui y vi son las i-ésimas columnas de U y V, resp. Nuestra suma solo llega a min{m, n} ya que sabemos que las columnas restantes de U o V serán puestas a cero por Σ.

Esta expresión muestra que cualquier matriz se puede descomponer como la suma de los productos exteriores de vectores:

Definición 6.2 (Producto exterior). El producto exterior de u Rm y v Rn es la matriz u v ≡ uv Rm×n .

Supongamos que deseamos escribir el producto Ax. Entonces, en su lugar, podríamos escribir:

$$\text{hacha} = \sum_{yo=1} \sigma_i u_i v_i \text{ yo } x$$

$$= \sum_{yo=1} \sigma_i u_i (v_i \ X)$$

$$= \sum_{yo=1} \sigma_i (v_i \cdot x) u_i \text{ ya que } x \cdot y = xy$$

Entonces, aplicar A ax es lo mismo que combinar linealmente los vectores ui con pesos σi(vi · x). Esta estrategia para calcular Ax puede proporcionar ahorros considerables cuando el número de valores de σi distintos de cero es relativamente pequeño. Además, podemos ignorar los valores pequeños de σi , truncando efectivamente esta suma para aproximar Ax con menos trabajo.

De manera similar, a partir de §6.2.1 podemos escribir la pseudoinversa de A como:

$$A^{+} = \sum_{\sigma_i=0} \frac{v_i u_i^{\text{yo}}}{\sigma_i}.$$

Obviamente podemos aplicar el mismo truco para evaluar A +x, y de hecho podemos aproximar A +x evaluando solo aquellos términos en la suma para los cuales σi es relativamente pequeño. En la práctica, calculamos los valores singulares σi como raíces cuadradas de los valores propios de AA o AA, y se pueden usar métodos como la iteración de potencia para revelar un conjunto parcial de valores propios en lugar de un conjunto completo. Por lo tanto, si vamos a tener que resolver una serie de problemas de mínimos cuadrados Axi ≈ bi para diferentes bi y estamos satisfechos con una aproximación de xi , puede ser valioso calcular primero los valores más pequeños de σi y usar la aproximación anterior. Esta estrategia también evita tener que calcular o almacenar la matriz A + completa y puede ser precisa cuando A tiene una amplia gama de valores singulares.

Volviendo a nuestra notación original A = UΣV, nuestro argumento anterior muestra efectivamente que una aproximación potencialmente útil de A es A˜ ≡ UΣ˜ V donde Σ˜ redondea los valores pequeños de Σ a cero. Es fácil comprobar que el espacio columna de A˜ tiene una dimensión igual al número de valores distintos de cero en la diagonal de Σ˜. De hecho, esta aproximación no es una estimación ad hoc sino que resuelve un problema de optimización difícil publicado por el famoso siguiente teorema (enunciado sin demostración):

Teorema 6.1 (Eckart-Young, 1936). Supongamos que A se obtiene de A = UΣV truncando todo menos A − A˜Fro y más grandes σi de A a cero. Entonces A˜ minimiza ambas restricciones A − A˜2 sujeto a los k valores singulares de que el espacio de columna de A tiene como máximo la dimensión k.

6.2.3 Normas de matriz

La construcción de la SVD también nos permite volver a nuestra discusión de las normas de matriz de §3.3.1. Por ejemplo, recuerde que definimos la norma de Frobenius de A como

$$\|A\|_{\text{Para}}^2 \equiv \sum_{yo} un_{ij}^2.$$

Si escribimos A = UΣV , podemos simplificar esta expresión:

$$\begin{aligned}
\|A\|_{\text{Para}}^2 &= \sum_j \|Aej\|^2 \text{ ya que este producto es la j-ésima columna de A} \\
&= \sum_j \|U\Sigma Vej\|_2^2, \text{ sustituyendo la SVD} \\
&= \sum_j \|V\Sigma Vej\|_2^2 \text{ desde } \|x\|_2^2 = xx \text{ y U es ortogonal} \\
&= \|\Sigma V\|_F^2 \text{ Por la misma lógica} \\
&= \|V\Sigma\|_{\text{Para}}^2 \text{ ya que una matriz y su transpuesta tienen la misma norma de Frobenius} \\
&= \sum_j \|V\Sigma ej\|^2 = \sum_j \sigma_j^2 \|Vej\|^2 \text{ por la diagonal de } \Sigma \\
&= \sum_j \sigma_j^2 \text{ ya que V es ortogonal}
\end{aligned}$$

Así, la norma de Frobenius de A Rm×n es la suma de los cuadrados de sus valores singulares.

Este resultado es de interés teórico, pero en la práctica, la definición básica de la norma de Frobe nius ya es sencilla de evaluar. Más interesante aún, recuerde que la norma dos inducida de A está dada por

$$\|A\|_2^2 = \max\{\lambda : \text{existe } x \quad R \quad \text{con } A\ Ax = \lambda x\}.$$

Ahora que hemos estudiado los problemas de valores propios, nos damos cuenta de que este valor es la raíz cuadrada del valor propio más grande de AA, o de manera equivalente

$$\|A\|_2 = \max\{\sigma_i\}.$$

En otras palabras, podemos leer la norma de dos de A directamente de sus valores propios.

De manera similar, recuerde que el número de condición de A está dado por cond A = A2A −12. Por nuestro derivación de A +, los valores singulares de A $^{-1}$ deben ser los recíprocos de los valores singulares de A. Combinando esto con nuestra simplificación de rendimientos A2 :

$$\text{cond } A = \frac{\sigma_{\max}}{\sigma_{\min}}.$$

Esta expresión produce una estrategia para evaluar el condicionamiento de A. Por supuesto, calcular σmin requiere resolver sistemas Ax = b, un proceso que en sí mismo puede sufrir de un mal condicionamiento de A; si esto es un problema, el condicionamiento se puede acotar y aproximar usando varias aproximaciones de los valores singulares de A.

6.2.4 El problema de Procrustes y la alineación

Muchas técnicas de visión artificial implican la alineación de formas tridimensionales. Por ejemplo, supongamos que tenemos un escáner tridimensional que recopila dos nubes de puntos del mismo objeto rígido desde diferentes puntos de vista. Una tarea típica podría ser alinear estas dos nubes de puntos en un solo marco de coordenadas.

Dado que el objeto es rígido, esperamos que haya alguna matriz de rotación R y traslaciónt R3 tal que rotar la primera nube de puntos por R y luego trasladar por byt alinea los dos conjuntos de datos.
Nuestro trabajo es estimar estos dos objetos.

Si los dos escaneos se superponen, el usuario o un sistema automatizado puede marcar n puntos correspondientes que correspondan entre los dos escaneos; podemos almacenarlos en dos matrices X1, X2 R3×n . Entonces, para cada columna x1i de X1 y x2i de X2, esperamos Rx1i +t = x2i . Podemos escribir una función de energía que mida cuánto se cumple esta relación:

$$mi \equiv \sum_i \quad Rx1i + t - x2i \qquad ^2.$$

Si fijamos R y minimizamos con respecto a tot, la optimización de E obviamente se convierte en un problema de mínimos cuadrados. Ahora, supongamos que optimizamos para R sin fijar. Esto es lo mismo que minimizar $^t_{2\ \text{para,}}$ RX1 − X donde las columnas$_2$ son los de X2 traducidos port, sujeto a que R sea una matriz de rotación de 3 × 3, de X es decir, que RR = I3×3. Esto se conoce como el problema ortogonal de Procrustes.

Para resolver este problema, introduciremos la traza de una matriz cuadrada de la siguiente manera:

Definición 6.3 (Rastro). La traza de A Rn×n es la suma de su diagonal:

$$tr(A) \equiv \sum_i \quad ai.$$

Es sencillo comprobar que A $^2_{\text{Para}}$ = tr(A A). Por lo tanto, podemos simplificar E de la siguiente manera:

$$\begin{aligned} RX1 - X \ {}^t_2 \ {}^2_{\text{Para}} &= tr((RX1 - X \ {}^t_2)\,(RX1 - X \ {}^t_2)) \\ &= tr(X\,1 \ X1 - X \ {}_1 \ RX \ {}^t_2 \ {}^t_2 \ X \ {}^t \ RX1 + X \ {}^t_{2X2}) \\ &= \text{constante} - 2tr(X\,2 \\ &\qquad RX1) \text{ ya que } tr(A + B) = tr\,A + tr\,B \text{ y } tr(A\ ) = tr(A) \end{aligned}$$

Por lo tanto, deseamos maximizar tr(X $^t_2$RX1) con RR = I3×3. En los ejercicios demostrarás que tr(AB) = tr(BA). Por lo tanto, nuestro objetivo puede simplificarse ligeramente a tr(RC) con C ≡ X1X $^t_2$. Aplicando la SVD, si descomponemos C = UΣV entonces podemos simplificar aún más:

$$\begin{aligned} tr(RC) = tr(RU\Sigma V\ ) &\text{ por definición } = tr((V \\ &RU)\Sigma) \text{ ya que } tr(AB) = tr(BA) = tr(\tilde{R}\Sigma) \text{ si} \\ &\text{definimos } \tilde{R} = V\,RU, \text{ que también es ortogonal} = \sum_i \sigma i \tilde{r}ii \text{ ya que } \Sigma \text{ es} \\ &\text{diagonal} \end{aligned}$$

Como R˜ es ortogonal, todas sus columnas tienen longitud unitaria. Esto implica que r˜ii ≤ 1, ya que de lo contrario la norma de la columna i sería demasiado grande. Dado que σi ≥ 0 para todo i, este argumento muestra que podemos maximizar tr(RC) tomando R˜ = I3×3. Al deshacer nuestras sustituciones, se muestra R = VRU˜ = VU.

De manera más general, hemos mostrado lo siguiente:

Teorema 6.2 (Procusto ortogonal). La matriz ortogonal R que minimiza RX − Y VU, donde SVD ^{2} es dado por se aplica al factor XY = UΣV .

Volviendo al problema de la alineación, una estrategia típica es un enfoque alternativo:

1. Fijar R y minimizar E con respecto a tot.
2. Fijar la resultantet y minimizar E con respecto a R sujeta a RR = I3×3.
3. Regrese al paso 1.

La energía E disminuye con cada paso y por lo tanto converge a un mínimo local. Dado que nunca optimizamos t y R simultáneamente, no podemos garantizar que el resultado sea el valor más pequeño posible de E, pero en la práctica este método funciona bien.

6.2.5 Análisis de componentes principales (PCA)

Recuerde la configuración de §5.1.1: Deseamos encontrar una aproximación de baja dimensión de un conjunto de puntos de datos, que podemos almacenar en una matriz X Rn×k para k observaciones en n dimensiones. Previamente mostramos que si se nos permite una sola dimensión, la mejor dirección posible viene dada por el vector propio dominante de XX.

Supongamos que, en cambio, se nos permite proyectar en el intervalo de d vectores con d ≤ min{k, n} y deseamos elegir estos vectores de manera óptima. Podríamos escribirlos en una matriz C de n × d; ya que podemos aplicar Gram-Schmidt a cualquier conjunto de vectores, podemos suponer que las columnas de C son ortonormales, mostrando CC = Id×d . Dado que C tiene columnas ortonormales, por las ecuaciones normales la proyección de X sobre el espacio de columnas de C viene dada por CCX.

En esta configuración, deseamos minimizar X − CCXFro sujeto a CC = Id×d . podemos simplificar nuestro problema un poco:

$$\begin{aligned} \|X - CCX\|_{\text{Para}}^2 &= \operatorname{tr}((X - CCX)^\top (X - CCX)) \text{ ya que } A = \|A\|_{\text{Para}}^2 = \operatorname{tr}(A^\top A) \\ &= \operatorname{tr}(X^\top X - 2X^\top CCX + X^\top CCCCX) = \\ &= \text{const.} - \operatorname{tr}(X^\top CCX) \text{ ya que } CC = I_{d\times d} \\ &= -\|CX\|_{\text{Para}}^2 + \text{const.} \end{aligned}$$

Entonces, de manera equivalente podemos ^{2} para; para los estadísticos, esto muestra cuándo las filas de X tienen maximizar CX media cero que deseamos maximizar la varianza de la proyección C X.

Ahora, supongamos que factorizamos X = UΣV . Entonces, deseamos maximizar C UΣV = C˜ΣFro = Fro Σ˜ CFro por la ortogonalidad de V si tomamos C˜ = CU. Si los elementos de C˜ son c˜ij, al expandir esta norma se obtiene

$$\|\tilde{C}\Sigma\|_{\text{Para}}^2 = \sum_i \sigma_i^2 \sum_j \tilde{c}_{ij}^2.$$

Por la ortogonalidad de las columnas de C˜, sabemos que ∑i c˜ ^{2} = 1 para todo j y, dado que C˜ puede tener menos de n columnas, ∑j c˜ ^{2} es como máximo 1 en la suma anterior, por lo que si entonces claramente el máximo se logra tomando Al deshacer nuestro cambio de , las columnas de C˜ para ordenar de tal manera que σ1 ≥ σ2 ≥ · · · bee1, . . . ,ed . coordenadas, vemos que nuestra elección de C debería ser las primeras d columnas de U.

Hemos demostrado que la SVD de X se puede utilizar para resolver un problema de análisis de componentes principales (PCA). En la práctica, las filas de X generalmente se desplazan para tener media cero antes de realizar el SVD; como se muestra en la Figura NÚMERO, esto centra el conjunto de datos sobre el origen, proporcionando vectores PCA ui más significativos .

6.3 Problemas

Parte III

Técnicas no lineales

Capítulo 7

Sistemas no lineales

Por mucho que lo intentemos, simplemente no es posible expresar todos los sistemas de ecuaciones en el marco de trabajo lineal que hemos desarrollado en los últimos capítulos. No es necesario motivar el uso de logaritmos, exponenciales, funciones trigonométricas, valores absolutos, polinomios, etc. en problemas prácticos, pero excepto en unos pocos casos especiales, ninguna de estas funciones es lineal. Cuando aparecen estas funciones, debemos emplear un conjunto de maquinaria más general aunque menos eficiente.

7.1 Problemas de una sola variable

Comenzamos nuestra discusión considerando problemas de una sola variable escalar. En particular, dada una función $f(x) : R \rightarrow R$, deseamos desarrollar estrategias para encontrar puntos $x \in R$ tales que $f(x) = 0$; llamamos x a raíz de f. Los problemas de una sola variable en álgebra lineal no son particularmente = b/a. Sin embargo, resolver un cos y − interesante; después de todo, podemos resolver la ecuación $ax - b = 0$ en forma cerrada como $x = 3 = 0$ es mucho ecuación no lineal como y es $^{2 + mi}$ menos obvio (por cierto, la solución $y = \pm 1.30246\ldots$).

7.1.1 Caracterización de problemas

Ya no podemos suponer que f es lineal, pero sin ninguna suposición sobre su estructura es poco probable que avancemos en la resolución de sistemas de una sola variable. Por ejemplo, se garantiza que un solucionador no encontrará ceros de f(x) dados por

$$f(x) = \begin{array}{l} -1 \; x \leq 0 \\ 1 \; x > 0 \end{array}$$

O peor:

$$f(x) = \begin{array}{l} -1 \; x \in Q \\ 1 \; \text{de lo contrario} \end{array}$$

Estos ejemplos son triviales en el sentido de que es poco probable que un cliente racional de software de búsqueda de raíces espere que tenga éxito en este caso, pero los casos mucho menos obvios no son mucho más difíciles. para construir.

Por esta razón, debemos agregar algunos supuestos de "regularización" acerca de que f proporciona un punto de apoyo en la posibilidad de diseñar técnicas de búsqueda de raíces. Estos supuestos típicos se encuentran a continuación, enumerados en orden creciente de fuerza:

- Continuidad: Una función f es continua si se puede dibujar sin levantar un bolígrafo; más formalmente, f es continua si la diferencia f(x) − f(y) desaparece cuando x → y.
- Lipschitz: Una función f es Lipschitz continua si existe una constante C tal que | f(x) − f(y)| ≤ C|x − y|; Las funciones de Lipschitz no necesitan ser diferenciables pero están limitadas en sus tasas de cambio.
- Diferenciabilidad: Una función f es derivable si su derivada f existe para todo x.
- C : Una función es C ∞ indica que todas las derivadas de f existen y son continuas; C k indica que existen k derivadas sucesivas y cada una de esas k derivadas es continua;

A medida que agregamos suposiciones cada vez más sólidas sobre f, podemos diseñar algoritmos más efectivos para resolver f(x) = 0. Ilustraremos este efecto considerando algunos algoritmos a continuación.

7.1.2 Continuidad y bisección

Supongamos que todo lo que sabemos sobre f es que es continua. En este caso, podemos enunciar un teorema intuitivo del cálculo estándar de una sola variable:

Teorema 7.1 (Teorema del valor intermedio). Supongamos que f : [a, b] → R es continua. Supongamos que f(x) < u < f(y). Entonces, existe z entre xey tal que f(z) = u.

En otras palabras, la función f debe alcanzar todos los valores entre f(x) y f(y).

Supongamos que se nos da como entrada la función f así como dos valores y r tales que f() · f(r) < 0; fíjate que esto significa que f() y f(r) tienen signos opuestos. Entonces, por el Teorema del Valor Intermedio sabemos que en algún lugar entre y r hay una raíz de f ! Esto proporciona una estrategia de bisección obvia para encontrar x

1. Calcule c = +r/2.
2. Si f(c) = 0, devuelve x = do.
3. Si f() · f(c) < 0, toma r ← c. De lo contrario, tome ← c.
4. Si |r − | < ε, devuelve x ≈ C.
5. Vuelva al paso 1

Esta estrategia simplemente divide el intervalo [,r] por la mitad iterativamente, manteniendo cada vez el lado en el que se sabe que existe una raíz. Claramente por el Teorema del Valor Intermedio converge incondicionalmente, en el sentido de que mientras f() · f(r) < 0 eventualmente se garantiza que ambos y r convergen en una raíz válida x .

7.1.3 Análisis de búsqueda de raíces

La bisección es la técnica más simple pero no necesariamente la más efectiva para encontrar raíces. Al igual que con la mayoría de los métodos de valores propios, la bisección es inherentemente iterativa y es posible que nunca proporcione una solución exacta. Sin embargo, podemos preguntar qué tan cerca está el valor ck de c en la k-ésima iteración de la raíz x que esperamos calcular. Este análisis proporcionará una línea de base para la comparación con otros métodos.

En general, supongamos que podemos establecer un límite de error Ek tal que la estimación xk de la raíz durante la k-
X ésima iteración de un método de búsqueda de raíces satisface |xk − x | < Ek . Obviamente cualquier algoritmo con Ek → 0
representa un esquema convergente; la velocidad de convergencia, sin embargo, se puede caracterizar por la velocidad a la que
Ek se aproxima a 0. están en el intervalo [k ,rk], un límite superior
dados por Ek ≡ | rk − k |. Como dividimos el intervalo por la de Por ejemplo, en bisección ya que tanto ck como x error están
mitad en cada iteración, sabemos que Ek+1 = 1/2Ek . Dado que Ek+1 es lineal en Ek , decimos que la bisección exhibe
convergencia lineal.

7.1.4 Iteración de punto fijo

Se garantiza que la bisección convergerá a una raíz para cualquier función continua f , pero si sabemos más sobre f podemos
formular algoritmos que puedan converger más rápidamente.

Como ejemplo, supongamos que deseamos encontrar x que satisface g(x) = x ; por supuesto, esta configuración
es equivalente al problema de búsqueda de raíces ya que resolver f(x) = 0 es lo mismo que resolver f(x) + x = x.
Sin embargo, como información adicional, también podemos saber que g es Lipschitz con constante C < 1.

El sistema g(x) = x sugiere una estrategia potencial que podríamos plantear como hipótesis:

1. Tome x0 como una suposición inicial de una raíz.
2. Iterar xk = g(xk−1).

Si esta estrategia converge, claramente el resultado es un punto fijo de g que satisface los criterios anteriores.

Afortunadamente, la propiedad de Lipschitz asegura que esta estrategia converja a una raíz, si existe.
Si tomamos Ek = |xk − x |, entonces tenemos la siguiente propiedad:

$$
\begin{aligned}
E_k = |x_k - x^\star| &= |g(x_{k-1}) - g(x^\star)| \text{ por diseño del esquema iterativo y definición de } x^\star \\
&\leq C|x_{k-1} - x^\star| \text{ ya que } g \text{ es Lipschitz} \\
&= CE_{k-1}
\end{aligned}
$$

La aplicación inductiva de esta declaración muestra que Ek ≤ C k |E0| → 0 cuando k → ∞. Por lo tanto, la iteración de punto fijo
converge a la x deseada !

De hecho, si g es Lipschitz con constante C < 1 en una vecindad [x − δ, x + δ], entonces, siempre que
se elija x0 en este intervalo, la iteración de punto fijo convergerá. Esto es cierto ya que nuestra expresión anterior para Ek muestra
que se reduce en cada iteración. y |g (x)| < 1. Por

Un caso importante ocurre cuando g es C sabemos continuidad de g en este caso, tenemos + δ] en el que |g (x)| < 1 − ε para
que existe algún entorno N = [x − δ, x para alguna elección de un ε cualquier x N,
> 0.1 suficientemente pequeño.1 Tome cualquier x, y N. Entonces, tenemos

$$
\begin{aligned}
|g(x) - g(y)| &= |g'(\theta)| \cdot |x - y| \text{ por el teorema del valor medio del cálculo básico, para algunos } \theta \in [x, y] \\
&< (1 - \varepsilon)|x - y|
\end{aligned}
$$

Esto muestra que g es Lipschitz con constante 1 − ε < 1 en N. Por lo tanto, cuando g es continuamente diferenciable y g (x) <
1, la iteración de punto fijo convergerá a x cuando la suposición inicial x0 esté cerca.

1Esta declaración es difícil de analizar: ¡asegúrese de entenderla!

Hasta ahora tenemos pocas razones para usar la iteración de punto fijo: hemos demostrado que se garantiza la convergencia solo cuando g es Lipschitz, y nuestro argumento sobre las Ek muestra una convergencia lineal como la bisección. Hay un caso, sin embargo, en el que la iteración de punto fijo proporciona una ventaja.

Supongamos que g es derivable con g (x) = 0. Entonces, el término de primer orden desaparece en la serie de Taylor para g, dejando atrás:

$$\text{gramo}(xk) = \text{gramo}(x\ ) + \text{gramo}\,(x\ )(xk - x\ )\,\frac{1}{2}\,^{2} + O\,(xk - x\ )^{3}.$$

Así, en este caso tenemos:

$$\begin{aligned} Ek &= |xk - x \\ &| = |g(xk{-}1) - g(x\ )| \text{ como antes} \\ &= \tfrac{1}{1}\,|g\,(x\ )|(xk{-}1 - x\ )\,2^{2} + O((xk{-}1 - x\ )^{3}) \text{ del argumento de Taylor} \\ &\leq \bar{}\,(|g\,(x\ )| + \varepsilon)|(xk{-}1 - x\ )^{2} \text{ para algún } \varepsilon \text{ siempre que } xk{-}1 \text{ esté cerca de } x \\ &= \tfrac{2}{2}\,1\,(|g\,(x\ )| + \varepsilon)E_{k-1}^{2} \end{aligned}$$

Por lo tanto, en este caso Ek es cuadrático en Ek−1 , por lo que decimos que la iteración de punto fijo puede tener convergencia cuadrática; observe que esta prueba de convergencia cuadrática solo es válida porque ya sabemos Ek → 0 de nuestra prueba de convergencia más general. Esto implica que Ek → 0 mucho más rápido, por lo que necesitaremos menos iteraciones para llegar a una raíz razonable.

Ejemplo 7.1 (Convergencia de iteración de punto fijo).

7.1.5 Método de Newton

Reforzamos nuestra clase de funciones una vez más para derivar un método que tenga una convergencia cuadrática más consistente. Ahora, supongamos nuevamente que deseamos resolver f(x) = 0, pero ahora asumimos que f es una ^{1}, a condición ligeramente más estricta que la de Lipschitz.

En un punto xk R, ya que f ahora es diferenciable podemos aproximarla usando una recta tangente: f(x) ≈

$$f(xk) + f\,(xk)(x - xk)$$

Resolviendo esta aproximación para f(x) ≈ 0 se obtiene una raíz

$$xk{+}1 = xk - \frac{f(xk)\,f}{(xk)}.$$

La iteración de esta fórmula se conoce como el método de Newton para encontrar raíces y equivale a resolver iterativamente una aproximación lineal del problema no lineal.

Note que si definimos

$$g(x) = x - f\,(x)\,\frac{f(x)}{\quad},$$

entonces el método de Newton equivale a una iteración de punto fijo en g. Derivando, encontramos: −

$$\begin{aligned} \text{gramo}\,(x) &= 1 - \frac{f\,(x)^{2}\,f(x)f\,(x) \text{ por la}}{f(x)f^{2}} \text{regla del cociente } f\,(x) \\ &= \frac{(x)}{f\,(x)^{2}} \end{aligned}$$

Supongamos es una raíz simple, lo que significa que f (x) = 0. Entonces, g (x) = 0, y por nuestra derivación de que x la iteración de punto fijo anterior sabemos que el método de Newton converge cuadráticamente a x para un punto suficientemente cercano estimación inicial. Por lo tanto, cuando f es diferenciable con una raíz simple, el método de Newton proporciona una fórmula de iteración de punto fijo que garantiza la convergencia cuadrática; sin embargo, es cuando x no es simple, la convergencia puede ser lineal o peor.

La derivación del método de Newton sugiere otros métodos derivados del uso de más términos en la serie de Taylor. Por ejemplo, el "método de Halley" agrega términos que involucran f a las iteraciones, y una clase de "métodos domésticos" toma un número arbitrario de derivadas. Estas técnicas ofrecen una convergencia de orden aún mayor a costa de tener que evaluar iteraciones más complejas y la posibilidad de modos de falla más exóticos. Otros métodos reemplazan las series de Taylor con otras formas básicas; por ejemplo, la interpolación fraccionaria lineal utiliza funciones racionales para aproximar mejor las funciones con estructura de asíntota.

7.1.6 Método de la secante

Una preocupación de eficiencia que aún no hemos abordado es el costo de evaluar f y sus derivados.
Si f es una función muy complicada, es posible que deseemos minimizar el número de veces que tenemos que calcular f o peor f . Los órdenes de convergencia más altos ayudan con este problema, pero también podemos diseñar métodos numéricos que eviten evaluar derivadas costosas.

Ejemplo 7.2 (Diseño). Supongamos que estamos diseñando un cohete y deseamos saber cuánto combustible agregar al motor. Para un número dado de galones x, podemos escribir una función f(x) que dé la altura máxima del cohete; nuestros ingenieros han especificado que deseamos que el cohete alcance una altura h, por lo que debemos resolver f(x) = h. Evaluar f(x) implica simular un cohete a medida que despega y monitorear su consumo de combustible, lo cual es una propuesta costosa, y aunque podríamos sospechar que f es diferenciable, es posible que no podamos evaluar f en un tiempo práctico.

Una estrategia para diseñar métodos de bajo impacto es reutilizar los datos tanto como sea posible. Para ejemplo, fácilmente podríamos aproximar:

$$(xk) \approx \frac{f(xk) - f(xk-1)\ f}{xk - xk-1}.$$

Es decir, dado que tuvimos que calcular f(xk−1) en la iteración anterior, simplemente usamos la pendiente de f(xk) para aproximar la derivada. Ciertamente, esta aproximación funciona bien, especialmente cuando las xk están cerca de la convergencia.

Conectar nuestra aproximación al método de Newton revela un nuevo esquema iterativo:

$$xk+1 = xk - \frac{f(xk)(xk - xk-1)}{f(xk) - f(xk-1)}$$

Tenga en cuenta que el usuario tendrá que proporcionar dos conjeturas iniciales x0 y x−1 para iniciar este esquema, o puede ejecutar una sola iteración de Newton para comenzar.

Analizar el método de la secante es un poco más complicado que los otros métodos que consideramos porque usa tanto f(xk) como f(xk−1); la prueba de su convergencia está fuera del alcance de nuestra discusión. Curiosamente, el análisis de errores revela que el error disminuye a una tasa de $1+\sqrt{5}/2$ (la "proporción áurea"), entre lineal y cuadrático; dado que la convergencia es cercana a la del método de Newton sin necesidad de evaluar f , el método de la secante puede proporcionar una sólida alternativa.

7.1.7 Técnicas híbridas

Se puede llevar a cabo ingeniería adicional para intentar combinar las ventajas de diferentes algoritmos de búsqueda de raíces. Por ejemplo, podríamos hacer las siguientes observaciones sobre dos métodos que hemos discutido:

- La bisección está garantizada para converger incondicionalmente, pero solo lo hace a una velocidad lineal.
- El método de la secante converge más rápido cuando llega a una raíz, pero en algunos casos puede que no lo haga. converger.

Supongamos que hemos puesto entre paréntesis una raíz de f(x) en un intervalo [k ,rk] como en la bisección. Podemos decir que estimación actual de x hacemos está dado por xk = k cuando | f(k)| < | f(rk)| y xk = rk en caso contrario. Si en nuestra un seguimiento de xk y xk−1 , entonces podríamos tomar xk+1 como la próxima estimación de la raíz dada por el método de la secante. Sin embargo, si xk está fuera del intervalo [k ,rk], podemos reemplazarlo con k+rk/2. Esta corrección garantiza que xk+1 [k ,rk], e independientemente de la elección, podemos actualizar a un corchete válido [k+1 ,rk+1] como en la bisección examinando el signo de f(xk+1). Este algoritmo se conoce como "método de Dekker".

La estrategia anterior intenta combinar la convergencia incondicional de la bisección con las estimaciones de raíz más fuertes del método de la secante. En muchos casos tiene éxito, pero su tasa de convergencia es algo difícil de analizar; los modos de falla especializados pueden reducir este método a la convergencia lineal o peor; de hecho, en algunos casos, ¡sorprendentemente, la bisección puede converger más rápidamente! Otras técnicas, por ejemplo, el “método de Brent”, realizan pasos de bisección con más frecuencia para evitar este caso y pueden exhibir un comportamiento garantizado a costa de una implementación algo más compleja.

7.1.8 Caso de variable única: resumen

Ahora hemos presentado y analizado varios métodos para resolver f(x) = 0 en el caso de una sola variable. Probablemente sea obvio en este punto que solo hemos raspado la superficie de tales técnicas; existen muchos esquemas iterativos para la búsqueda de raíces, todos con diferentes garantías, tasas de convergencia y advertencias. De todos modos, a través de nuestras experiencias podemos hacer una serie de observaciones:

- Debido a la posible forma genérica de f , es poco probable que podamos encontrar las raíces x exactamente y en su lugar nos conformemos con esquemas iterativos.
- Deseamos que la secuencia xk de raíces estimadas alcance x lo más rápido posible. Si Ek es una cota de error, podemos caracterizar varias situaciones de convergencia suponiendo que Ek → 0 como k → ∞. A continuación se muestra una lista completa de las condiciones que deben cumplirse cuando k es lo suficientemente grande:
  1. Convergencia lineal: Ek+1 ≤ CEk para algún C < 1 2. Convergencia
  superlineal: Ek+1 ≤ CEr para r > 1 (ahora no requerimos C < 1 ya que si Ek es lo suficientemente pequeño, la potencia k r puede cancelar los efectos de C)
  3. Convergencia cuadrática: Ek+1 ≤ CE2 4. k
  Convergencia cúbica: Ek+1 ≤ CE3 k (etcétera)
- Un método puede converger más rápidamente, pero durante cada iteración individual requiere cálculos adicionales; por esta razón, puede ser preferible hacer más iteraciones de un método más simple que menos iteraciones de uno más complejo.

7.2 Problemas multivariables

Algunas aplicaciones pueden requerir resolver un problema más general f(x) =0 para una función f : Rn → Rm. Ya vimos una instancia de este problema al resolver Ax =b, que es equivalente a encontrar raíces de f(x) ≡ Ax −b, pero el caso general es considerablemente más difícil. En particular, las estrategias como la bisección son difíciles de extender ya que ahora garantizamos que m valores diferentes son todos cero simultáneamente.

7.2.1 Método de Newton

Afortunadamente, una de nuestras estrategias se extiende de una manera directa. Recuerda que para f : Rn → Rm podemos escribir la matriz jacobiana, que da la derivada de cada componente de f en cada una de las direcciones de las coordenadas:

$$(Df)_{ij} \equiv \frac{d\text{-}\, fi}{dxj}$$

Podemos usar el jacobiano de f para extender nuestra derivación del método de Newton a múltiples dimensiones. En particular, la aproximación de primer orden de f viene dada por:

$$f(x) \approx f(xk) + D\, f(xk) \cdot (x - xk).$$

Sustituyendo el f(x) =0 deseado se obtiene el siguiente sistema lineal para la siguiente iteración xk+1 :

$$\text{re}\, f(xk) \cdot (xk+1 - xk) = -f(xk)$$

Esta ecuación se puede resolver usando la pseudoinversa cuando m < n; cuando m > n se pueden intentar los mínimos cuadrados, pero la existencia de una raíz y la convergencia de esta técnica son poco probables. Sin embargo, cuando D f es cuadrada, corresponde al método f : , obtenemos la iteración típica de Newton Rn → Rn :

$$xk+1 = xk - [D\, f(xk)]^{-1} f(xk),$$

donde, como siempre, no calculamos explícitamente la matriz [D f(xk)]−1 sino que la usamos para señalar la resolución de un sistema lineal.

La convergencia de métodos de punto fijo como el método de Newton que itera xk+1 = g(xk) requiere que el valor propio de magnitud máxima del jacobiano Dg sea menor que 1. Después de verificar esa suposición, un argumento similar al caso unidimensional muestra que el método de Newton puede para el cual D f(x) no cuadrática cerca de las raíces x es singular. tienen convergencia

7.2.2 Haciendo que Newton sea más rápido: Cuasi-Newton y Broyen

A medida que aumentan m y n, el método de Newton se vuelve muy costoso. Para cada iteración, se debe invertir una matriz D f(xk) diferente; debido a que cambia tan a menudo, la prefactorización de D f(xk) = LkUk no ayuda.

Algunas estrategias cuasi-Newton intentan aplicar diferentes estrategias de aproximación para simplificar las iteraciones individuales. Por ejemplo, un enfoque sencillo podría reutilizar D f de iteraciones anteriores mientras se vuelve a calcular f(xk) bajo el supuesto de que la derivada no cambia muy rápidamente. Volveremos a estas estrategias cuando analicemos la aplicación del método de Newton a la optimización.

Otra opción es intentar hacer un paralelo con nuestra derivación del método de la secante. Así como el método de la secante todavía contiene división, tales aproximaciones no aliviarán necesariamente la necesidad de invertir una matriz, pero permiten llevar a cabo la optimización sin calcular explícitamente el jacobiano D f . Tales extensiones no son totalmente obvias, ya que las diferencias divididas no producen una matriz jacobiana aproximada completa.

Recuerde, sin embargo, que la derivada direccional de f en la dirección v está dada por Dv f = D f · v. Al igual que con el método de la secante, podemos usar esta observación a nuestro favor al pedir que nuestra aproximación J de un jacobiano satisfaga

$$J \cdot (x_k - x_{k-1}) \approx f(x_k) - f(x_{k-1}).$$

El método de Broyden es una de esas extensiones del método de la secante que realiza un seguimiento no solo de una estimación x_k de x sino también de una matriz J_k que estima el jacobiano; Se deben suministrar las estimaciones iniciales J_0 y x_0 . Supongamos que tenemos una estimación previa J_{k-1} del jacobiano de la iteración anterior. Ahora tenemos un nuevo punto de datos x_k en el que hemos evaluado $f(x_k)$, por lo que nos gustaría actualizar J_{k-1} a un nuevo jacobiano J_k teniendo en cuenta esta nueva observación. Un modelo razonable es pedir que la nueva aproximación sea similar a la anterior excepto en la dirección $x_k - x_{k-1}$:

$$\begin{aligned} &\text{minimizar}_{J_k} \; \|J_k - J_{k-1}\|_{\text{Para}}^2 \\ &\text{tal que } J_k \cdot (x_k - x_{k-1}) = f(x_k) - f(x_{k-1}) \end{aligned}$$

Para resolver este problema, defina $\Delta J \equiv J_k - J_{k-1}$ y $d \equiv f(x_k) - f(x_{k-1}) - J_{k-1} \cdot \Delta x$. $\Delta x \equiv x_k - x_{k-1}$, sustituciones se obtiene la siguiente forma:

$$\begin{aligned} &\text{minimizar}_{\Delta J} \; \|\Delta J\|_{\text{Para}}^2 \\ &\text{tal que } \Delta J \cdot \Delta x = d \end{aligned}$$

Si tomamos λ como un multiplicador de Lagrange, esta minimización es equivalente a encontrar puntos críticos del Lagrangiano Λ:

$$\Lambda = \|\Delta J\|_{\text{Para}}^2 + \lambda \, (\Delta J \cdot \Delta x - d)$$

Diferenciar con respecto a $(\Delta J)_{ij}$ muestra:

$$0 = \frac{\partial \Lambda}{\partial (\Delta J)_{ij}} = 2(\Delta J)_{ij} + \lambda_i (\Delta x)_j \quad \Rightarrow \quad \Delta J = -\frac{\lambda (\Delta x)}{2}$$

Sustituyendo en $\Delta J \cdot \Delta x = d$ muestra $\lambda(\Delta x)\,(\Delta x) = -2d$, o equivalentemente $\lambda = -2d/\|\Delta x\|^2$. Finalmente,podemos sustituir para encontrar:

$$\Delta J = -\frac{1}{2}\lambda (\Delta x) = \Delta x \, \frac{d(\Delta x)}{\|\Delta x\|^2}$$

Expandiendo nuestros programas de sustitución:

$$\begin{aligned} J_k &= J_{k-1} + \Delta J \\ &= J_{k-1} + \frac{d(\Delta x)}{\|\Delta x\|^2} \\ &= J_{k-1} + \frac{(f(x_k) - f(x_{k-1}) - J_{k-1} \cdot \Delta x)}{\|x_k - x_{k-1}\|^2} (x_k - x_{k-1}) \end{aligned}$$

Por lo tanto, el método de Broyden simplemente alterna entre esta actualización y la correspondiente Newton f(xk). En paso xk+1 = xk − J $_k^{-1}$ algunos casos, se puede obtener eficiencia adicional haciendo un seguimiento de la matriz J $_k^{-1}$ explícitamente en lugar de la matriz Jk , que se puede actualizar usando una fórmula similar.

7.3 Acondicionamiento

Ya mostramos en el Ejemplo 1.7 que el número de condición de búsqueda de raíces en una sola variable es:

$$\text{condx} \quad f = \frac{1}{|\, f(x) \,|}$$

Como se ilustra en la Figura NÚMERO, este número de condición muestra que la mejor situación posible para encontrar raíces ocurre cuando f cambia rápidamente cerca de x, ya que en este caso perturbar x hará que f tome valores lejos de 0.

La aplicación de un argumento idéntico cuando f es multidimensional muestra un número de condición de D f(x) $^{-1}$. Observe que cuando D f no es invertible, el número de condición es infinito. Esta rareza preserva f(x) = ocurre porque la perturbación de primer orden x 0, y de hecho tal condición puede crea casos desafiantes de búsqueda de raíces como el que se muestra en la Figura NÚMERO.

7.4 Problemas

Muchas posibilidades, incluyendo:

- Muchos posibles esquemas de iteración de punto fijo para un problema dado de búsqueda de raíces, versión gráfica de la iteración de punto fijo
- Iteración de campo medio en ML
- Método de Muller: raíces complejas
- Métodos iterativos de orden superior: métodos de jefe de hogar
- Interpretación de elementos propios como búsqueda de raíces
- Convergencia del método de la secante
- Raíces de polinomios
- Método de Newton-Fourier (!)
- “Método de Newton modificado en caso de convergencia no cuadrática”
- Convergencia –¿ radio espectral para Newton multidimensional; convergencia cuadrática
- Actualización de Sherman-Morrison para Broyden

Capítulo 8

Optimización sin restricciones

En capítulos anteriores, hemos optado por adoptar un enfoque en gran medida variacional para derivar algoritmos estándar para el álgebra lineal computacional. Es decir, definimos una función objetivo, posiblemente con restricciones, y planteamos nuestros algoritmos como un problema de minimización o maximización. Una muestra de nuestra discusión anterior se enumera a continuación:

Problema	Objetivo	Restricciones
mínimos cuadrados	$E(x) = Ax - b$ ^{2}	Ninguno
Proyectob sobrea	$E(c) = ca - b$	Ninguno
Vectores propios de matriz simétrica	$E(x) = x\ Ax$	$X = 1$
pseudoinverso	$E(x) = x$ ^{2}	$A\ Ax = A\ b$
Análisis de componentes principales	$E(C) = X - CCXFro\ CC = Id_{\times d}$	
paso Broyden	$E(Jk) = Jk - Jk-1$ ^{2} Para	$Jk \cdot (xk - xk-1) = f(xk) - f(xk-1)$

Obviamente, la formulación de problemas de esta manera es un enfoque poderoso y general. Por esta razón, es valioso diseñar algoritmos que funcionen en ausencia de una forma especial para la energía E, de la misma manera que desarrollamos estrategias para encontrar raíces de f sin conocer la forma a priori.

8.1 Optimización sin restricciones: motivación

En este capítulo, consideraremos problemas sin restricciones, es decir, problemas que se pueden plantear como minimizar o maximizar una función $f : Rn \to R$ sin ningún requisito en la entrada. No es difícil encontrar tales problemas en la práctica; enumeramos algunos ejemplos a continuación.

Ejemplo 8.1 (Mínimos cuadrados no lineales). Supongamos que nos dan un número de pares (xi , yi) tales que $f(xi) \approx yi$, y deseamos encontrar la mejor aproximación de f dentro de una clase en particular. Por ejemplo, podemos esperar que f sea exponencial, en cuyo caso deberíamos poder escribir $f(x) = ceax$ para alguna c y alguna a; nuestro trabajo es encontrar estos parámetros. Una estrategia simple podría ser intentar minimizar la siguiente energía:

$$E(a, c) = \sum_i (yi - ceaxi)^2.$$

Esta forma de E no es cuadrática en a, por lo que no se aplican nuestros métodos de mínimos cuadrados lineales.

Ejemplo 8.2 (Estimación por máxima verosimilitud). En el aprendizaje automático, el problema de la estimación de parámetros implica examinar los resultados de un experimento aleatorio y tratar de resumirlos usando una distribución de probabilidad de una forma particular. Por ejemplo, podríamos medir la altura de cada estudiante en una clase, dando una lista de alturas hi para cada estudiante i. Si tenemos muchos estudiantes, podríamos modelar la distribución de las alturas de los estudiantes usando una distribución normal:

$$g(h; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(h-\mu)^2/2\sigma^2},$$

donde μ es la media de la distribución y σ es la desviación estándar.

Bajo esta distribución normal, la probabilidad de que observemos la estatura hi del estudiante i viene dada por g(hi ; μ, σ), y bajo la suposición (razonable) de que la estatura del estudiante i es probabilísticamente independiente de la del estudiante j, la probabilidad de observar todo el conjunto de alturas observadas viene dada por el producto

$$P(\{h1, \ldots, hn\}; \mu, \sigma) = \prod_i g(hi ; \mu, \sigma).$$

Un método común para estimar los parámetros μ y σ de g es maximizar P visto como una función de μ y σ con {hi} fijo; esto se denomina estimación de máxima verosimilitud de μ y σ. En la práctica, normalmente optimizamos el logaritmo de verosimilitud (μ, σ) ≡ log P({h1, . . . , hn}; μ, σ); esta función tiene los mismos máximos pero disfruta de mejores propiedades numéricas y matemáticas.

Ejemplo 8.3 (Problemas geométricos). Muchos problemas de geometría encontrados en gráficos y visión no se reducen a energías de mínimos cuadrados. Por ejemplo, supongamos que tenemos un número de puntos x1, . . . ,xk R3 . Si deseamos agrupar estos puntos, podríamos resumirlos con una sola x minimizando:

$$E(x) \equiv \sum_i x - xi2.$$

La x R3 que minimiza E se conoce como la mediana geométrica de {x1, . . . ,xk}. Observe que la norma de la diferencia x −xi en E no está al cuadrado, por lo que la energía ya no es cuadrática en los componentes de x.

Ejemplo 8.4 (Equilibrios físicos, adaptado de CITE). Supongamos que adjuntamos un objeto a un conjunto de resortes; cada resorte está anclado en el punto xi R3 y tiene longitud natural Li y constante ki . En ausencia de gravedad, si nuestro objeto está ubicado en la posición p R3, la red de manantiales tiene energía potencial

$$E(p) = \frac{1}{2} \sum_i ki \left(p - xi2 - Li\right)^2$$

Los equilibrios de este sistema están dados por mínimos de E y reflejan puntos p en los que las fuerzas del resorte están equilibradas. Dichos sistemas de ecuaciones se utilizan para visualizar gráficos G = (V, E), uniendo vértices en V con resortes para cada par en E.

8.2 Optimalidad

Antes de discutir cómo minimizar o maximizar una función, debemos tener claro qué es lo que buscamos; observe que maximizar f es lo mismo que minimizar −f , por lo que el problema de minimización es suficiente para nuestra consideración. Para un f particular: Rn → R y x Rn necesitamos derivar , tiene el valor f(x) optimalidad que verifican que x Por más bajo posible. condiciones de

supuesto, idealmente nos gustaría encontrar óptimos globales de f :

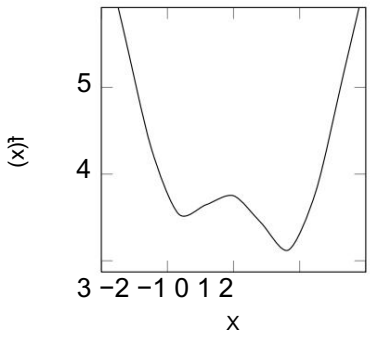


Figura 8.1: Una función f(x) con múltiples óptimos.

Definición 8.1 (Mínimo global). El punto x f(x) ≤ Rn es un mínimo global de f : Rn → R si f(x) para todo x Rn .

Encontrar un mínimo global de f sin ninguna información sobre la estructura de f requiere efectivamente buscar en la oscuridad. Por ejemplo, suponga que un algoritmo de optimización identifica el mínimo local cerca de x = −1 en la función de la figura 8.1. Es casi imposible darse cuenta de que hay un segundo mínimo más bajo cerca de x = 1 simplemente adivinando los valores de x; por lo que sabemos, ¡puede haber un tercer mínimo aún más bajo de f en x = 1000!

Así, en muchos casos nos satisfacemos encontrando un mínimo local:

Definición 8.2 (Mínimo local). El punto x f(x) para Rn es un mínimo local de f : Rn → R si f(x) ≤ < todo x Rn que satisface x −x ε para algún ε > 0.

definición requiere que x radio ε. alcanza el valor más pequeño en alguna vecindad definida por la Esta Note que los algoritmos de optimización local tienen una severa limitación de que no pueden garantizar que produzcan el valor más bajo posible de f , como en la figura 8.1 si se alcanza el mínimo local izquierdo; se aplican muchas estrategias, heurísticas y de otro tipo, para explorar el panorama de posibles valores de x para ayudar a ganar confianza en que un mínimo local tiene el mejor valor posible.

8.2.1 Optimalidad diferencial

Una historia familiar del cálculo de una y varias variables es que encontrar mínimos y máximos potenciales de una función f : Rn → R es más sencillo cuando f es derivable. Recuerda que el vector gradiente f = (∂ f/∂x1, . . . , ∂ f/∂xn) apunta en la dirección en la que f aumenta más; el vector − f apunta en la dirección de mayor disminución. Una forma de ver esto es recordar que cerca de a , f se parece a la función lineal punto x0 Rn

$$f(x) \approx f(x0) + \quad f(x0) \cdot (x - x0).$$

Si tomamos x −x0 = α f(x0), entonces encontramos:

$$f(x0 + \alpha \quad f(x0)) \approx f(x0) + \alpha \quad f(x0)^{2}$$

Cuando f(x0) > 0, el signo de α determina si f crece o decrece.

No es difícil formalizar el argumento anterior para mostrar que si x0 es un mínimo local, entonces debemos tener f(x0) =0. Note que esta condición es necesaria pero no suficiente: maxima y silla de montar

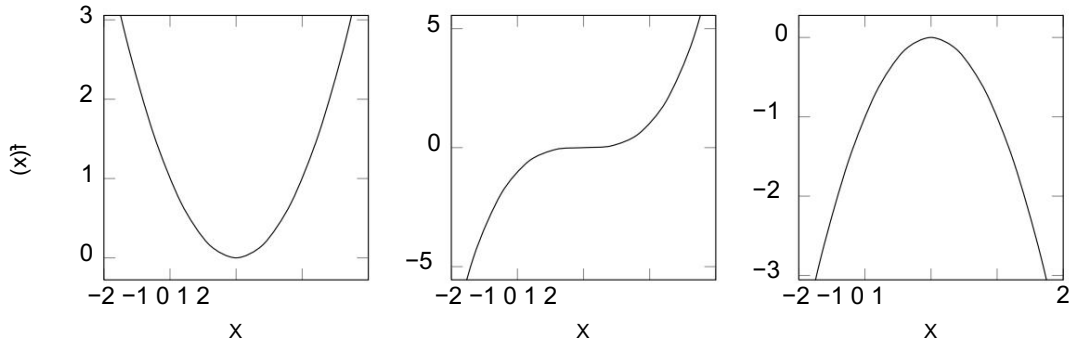


Figura 8.2: Los puntos críticos pueden tomar muchas formas; aquí mostramos un mínimo local, un punto silla y un máximo local.

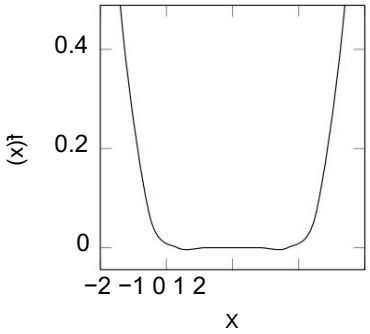


Figura 8.3: Una función con muchos puntos estacionarios.

los puntos también tienen f(x0) = 0 como se ilustra en la figura 8.2. Aun así, esta observación sobre los mínimos de funciones diferenciables produce una estrategia común para encontrar raíces:

1. Encuentra puntos xi que satisfagan f(xi) =0.

2. Verifique cuál de estos puntos es un mínimo local en oposición a un máximo o punto silla.

Dado su importante papel en esta estrategia, damos a los puntos que buscamos un nombre especial:

Definición 8.3 (Punto estacionario). Un punto estacionario de f : Rn → R es un punto x Rn que satisface f(x) =0.

Es decir, nuestra estrategia de minimización puede ser encontrar puntos estacionarios de f y luego eliminar aquellos que no son mínimos.

Es importante tener en cuenta cuándo podemos esperar que nuestras estrategias de minimización tengan éxito. En la mayoría de los casos, como los que se muestran en la figura 8.2, los puntos estacionarios de f están aislados, lo que significa que podemos escribirlos en una lista discreta {x0,x1, . . .}. Sin embargo, en la figura 8.3 se muestra un caso degenerado; aquí, todo el intervalo [−1/2, 1/2] está compuesto de puntos estacionarios, lo que hace imposible considerarlos uno por uno. En su mayor parte, ignoraremos cuestiones como los casos degenerados, pero volveremos a ellos cuando consideremos el condicionamiento del problema de minimización.

Supongamos que identificamos un punto x R como un punto estacionario de f y ahora queremos comprobar si es un mínimo local. Si f es dos veces diferenciable, una estrategia que podemos emplear es escribir su hessiana

matriz:

$$\mathrm{Hf}(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Podemos agregar otro término a nuestra expansión de Taylor de f para ver el papel de Hf :

$$f(x) \approx f(x_0) + f(x_0) \cdot (x - x_0) + \frac{1}{2} (x - x_0) \mathrm{Hf}(x - x_0)$$

Si sustituimos un punto estacionario x , entonces por definición sabemos:

$$f(x) \approx f(x) + \frac{1}{2} (x - x) \mathrm{Hf}(x - x)$$

Si Hf es definida positiva, entonces esta expresión muestra $f(x) \geq f(x)$, y por lo tanto x es un mínimo local.
En términos más generales, puede ocurrir una de algunas situaciones:

- Si Hf es definida positiva, entonces x es un mínimo local de f .
- Si Hf es definida negativa, entonces x es un máximo local de f .
- Si Hf es indefinido, entonces x es un punto de silla de f .
- Si Hf no es invertible, entonces pueden ocurrir rarezas como la función de la figura 8.3.

Se puede comprobar si una matriz es definida positiva comprobando si existe su factorización de Cholesky o, más lentamente, comprobando que todos sus valores propios son positivos. Por lo tanto, cuando se conoce el hessiano de f, podemos verificar la optimización de los puntos estacionarios usando la lista anterior; muchos algoritmos de optimización, incluidos los que discutiremos, simplemente ignoran el caso final y notifican al usuario, ya que es relativamente poco probable.

8.2.2 Optimalidad a través de las propiedades de la función

Ocasionalmente, si conocemos más información acerca de $f : \mathbb{R}^n \to \mathbb{R}$ podemos proporcionar condiciones de optimalidad que son más fuertes o más fáciles de verificar que las anteriores.

Una propiedad de f que tiene fuertes implicaciones para la optimización es la convexidad, ilustrada en la figura NÚMERO:

Definición 8.4 (Convexo). Una función $f : \mathbb{R}^n \to \mathbb{R}$ es convexa cuando para todo $x, y \in \mathbb{R}^n$ y $\alpha \in (0, 1)$ se cumple la siguiente relación:

$$f((1 - \alpha)x + \alpha y) \leq (1 - \alpha) f(x) + \alpha f(y).$$

Cuando la desigualdad es estricta, la función es estrictamente convexa.

La convexidad implica que si conectas en $\mathbb{R}^n$ dos puntos con una línea, los valores de f a lo largo de la línea son menores o iguales a los que obtendrías por interpolación lineal.

Las funciones convexas disfrutan de muchas propiedades sólidas, la más básica de las cuales es la siguiente:

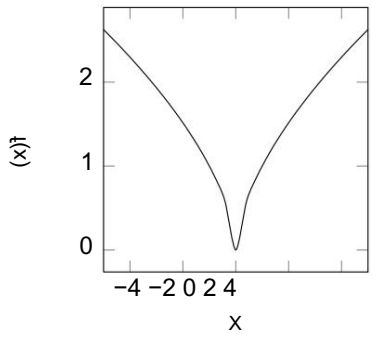


Figura 8.4: Una función cuasiconvexa.

Proposición 8.1. Un mínimo local de una función convexa f : Rn → R es necesariamente un mínimo global.

Prueba. Tome x como tal mínimo local y suponga que existe x Entonces, para α = x con f(x) < f(x). (0, 1),

$$f(x + \alpha(x - x)) \leq (1 - \alpha)f(x) + \alpha f(x) \text{ por convexidad}$$
$$< f(x) \text{ ya que } f(x) < f(x)$$

Pero tomar α → 0 muestra que x no puede ser un mínimo local. □

Esta proposición y las observaciones relacionadas muestran que es posible verificar si se ha alcanzado un mínimo global de una función convexa simplemente aplicando la optimización de primer orden. Por lo tanto, es valioso verificar a mano si una función que se está optimizando resulta ser convexa, una situación que ocurre sorprendentemente a menudo en la computación científica; una condición suficiente que puede ser más fácil de verificar cuando f es dos veces diferenciable es que Hf es definida positiva en todas partes.

Otras técnicas de optimización tienen garantías bajo otros supuestos sobre f . Para examen
Por ejemplo, una versión más débil de la convexidad es la cuasi-convexidad, que se logra cuando

$$f((1 - \alpha)x + \alpha y) \leq \max(f(x), f(y)).$$

En la figura 8.4 se muestra un ejemplo de una función cuasiconvexa; aunque no tiene la forma característica de "tazón" de una función convexa, tiene un óptimo único.

8.3 Estrategias unidimensionales

Como en el último capítulo, comenzaremos con la optimización unidimensional de f : R → R y luego expandiremos nuestras estrategias a funciones más generales f : Rn → R.

8.3.1 Método de Newton

Nuestra estrategia principal para minimizar funciones derivables f : Rn → R será encontrar sta que satisfaga estacionarios x f(x) = 0. Suponiendo que podemos verificar si los puntos estacionarios son puntos máximos, mínimos o puntos de silla como resultado paso de procesamiento, nos centraremos en el problema de encontrar los puntos estacionarios x ·

Para ello, supongamos que f : R → R es derivable. Entonces, como en nuestra derivación de Newton método para encontrar raíces, podemos aproximar:

$$1\ f(x) \approx f(xk) + f\,(xk)(x - xk) + f\,(xk)(x - xk)\ 2^{2}.$$

La aproximación del lado derecho es una parábola cuyo vértice se encuentra en xk − f (xk)/f (xk).
Por supuesto, en realidad f no es necesariamente una parábola, por lo que el método de Newton simplemente itera la fórmula

$$xk+1 = xk - \frac{f\,(xk)}{f\,(xk)}.$$

Esta técnica se analiza fácilmente dado el trabajo que ya hemos realizado para comprender el método de Newton para encontrar raíces en el capítulo anterior. En particular, una forma alternativa de derivar la fórmula anterior proviene de la búsqueda de raíces en f (x), ya que los puntos estacionarios satisfacen f (x) = 0.
Por lo tanto, en la mayoría de los casos, el método de optimización de Newton exhibe convergencia cuadrática, siempre que la suposición inicial x0 esté lo suficientemente cerca de x .

Una pregunta natural es si el método de la secante se puede aplicar de manera análoga.
Nuestra derivación del método de Newton anterior encuentra raíces de f , por lo que el método de la secante podría usarse para eliminar la evaluación de f pero no de f ; las situaciones en las que sabemos f pero no f son relativamente raras. Un paralelo más adecuado es reemplazar los segmentos de línea usados para aproximar f en el método de la secante con parábolas. Esta estrategia, conocida como interpolación parabólica sucesiva, también minimiza una aproximación cuadrática de f en cada iteración, pero en lugar de usar f(xk), f (xk) y f (xk) para construir la aproximación, usa f(xk), f(xk−1), y f(xk−2). La derivación de esta técnica es relativamente sencilla y converge superlinealmente.

8.3.2 Búsqueda de la Sección Dorada

Nos saltamos la bisección en nuestro paralelo de técnicas de búsqueda de raíces de una sola variable. Hay muchas razones para esta omisión. Nuestra motivación para la bisección fue que empleó solo la suposición más débil sobre f necesaria para encontrar las raíces: la continuidad. Sin embargo, el teorema del valor intermedio no se aplica a los mínimos de forma intuitiva, por lo que parece que no existe un enfoque tan sencillo.

Sin embargo, es valioso tener disponible al menos una estrategia de minimización que no requiera diferenciabilidad de f como suposición subyacente; después de todo, hay funciones no diferenciables que tienen mínimos claros, como f(x) ≡ |x| en x = 0. Con este fin, una suposición alternativa podría ser que f es unimodular:

Definición 8.5 (Unimodular). Una función f : [a, b] → R es unimodular si existe x [a, b] tal que f es decreciente para x [a, x] y creciente para x [x , b].

En otras palabras, una función unimodular decrece durante algún tiempo y luego comienza a aumentar; no se permiten mínimos localizados. Observe que funciona como |x| no son diferenciables pero siguen siendo unimodulares.

Supongamos que tenemos dos valores x0 y x1 tales que a < x0 < x1 < b. Podemos hacer dos observaciones que nos ayudarán a formular una técnica de optimización:

- Si f(x0) ≥ f(x1), entonces sabemos que f(x) ≥ f(x1) para todo x [a, x0]. Así, el intervalo [a, x0] puede descartarse en nuestra búsqueda de un mínimo de f .

- Si f(x1) ≥ f(x0), entonces sabemos que f(x) ≥ f(x0) para todo x [x1, b], y por lo tanto podemos descartar [x1, b].

Esta estructura sugiere una estrategia potencial para la minimización que comienza con el intervalo [a, b] y elimina iterativamente las piezas de acuerdo con las reglas anteriores.

Sin embargo, queda un detalle importante. Nuestra garantía de convergencia para el algoritmo de bisección provino del hecho de que podíamos eliminar la mitad del intervalo en cuestión en cada iteración.
Podríamos proceder de manera similar, eliminando un tercio del intervalo cada vez; esto requiere dos evaluaciones de f durante cada iteración en las nuevas ubicaciones x0 y x1 . Sin embargo, si evaluar f es costoso, es posible que deseemos reutilizar la información de iteraciones anteriores para evitar al menos una de esas dos evaluaciones.

Por ahora a = 0 yb = 1; las estrategias que derivamos a continuación funcionarán de manera más general al cambiar y escalar. A falta de más información sobre f , también podríamos hacer una elección simétrica x0 = α y x1 = 1 − α para alguna α (0, 1/2). Supongamos que nuestra iteración elimina el intervalo más a la derecha [x1, b]. Entonces, el intervalo de búsqueda se convierte en [0, 1 − α], y conocemos f(α) de la iteración anterior. La siguiente iteración dividirá [0, 1 − α] de manera que x0 = α(1 − α) y x1 = (1 − α)^2. Si nosotros deseamos reutilizar f(α) de la iteración anterior, podríamos establecer (1 − α)^2 = α, dando:

$$\alpha = \frac{1}{2}(3 - \sqrt{5})$$

$$1 - \alpha = 2^{-1}(\sqrt{5} - 1)$$

¡El valor de 1 − α ≡ τ anterior es la proporción áurea! Permite la reutilización de una de las evaluaciones de funciones de las iteraciones anteriores; un argumento simétrico muestra que la misma elección de α funciona si hubiéramos eliminado el intervalo izquierdo en lugar del derecho.

El algoritmo de búsqueda de la sección áurea hace uso de esta construcción (CITE):

1. Toma τ ≡ $\frac{1}{2}(\sqrt{5} - 1)$. e inicializamos a y b para que f sea unimodular en [a, b].
2. Haz una subdivisión inicial x0 = a + (1 − τ)(b − a) y x1 = a + τ(b − a).
3. Inicializar f0 = f(x0) y f1 = f(x1).
4. Iterar hasta que b − a sea lo suficientemente pequeño:
   (a) Si f0 ≥ f1, elimine el intervalo [a, x0] de la siguiente manera:
   - Mover el lado izquierdo: a ← x0
   - Reutilizar la iteración anterior: x0 ← x1, f0 ← f1
   - Generar una nueva muestra: x1 ← a + τ(b − a), f1 ← f(x1)

   (b) Si f1 > f0, elimine el intervalo [x1, b] de la siguiente manera:
   - Mover el lado derecho: b ← x1
   - Reutilizar la iteración anterior: x1 ← x0, f1 ← f0
   - Generar una nueva muestra: x0 ← a + (1 − τ)(b − a), f0 ← f(x0)

Este algoritmo claramente converge incondicional y linealmente. Cuando f no es globalmente unimodal, puede ser difícil encontrar [a, b] tal que f sea unimodal en ese intervalo, lo que limita un poco las aplicaciones de esta técnica; generalmente [a, b] se adivina intentando entre paréntesis un mínimo local de f .

8.4 Estrategias multivariables

Continuamos en nuestro paralelo de nuestra discusión sobre la búsqueda de raíces ampliando nuestra discusión a problemas de múltiples variables. Al igual que con la búsqueda de raíces, los problemas de múltiples variables son considerablemente más difíciles que los problemas de una sola variable, pero aparecen tantas veces en la práctica que vale la pena considerarlos cuidadosamente.

Aquí, consideraremos solo el caso de que f : Rn → R sea diferenciable. Los métodos de optimización más similares a la búsqueda de la sección áurea para funciones no diferenciables tienen aplicaciones limitadas y son difíciles de formular.

8.4.1 Descenso de gradiente

Recuerde de nuestra discusión anterior que f(x) apunta en la dirección del "ascenso más pronunciado" de f en x; de manera similar, el vector − f(x) es la dirección del "descenso más pronunciado". Si nada más, esta definición garantiza que cuando f(x) =0, para α pequeño > 0 debemos tener

$$f(x - \alpha \nabla f(x)) \leq f(x).$$

Supongamos que nuestra estimación actual de la ubicación del mínimo de f es xk . Entonces, podríamos desear elegir xk+1 para que f(xk+1) < f(xk) para una estrategia de minimización iterativa. Una forma de simplificar la búsqueda de xk+1 sería usar uno de nuestros algoritmos unidimensionales de §8.3 en un problema más simple. En particular, considere la función gk(t) ≡ f(xk − t f(xk)), que restringe f a la línea que pasa por xk paralela a f(xk). Gracias a nuestra discusión sobre el gradiente, sabemos que t pequeño producirá una disminución en f .

El algoritmo de descenso de gradiente resuelve iterativamente estos problemas unidimensionales para mejorar nuestra estimación de xk :

1. Elija una estimación inicial x0
2. Iterar hasta la convergencia de xk :
   (a) Tome gk(t) ≡ f(xk − t f(xk)) (b) Use un algoritmo unidimensional para encontrar t minimizando gk sobre todo t ≥ 0 ("búsqueda de línea") (c) Tome xk+1 ≡ xk − t f(xk)

Cada iteración del descenso del gradiente disminuye f(xk), por lo que los valores objetivos convergen. El algoritmo solo termina cuando f(xk) ≈ 0, lo que demuestra que el descenso del gradiente debe alcanzar al menos un mínimo local; Sin embargo, la convergencia es lenta para la mayoría de las funciones f. El proceso de búsqueda de línea puede ser reemplazado por un método que simplemente disminuye el objetivo en una cantidad no despreciable aunque subóptima, aunque es más difícil garantizar la convergencia en este caso.

8.4.2 Método de Newton

Paralelamente a nuestra derivación del caso de una sola variable, podemos escribir una aproximación en serie de Taylor de f : Rn → R usando su Hessian Hf :

$$f(x) \approx f(xk) + \nabla f(xk) \cdot (x - xk) + \frac{1}{2}(x - xk) \cdot Hf(xk) \cdot (x - xk)$$

Derivando con respecto ax e igualando el resultado a cero se obtiene el siguiente esquema iterativo:

$$x_{k+1} = x_k - [Hf(x_k)]^{-1} \quad f(x_k)$$

Es fácil volver a comprobar que esta expresión es una generalización de la de §8.3.1, y una vez más converge cuadráticamente cuando x0 está cerca de un mínimo.

El método de Newton puede ser más eficiente que el descenso de gradiente según el objetivo de optimización f . Recuerde que cada iteración del descenso de gradiente requiere potencialmente muchas evaluaciones de f durante el procedimiento de búsqueda de línea. Por otro lado, debemos evaluar e invertir el Hessian Hf durante cada iteración del método de Newton. Tenga en cuenta que estos factores no afectan la cantidad de iteraciones, pero sí afectan el tiempo de ejecución: esta es una compensación que puede no ser obvia a través del análisis tradicional.

Es intuitivo por qué el método de Newton converge rápidamente cuando está cerca de un óptimo. En particular, el descenso de gradiente no tiene conocimiento de Hf ; procede de manera análoga a caminar cuesta abajo mirando solo a tus pies. Al usar Hf , el método de Newton tiene una imagen más grande de la forma de f cercana.

Sin embargo, cuando Hf no es definida positiva, el objetivo localmente puede parecer una silla de montar o un pico en lugar de un cuenco. En este caso, saltar a un punto estacionario aproximado podría no tener sentido.
Por lo tanto, las técnicas adaptativas podrían verificar si Hf es definida positiva antes de aplicar un paso de Newton; si no es definida positiva, los métodos pueden volver al descenso de gradiente para encontrar una mejor aproximación del mínimo. Alternativamente, pueden modificar Hf , por ejemplo, proyectándose sobre la matriz definida positiva más cercana.

8.4.3 Optimización sin Derivadas: BFGS

El método de Newton puede ser difícil de aplicar a funciones complicadas f : Rn → R. La segunda derivada de f puede ser considerablemente más complicada que la forma de f , y Hf cambia con cada iteración, lo que dificulta la reutilización del trabajo de iteraciones anteriores. Además, Hf tiene un tamaño n × n, por lo que almacenar Hf requiere O(n ^{2}) espacio, que puede ser inaceptable.

Al igual que en nuestra discusión sobre la búsqueda de raíces, las técnicas de minimización que imitan el método de Newton pero usan derivadas aproximadas se denominan métodos cuasi-Newton. A menudo, pueden tener propiedades de convergencia fuertes similares sin la necesidad de una reevaluación explícita e incluso la factorización de la arpillera en cada iteración. En nuestra discusión, seguiremos el desarrollo de (CITE NO CEDAL AND WRIGHT).

Supongamos que deseamos minimizar f : Rn → R usando un esquema iterativo. Cerca de la estimación actual xk de la raíz, podríamos estimar f con un modelo cuadrático:

$$f(x_k + \delta x) \approx f(x_k) + \quad f(x_k) \cdot \delta x + \quad \frac{1}{2} (\delta x)\, B_k(\delta x).$$

Observe que hemos pedido que nuestra aproximación concuerde con f de primer orden en xk ; Sin embargo, al igual que en el método de Broyden para encontrar raíces, permitiremos que nuestra estimación de Hessian Bk varíe.

Este modelo cuadrático se minimiza tomando $\delta x = -B_k^{-1} \quad f(x_k)$. En caso de que δx2 sea grande y no deseemos dar un paso tan considerable, nos permitiremos escalar esta diferencia con un tamaño de paso αk , resultando

$$x_{k+1} = x_k - \alpha_k B_k^{-1} \quad f(x_k).$$

Nuestro objetivo es encontrar una estimación razonable de Bk+1 actualizando Bk , de modo que podamos repetir este proceso.

La hessiana de f no es más que la derivada de f , por lo que podemos escribir una condición de estilo secante en Bk+1 :

$$B_{k+1}(x_{k+1} - x_k) = f(x_{k+1}) - f(x_k).$$

Sustituiremos $s_k \equiv x_{k+1} - x_k$ y $y_k \equiv f(x_{k+1}) - f(x_k)$, dando una condición equivalente $B_{k+1}s_k = y_k$.

Dada la optimización en cuestión, deseamos que Bk tenga dos propiedades:

- Bk debe ser una matriz simétrica, como la hessiana Hf .
- Bk debe ser positivo (semi-)definido, por lo que estamos buscando mínimos en lugar de máximos o puntos de silla.

La condición de simetría es suficiente para eliminar la posibilidad de usar la estimación de Broyden que desarrollamos en el capítulo anterior.

La restricción definida positiva pone implícitamente una condición sobre la relación entre sk y En particular, premultiplicando la relación $B_{k+1}s_k = y_k$ por s $B_{k+1}s_k = s_k y_k$. Para y_k . $_k$ muestra s_k
Para que Bk+1 sea definida positiva, debemos tener $s_k \cdot y_k > 0$. Esta observación puede guiar nuestra elección de α_k ; es fácil ver que se cumple para $\alpha_k > 0$ suficientemente pequeños.

Suponga que sk e yk satisfacen nuestra condición de compatibilidad. Con esto en su lugar, podemos escribir abajo una optimización de estilo Broyden que conduce a una posible aproximación Bk+1 :

$$\begin{aligned} \underset{B_{k+1}}{\text{minimizar}} \quad & \|B_{k+1} - B_k\| \\ \text{tal que} \quad & B_{k+1} = \text{fondo}+1 \\ & B_{k+1}s_k = y_k \end{aligned}$$

Para elección adecuada de normas · Fletcher- ， esta optimización produce el conocido DFP (Davidon Powell) esquema iterativo.

En lugar de trabajar en los detalles del esquema DFP, pasamos a un método más popular conocido como la fórmula BFGS (Broyden-Fletcher-Goldfarb-Shanno), que aparece en muchos sistemas modernos. Observe que, ignorando nuestra elección de α_k por ahora, nuestra aproximación de segundo orden se minimizó tomando $\delta x = -B_k^{-1} \nabla f(x_k)$. Así, al final, el comportamiento de nuestro esquema iterativo está dictada por la matriz inversa B_k^{-1} . Preguntar que $B_{k+1} - B_k$ es pequeño aún puede implicar diferencias relativamente malas entre la acción de B_k^{-1} y el de B_{k+1}^{-1}

Con esta observación en mente, el esquema BFGS hace una pequeña alteración a la derivación anterior. En lugar de calcular Bk en cada iteración, podemos calcular su inversa $H_k \equiv B_k^{-1}$ directamente.
Ahora nuestra condición $B_{k+1}s_k = y_k$ se invierte $s_k = H_{k+1}y_k$; la condición de que Bk sea simétrico es lo mismo que pedir que Hk sea simétrico. Resolvemos una optimización

$$\begin{aligned} \underset{H_{k+1}}{\text{minimizar}} \quad & \|H_{k+1} - H_k\| \\ \text{tal que} \quad & H_{k+1} = H_{k+1}k \\ & +1s_k = H_k +1y_k \end{aligned}$$

Esta construcción tiene el beneficio adicional de no requerir la inversión de la matriz para calcular $\delta x = -H_k \nabla f(x_k)$.

Para derivar una fórmula para Hk+1 , debemos decidir sobre una norma matricial · . Al igual que con nuestra discusión anterior, la norma de Frobenius se parece más a la optimización de mínimos cuadrados, lo que hace probable que podamos generar una expresión de forma cerrada para Hk+1 en lugar de tener que resolver la minimización anterior como una subrutina de optimización BFGS.

La norma de Frobenius, sin embargo, tiene un serio inconveniente para las matrices hessianas. Recuerde que la matriz hessiana tiene entradas (Hf)ij = ∂ fi/∂xj . A menudo, las cantidades xi para diferentes i pueden tener diferentes unidades; Por ejemplo, considere maximizar la ganancia (en dólares) obtenida vendiendo una hamburguesa con queso de radio r (en pulgadas) y precio p (en dólares), lo que lleva a f : (pulgadas, dólares) → dólares. Elevar al cuadrado estas diferentes cantidades y sumarlas no tiene sentido.

Supongamos que encontramos una matriz definida positiva simétrica W tal que Wsk = yk ; comprobaremos en los ejercicios que tal matriz existe. Tal matriz lleva las unidades de sk = xk+1 − xk a las de yk = f(xk+1) − f(xk). Inspirándonos en nuestra expresión $A = \text{Tr}(A^{\text{Para}} A)$, podemos definir una norma de Frobenius ponderada de una matriz A como

$$\|A\|_W^2 \equiv \text{Tr}(A \, WAW)$$

Es sencillo comprobar que esta expresión tiene unidades consistentes cuando se aplica a nuestra optimización para Hk+1 . Cuando tanto W como A son simétricas con las columnas w i andai , respectivamente, expandir la expresión anterior muestra:

$$\|A\|_W^2 = \sum_{yo} (w_i \cdot aj)(w_j \cdot ai).$$

Esta elección de norma combinada con la elección de W produce una fórmula particularmente limpia para Hk+1 y yk : dada Hk ,sk ,

$$Hk+1 = (\text{In}\times n - \rho k s k y_k)\text{Hk}(\text{In}\times n - \rho k y k s_k) + \rho k s k s_k ,$$

donde ρk ≡ 1/y·s. En el apéndice de este capítulo mostramos cómo derivar esta fórmula.

El algoritmo BFGS evita la necesidad de calcular e invertir una matriz hessiana para f , pero aún requiere O(n Una variante útil conocida como L-BFGS ("BFGS de memoria limitada") evita este problema al mantener un historial limitado de vectores yk andsk y aplicando Hk expandiendo su fórmula recursivamente. Este enfoque en realidad puede tener mejores propiedades numéricas a pesar de su uso compacto del espacio; en particular, los vectores antiguos yk ysk pueden ya no ser relevantes y deben ignorarse.

8.5 Problemas

Lista de ideas:

- Derivar Gauss-Newton
- Métodos estocásticos, AdaGrad
- Algoritmo VSCG
- Condiciones de Wolfe para descenso de gradiente; conectar a BFGS
- Fórmula de Sherman-Morrison-Woodbury para Bk para BFGS
- Demostrar convergencia de BFGS; mostrar la existencia de una matriz W
- Algoritmo de gradiente reducido (generalizado)
- Número de condición para la optimización

Apéndice: Derivación de BFGS Update1

Nuestra optimización para Hk+1 tiene la siguiente expresión del multiplicador de Lagrange (para facilitar la notación, tomamos Hk+1 ≡ H y Hk = H):

$$\Lambda \equiv \sum_{yo} (w\ yo \cdot (hj - h_j))(w\ j \cdot (hola - h_i)) - \sum_{yo<j} \alpha ij(Hij - Hji) - \lambda\ (Hyk - sk)$$

$$= \sum_{yo} (w\ yo \cdot (hj - h_j))(w\ j \cdot (hola - h_i)) - \sum_{yo} \alpha ijHij - \lambda\ (Hyk - sk) \text{ si suponemos } \alpha ij = -\alpha ji$$

Tomar derivadas para encontrar puntos críticos muestra (para y ≡ yk ,s ≡sk):

$$0 = \frac{\partial \Lambda}{\partial Hij} = \sum 2wi(w\ j \cdot (h - h)) - \alpha ij - \lambda iyj$$

$$= 2\sum wi(W(H - H))j - \alpha ij - \lambda iyj$$

$$= 2\sum (W(H - H))jwi - \alpha ij - \lambda iyj \text{ por simetría de W}$$

$$= 2(W(H - H)W)ji - \alpha ij - \lambda iyj$$

$$= 2(W(H - H)W)ij - \alpha ij - \lambda iyj \text{ por simetría de W y H}$$

Entonces, en forma matricial tenemos la siguiente lista de hechos:

$$0 = 2W(H - H)W - A - \lambda y \text{ , donde } A_{ij} = \alpha ij$$

$$A = -A,\ W = W,\ H = H,(H) = H$$

$$Hy = s,\ Ws = y$$

Podemos lograr un par de relaciones usando transposición combinada con simetría de H y W y asimetría de A:

$$0 = 2W(H - H)W - A - \lambda y$$

$$0 = 2W(H - H)W + A - y\lambda = 0 =$$

$$4W(H - H)W - \lambda y - y\lambda$$

Posterior a la multiplicación de esta relación por muestra:

$$0 = 4(y - WH\ y) - \lambda(y \cdot s) - y(\lambda \cdot s)$$

Ahora, toma el producto punto con:

$$0 = 4(y \cdot s) - 4(y\ H\ y) - 2(y \cdot s)(\lambda \cdot s)$$

Esta espectáculos:

$$\lambda \cdot s = 2\rho y\ (s - H\ y), \text{ para } \rho \equiv 1/y \cdot s$$

1Agradecimiento especial a Tao Du por depurar varias partes de esta derivación.

Ahora, sustituimos esto en nuestra igualdad vectorial:

$$\begin{aligned} 0 &= 4(y - WH\quad y) - \lambda(y \cdot s) - y(\lambda \cdot s) \text{ desde antes} \\ &= 4(y - WH\quad y) - \lambda(y \cdot s) - y[2\rho y\ (s - H\quad y)] \text{ de nuestra simplificación} \\ = \quad \lambda &= 4\rho(y - WH\quad y) - 2\rho\ ^2 y\ (s - H\quad y)y \end{aligned}$$

Después de multiplicar por y muestra:

$$\lambda y = 4\rho(y - WH\quad y)y - 2\rho\quad ^2 y\ (s - H\quad y)yy$$

Tomando la transposición,

$$y\lambda = 4\rho y(y - y\ H\quad W) - 2\rho\quad ^2 y\ (s - H\quad y)yy$$

La combinación de estos resultados y la división por cuatro muestra:

$$\frac{1}{4}(\lambda y + y\lambda) = \rho(2yy - WH\quad yy - yy\ H\quad W) - \rho\quad ^2 y\ (s - H\quad y)yy$$

Ahora, pre- y post-multiplicaremos por W^{-1} . Como $Ws = y$, podemos escribir de manera equivalente $= W^{-1}y$; además, por la simetría de W sabemos que $y\ W^{-1} = s$. La aplicación de estas identidades a la expresión anterior muestra:

$$\begin{aligned} \frac{1}{4}W^{-1}(\lambda y + y\lambda)W^{-1} &= 2\rho ss - \rho H\quad ys - \rho sy\ H\quad - \rho\ ^2(y\ s)\ ss + \rho\quad ^2(yH\quad y)ss \\ &= 2\rho ss - \rho H\quad ys - \rho sy\ H\quad - \rho ss + s\rho\quad ^2(y\ H\quad y)s \text{ por definición de } \rho \\ &= \rho ss - \rho H\quad ys - \rho sy\ H\quad + s\rho\ ^2(yH\quad y)s \end{aligned}$$

Finalmente, podemos concluir nuestra derivación del paso BFGS de la siguiente manera:

$$\begin{aligned} 0 &= 4W(H - H\quad)W - \lambda y - y\lambda \text{ de antes} \\ = \quad H &= 4\ \frac{1}{=} W^{-1}(\lambda y + y\lambda)W^{-1} + H \\ &\quad \rho ss - \rho H\quad ys - \rho sy\ H\quad + s\rho\ ^2(y\ H\quad y)s + H\quad \text{del último párrafo} \\ &= H\quad (I - \rho ys) + \rho ss - \rho sy\ H\quad (I\quad + (\rho sy)H\quad (\rho ys) \\ &= H\quad - \rho ys) + \rho ss - \rho sy\ H\quad (I - \rho ys) = \rho ss + (I \\ &\quad - \rho sy)H\quad (I - \rho ys) \end{aligned}$$

Esta expresión final es exactamente el paso BFGS presentado en el capítulo.

Capítulo 9

Optimización con restricciones

Continuamos nuestra consideración de los problemas de optimización estudiando el caso restringido. Estos problemas toman la siguiente forma general:

$$\text{minimizar } f(x) \text{ tal}$$
$$\text{que } g(x) = 0 \; h(x) \geq 0$$

Aquí, $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$ y $h : \mathbb{R}^n \to \mathbb{R}^p$. Obviamente, esta forma es extremadamente genérica, por lo que no es difícil predecir que los algoritmos para resolver tales problemas en ausencia de suposiciones adicionales sobre f, g o h pueden ser difíciles de formular y están sujetos a degeneraciones como mínimos locales y falta de convergencia. De hecho, esta optimización codifica otros problemas que ya hemos considerado; si tomamos $f(x) \equiv 0$, entonces esta optimización restringida se convierte en búsqueda de raíces en g, mientras que si tomamos $g(x) = h(x) \equiv 0$, entonces se reduce a una optimización sin restricciones en f .

A pesar de esta perspectiva algo sombría, las optimizaciones para el caso general restringido pueden ser valiosas cuando f, g y h no tienen una estructura útil o son demasiado especializadas para merecer un tratamiento especializado. Además, cuando f es heurística de todos modos, simplemente encontrar un factiblex para el cual $f(x) < f(x_0)$ para una suposición inicial x_0 es valioso. Una aplicación simple en este dominio sería un sistema económico en el que f mide costos; obviamente deseamos minimizar los costos, pero si x_0 representa la configuración actual, cualquier x que disminuya f es un resultado valioso.

9.1 Motivación

No es difícil encontrar problemas de optimización con restricciones en la práctica. De hecho, ya enumeramos muchas aplicaciones de estos problemas cuando discutimos los vectores propios y los valores propios, ya que este problema se puede plantear como encontrar puntos críticos de $x^\top Ax$ sujeto a $\|x\|_2^2 = 1$; por supuesto, el caso particular del cálculo de valores propios admite algoritmos especiales que lo hacen un problema más simple.

Aquí enumeramos otras optimizaciones que no disfrutan de la estructura de los problemas de valores propios:

Ejemplo 9.1 (Proyección geométrica). Muchas superficies S en $\mathbb{R}^3$ se pueden escribir implícitamente en la forma $g(x) = 0$ para alguna g. Por ejemplo, la esfera unitaria resulta de tomar $g(x) \equiv \|x\|_2^2 - 1$, mientras que un cubo puede

construirse tomando g(x) = x1 − 1. De hecho, algunos entornos de modelado 3D permiten a los usuarios especificar objetos "blobby", como en la Figura NÚMERO, como sumas

$$g(x) \equiv c + \sum_i a_i e^{-b_i \|x - x_i\|_2^2}.$$

Supongamos que nos dan un punto y R3 y deseamos encontrar el punto más cercano en S a y. Este problema se resuelve usando la siguiente minimización restringida:

$$\begin{aligned} &\text{minimizar}_x \ \|x - y\|_2 \text{ tal que} \\ &g(x) = 0 \end{aligned}$$

Ejemplo 9.2 (Fabricación). Suponga que tiene m materiales diferentes; tienes si unidades de cada material i en stock. Puede fabricar k productos diferentes; el producto j te da una ganancia pj y usa cij del material i para fabricarlo. Para maximizar las ganancias, puede resolver la siguiente optimización para la cantidad total xj que debe fabricar de cada artículo j:

$$\begin{aligned} &\text{maximizar}_x \ p_j x_j \sum_{j=1}^{k} \\ &\text{tal que } x_j \geq 0 \quad j \quad \{1, \ldots, k\} \\ &\sum_{j=1}^{k} c_{ij} x_j \leq s_i \quad i \quad \{1, \ldots, m\} \end{aligned}$$

La primera restricción asegura que no haga números negativos de ningún producto, y la segunda asegura que no use más que su stock de cada material.

Ejemplo 9.3 (Mínimos cuadrados no negativos). Ya hemos visto numerosos ejemplos de problemas de mínimos cuadrados, pero a veces los valores negativos en el vector solución pueden no tener sentido. Por ejemplo, en gráficos por computadora, un modelo animado podría expresarse como una estructura ósea deformante más una “piel” enredada; para cada punto de la piel se puede calcular una lista de pesos para aproximar la influencia de las posiciones de las articulaciones óseas sobre la posición de los vértices de la piel (CITE). Dichos pesos deben limitarse a ser no negativos para evitar un comportamiento degenerado mientras la superficie se deforma. En tal caso, podemos resolver el problema de los “mínimos cuadrados no negativos”:

$$\begin{aligned} &\text{minimizar}_x \ \|Ax - b\|_2 \text{ tal que} \\ &x_i \geq 0 \quad i \end{aligned}$$

Investigaciones recientes implican caracterizar la escasez de soluciones de mínimos cuadrados no negativos, que a menudo tienen varios valores xi que satisfacen xi = 0 exactamente (CITE).

Ejemplo 9.4 (Ajuste de paquete). En visión artificial, supongamos que tomamos una fotografía de un objeto desde varios ángulos. Una tarea natural es reconstruir la forma tridimensional del objeto. Para hacerlo, podríamos marcar un conjunto correspondiente de puntos en cada imagen; en particular, podemos tomar xij R2 como la posición del punto característico j en la imagen i. En realidad, cada punto característico tiene una posición yj R3 en el espacio, que nos gustaría calcular. Además, debemos encontrar las posiciones de las propias cámaras, que podemos representar como

matrices de proyección desconocidas Pi . Este problema, conocido como ajuste de paquete, se puede abordar mediante una estrategia de optimización:

$$\text{minimizaryj ,Pi} \sum_{yo} \text{Piyj} - \text{xij}_{2}^{2}$$

$$\text{tal que Pi es ortogonal} \quad i$$

La restricción de ortogonalidad asegura que las transformaciones de la cámara sean razonables.

9.2 Teoría de la Optimización Restringida

En nuestra discusión, supondremos que f, g y h son diferenciables. Existen algunos métodos que solo hacen suposiciones de continuidad débil o de Lipschitz, pero estas técnicas son bastante especializadas y requieren una consideración analítica avanzada.

Aunque todavía no hemos desarrollado algoritmos para la optimización restringida general, implícitamente hemos hecho uso de la teoría de tales problemas al considerar los métodos de valor propio. Específicamente, recuerde el método de los multiplicadores de Lagrange, presentado en el Teorema 0.1. En esta técnica, los puntos críticos f(x) sujetos a g(x) se caracterizan como puntos críticos de la función multiplicadora de La grange sin restricciones $\Lambda(x,\lambda) \equiv f(x) - \lambda \cdot g(x)$ con respecto a ambos λ yx simultáneamente.

Este teorema nos permitió proporcionar interpretaciones variacionales de los problemas de valores propios; más generalmente, proporciona un criterio alternativo (necesario pero no suficiente) para que x sea un punto crítico de una optimización con restricciones de igualdad.

Sin embargo, simplemente encontrar una x que satisfaga las restricciones puede ser un desafío considerable. Podemos separar estos temas haciendo algunas definiciones:

Definición 9.1 (Punto factible y conjunto factible). Un punto factible de un problema de optimización con restricciones es cualquier punto x que satisfaga g(x) =0 y h(x) ≥0. El conjunto factible es el conjunto de todos los puntos x que satisfacen estas restricciones.

Definición 9.2 (Punto crítico de optimización restringida). Un punto crítico de una optimización restringida es uno que satisface las restricciones que también es un máximo, mínimo o punto de silla local de f dentro del conjunto factible.

Las optimizaciones restringidas son difíciles porque resuelven simultáneamente problemas de búsqueda de raíces (la restricción g(x) =0), problemas de satisfacibilidad (la restricción h(x) ≥0) y minimización (la función f). Aparte de esto, para llevar nuestras técnicas diferenciales a una generalidad completa, debemos encontrar una manera de agregar restricciones de desigualdad al sistema multiplicador de Lagrange. Supongamos que hemos encontrado el mínimo de la optimización, denotado x. Para cada restricción de desigualdad hi(x) ≥ 0, tenemos dos opciones:

- hi(x) = 0: dicha restricción está activa, lo que probablemente indica que si se eliminara la restricción, el óptimo podría cambiar.
- hi(x) > 0: Tal restricción está inactiva, lo que significa que en una vecindad de x si hubiésemos eliminado esta restricción aún habríamos alcanzado el mismo mínimo.

Por supuesto, no sabemos qué restricciones estarán activas o inactivas en x hasta que se calcule.

Si todas nuestras restricciones estuvieran activas, entonces podríamos cambiar nuestra restricción h(x) ≥0 a una igualdad sin afectar el mínimo. Esto podría motivar a estudiar el siguiente sistema multiplicador de Lagrange:

$$\Lambda(x,\lambda,\mu) \equiv f(x) - \lambda \cdot g(x) - \mu \cdot h(x)$$

Sin embargo, ya no podemos decir que x es un punto crítico de Λ porque las restricciones inactivas eliminarían los términos anteriores. Ignorando este (¡importante!) problema por el momento, podríamos proceder a ciegas y pedir puntos críticos de este nuevo Λ con respecto a x, que satisfagan lo siguiente:

$$0 = f(x) - \sum_i \lambda_i g_i(x) - \sum_j \mu_j h_j(x)$$

Aquí hemos separado los componentes individuales de g y h y los hemos tratado como funciones escalares para evitar una notación compleja.

Un truco inteligente puede extender esta condición de optimalidad a los sistemas con restricciones de desigualdad. Note que si hubiéramos tomado μj = 0 siempre que hj esté inactiva, entonces esto elimina los términos irrelevantes de las condiciones de optimización. En otras palabras, podemos agregar una restricción a los multiplicadores de Lagrange:

$$\mu_j h_j(x) = 0.$$

Con esta restricción en su lugar, sabemos que al menos uno de μj y hj(x) debe ser cero y, por lo tanto, ¡nuestra condición de optimalidad de primer orden aún se cumple!

Hasta ahora, nuestra construcción no ha distinguido entre la restricción hj(x) ≥ 0 y la restricción hj(x) ≤ 0. Si la restricción está inactiva, podría haberse eliminado sin afectar el resultado de la optimización localmente, por lo que considere el caso cuando la restricción está activa. Intuitivamente,[1] en este caso esperamos que haya una manera de disminuir f violando la restricción. Localmente, la dirección en la que f decrece es $-\nabla f(x^*)$ y la dirección en la que hj decrece es $-\nabla h_j(x^*)$. Así, comenzando en x* podemos disminuir f aún más violando la restricción hj(x) ≥ 0 cuando $\nabla f(x^*) \cdot \nabla h_j(x^*) \geq 0$.

Por supuesto, es difícil trabajar con productos de gradientes de f y hj . Sin embargo, recordemos que en nuestra condición de optimalidad de primer orden nos dice:

$$\nabla f(x^*) = \sum_i \lambda_i^* \nabla g_i(x^*) + \sum_{\mu_j \text{ activo}} \mu_j \nabla h_j(x^*)$$

Los valores μj inactivos son cero y se pueden eliminar. De hecho, podemos eliminar las restricciones g(x) = 0 agregando restricciones de desigualdad g(x) ≥ 0 y g(x) ≤ 0 a h; esta es una conveniencia matemática para escribir una prueba en lugar de una maniobra numérica. Luego, tomando productos escalares con ∇h_k para cualquier k fijo, se muestra:

$$\sum_{\mu_j \text{ activo}} \mu_j \nabla h_j(x^*) \cdot \nabla h_k(x^*) = \nabla f(x^*) \cdot \nabla h_k(x^*) \geq 0$$

Vectorizar esta expresión muestra $Dh(x^*)Dh(x^*)^\mu$ inite, esto ≥ 0. Como $Dh(x^*)Dh(x^*)^\mu$ es semidef positivo implica μ el hecho de que ≥ 0. Así, la observación $\nabla f(x^*) \cdot \nabla h_j(x^*) \geq 0$ se manifiesta simplemente por μj ≥ 0.

Nuestras observaciones se pueden formalizar para probar una condición de optimalidad de primer orden para optimizaciones con restricciones de desigualdad:

[1] No debe considerar nuestra discusión como una prueba formal, ya que no estamos considerando muchos casos límite.

Teorema 9.1 (condiciones de Karush-Kuhn-Tucker (KKT)). El vector x Rn es un punto crítico para minimizar f sujeto a g(x) =0 y h(x) ≥0 cuando existe λ Rm y μ Rp tal que:

- 0 = f(x) − ∑i λi gi(x) − ∑j μj hj(x) (“estacionariedad”)
- g(x) =0 y h(x) ≥0 (“factibilidad primal”) •
- μjhj(x) = 0 para todo j (“holgura complementaria”) •
- μj ≥ 0 para todo j (“factibilidad dual ”)

Observe que cuando se elimina h, este teorema se reduce al criterio del multiplicador de Lagrange.

Ejemplo 9.5 (Optimización simple2). Supongamos que deseamos resolver

$$\begin{aligned} &\text{maximizar } xy \\ &\text{tal que } x + yx, y^2 \le 2 \\ &\ge 0 \end{aligned}$$

En este caso no tendremos λ y tres μ. Tomamos f(x, y) = −xy, h1(x, y) ≡ 2− x − y y h3(x, y) = y. Las ^{2}, h2(x, y) = x, condiciones de KKT son:

$$\begin{aligned} &\text{Estacionariedad: } 0 = -y + \mu 1 - \\ &\qquad \mu 2\ 0 = -x + 2\mu 1y - \mu 3 \\ &\text{Viabilidad primaria: } x + yx,^2 \le 2 \\ &\qquad y \ge 0 \\ &\text{Holgura complementaria: } \mu 1(2 - x - y\ \mu 2x = {}^2) = 0 \\ &\qquad 0\ \mu 3y = 0 \\ &\text{Viabilidad dual: } \mu 1, \mu 2, \mu 3 \ge 0 \end{aligned}$$

Ejemplo 9.6 (Programación lineal). Considere la optimización:

$$\begin{aligned} &\text{minimizarx } b \cdot x \\ &\text{tal que } Ax \ge c \end{aligned}$$

Observe que el ejemplo 9.2 se puede escribir de esta manera. Las condiciones KKT para este problema son:

$$\begin{aligned} &\text{Estacionariedad: } A\mu = b \\ &\text{Viabilidad primaria: } Ax \ge c \\ &\text{Holgura complementaria: } \mu i(ai \cdot x - ci) = 0 \quad i, \text{dondea}_{\ i} \text{ es la fila i de A} \\ &\text{Viabilidad dual: } \mu \ge 0 \end{aligned}$$

Al igual que con el caso de los multiplicadores de Lagrange, no podemos suponer que cualquier x que satisfaga las condiciones KKT minimice automáticamente f sujeto a las restricciones, incluso localmente. Una forma de verificar la optimización local es examinar la hessiana de f restringida al subespacio de Rn en el que x puede moverse sin violar las restricciones; si este hessiano "reducido" es definido positivo, entonces la optimización ha alcanzado un mínimo local.

2De http://www.math.ubc.ca/~israel/m340/kkt2.pdf

9.3 Algoritmos de optimización

Una consideración cuidadosa de los algoritmos para la optimización restringida está fuera del alcance de nuestra discusión; afortunadamente existen muchas implementaciones estables de estas técnicas y mucho se puede lograr como un "cliente" de este software en lugar de volver a escribirlo desde cero. Aun así, es útil esbozar algunos enfoques potenciales para ganar algo de intuición sobre cómo funcionan estas bibliotecas.

9.3.1 Programación Cuadrática Secuencial (SQP)

Similar a BFGS y otros métodos que consideramos en nuestra discusión de la optimización sin restricciones, una estrategia típica para la optimización restringida es aproximar f, g y h con funciones más simples, resolver la optimización aproximada e iterar.

Supongamos que tenemos una conjetura xk de la solución al problema de optimización con restricciones. Podríamos aplicar una expansión de Taylor de segundo orden a f y una aproximación de primer orden a g y h para definir una siguiente iteración como la siguiente:

$$x_{k+1} \equiv x_k + \underset{d}{\text{argumento mínimo}} \; \frac{1}{2} d H f(x_k) d + \quad f(x_k) \cdot d + f(x_k)$$

$$\text{tal que } g_i(x_k) + \quad g_i(x_k) \cdot d = 0$$

$$h_i(x_k) + \quad h_i(x_k) \cdot d \geq 0$$

La optimización para encontrar d tiene un objetivo cuadrático con restricciones lineales, para lo cual la optimización puede ser considerablemente más fácil usando una de muchas estrategias. Se conoce como un programa cuadrático.

Por supuesto, esta aproximación de Taylor solo funciona en una vecindad del punto óptimo. Cuando no se dispone de una buena suposición inicial x0 , es probable que estas estrategias fracasen.

Restricciones de igualdad Cuando las únicas restricciones son las igualdades y se elimina h, el programa cuadrático para d tiene condiciones de optimalidad del multiplicador de Lagrange derivadas de la siguiente manera:

$$1\ \Lambda(d,\lambda) \equiv \quad \frac{1}{2} d H f(x_k) d + \quad f(x_k) \cdot d + f(x_k) + \lambda \left(g(x_k) + Dg(x_k) d\right)$$

$$= \quad 0 = \quad d\Lambda = Hf(x_k) d + \quad f(x_k) + [Dg(x_k)]\lambda$$

Combinando esto con la condición de igualdad se obtiene un sistema lineal:

$$\begin{bmatrix} Hf(x_k) & [Dg(x_k)] \\ Dg(x_k) & 0 \end{bmatrix} \begin{matrix} d \\ \lambda \end{matrix} = \begin{matrix} - \quad f(x_k) \\ -g(x_k) \end{matrix}$$

Por lo tanto, cada iteración de la programación cuadrática secuencial en presencia de solo restricciones de igualdad puede lograrse resolviendo este sistema lineal en cada iteración para obtener xk+1 ≡ xk + d. Es importante tener en cuenta que el sistema lineal anterior no es definido positivo, por lo que a gran escala puede ser difícil de resolver.

Las extensiones de esta estrategia funcionan como BFGS y aproximaciones similares funcionan para la optimización sin restricciones, mediante la introducción de aproximaciones de Hessian Hf . La estabilidad también se puede introducir limitando la distancia recorrida en una sola iteración.

Restricciones de desigualdad Existen algoritmos especializados para resolver programas cuadráticos en lugar de programas no lineales generales, y estos pueden usarse para generar pasos de SQP. Una estrategia notable es mantener un "conjunto activo" de restricciones que estén activas al mínimo con respecto a d; entonces los métodos con restricciones de igualdad anteriores se pueden aplicar ignorando las restricciones inactivas. Las iteraciones de la optimización del conjunto activo actualizan el conjunto activo de restricciones agregando restricciones violadas al conjunto activo y eliminando las restricciones de desigualdad hj para las cuales f · hj ≤ 0 como en nuestra discusión de las condiciones KKT.

9.3.2 Métodos de barrera

Otra opción para minimizar en presencia de restricciones es cambiar las restricciones a términos de energía. Por ejemplo, en el caso de igualdad restringida, podríamos minimizar un objetivo "aumentado" de la siguiente manera:

$$f_\rho(x) = f(x) + \rho g(x)^2_2$$

Note que tomar $\rho \to \infty$ obligará a g(x) a ser lo más pequeño posible, por lo que eventualmente llegaremos a g(x) ≈0. Por lo tanto, el método de barrera de la optimización restringida aplica técnicas iterativas de optimización sin restricciones a fρ y verifica qué tan bien se satisfacen las restricciones; si no están dentro de una tolerancia dada, se incrementa ρ y la optimización continúa utilizando la iteración anterior como punto de partida.

Los métodos de barrera son simples de implementar y usar, pero pueden exhibir algunos modos de falla perniciosos. En particular, a medida que aumenta ρ , la influencia de f sobre la función objetivo disminuye y la hessiana de fρ se vuelve cada vez más pobremente condicionada.

Los métodos de barrera también se pueden aplicar a las restricciones de desigualdad. Aquí debemos asegurarnos de que hi(x) ≥ 0 para todo i; Las elecciones típicas de funciones de barrera pueden incluir 1/hi(x) (la “barrera inversa”) o − log hi(x) (la “barrera logarítmica”).

9.4 Programación convexa

En términos generales, los métodos como los que hemos descrito para la optimización restringida vienen con pocas o ninguna garantía sobre la calidad de la salida. Ciertamente, estos métodos no pueden obtener mínimos globales sin una buena suposición inicial x0, y en ciertos casos, por ejemplo, cuando el Hessian
cerca de x no es definida positiva, es posible que no converjan en absoluto.

Hay una notable excepción a esta regla, que aparece en una serie de optimizaciones importantes: la programación convexa. La idea aquí es que cuando f es una función convexa y el propio conjunto factible es convexo, entonces la optimización posee un mínimo único. Ya hemos definido una función convexa, pero necesitamos entender qué significa que un conjunto de restricciones sea convexo:

Definición 9.3 (Conjunto convexo). Un conjunto S Rn es convexo si para cualquier x,y S, el punto tx + (1 − t)y también está en S para cualquier t [0, 1].

Como se muestra en la Figura NÚMERO, intuitivamente un conjunto es convexo si la forma de su límite no puede doblarse tanto hacia adentro como hacia afuera.

Ejemplo 9.7 (Círculos). El disco {x Rn : x2 ≤ 1} es convexo, mientras que el círculo unitario {x Rn : x2 = 1} no lo es.

Es fácil ver que una función convexa tiene un mínimo único incluso cuando esa función está restringida a un dominio convexo. En particular, si la función tuviera dos mínimos locales, entonces la línea de puntos entre esos mínimos debe dar valores de f no mayores que los de los extremos.

Hay fuertes garantías de convergencia disponibles para optimizaciones convexas que garantizan encontrar el mínimo global siempre que f sea convexa y las restricciones sobre g y h formen un conjunto factible convexo. Por lo tanto, un ejercicio valioso para casi cualquier problema de optimización es verificar si es convexo, ya que dicha observación puede aumentar la confianza en la calidad de la solución y las posibilidades de éxito en gran medida.

Un nuevo campo llamado programación convexa disciplinada intenta encadenar reglas simples sobre la convexidad para generar optimizaciones convexas (CITE CVX), lo que permite al usuario final combinar términos y restricciones de energía convexa simples siempre que satisfagan los criterios que hacen que la optimización final sea convexa. Las declaraciones útiles sobre la convexidad en este dominio incluyen las siguientes:

- La intersección de conjuntos convexos es convexa; por lo tanto, agregar múltiples restricciones convexas es una operación permitida.
- La suma de funciones convexas es convexa.
- Si f y g son convexas, también lo es $h(x) \equiv \max\{ f(x), g(x)\}$.
- Si f es una función convexa, el conjunto $\{x : f(x) \leq c\}$ es convexo.

Herramientas como la biblioteca CVX ayudan a separar la implementación de una variedad de objetivos convexos de su minimización.

Ejemplo 9.8 (Programación convexa).

- El problema de mínimos cuadrados no negativos del ejemplo 9.3 es convexo porque $Ax - b_2$ es convexa función de x y el conjunto $x \geq 0$ es convexo.
- El problema de programación lineal del ejemplo 9.6 es convexo porque tiene un objetivo lineal y restricciones
- Podemos incluir x1 en un objetivo de optimización convexo introduciendo una variable y. Para hacerlo, agregamos restricciones $y_i \geq x_i$ y $y_i \geq -x_i$ para cada i y un objetivo $\sum_i y_i$. Esta suma tiene términos que son al menos tan grandes como $|x_i|$ y que la energía y las restricciones son convexas. Como mínimo debemos tener $y_i = |x_i|$ ya que hemos restringido $y_i \geq |x_i|$ y deseamos minimizar la energía. Las bibliotecas convexas "disciplinadas" pueden realizar tales operaciones entre bastidores sin revelar tales sustituciones al usuario final.

Un ejemplo particularmente importante de optimización convexa es la programación lineal del ejemplo 9.6.
El famoso algoritmo simplex realiza un seguimiento de las restricciones activas, resuelve la x resultante utilizando un sistema lineal y verifica si el conjunto activo debe actualizarse; no se necesitan aproximaciones de Taylor porque el conjunto objetivo y factible están dados por maquinaria lineal. Las estrategias de programación lineal de punto interior como el método de barrera también son exitosas para estos problemas. Por esta razón, los programas lineales se pueden resolver a gran escala (¡hasta millones o miles de millones de variables!) y, a menudo, aparecen en problemas como la programación o la fijación de precios.

9.5 Problemas

- ¿Derivar símplex?
- Dualidad de programación lineal

Capítulo 10

Solucionadores lineales iterativos

En los dos capítulos anteriores, desarrollamos estrategias para resolver una nueva clase de problemas que involucran la minimización de una función f(x) con o sin restricciones en x. Al hacerlo, relajamos nuestro punto de vista del álgebra lineal numérica y, en particular, de la eliminación gaussiana de que debemos encontrar una solución exacta para un sistema de ecuaciones y, en cambio, recurrimos a esquemas iterativos que garantizan aproximar el mínimo de una función cada vez mejor a medida que lo hacen. iterar más y más. Incluso si nunca encontramos exactamente el mínimo, sabemos que eventualmente encontraremos un x0 con f(x0) ≈0 con niveles de calidad arbitrarios, dependiendo del número de iteraciones que ejecutemos.

Ahora tenemos una repetición de nuestro problema favorito del álgebra lineal numérica, resolviendo Ax =b para x, pero aplicamos un enfoque iterativo en lugar de esperar encontrar una solución en forma cerrada. Esta estrategia revela una nueva clase de solucionadores de sistemas lineales que pueden encontrar aproximaciones confiables de x en sorprendentemente pocas iteraciones. Ya hemos sugerido cómo abordar esto en nuestra discusión de álgebra lineal, sugiriendo que las soluciones a los sistemas lineales son mínimos de la energía Ax −b entre otros. 2^2,

¿Por qué molestarse en derivar otra clase de solucionadores de sistemas lineales? Hasta ahora, la mayoría de nuestros enfoques directos requieren que representemos A como una matriz completa de n × n, y algoritmos como LU, QR o la factorización de Cholesky toman alrededor de $O(n^3)$ tiempo. Hay dos casos a tener en cuenta para poten
(razones potenciales para probar esquemas iterativos:

1. Cuando A es escaso, los métodos como la eliminación gaussiana tienden a inducir el relleno, lo que significa que incluso) si A contiene O(n) valores distintos de cero, los pasos intermedios de eliminación pueden introducir O(n valores ^{2} distintos de cero. Esta propiedad puede causar rápidamente que los sistemas de álgebra lineal se queden sin memoria. Por el contrario, los algoritmos de este capítulo solo requieren que usted pueda aplicar A a vectores, lo que se puede hacer en un tiempo proporcional al número de valores distintos de cero en una matriz.

2. Es posible que deseemos derrotar al $O(n^3)$ tiempo de ejecución de las técnicas estándar de factorización de matrices. En particular, si un esquema iterativo puede descubrir una solución bastante precisa para Ax = b en unas pocas iteraciones, los tiempos de ejecución pueden reducirse considerablemente.

Además, tenga en cuenta que muchos de los métodos de optimización no lineal que hemos discutido, en particular aquellos que dependen de un paso tipo Newton, requieren resolver un sistema lineal en cada iteración. Por lo tanto, formular el solucionador más rápido posible puede marcar una diferencia considerable al implementar métodos de optimización a gran escala que requieren una o más soluciones lineales por iteración. De hecho, en este caso, una solución imprecisa pero rápida de un sistema lineal podría ser aceptable, ya que de todos modos alimenta una técnica iterativa más grande.

Tenga en cuenta que gran parte de nuestra discusión se debe a CITE, aunque nuestro desarrollo puede ser algo más corto dado el desarrollo en los capítulos anteriores.

10.1 Descenso de gradiente

Centraremos nuestra discusión en resolver Ax =b donde A tiene tres propiedades:

1. A Rn×n es cuadrado
2. A es simétrica, es decir, A = A
3. A es definida positiva, es decir, para todo x = 0, x Ax > 0

Hacia el final de este capítulo relajaremos estas suposiciones. Por ahora, observe que podemos reemplazar Ax = b con las ecuaciones normales A Ax = A b para satisfacer estos criterios, aunque como hemos discutido, esta sustitución puede crear problemas de condicionamiento numérico.

10.1.1 Derivación del esquema iterativo

En este caso, es fácil comprobar que las soluciones de Ax = b son mínimos de la función f(x) dada por la forma cuadrática

$$f(x) \equiv \frac{1}{2} x Ax - bx + c \text{ para}$$

cualquier c R. En particular, tomando la derivada de f muestra

$$f(x) = Ax - b,$$

y establecer f(x) =0 produce el resultado deseado.

En lugar de resolver f(x) = 0 directamente como lo hemos hecho en el pasado, supongamos que aplicamos el estrategia de descenso de gradiente a esta minimización. Recuerde el algoritmo de descenso de gradiente básico:

1. Calcule la dirección de búsqueda d k ≡ − f(xk−1) =b − Axk−1 .
2. Defina xk ≡ xk−1 + αkd k , donde αk se elige tal que f(xk) < f(xk−1)

Para una función genérica f , decidir el valor de αk puede ser un problema unidimensional difícil de “búsqueda de línea”, que se reduce a minimizar f(xk−1 + αkd k) como una función de una sola variable αk ≥ 0. Para nuestra elección particular de la forma cuadrática f(x) = $\frac{1}{2}$x Ax −bx + c, sin embargo, podemos hacer búsqueda lineal en forma cerrada. En particular, definir

$$\begin{aligned} g(\alpha) &\equiv f(x + \alpha d) \\ &= \frac{1}{2}(x + \alpha d) A(x + \alpha d) - b(x + \alpha d) + c \\ &= \frac{1}{2}(x Ax + 2\alpha x Ad + \alpha^2 d Ad) - bx - \alpha b d + c \text{ por simetría de A} \\ &= \frac{\alpha^2}{2} dAd + \alpha(x Ad - b d) + \text{const.} \\ \frac{dg}{d\alpha}(\alpha) &= \alpha dAd + d(Ax - b) \end{aligned}$$

Así, si deseamos minimizar g con respecto a α, simplemente elegimos

$$\alpha = \frac{d(b - Ax)\, dAd}{}$$

En particular, para el descenso de gradiente elegimos d $k = b - Axk$, por lo que, de hecho, αk adopta una forma agradable:

$$ak = \frac{dd\ kk}{d_k \quad \text{Anuncio}}$$

Al final, nuestra fórmula para la búsqueda de líneas produce el siguiente esquema de descenso de gradiente iterativo para resolver Ax =b en el caso definido positivo simétrico:

$$d_k = b - Axk-1$$

$$ak = \frac{dd\ kk}{dk \quad \text{Anuncio}_k}$$

$$xk = xk-1 + \alpha k d\ k$$

10.1.2 Convergencia

Por construcción, nuestra estrategia para el descenso de gradientes disminuye f(xk) cuando k → ∞. Aun así, no hemos demostrado que el algoritmo alcance el valor mínimo posible de f , y no hemos podido caracterizar cuántas iteraciones debemos ejecutar para alcanzar un nivel de confianza razonable de que Axk ≈b.

Una estrategia simple para comprender la convergencia del algoritmo de descenso de gradiente para nuestra elección de f es examinar el cambio en el error hacia atrás de una iteración a otra.[1] Supongamos que es la solución que buscamos, es decir, Ax $= b$. Entonces, podemos x estudiar la relación de retroceso error de iteración en iteración:

$$Rk \equiv \frac{f(xk) - f(x \quad)}{f(xk-1) - f(x \quad)}$$

Obviamente, acotar Rk < β < 1 para algunos β muestra que el gradiente descendente converge.

Por conveniencia, podemos expandir f(xk):

$$\begin{aligned} f(xk) &= f(xk-1 + \alpha k d\ k) \text{ por nuestro esquema iterativo} \\ &= \frac{1}{2}(xk-1 + \alpha k d\ k)\, A(xk-1 + \alpha k d\ k) - b\ (xk-1 + \alpha k d\ k) + c \\ &= f(xk-1) + \alpha k d\ k\, Axk-1 + \alpha\ k\ 2\ 1\ (b\ {}^{dk}\_\ Ad\ k - \alpha k\, bd\ k \text{ por definición de f} \\ &= f(xk-1) + \alpha k d \quad k\ d\ k) + \quad {}^{-}\_{k\ 2}{}^{2}{}_{a}\ d \quad Ad\ k - \alpha k\, bd\ k \text{ desde } d\ k\ k = b - Axk-1 \\ &= f(xk-1) - \alpha k d \quad {}^{1}{}_{\text{re}}{}^{2}\ k + \alpha\ \overline{kk}\ 2 \quad d_k \quad \text{Anuncio} \end{aligned}$$

---

[1] Este argumento se presenta, por ejemplo, en http://www-personal.umich.edu/~mepelman/teaching/IOE511/Handouts/511notas07-7.pdf.

$$= f(xk-1) - re \frac{dd\,kk}{k\ \text{Anuncio}_k}\,1\,2\,dd\,k + k - \frac{dd\,kk}{d_k\ \text{Anuncio}_k}^2 d\ \text{Ad}_k \text{ por definición de } \alpha k\ k$$

$$= f(xk-1) - \frac{(d_k\ dk)^2}{d_k\ \text{Anuncio}} + \frac{1}{2\,re}\frac{(re\,k\ dk)^2}{k\ \text{Anuncio}} = f(xk-1) - 2d\ \frac{(d_k\ dk)^2}{k\ \text{Anuncio}}$$

Así, podemos volver a nuestra fracción:

$$Rk = \frac{f(xk-1) - \frac{(d_k\ dk)^2}{2dk\ \text{Anuncio}_k} - f(x\quad)}{f(xk-1) - f(x\quad)} \text{ por nuestra fórmula para } f(xk)$$

$$= 1 - \frac{(d_k\ dk)^2}{2d\,k\ \text{Anuncio}\,k(f(xk-1) - f(x\quad))}$$

Note que Ax =b, entonces podemos escribir:

$$f(xk-1) - f(x\quad) = \frac{1}{2}x\,k-1Axk-1 - b\,xk-1 + c - \frac{1}{2}(x\quad)\ \text{segundo} - bx + c$$

$$= \frac{1}{2}x\,k-1Axk-1 - b\,xk-1 - 2\,1\ \frac{1}{2}b\,A - 1b\,2$$

$$= \frac{1}{2}(Axk-1 - b)\,A\,2\,1^{-1}(Axk-1 - b) \text{ por simetría de A}$$

$$= \frac{re}{2}\,k\ un^{-1}\,d\,k \text{ por definición de } d\quad k$$

De este modo,

$$Rk = 1 - 2d\,k\ \frac{(d_k\ dk)^2}{\text{Anuncio}\,k(f(xk-1) - f(x\quad))}$$

$$= 1 - re\ \frac{(d_k\ re\,k)^2}{k\ \text{Anuncio}\,k \cdot d\,k\ A - 1dk} \text{ por nuestra última simplificación}$$

$$= 1 - re\ \frac{d_k\ dk}{k\ \text{Anuncio}_k\ d\quad A - 1dk\ k} \cdot \frac{d_k\ dk}{}$$

$$\leq 1 - \min dAd\ \frac{1}{d=1} \quad \min_{d=1}\ \frac{1}{dA-1d} \text{ ya que esto hace que el segundo término sea más pequeño}$$

$$1 - \max_{d=1} dAd = \max_{d=1} dA^{-1}\,d$$

$$\sigma\text{mín} = \frac{1}{\sigma\text{máx}} \text{ donde } \sigma\text{min y } \sigma\text{max son los valores singulares mínimo y máximo de A}$$

$$= 1 - \frac{1}{\text{cond A}}$$

Tomó una cantidad considerable de álgebra, pero probamos un hecho importante:

La convergencia del descenso del gradiente en f depende del condicionamiento de A.

Es decir, cuanto mejor acondicionado esté A, más rápido convergerá el descenso del gradiente. Además, dado que cond A ≥ 1, sabemos que nuestra estrategia de descenso de gradiente anterior converge incondicionalmente a x, , aunque la convergencia puede ser lenta cuando A está mal condicionado.

La figura NÚMERO ilustra el comportamiento del descenso del gradiente para matrices bien y mal acondicionadas. Como puede ver, el descenso de gradiente puede tener dificultades para encontrar el mínimo de nuestra función cuadrática f cuando los valores propios de A tienen una amplia dispersión.

10.2 Gradientes conjugados

Recuerde que resolver Ax = b para A Rn×n tomó la estrategia $O(n^3)$ tiempo. Reexaminar el descenso del gradiente anterior, vemos que cada iteración toma $O(n^2)$ tiempo, ya que debemos calcular matriz-vector productos entre A, xk−1 y d k . Por lo tanto, si el descenso de gradiente toma más de n iteraciones, también podría haber aplicado la eliminación gaussiana, que recuperará la solución exacta en la misma cantidad de tiempo.Desafortunadamente, no podemos demostrar que el descenso de gradiente tiene que tomar un número finito de iteraciones y, de hecho, en casos mal condicionados puede tomar una gran cantidad de iteraciones para encontrar el mínimo.

Por esta razón, diseñaremos un algoritmo que garantice la convergencia en un máximo de n pasos,) preservando el $O(n^3$ sincronización en el peor de los casos para resolver sistemas lineales. En el camino encontraremos que este algoritmo, de hecho, exhibe mejores propiedades de convergencia en general, lo que lo convierte en una opción razonable incluso si no lo ejecutamos hasta el final.

10.2.1 Motivación

Nuestra derivación del algoritmo de gradientes conjugados está motivada por una observación bastante directa. Supongamos que conocemos la solución x de otra f en Ax =b. Entonces, podemos escribir nuestra forma cuadrática manera:

$$
\begin{aligned}
f(x) &= \tfrac{1}{2}(x - x^*)^T A(x - x^*) + x^T A x^* - b^T x + c \quad \text{por definición} \\
&= \tfrac{1}{2}(x - x^*)^T A(x - x^*) + x^T A x^* - \tfrac{1}{2}(x^*)^T A x^* - b^T x + c \quad \text{sumando y restando los mismos términos} \\
&= \tfrac{1}{2}(x - x^*)^T A(x - x^*) + x^T b - (x^*)^T b - b^T x + c \quad \text{ya que } Ax^* = b \\
&= \tfrac{1}{2}(x - x^*)^T A(x - x^*) + \text{const.} \quad \text{ya que los términos } xb \text{ se cancelan}
\end{aligned}
$$

hacemos. Por lo tanto, hasta un cambio constante f es lo mismo $\frac{1}{2}(x - x^*)^T A(x - x^*)$.Por supuesto, lo que el producto, x pero esta observación nos muestra la naturaleza de f ; simplemente está midiendo la distancia ≡ de x a x con respecto a la “norma A” v De hecho, $\|v\|_A^2 = v^T Av$.

dado que A es simétrico y definido positivo, aunque puede ser lento de llevar a cabo en la práctica, sabemos que se puede factorizar usando la estrategia de Cholesky como A = LL. Con esta factorización en la mano, f toma una forma aún mejor:

$$f(x) = \frac{1}{2}\|L^T(x - x^*)\|_2^2 + \text{const.}$$

Dado que L es una matriz invertible, esta norma es verdaderamente una medida de distancia entre x y x .

Definir y ≡ L x y y ≡ L x Entonces, desde este nuevo punto de vista, estamos minimizando 2 . Por f(y) = . 2 y −y supuesto, si realmente pudiéramos llegar a este punto a través de la factorización de Cholesky, optimizando f sería extremadamente fácil, pero para derivar un esquema para esta minimización sin L, consideramos la posibilidad de minimizar f usando solo búsquedas lineales derivadas en §10.1.1.

Hacemos una simple observación sobre cómo minimizar nuestra función simplificada f usando tal estrategia ej., ilustrada en la Figura NÚMERO:

Proposición 10.1. Suponga que {w 1, . . . , w n} son ortogonales en Rn . Entonces, f se minimiza en un máximo de n pasos mediante la búsqueda de líneas en la dirección w 1, luego en la dirección w 2, y así sucesivamente.

Prueba. Tome las columnas de Q Rn×n como los vectores w i ; Q es una matriz ortogonal. Como Q f(y) = y −y es ortogonal, podemos palabras, rotamos para que w 1 sea el primer vector base estándar, w 2 sea el segundo, y así sucesivamente. Si escribimos z ≡ Qy y después de la segunda iteración

z ≡ Qy y así , entonces claramente después de la primera iteración debemos tener z1 = z 1 , z2 = z2 , sucesivamente. Después de n pasos llegamos a zn = z n , dando el resultado deseado. □

Por lo tanto, la optimización de f siempre se puede lograr utilizando búsquedas de n líneas, siempre que esas búsquedas se realicen en direcciones ortogonales.

Todo lo que hicimos para pasar de f a f fue rotar las coordenadas usando L . Tal transformación lineal líneas rectas a líneas rectas, por lo que hacer una búsqueda de línea en hacer una f a lo largo de algún vector w es equivalente lleva búsqueda de línea a lo largo de (L) −1w en nuestra función cuadrática original f . Por el contrario, si hacemos búsquedas de n líneas en f en direcciones vi tales que L vi ≡ w i son ortogonales, entonces por la Proposición 10.1 debemos haber encontrado x . Note que preguntar w i · w = 0 es lo mismo que preguntar j

$$0 = w_i \cdot w_j = (L v_i) \cdot (L v_j) = v_i (L^T L) v_j = v_i A v_j .$$

Acabamos de argumentar un importante corolario de la Proposición 10.1. Defina vectores conjugados como sigue:

Definición 10.1 (vectores A-conjugados). Dos vectores v, w son A-conjugados si v Aw = 0.

Luego, en base a nuestra discusión, hemos mostrado:

Proposición 10.2. Supongamos {v1, . . . ,vn} son conjugados A. Entonces, f se minimiza en un máximo de n pasos mediante la búsqueda de líneas en la dirección v1, luego en la dirección v2, y así sucesivamente.

En un nivel alto, el algoritmo de gradientes conjugados simplemente aplica esta proposición, generando y buscando a lo largo de direcciones A conjugadas en lugar de moverse a lo largo de − f . Tenga en cuenta que este resultado puede parecer algo contrario a la intuición: no necesariamente nos movemos a lo largo de la dirección de descenso más empinada, sino que pedimos que nuestro conjunto de direcciones de búsqueda satisfaga un criterio global para asegurarnos de que no repetimos el trabajo. Esta configuración garantiza la convergencia en un número finito de iteraciones y reconoce la estructura de f en términos de f discutida anteriormente.

Recuerde que motivamos las direcciones A-conjugadas notando que son ortogonales después de aplicar L de la factorización A = LL. Desde este punto de vista, estamos tratando con dos productos punto: xi · xj y yi ·yj ≡ (L xi) · (L xj) = x i LLxj = x i Axj . Estos dos productos figurarán en nuestra discusión posterior en cantidades iguales, por lo que denotaremosel "producto interno A" como

$$\langle u, v \rangle_A \equiv (L u) \cdot (L v) = u^T A v.$$

10.2.2 Subóptimo del Descenso de Gradiente

Hasta ahora, sabemos que si podemos encontrar n direcciones de búsqueda conjugadas A, podemos resolver Ax =b en n pasos a través de búsquedas lineales a lo largo de estas direcciones. Lo que queda es descubrir una estrategia para encontrar estas direcciones de la manera más eficiente posible. Para hacerlo, examinaremos una propiedad más del algoritmo de descenso de gradiente que inspirará un enfoque más refinado.

Supongamos que estamos en xk durante un método de búsqueda de línea iterativa en f(x); llamaremos residual rk ≡ b − Axk a la dirección de mayor descenso de f en xk . Es posible que no decidamos hacer una búsqueda de línea a lo largo de rk como en el descenso de gradiente, ya que las direcciones de gradiente no son necesariamente A-conjugadas. Entonces, generalizando un poco, encontraremos xk+1 a través de una búsqueda de línea a lo largo de una dirección vk+1 aún no determinada .

A partir de nuestra derivación del descenso de gradiente, debemos elegir xk+1 = xk + αk+1vk+1 , donde αk+1 viene dado por

$$ak+1 = \frac{v_{k+1}}{rkv\ k\ +1Avk+1}.$$

Aplicando esta expansión de xk+1 , podemos escribir una fórmula de actualización alternativa para el residuo:

$$\begin{aligned} rk+1 &= b - Axk+1 = b - \\ &\quad A(xk + \alpha k+1vk+1) \text{ por definición de } xk+1 = (b - Axk) - \\ &\quad \alpha k+1Avk+1 = rk - \alpha k+1Avk+1 \text{ por} \\ &\quad \text{definición ofrk} \end{aligned}$$

Esta fórmula se mantiene independientemente de nuestra elección de vk+1 y se puede aplicar a cualquier método de búsqueda de línea iterativa.

Sin embargo, en el caso del descenso de gradiente, elegimos vk+1 ≡ rk . Esta elección da una relación de recurrencia:

$$rk+1 = rk - \alpha k+1Arca .$$

Esta simple fórmula conduce a una proposición instructiva:

Proposición 10.3. Al realizar un descenso de gradiente en f , span{r0, . . . ,rk} = intervalo{r0, Ar0, . . . , Un kr0}.

Prueba. Esta declaración se sigue inductivamente de nuestra fórmula forrk+1 anterior. □

La estructura que estamos descubriendo comienza a parecerse mucho a los métodos subespaciales de Krylov mencionados en el Capítulo 5: ¡Esto no es un error!

El descenso de gradiente llega a xk moviéndose a lo largo de r0, luego r1, y así sucesivamente hasta rk . Así, al final sabemos que la iteración xk del descenso del gradiente en f se encuentra en algún lugar del plano x0 + A k−1r0}, por la Proposición 10.3. amplitud {r0, Ar0, . . . , no es cierto que si ejecutamos gradiente Desafortunadamente, es un intervalo {r0,r1, . . . ,rk−1} = x0 + descendente, la iteración xk es óptima en este subespacio. En otras palabras, en general puede darse el caso de que

$$-x0 = \min_{v \in \text{span}\{r0,Ar0,...,Ak-1r0\}} xk \quad f(x0 + v) \arg$$

Idealmente, cambiar esta desigualdad a una igualdad garantizaría que generar xk+1 a partir de xk no “cancele” ningún trabajo realizado durante las iteraciones 1 a k − 1.

Si reexaminamos nuestra prueba de la Proposición 10.1 teniendo presente este hecho, podemos hacer una observación que sugiera cómo podríamos usar la conjugación para mejorar el gradiente descendente. En particular, una vez que zi cambia a z, nunca cambia de valor en una iteración futura. En otras palabras, después de rotar de z a i , x se cumple la siguiente proposición:

Proposición 10.4. Tome xk como la k-ésima iteración del proceso de la Proposición 10.1 después de buscar a lo largo de vk . Entonces,

$$x_k - x_0 = \underset{v \;\; \text{span}\{v_1,\ldots,v_k\}}{\text{argumento mínimo}} \quad f(x_0 + v)$$

Por lo tanto, en el mejor de los mundos posibles, en un intento de superar el descenso de gradiente, podríamos esperar que A encuentre direcciones A conjugadas {v1, . . . ,vn} tales que abarcan {v1, . . . ,vk} = intervalo {r0, Ar0, . . . , para cada k; k−1r0} entonces se garantiza que nuestro esquema iterativo no funcionará peor que el descenso de gradiente durante cualquier iteración dada. Pero, con avidez, deseamos hacerlo sin ortogonalización o sin almacenar más de un número finito de vectores a la vez.

10.2.3 Generación de direcciones conjugadas A

Por supuesto, dado cualquier conjunto de direcciones, podemos hacerlas A-ortogonales utilizando un método como la ortogonalización de Gram Schmidt. Desafortunadamente, ortogonalizar {r0, Ar0, . . .} encontrar el conjunto de direcciones de búsqueda es costoso y requeriría mantener una lista completa de direcciones vk ; esta construcción probablemente excedería los requisitos de tiempo y memoria incluso de la eliminación gaussiana.
Revelaremos una observación final sobre Gram-Schmidt que hace que los gradientes conjugados sean manejables al generar direcciones conjugadas sin un costoso proceso de ortogonalización.

Ignorando estos problemas, podríamos escribir un "método de direcciones conjugadas" de la siguiente manera:

$$\text{Actualizar dirección de búsqueda (paso de Gram-Schmidt incorrecto): } v_k = A^{k-1} r_0 - \sum_{yo<k} \frac{A^{k-1} r_0, v_i A}{v_i, v_i a} v_i$$

$$\text{Búsqueda de línea: } \alpha_k = \frac{v_k \, r_{k-1}}{v_k \; Av_k}$$

$$\text{Actualizar estimación: } x_k = x_{k-1} + \alpha_k v_k$$

$$\text{Actualizar residual: } r_k = r_{k-1} - \alpha_k A v_k$$

Aquí, calculamos la k-ésima dirección de búsqueda vk simplemente proyectando v1, . . . ,vk−1 del vector A k−1r0. Este algoritmo obviamente tiene la propiedad span {v1, . . . ,vk} = intervalo {r0, Ar0, . . . , A k−1r0} sugerido en §10.2.2, pero tiene dos problemas:

1. De manera similar a la iteración de potencia para vectores propios, es probable que la potencia A k−1r0 se parezca principalmente al primer vector propio de A, lo que hace que la proyección esté cada vez menos condicionada
2. Tenemos que mantener v1, . . . ,vk−1 para calcular vk ; por lo tanto, cada iteración de este algoritmo necesita más memoria y tiempo que la anterior.

Podemos solucionar el primer problema de una manera relativamente sencilla. En particular, ahora mismo proyectamos las direcciones de búsqueda anteriores desde A k−1r0, pero en realidad podemos proyectar direcciones anteriores desde cualquier vector w siempre que

$$w \quad \text{tramo}\{r_0, Ar_0, \ldots, A^{k-1} r_0\} \backslash \text{span}\{r_0, Ar_0, \ldots, A^{k-2} r_0\},$$

es decir, siempre que w tenga alguna componente en la parte nueva del espacio.

Una elección alternativa de w con esta propiedad es el residualrk−1 . Esta propiedad se sigue de la actualización residual rk = rk−1 − αkAvk ; en esta expresión multiplicamos vk por A, introduciendo la nueva potencia de A que necesitamos. Esta elección también imita más de cerca el algoritmo de descenso de gradiente, que tomó vk =rk−1 . Por lo tanto, podemos actualizar un poco nuestro algoritmo:

$$\text{Actualizar la dirección de búsqueda (mal Gram-Schmidt en residual): } vk = rk-1 - \sum_{yo<k} \frac{rk-1\ ,viA\ vi}{vi\ ,via}$$

$$\text{Búsqueda de línea: } \alpha k = \frac{v k\ rk-1}{v_k\ Avk}$$

$$\text{Actualizar estimación: } xk = xk-1 + \alpha kvk$$

$$\text{Actualizar residual: } rk = rk-1 - \alpha kAvk$$

Ahora no hacemos aritmética que involucre el vector mal condicionado A k−1r0 pero todavía tenemos el problema de "memoria" anterior.

De hecho, la observación sorprendente sobre el paso anterior de ortogonalización es que la mayoría de los términos de la suma son exactamente cero. Esta sorprendente observación permite que ocurra cada iteración de gradientes conjugados sin aumentar el uso de la memoria. Conmemoramos este resultado en una proposición:

Proposición 10.5. En el método de “dirección conjugada” anterior, rk ,vA = 0 para todo < k.

Prueba. Procedemos inductivamente. No hay nada que probar para el caso base k = 1, así que suponga que k > 1 y que el resultado se cumple para todo k < k. Por la fórmula de actualización residual, sabemos:

$$rk\ ,vA = rk-1\ ,vA - \alpha kAvk\ ,vA = rk-1\ ,vA - \alpha kvk\ ,\ AvA,$$

donde la segunda igualdad se sigue de la simetría de A.

Primero, suponga < k − 1. Entonces el primer término de la diferencia anterior es cero por inducción. Además, por construcción Av span {v1, . . . ,v+1}, por lo que como hemos construido nuestras direcciones de búsqueda para que sean A-conjugadas, sabemos que el segundo término también debe ser cero.

Para concluir la prueba, consideramos el caso = k − 1. Usando la fórmula de actualización residual, sabemos:

$$Avk-1 = \frac{1}{\alpha k-1}\ (rk-2\ -rk-1)$$

Byrk premultiplicando muestra:

$$rk\ ,vk-1A = \frac{1}{\alpha k-1}\ {}_{rk}\ _{-}\ (rk-2\ -rk-1)$$

La diferencia rk−2 −rk−1 vive en el intervalo {r0, Ar0, . . . , la A k−1r0}, por la fórmula de actualización residual. Proposición 10.4 muestra que xk es óptima en este subespacio. Como rk = − f(xk), esto implica A k−1r0}, ya que de {r0, Ar0, . . . , en el subespacio para pasar de xk a lo contrario existiría una dirección que debemos tenerrk span disminuir f . En particular, esto muestra el producto interno por encima de rk ,vk−1A = 0, como se desea. □

Por lo tanto, nuestra prueba anterior muestra que podemos encontrar una nueva dirección vk de la siguiente manera:

$$vk = rk-1 - \sum_{yo<k} \frac{rk-1\ ,viaA}{vi\ ,via} vi \text{ por la fórmula de Gram-Schmidt}$$

$$= rk-1 - \frac{rk-1\ ,vk-1A}{vk-1\ ,vk-1A} vk-1 \text{ porque los términos restantes se anulan}$$

Dado que la suma sobre i desaparece, el costo de calcular vk no depende de k.

10.2.4 Formulación del algoritmo de gradientes conjugados

Ahora que tenemos una estrategia que produce direcciones de búsqueda conjugadas A con un esfuerzo computacional relativamente pequeño, simplemente aplicamos esta estrategia para formular el algoritmo de gradientes conjugados.

En particular, suponga que x0 es una suposición inicial de la solución de Ax = b, y tome r0 ≡ b − Ax0.

Por conveniencia, tomemos v0 ≡ 0. Luego, iterativamente actualizamos xk−1 a xk usando una serie de pasos para k = 1, 2, . . .:

$$\text{Actualizar dirección de búsqueda: } vk = rk-1 - \frac{rk-1\ ,vk-1A}{vk-1\ ,vk-1A} vk-1$$

$$\text{Búsqueda de línea: } \alpha k = \frac{v_k\ rk-1}{v_k\ Avk}$$

$$\text{Actualizar estimación: } xk = xk-1 + \alpha k vk$$

$$\text{Actualizar residual: } rk = rk-1 - \alpha k Avk$$

Este esquema iterativo es solo un ajuste menor al algoritmo de descenso de gradiente pero tiene muchas propiedades deseables por construcción:

- f(xk) tiene un límite superior por el de la iteración k-ésima del descenso del gradiente
- El algoritmo converge a x en n pasos
- En cada paso, la iteración xk es óptima en el subespacio abarcado por las primeras k direcciones de búsqueda

Con el fin de exprimir al máximo la calidad numérica de los gradientes conjugados, podemos intentar simplificar los valores numéricos de las expresiones anteriores. Por ejemplo, si conectamos la actualización de la dirección de búsqueda en la fórmula para αk , por ortogonalidad podemos escribir:

$$ak = \frac{r_{k-1}\ rk-1}{v_k\ Avk}$$

Ahora se garantiza que el numerador de esta fracción no es negativo sin precisión numérica asuntos.

De manera similar, podemos definir una constante βk para dividir la actualización de la dirección de búsqueda en dos pasos:

$$\beta k \equiv - \frac{rk-1\ ,vk-1A}{vk-1\ ,vk-1A}$$

$$vk = rk-1 + \beta k vk-1$$

Podemos simplificar nuestra fórmula para βk :

$$\beta_k \equiv -\frac{r_{k-1}, v_{k-1}A}{v_{k-1}, v_{k-1}A}$$

$$= -\frac{r_{k-1}Av_{k-1}}{v_{k-1}Av_{k-1}} \text{ por definición de } \cdot, \cdot A$$

$$= -\frac{r_{k-1}(r_{k-2} - r_{k-1})}{\alpha_{k-1}v_{k-1}Av_{k-1}} \text{ since } r_k = r_{k-1} - \alpha_k Av_k$$

$$= \frac{r_{k-1}r_{k-1}}{\alpha_{k-1}v_{k-1}Av_{k-1}} \text{ por un cálculo por debajo de } \alpha_{k-1}v$$

$$= \frac{r_{k-1}r_{k-1}}{r_{k-2}r_{k-2}} \text{ por nuestra última fórmula para } \alpha_k$$

Esta expresión revela que βk ≥ 0, una propiedad que podría no haberse cumplido después de problemas de precisión numérica. Tenemos un cálculo restante a continuación:

$$r_{k-2}r_{k-1} = r_{k-2}(r_{k-2} - \alpha_{k-1}Av_{k-1}) \text{ por nuestra fórmula de actualización residual}$$

$$= r_{k-2}r_{k-2} - \frac{r_{k-2}r_{k-2}}{v_{k-1}Av_{k-1}} r_{k-2}Av_{k-1} \text{ por nuestra fórmula para } \alpha_k$$

$$= r_{k-2}r_{k-2} - v_{k-1}Av_{k-1} \frac{r_{k-2}r_{k-2}}{v_{k-1}Av_{k-1}} \text{ por la actualización de } v_k \text{ y A-conjugacy de } v_k\text{'s}$$

$$= 0, \text{ según sea necesario.}$$

Con estas simplificaciones, tenemos una versión alternativa de gradientes conjugados:

Actualizar dirección de búsqueda: $\beta_k = \frac{r_{k-1}r_{k-1}}{r_{k-2}r_{k-2}}$

$$v_k = r_{k-1} + \beta_k v_{k-1}$$

Búsqueda de línea: $\alpha_k = \frac{r_{k-1}r_{k-1}}{v_k Av_k}$

Actualizar estimación: $x_k = x_{k-1} + \alpha_k v_k$

Actualizar residual: $r_k = r_{k-1} - \alpha_k Av_k$

Por razones numéricas, en ocasiones, en lugar de usar la fórmula de actualización fork, es recomendable usar la fórmula residual =b − Axk . Esta fórmula requiere una multiplicación matriz-vector adicional, pero repara la "desviación" numérica causada por el redondeo de precisión finita. Tenga en cuenta que no es necesario almacenar una larga lista de residuos anteriores o direcciones de búsqueda: los gradientes conjugados ocupan una cantidad constante de espacio de una iteración a otra.

10.2.5 Condiciones de convergencia y parada

Por construcción, se garantiza que el algoritmo de gradientes conjugados (CG) no convergerá más lentamente que el descenso de gradiente en f , mientras que no es más difícil de implementar y tiene una serie de otros

propiedades positivas. Una discusión detallada de la convergencia de CG está fuera del alcance de nuestra discusión, pero en general el algoritmo se comporta mejor en matrices con valores propios distribuidos uniformemente en un rango pequeño. Una estimación aproximada paralela a nuestra estimación en §10.1.2 muestra que el algoritmo CG satisface:

$$\frac{f(x_k) - f(x^*)}{f(x_0) - f(x^*)} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k$$

donde $\kappa \equiv \text{cond } A$. De manera más general, el número de iteraciones necesarias para que el gradiente conjugado alcance un valor de error dado generalmente puede estar limitado por una función de $\sqrt{\kappa}$, mientras que los límites para la convergencia del descenso del gradiente son proporcionales a κ.

Sabemos que se garantiza que los gradientes conjugados convergen a x^* exactamente en n pasos, pero cuando n es grande, puede ser preferible detenerse antes. De hecho, la fórmula para β_k se dividirá por cero cuando el residuo sea muy corto, lo que puede causar problemas de precisión numérica cerca del mínimo de f. Por lo tanto, en la práctica, CG generalmente se detiene cuando la relación r_k/r_0 es lo suficientemente pequeña.

10.3 Preacondicionamiento

Ahora tenemos dos poderosos esquemas iterativos para encontrar soluciones para $Ax = b$ cuando A es simétrica y definida positiva: pendiente de gradiente y gradientes conjugados. Ambas estrategias convergen incondicionalmente, lo que significa que independientemente de la suposición inicial x_0 con suficientes iteraciones, nos acercaremos arbitrariamente a la verdadera solución x^* exactamente en un número finito de iteraciones. Por supuesto, el tiempo necesario para llegar a una solución de $Ax = b$ para ambos métodos es directamente proporcional al número de iteraciones necesarias para llegar a x^* dentro de una tolerancia aceptable. Por lo tanto, tiene sentido ajustar una estrategia tanto como sea posible para minimizar el número de iteraciones para la convergencia.

Con este fin, notamos que podemos caracterizar las tasas de convergencia de ambos algoritmos y muchas más técnicas iterativas relacionadas en términos del número de condición $\text{cond } A$. Es decir, cuanto menor sea el valor de $\text{cond } A$, menos tiempo debería tomar para resolver $Ax = b$. Observe que esta situación es algo diferente para la eliminación gaussiana, que toma la misma cantidad de pasos independientemente de A; en otras palabras, el condicionamiento de A afecta no solo la calidad de la salida de los métodos iterativos, sino también la velocidad a la que x se acerca.

Por supuesto, para cualquier matriz invertible P, se da el caso de que resolver $PAx = Pb$ es equivalente a resolver $Ax = b$. El truco, sin embargo, es que el número de condición de PA no necesita ser el mismo que el de A; en el extremo (inalcanzable), por supuesto, si tomamos $P = A^{-1}$, eliminaríamos los problemas de condicionamiento por completo. De manera más general, supongamos que $P \approx A^{-1}$ cond A, por lo que puede ser recomendable . Entonces, esperamos cond PA aplicar P antes de resolver el sistema lineal. En este caso, llamaremos a P un preacondicionador.

Si bien la idea del preacondicionamiento parece atractiva, quedan dos problemas:

1. Si bien A puede ser simétrico y definido positivo, el producto PA en general no disfrutará estas propiedades.
2. Necesitamos encontrar $P \approx A^{-1}$ que es más fácil de calcular que A^{-1} en sí mismo.

Abordamos estos problemas en las siguientes secciones.

10.3.1 CG con preacondicionamiento

Centraremos nuestra discusión en gradientes conjugados ya que tiene mejores propiedades de convergencia, aunque la mayoría de nuestras construcciones también se aplicarán con bastante facilidad al descenso de gradiente. Desde este punto de vista, si revisamos nuestras construcciones en §10.2.1, está claro que nuestra construcción de CG depende en gran medida tanto de la simetría como de la definición positiva de A, por lo que ejecutar CG en PA generalmente no convergerá fuera de la caja.

Supongamos, sin embargo, que el precondicionador P es en sí mismo simétrico y definido positivo. Esta es una suposición razonable ya que A debe satisfacer estas propiedades. Entonces, nuevamente podemos escribir $P^{-1} = EE$. Factorización de Cholesky de la inversa P^{-1} Hacemos la siguiente observación:

Proposición 10.6. El número de condición de PA es el mismo que el de $E^{-1}AE^{-}$.

Prueba. Mostramos que PA y $E^{-1}AE^{-}$ tienen los mismos valores singulares; el número de condición es la relación entre el valor singular máximo y el mínimo, por lo que esta declaración es más que suficiente. En particular, está claro que $E^{-1}AE^{-}$ es simétrico y definido positivo, por lo que sus vectores propios son sus valores singulares. Por lo tanto, suponga que $E^{-1}AE^{-}x = \lambda x$. Conocemos $P^{-1} = EE$, entonces $P = E^{-}E^{-1}$. Por lo tanto, si premultiplicamos ambos lados de nuestra expresión de vector propio por E^{-} encontramos $PAE^{-}x = \lambda E^{-}x$.
Definir $y \equiv E^{-}x$ muestra $PAy = \lambda y$. Por lo tanto, PA y $E^{-1}AE^{-}$ tienen espacios propios completos y valores propios idénticos. □

Esta proposición implica que si hacemos CG en la matriz definida positiva simétrica $E^{-1}AE^{-}$, recibiremos los mismos beneficios condicionantes que tendríamos si pudiéramos iterar en PA. Como en nuestra prueba de la Proposición 10.6, podríamos lograr nuestra nueva solución para $y = E\,x$ en dos pasos:

1. Resuelve $E^{-1}AE^{-}y = E^{-1}b$.
2. Resuelve $x = E^{-}y$.

Encontrar E sería parte integral de esta estrategia, pero probablemente sea difícil, pero demostraremos en breve que no es necesario.

Ignorando el cálculo de E, podríamos realizar el paso 1 usando CG de la siguiente manera:

$$\text{Actualizar dirección de búsqueda: } \beta_k = \frac{r_{k-1}\, r_{k-1}}{r_{k-2}\, r_{k-2}}$$

$$v_k = r_{k-1} + \beta_k v_{k-1}$$

$$\text{Búsqueda de línea: } \alpha_k = \frac{r_{k-1}\, r_{k-1}}{v_k\, E^{-1}AE^{-}v_k}$$

$$\text{Actualizar estimación: } y_k = y_{k-1} + \alpha_k v_k$$

$$\text{Actualizar residual: } r_k = r_{k-1} - \alpha_k E^{-1}AE^{-}v_k$$

Este esquema iterativo convergerá según el condicionamiento de nuestra matriz $E^{-1}AE^{-}$.

Defina $\tilde{r}_k \equiv Er_k$, $\tilde{v}_k \equiv E^{-}v_k$ y $x_k \equiv Ey_k$. Si recordamos la relación $P^{-1} = E^{-}E$, reescribamos, podemos nuestra iteración de gradientes conjugados precondicionados usando estas nuevas variables:

$$\text{Actualizar dirección de búsqueda: } \beta_k = \frac{\tilde{r}_{k-1} P \tilde{r}_{k-1}}{\tilde{r}_{k-2} P \tilde{r}_{k-2}}$$

$$\tilde{v}_k = P\tilde{r}_{k-1} + \beta_k \tilde{v}_{k-1}$$

$$\text{Búsqueda de línea: } \alpha_k = \frac{\tilde{r}_{k-1}^{T} P r_{k-1}}{\tilde{v}_k^{T} A \tilde{v}_k}$$

$$\text{Actualizar estimación: } x_k = x_{k-1} + \alpha_k \tilde{v}_k$$

$$\text{Actualización residual: } \tilde{r}_k = \tilde{r}_{k-1} - \alpha_k A \tilde{v}_k$$

Esta iteración no depende de la factorización de Cholesky de P realizada usando únicamente P^{-1} en absoluto, pero en cambio puede aplicaciones de P y A. Es fácil ver que el esquema $x_k \to x$ disfruta de los beneficios del precondicionamiento , ser así de hecho esto sin necesidad de factorizar el precondicionador.

Como nota al margen, se puede llevar a cabo un preacondicionamiento aún más efectivo reemplazando A con PAQ por una segunda matriz Q, aunque esta segunda matriz requerirá cálculos adicionales para aplicar. Este ejemplo representa una compensación común: si un acondicionador previo toma demasiado tiempo para aplicarse en una sola iteración de CG u otro método, es posible que no valga la pena el número reducido de iteraciones.

10.3.2 Precondicionadores comunes

Encontrar buenos preacondicionadores en la práctica es tanto un arte como una ciencia. Encontrar el mejor depende de la aproximación P de A y así P^{-1} estructura de A, la aplicación particular en cuestión y sucesivamente. Sin embargo, incluso las aproximaciones aproximadas pueden ayudar considerablemente a la convergencia, por lo que rara vez aparecen aplicaciones de CG que no utilizan un preacondicionador.

La mejor estrategia para formular P a menudo es específica de la aplicación, y un problema de aproximación de ingeniería interesante implica diseñar y probar varias P para obtener el mejor preacondicionador.
Dos estrategias comunes son las siguientes:

- Un precondicionador diagonal (o “Jacobi”) simplemente toma P como la matriz obtenida al invertir los elementos diagonales de A; es decir, P es la matriz diagonal con entradas 1/aii. Esta estrategia puede aliviar el escalado no uniforme de fila a fila, que es una causa común de mal acondicionamiento.

- El precondicionador inverso aproximado escaso se formula resolviendo un subproblema $\min_{P \in S} \|AP - I\|_{Fro}$, donde P se restringe a estar en un conjunto S de matrices sobre las que es menos difícil optimizar dicho objetivo. Por ejemplo, una restricción común es prescribir un patrón de dispersión para P, por ejemplo, solo ceros en la diagonal o donde A tiene ceros.

- Los factores condicionantes de Cholesky incompletos $A \approx L L^{T}$ y luego aproxima A^{-1} resolviendo los problemas apropiados de sustitución hacia adelante y hacia atrás. Por ejemplo, una estrategia popular implica pasar por los pasos de la factorización de Cholesky pero solo guardar la salida en las posiciones (i, j) donde $a_{ij} = 0$.

- Los valores distintos de cero en A pueden considerarse un gráfico, y la eliminación de bordes en el gráfico o la agrupación de nodos puede desconectar varios componentes; el sistema resultante es una diagonal de bloque después de permutar filas y columnas y, por lo tanto, puede resolverse utilizando una secuencia de soluciones más pequeñas. Tal estrategia de descomposición de dominios puede ser efectiva para sistemas lineales que surgen de ecuaciones diferenciales como las que se consideran en el Capítulo NÚMERO.

Algunos precondicionadores vienen con límites que describen cambios en el condicionamiento de A después de reemplazarlo con PA, pero en su mayor parte estas son estrategias heurísticas que deben probarse y refinarse.

10.4 Otros esquemas iterativos

Los algoritmos que hemos desarrollado en detalle en este capítulo se aplican para resolver Ax = b cuando A es cuadrada, simétrica y definida positiva. Nos hemos centrado en este caso porque aparece muy a menudo en la práctica, pero hay casos en los que A es asimétrica, indefinida o incluso rectangular. Está fuera del alcance de nuestra discusión derivar algoritmos iterativos en cada caso, ya que muchos requieren un análisis especializado o desarrollo avanzado, pero aquí resumimos algunas técnicas de alto nivel (CITAR CADA UNO):

- Los métodos de división descomponen A = M − N y observa que Ax = b es equivalente a Mx = Nx +b. Si M es fácil de invertir, entonces se puede derivar un esquema de punto fijo escribiendo Mxk = Nxk−1 +b (CITE); estas técnicas son fáciles de implementar pero tienen convergencia dependiendo del espectro de la matriz G = M−1N y en particular pueden divergir cuando el radio espectral de G es mayor que uno. Una opción popular de M es la diagonal de A. Los métodos como la relajación excesiva sucesiva (SOR) ponderan estos dos términos para lograr una mejor convergencia.

- El método residual de la ecuación normal del gradiente conjugado (CGNR) simplemente aplica el algoritmo CG a las ecuaciones normales A Ax = A b. Este método es simple de implementar y garantiza la convergencia siempre que A sea de rango completo, pero la convergencia puede ser lenta debido al mal condicionamiento de AA como se analiza en el Capítulo NÚMERO.

- El método del error de la ecuación normal del gradiente conjugado (CGNE) resuelve de manera similar AAy =b; entonces la solución de Ax =b es simplemente A y.

- Métodos como MINRES y SYMMLQ se aplican a matrices definidas A simétricas pero no necesariamente positivas reemplazando nuestra forma cuadrática f(x) con g(x) ≡ b − Ax2; esta función g se minimiza en las soluciones de Ax =b independientemente de la definición de A.

- Dado el mal acondicionamiento de CGNR y CGNE, los algoritmos LSQR y LSMR también minimizan g(x) con menos suposiciones sobre A, en particular permitiendo la solución de sistemas de mínimos cuadrados.

- Los métodos generalizados que incluyen GMRES, QMR, BiCG, CGS y BiCGStab resuelven Ax =b con la única salvedad de que A es cuadrada e invertible. Optimizan energías similares, pero a menudo tienen que almacenar más información sobre iteraciones anteriores y pueden tener que factorizar matrices intermedias para garantizar la convergencia con tal generalidad.

- Finalmente, los métodos de Fletcher-Reeves, Polak-Ribi`ere y otros regresan al problema más general de minimizar una función no cuadrática f , aplicando pasos de gradiente conjugado para encontrar nuevas direcciones de búsqueda de líneas. Las funciones f que están bien aproximadas por cuadráticas se pueden minimizar de manera muy efectiva usando estas estrategias, aunque no necesariamente hacen uso de Hessian; por ejemplo, el método de Fletcher-Reeves simplemente reemplaza el residuo en las iteraciones de CG con el gradiente negativo − f . Es posible caracterizar la convergencia de estos métodos cuando se acompañan de estrategias de búsqueda de líneas suficientemente efectivas.

Muchos de estos algoritmos son casi tan fáciles de implementar como CG o descenso de gradiente, y existen muchas implementaciones que simplemente requieren ingresar A yb. Muchos de los algoritmos enumerados anteriormente requieren la aplicación de A y A, lo que puede ser un desafío técnico en algunos casos. Como regla general, cuanto más generalizado es un método, es decir, cuantas menos suposiciones hace un método sobre la estructura de la matriz A, más iteraciones es probable que necesite para compensar esta falta de suposiciones. Dicho esto, no existen reglas duras y rápidas simplemente observando el esquema iterativo más exitoso, aunque existe una discusión teórica limitada que compara las ventajas y desventajas de cada uno de estos métodos (CITE).

10.5 Problemas

- Derivar CGNR y/o CGNE
- Derivar MINRES
- Derivar Fletcher-Reeves
- Diapositiva 13 de http://math.ntnu.edu.tw/~min/matrix_computation/Ch4_Slide4_CG_2011.pdf

Parte IV

Funciones, Derivadas e Integrales

Capítulo 11

Interpolación

Hasta ahora hemos derivado métodos para analizar funciones f, por ejemplo, encontrar sus mínimos y raíces. Evaluar f(x) en un x Rn particular puede ser costoso, pero una suposición fundamental de los métodos que desarrollamos en capítulos anteriores es que podemos obtener f(x) cuando lo deseamos, independientemente de x.

Hay muchos contextos en los que esta suposición no es realista. Por ejemplo, si tomamos una fotografía con una cámara digital, recibimos una cuadrícula de n × m de valores de color de píxeles que muestra el continuo de luz que ingresa a la lente de la cámara. Podríamos pensar en una fotografía como una función continua desde la posición de la imagen (x, y) hasta el color (r, g, b), pero en realidad solo conocemos el valor de la imagen en ubicaciones separadas por nm en el plano de la imagen. De manera similar, en el aprendizaje automático y las estadísticas, a menudo solo recibimos muestras de una función en los puntos donde recopilamos datos, y debemos interpolarlos para tener valores en otros lugares; en un entorno médico, podemos monitorear la respuesta de un paciente a diferentes dosis de un fármaco, pero solo podemos predecir lo que sucederá con una dosis que no hayamos probado explícitamente.

En estos casos, antes de que podamos minimizar una función, encontrar sus raíces o incluso calcular valores f(x) en ubicaciones arbitrarias x, necesitamos un modelo para interpolar f(x) a todo Rn (o algún subconjunto del mismo) dada una colección de muestras f(xi). Por supuesto, las técnicas que resuelven este problema de interpolación son inherentemente aproximadas, ya que no conocemos los verdaderos valores de f, por lo que buscamos que la función interpolada sea uniforme y sirva como una predicción "razonable" de los valores de la función.

En este capítulo supondremos que los valores f(xi) se conocen con total certeza; en este caso, también podríamos pensar en el problema como una extensión de f al resto del dominio sin perturbar el valor en ninguna de las ubicaciones de entrada. En el Capítulo NÚMERO (ESCRÍBEME EN 2014), consideraremos el problema de regresión, en el que el valor de f(xi) se conoce con cierta incertidumbre, en cuyo caso podemos renunciar por completo a igualar f(xi) a favor de hacer que f sea más uniforme .

11.1 Interpolación en una sola variable

Antes de considerar el caso más general, diseñaremos métodos para interpolar funciones de una sola variable f : R → R. Como entrada, tomaremos un conjunto de k pares (xi , yi) con el supuesto f(xi) = yi ; nuestro trabajo es encontrar f(x) para x {x1, . . . , xk}.

Nuestra estrategia en esta sección y otras se inspirará en el álgebra lineal al escribir f(x) en una base. Es decir, el conjunto de todas las funciones posibles f : R → R es demasiado grande para trabajar con él e incluye muchas funciones que no son prácticas en un entorno computacional. Por lo tanto, simplificamos el espacio de búsqueda al obligar a f a escribirse como una combinación lineal de un bloque de construcción más simple

funciones de base. Esta estrategia ya es familiar del cálculo básico: la expansión de Taylor escribe funciones en base a polinomios, mientras que las series de Fourier usan seno y coseno.

11.1.1 Interpolación de polinomios

Quizás el interpolador más directo es asumir que f(x) está en R[x], el conjunto de polinomios. Los polinomios son suaves y es sencillo encontrar un polinomio de grado k − 1 a través de k puntos de muestra.

De hecho, el Ejemplo 3.3 ya resuelve los detalles de tal técnica de interpolación. Como un recordatorio, supongamos que deseamos encontrar $f(x) \equiv a0 + a1x + a2x^2 + \ldots + ak-1x^{k-1}$; aquí nuestras incógnitas son los valores a0, . . . , ak−1 . Introducir la expresión yi = f(xi) para cada i muestra que el vectora satisface el sistema k × k de Vandermonde:

$$\begin{bmatrix} 1 & x1 & \cdots & x1^{k-1} \\ 1 & x2 & \cdots & x2^{k-1} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & xk-1 & \cdots & xk-1^{k-1} \end{bmatrix} \begin{bmatrix} a0 \\ a1 \\ \vdots \\ ak-1 \end{bmatrix} = \begin{bmatrix} y0 \\ y1 \\ \vdots \\ yk-1 \end{bmatrix}$$

Por lo tanto, llevar a cabo una interpolación polinomial de grado k se puede lograr utilizando una solución lineal k × k aplicando nuestras estrategias genéricas de los capítulos anteriores, pero de hecho podemos hacerlo mejor.

Una forma de pensar en nuestra forma para f(x) es que está escrita en una base. Al igual que una base para Rn es un conjunto de n vectores linealmente independientes v1, . . . ,vn, aquí el espacio de polinomios de grado x k−1}. Puede el lapso de monomios {1, x, x R[x], pero nuestra elección $x^2, \ldots,$ ser que la base más obvia para k − 1 esté escrita en actual tiene pocas propiedades que la hagan útil para el problema de interpolación.
Una forma de ver este problema es graficar la secuencia de funciones 1, x, x k todas $x^2, x^3, \ldots$ para x [0, 1]; en esto intervalo, es fácil ver que a medida que k crece, las funciones x empiezan a verse similares.

Continuando con la aplicación de nuestra intuición del álgebra lineal, podemos elegir escribir nuestro polinomio en una base que se adapte mejor al problema en cuestión. Esta vez, recuerda que tenemos k pares (x1, y1), . . . ,(xk , yk). Usaremos estos puntos (fijos) para definir la base de interpolación de Lagrange φ1, . . . , φk escribiendo:

$$\varphi i(x) \equiv \frac{\prod j=i\ (x - xj)}{\prod j=i\ (xi - xj)}$$

Aunque no está escrito en base 1, x, x nominal de $x^2, \ldots, x^{k-1}$, es fácil ver que cada φi sigue siendo un poli grado k − 1. Además, la base de Lagrange tiene la siguiente propiedad deseable:

$$\varphi i(x) = \begin{cases} 1 \text{ cuando } = i \\ 0 \text{ en caso contrario.} \end{cases}$$

Por lo tanto, encontrar el polinomio único de grado k − 1 que se ajuste a nuestros pares (xi , yi) es fácil en la base de Lagrange:

$$f(x) \equiv \sum_i yi\varphi i(x)$$

En particular, si sustituimos x = xj encontramos: f(xj) =

$$\sum_i yi\varphi i(xj)$$

= yj por nuestra expresión para φi(x) anterior.

Así, en la base de Lagrange podemos escribir una fórmula cerrada para f(x) que no requiere resolver el sistema de Vandermonde. El inconveniente, sin embargo, es que cada φi(x) toma O(k) tiempo para evaluar usando la fórmula anterior para un x dado, por lo que encontrar f(x) toma O(n ai del sistema de ^{2}) tiempo; si encontramos los coeficientes Vandermonde explícitamente, sin embargo, la evaluación el tiempo puede reducirse a O(n).

Aparte del tiempo de cálculo, la base de Lagrange tiene un inconveniente numérico adicional. Observe que el denominador es el producto de un número de términos. Si los xi están muy juntos, entonces el producto puede incluir muchos términos cercanos a cero, por lo que estamos dividiendo por un número potencialmente pequeño.
Como hemos visto, esta operación puede crear problemas numéricos que deseamos evitar.

Una base para polinomios de grado k − 1 que intenta llegar a un compromiso entre la calidad numérica de los monomios y la eficiencia de la base de Lagrange es la base de Newton, definida de la siguiente manera:

$$\psi_i(x) = \prod_{j=1}^{i-1} (x - x_j)$$

Definimos ψ1(x) ≡ 1. Observa que ψi(x) es un polinomio de grado i − 1. Por definición de ψi , es claro que ψi(x) = 0 para todo < i. Si deseamos escribir f(x) = ∑i ciψi(x) y escribir esta observación más explícitamente, encontramos:

$$\begin{aligned} f(x_1) &= c_1\psi_1(x_1) \\ f(x_2) &= c_1\psi_1(x_2) + c_2\psi_2(x_2) \\ f(x_3) &= c_1\psi_1(x_3) + c_2\psi_2(x_3) + c_3\psi_3(x_3) \\ &\vdots \quad \vdots \end{aligned}$$

En otras palabras, podemos resolver la siguiente fuerza del sistema triangular inferior:

$$\begin{bmatrix} \psi_1(x_1) & 0 & 0 & \cdots & 0 \\ \psi_1(x_2) & \psi_2(x_2) & 0 & \cdots & 0 \\ \psi_1(x_3) & \psi_2(x_3) & \psi_3(x_3) & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \psi_1(x_k) & \psi_2(x_k) & \psi_3(x_k) & \cdots & \psi_k(x_k) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ s_i \end{bmatrix}$$

Este sistema se puede resolver en O(n ^{2}) tiempo usando sustitución hacia adelante, en lugar de O(n ^{3}) tiempo necesario para resolver el sistema de Vandermonde.

Ahora tenemos tres estrategias para interpolar k puntos de datos utilizando un polinomio de grado k − 1 escribiéndolo en las bases monomio, de Lagrange y de Newton. Los tres representan diferentes compromisos entre la calidad numérica y la velocidad. Sin embargo, una propiedad importante es que la función interpolada resultante f(x) es la misma en todos los casos. Más explícitamente, hay exactamente un polinomio de grado k − 1 que pasa por un conjunto de k puntos, por lo que dado que todos nuestros interpoladores son de grado k − 1, deben tener la misma salida.

11.1.2 Bases alternativas

Aunque las funciones polinómicas son particularmente adecuadas para el análisis matemático, no existe una razón fundamental por la que nuestra base de interpolación no pueda consistir en diferentes tipos de funciones. Por ejemplo, un resultado culminante del análisis de Fourier implica que una gran clase de funciones se aproximan bien mediante sumas de funciones trigonométricas cos(kx) y sin(kx) para k N. Una construcción

como el sistema de Vandermonde todavía se aplica en este caso, y de hecho el algoritmo Fast Fourier Transform (que merece una discusión más amplia) muestra cómo llevar a cabo dicha interpolación aún más rápido.

Una extensión más pequeña del desarrollo en §11.1.1 es a funciones racionales de la forma:

$$= 2 \qquad \frac{p0 + p1x + p2x\ f(x)^{2} + \cdots + pmx^{\text{metro}}}{q0 + q1x + q2x^{\text{norte}} + \cdots + qnx^{\text{norte}}}$$

Note que si nos dan k pares (xi , yi), entonces necesitaremos m + n + 1 = k para que esta función esté bien definida. Se debe fijar un grado de libertad adicional para tener en cuenta el hecho de que la misma función racional se puede expresar de múltiples maneras mediante una escala idéntica del numerador y el denominador.

Las funciones racionales pueden tener asíntotas y otros patrones que no se pueden lograr usando solo polinomios, por lo que pueden ser interpolantes deseables para funciones que cambian rápidamente o tienen polos. De hecho, una vez que se fijan m y n, los coeficientes pi y qi todavía se pueden encontrar usando técnicas lineales al multiplicar ambos lados por el denominador:

$$yi(q0 + q1xi + q2x_{i}^{2} + \cdots + qnx_{i}^{\text{norte}}) = p0 + p1xi + p2x_{i}^{2} + \cdots + pmx_{i}^{\text{metro}}$$

Nuevamente, las incógnitas en esta expresión son las p y las q.

Sin embargo, la flexibilidad de las funciones racionales puede causar algunos problemas. Por ejemplo, considere el siguiente ejemplo:

Ejemplo 11.1 (Fracaso de la interpolación racional, Bulirsch-Stoer §2.2). Supongamos que deseamos encontrar f(x) con los siguientes puntos de datos: (0, 1), (1, 2), (2, 2). Podríamos elegir m = n = 1. Entonces, nuestras condiciones lineales se convierten en:

$$q0 = p0$$

$$2(q0 + q1) = p0 + p1\ 2(q0 + 2q1) = p0 + 2p1$$

Una solución no trivial para este sistema es:

$$p0 = 0$$

$$p1 = 2$$

$$q0 = 0$$

$$q1 = 1$$

Esto implica la siguiente forma para f(x):

$$f(x) = \frac{2x}{X}$$

Esta función tiene una degeneración en x = 0 y, de hecho, al cancelar la x en el numerador y el denominador, no se obtiene f(0) = 1 como podríamos desear.

Este ejemplo ilustra un fenómeno mayor. Nuestro sistema lineal para encontrar las p y las q puede tener problemas cuando el denominador resultante $\sum px$ tiene una raíz en cualquiera de las xi fijas .
Se puede demostrar que cuando este es el caso, no existe ninguna función racional con la elección fija de m y n interpolando los valores dados. Una resolución parcial típica en este caso se presenta en (CITE), que incrementa m y n alternativamente hasta que existe una solución no trivial. Sin embargo, desde un punto de vista práctico, la naturaleza especializada de estos métodos es un buen indicador de que las estrategias de interpolación alternativas pueden ser preferibles cuando fallan los métodos racionales básicos.

11.1.3 Interpolación por tramos

Hasta ahora, hemos construido nuestras estrategias de interpolación combinando funciones simples en todo R. Sin embargo, cuando el número k de puntos de datos se vuelve alto, se hacen evidentes muchas degeneraciones. Por ejemplo, la Figura NÚMERO muestra ejemplos en los que ajustar polinomios de alto grado a los datos de entrada puede producir resultados inesperados. Además, la Figura NÚMERO ilustra cómo estas estrategias no son locales, lo que significa que cambiar cualquier valor único yi en los datos de entrada puede cambiar el comportamiento de f para todo x, incluso aquellos que están lejos del xi correspondiente . De alguna manera, esta propiedad no es realista: esperamos que solo los datos de entrada cerca de una determinada x afecten el valor de f(x), especialmente cuando hay una gran nube de puntos de entrada.

Por estas razones, cuando diseñamos un conjunto de funciones base φ1, . . . , φk , una propiedad deseable no es solo que sea fácil trabajar con ellos, sino también que tengan un soporte compacto:

Definición 11.1 (Soporte compacto). Una función g(x) tiene soporte compacto si existe C R tal que g(x) = 0 para cualquier x con |x| > c

Es decir, las funciones admitidas de forma compacta solo tienen un rango finito de puntos en los que pueden tomar valores distintos de cero.

Una estrategia común para construir bases de interpolación con soporte compacto es hacerlo por partes. En particular, gran parte de la literatura sobre gráficos por computadora depende de la construcción de polinomios por partes, que se definen dividiendo R en un conjunto de intervalos y escribiendo un polinomio diferente en cada intervalo. Para hacerlo, ordenaremos nuestros puntos de datos de modo que x1 < x2 < · · · < xk . Entonces, dos ejemplos simples de interpolantes por partes son los siguientes:

- Constante por partes (NÚMERO de figura): para una x dada, encuentre el punto de datos xi minimizando |x − xi | y define f(x) = yi .

- Lineal por partes (Figura NÚMERO): Si x < x1 tomar f(x) = y1, y si x > xk tomar f(x) = yk . De lo contrario, encuentre un intervalo con x [xi , xi+1] y defina

$$f(x) = y_{i+1} \cdot \frac{x - x_i}{x_{i+1} - x_i} + y_i \cdot 1 - \frac{x - x_i}{x_{i+1} - x_i}.$$

Más generalmente, podemos escribir un polinomio diferente en cada intervalo [xi , xi+1]. Observe nuestro patrón hasta ahora: los polinomios constantes por partes son discontinuos, mientras que las funciones lineales por partes son continuas. Es fácil ver que las cuadráticas por partes pueden ser C y así sucesivamente. Estas mayores continuidad y diferenciabilidad se produce a pesar de que cada yi tiene apoyo local; esta teoría se desarrolla en detalle en la construcción de "splines" o curvas que se interpolan entre puntos dados valores de funciones y tangentes.

Esta mayor continuidad, sin embargo, tiene sus propios inconvenientes. Con cada grado adicional de diferenciabilidad, ponemos una suposición de suavidad más fuerte en f . Esta suposición puede ser poco realista: muchos fenómenos físicos son realmente ruidosos o discontinuos, y esta mayor suavidad puede afectar negativamente los resultados de la interpolación. Un dominio en el que este efecto es particularmente claro es cuando la interpolación se usa junto con herramientas de simulación física. La simulación de flujos de fluidos turbulentos con funciones suavizadas puede eliminar fenómenos discontinuos como ondas de choque que son deseables como salida.

Aparte de estos problemas, los polinomios por partes todavía se pueden escribir como combinaciones lineales de funciones básicas. Por ejemplo, las siguientes funciones sirven como base para las funciones constantes por tramos:

$$\varphi_i(x) = \begin{cases} 1 \text{ cuando } x_{i-1}+x_i \le x < \frac{x_i+x_{i+1}}{2} \\ 0 \text{ de lo contrario} \end{cases}$$

Esta base simplemente pone la constante 1 cerca de xi y 0 en cualquier otro lugar; la interpolación constante por partes de un conjunto de puntos (xi , yi) se escribe como f(x) = ∑i yiφi(x). De manera similar, la llamada base de "sombrero" que se muestra en la Figura NÚMERO abarca el conjunto de funciones lineales por partes con bordes afilados en nuestros puntos de datos xi :

$$\psi_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}} & \text{cuando } x_{i-1} < x \le x_i \\ \frac{x_{i+1}-x}{x_{i+1}-x_i} & \text{cuando } x_i < x \le x_{i+1} \text{ en} \\ 0 & \text{caso contrario} \end{cases}$$

Una vez más, por construcción, la interpolación lineal por partes de los puntos de datos dados es f(x) = ∑i yiψi(x).

11.1.4 Procesos Gaussianos y Kriging

No cubierto en CS 205A, otoño de 2013.

11.2 Interpolación multivariable

Existen muchas extensiones de las estrategias anteriores para interpolar una función con puntos de datos dados (xi , yi) donde xi Rn ahora puede ser multidimensional. Sin embargo, las estrategias para la interpolación en este caso no son tan claras porque es menos obvio dividir Rn en un pequeño número de regiones alrededor de xi . Por esta razón, un patrón común es interpolar utilizando funciones de orden relativamente bajo, es decir, preferir estrategias de interpolación simplistas y eficientes a las que generan funciones C ∞ .

Si todo lo que tenemos es el conjunto de entradas y salidas (xi , yi), entonces una estrategia constante por partes para la interpolación es usar la interpolación del vecino más cercano. En este caso f(x) simplemente toma el valor yi correspondiente a xi minimizando x − xi2; las implementaciones simples iteran sobre todo i para encontrar este valor, aunque las estructuras de datos como los árboles kd pueden encontrar vecinos más cercanos más rápidamente. Así como las interpolaciones constantes por partes dividieron R en intervalos sobre los puntos de datos xi , la estrategia del vecino más cercano divide Rn en un conjunto de celdas de Voronoi:

Definición 11.2 (célula de Voronoi). Dado un conjunto de puntos S = {x1,x2, . . . ,xk} Rn , la celda de Voronoi correspondiente a un xi específico es el conjunto Vi ≡ {x : x −xi2 < x −xj2 para todo j = i}. Es decir, es el conjunto de puntos más cercano a xi que a cualquier otro xj en S.

La Figura NÚMERO muestra un ejemplo de las celdas de Voronoi sobre un conjunto de puntos de datos en las . Estos celdas R2 que tienen muchas propiedades favorables; por ejemplo, son polígonos convexos y están localizados alrededor de cada xi . De hecho, la conectividad de las celdas de Voronoi es un problema bien estudiado en geometría computacional que conduce a la construcción de la célebre triangulación de Delaunay.

Hay muchas opciones para la interpolación continua de funciones en Rn , cada una con sus propias ventajas y desventajas. Si deseamos extender nuestra estrategia de vecino más cercano anterior, por ejemplo, podríamos calcular varios vecinos más cercanos de x e interpolar f(x) en función de x −

xi2 para cada vecino más cercano xi . Ciertas estructuras de datos de "k-vecino más cercano" pueden acelerar las consultas en las que desea encontrar múltiples puntos en un conjunto de datos más cercano a una x dada.

Otra estrategia que aparece con frecuencia en la literatura de gráficos por computadora es la interpolación baricéntrica. Supongamos que tenemos exactamente n + 1 puntos de muestra (x1, y1), . . . ,(xn+1, yn+1), donde xi , Rn y como siempre deseamos interpolar los valores de y a todo Rn ; por ejemplo, en el plano se nos darían tres valores asociados a los vértices de un triángulo. Cualquier punto x Rn se puede escribir únicamente como una combinación lineal $x = \sum_{i=1}^{n+1} a_i x_i$ con una restricción adicional de que $\sum_i a_i = 1$; en otras palabras, escribimos x como un promedio ponderado de los puntos xi . La interpolación baricéntrica en este caso simplemente escribe $f(x) = \sum_i a_i(x) y_i$.

En el plano R2 , la interpolación baricéntrica tiene una interpolación geométrica directa en áreas triangulares giratorias, ilustradas en la Figura NÚMERO. Además, es fácil comprobar que la función interpolada resultante f(x) es afín, lo que significa que se puede escribir f(x) = c + d · x para algunos c R y d Rn .

En general, el sistema de ecuaciones que deseamos resolver por interpolación baricéntrica en algún x Rn es:

$$\sum_i a_i x_i = x$$

$$\sum_i a_i = 1 \_$$

En ausencia de degeneraciones, este sistema fora es invertible cuando hay n + 1 puntos xi . Sin embargo, en presencia de más xi , el sistema para a se vuelve subdeterminado. Esto significa que hay múltiples formas de escribir una x dada como un promedio ponderado de las xi .

Una solución a este problema es agregar más condiciones en el vector de pesos promediosa. Esta estrategia da como resultado coordenadas baricéntricas generalizadas, un tema de investigación en las matemáticas e ingeniería modernas. Las restricciones típicas de ona piden que sea suave como una función en Rn y no negativa en el interior del conjunto de xi cuando estos puntos definen un polígono o poliedro. La Figura NÚMERO muestra un ejemplo de coordenadas baricéntricas generalizadas calculadas a partir de puntos de datos en un polígono con más de n + 1 puntos.

Una resolución alternativa del problema indeterminado para coordenadas baricéntricas se relaciona con la idea de usar funciones por partes para la interpolación; restringiremos nuestra discusión aquí a xi R2 por simplicidad, aunque las extensiones a dimensiones más altas son relativamente obvias.
Muchas veces, no solo se nos da el conjunto de puntos xi , sino también una descomposición del dominio que nos interesa (en este caso, un subconjunto de R2) en n + objetos unidimensionales que usan esos puntos como vértices. Por ejemplo, la Figura NÚMERO muestra una teselación de una parte de R2 en triángulos.
La interpolación en este caso es sencilla: el interior de cada triángulo se interpola usando coordenadas baricéntricas.

Ejemplo 11.2 (Sombreado). En gráficos por computadora, una de las representaciones más comunes de una forma es como un conjunto de triángulos en una malla. En el modelo de sombreado por vértice, se calcula un color para cada vértice en una malla. Luego, para representar la imagen en la pantalla, esos valores por vértice se interpolan usando la interpolación baricéntrica al interior de los triángulos. Se utilizan estrategias similares para texturizar y otras tareas comunes. La Figura NÚMERO muestra un ejemplo de este modelo de sombreado simple. Aparte, un problema específico de los gráficos por computadora es la interacción entre las transformaciones de perspectiva y las estrategias de interpolación. La interpolación baricéntrica de color en una superficie 3D y luego proyectar ese color en el plano de la imagen no es lo mismo que proyectar triángulos en el plano de la imagen y luego interpolar colores en el interior del triángulo; por lo tanto, los algoritmos en este dominio deben aplicar la corrección de perspectiva para dar cuenta de este error.

Dado un conjunto de puntos en R2 , el problema de la triangulación está lejos de ser trivial, y los algoritmos para realizar este tipo de cálculo a menudo se extienden mal a Rn . Por lo tanto, en dimensiones más altas, las estrategias de regresión o vecino más cercano se vuelven preferibles (ver Capítulo NÚMERO).

La interpolación baricéntrica conduce a una generalización de las funciones hat lineales por tramos de §11.1.3 ilustradas en la Figura NÚMERO. Recuerde que nuestra salida interpoladora está completamente determinada por los valores yi en los vértices de los triángulos. De hecho, podemos pensar en f(x) como una combinación lineal ∑i yiφi(x), donde cada φi(x) es la función baricéntrica por partes obtenida al poner un 1 en xi y un 0 en cualquier otro lugar, como en la figura NÚMERO . Estas funciones de sombrero triangular forman la base del "método de elementos finitos de primer orden", que exploraremos en capítulos futuros; Las construcciones especializadas que utilizan polinomios de orden superior se conocen como "elementos de orden superior" y se pueden utilizar para garantizar la diferenciabilidad a lo largo de los bordes del triángulo.

Una descomposición alternativa e igualmente importante del dominio de f ocurre cuando los puntos xi ocurren en una cuadrícula regular en Rn . Los siguientes ejemplos ilustran situaciones en las que este es el caso:

Ejemplo 11.3 (Procesamiento de imágenes). Como se mencionó en la introducción, una fotografía digital típica se representa como una cuadrícula m × n de intensidades de color rojo, verde y azul. Podemos pensar en estos valores como viviendo en una red en Z × Z. Sin embargo, supongamos que deseamos rotar la imagen en un ángulo que no sea , un múltiplo de 90◦ . Luego, como se ilustra en la Figura NÚMERO, debemos buscar valores de imagen en posiciones potencialmente no enteras, lo que requiere la interpolación de colores a R × R.

Ejemplo 11.4 (Imágenes médicas). La salida típica de un dispositivo de imágenes por resonancia magnética (IRM) es una cuadrícula am × n × p de valores que representan la densidad del tejido en diferentes puntos; teóricamente, el modelo típico para esta función es f : R3 → R. Podemos extraer la superficie externa de un órgano en particular, que se muestra en la Figura NÚMERO, encontrando el conjunto de niveles {x : f(x) = c} para alguna c. Encontrar este conjunto de niveles requiere que extiendamos f a toda la cuadrícula de vóxeles para encontrar exactamente dónde cruza c.

Las estrategias de interpolación basadas en cuadrículas normalmente aplican las fórmulas unidimensionales de §11.1.3 una dimensión a la vez. Por ejemplo, los esquemas de interpolación bilineal en R2 interpolan linealmente una dimensión a la vez para obtener el valor de salida:

Ejemplo 11.5 (Interpolación bilineal). Supongamos que f toma los siguientes valores:

- f(0, 0) = 1
- f(0, 1) = −3
- f(1, 0) = 5
- f(1, 1) = −11

y el intermedio f se obtiene por interpolación bilineal. Para encontrar $f(\frac{1}{4}, \frac{1}{2})$, primero interpolamos en x1 para encontrar:

$$f\left(\frac{1}{4}, 0\right) = \frac{3}{4} f(0, 0) + \frac{1}{4} f(1, 0) = \frac{3}{4} 1 + \frac{1}{4} 5 = 2$$

$$f\left(\frac{1}{4}, 1\right) = \frac{3}{4} f(0, 1) + \frac{1}{4} f(1, 1) = \frac{3}{4}(-3) + \frac{1}{4}(-11) = -5$$

A continuación, interpolamos en x2:

$$F\left(\frac{1}{4}, \frac{1}{2}\right) = \frac{1}{2}\text{en}\left(\frac{1}{4}, 0 + 2^{-1} f\left(\frac{1}{4}, 3\right)\right) = -2$$

Una propiedad importante de la interpolación bilineal es que recibimos la misma salida interpolando primero en x2 y segundo en x1.

Los métodos de orden superior como la interpolación bicúbica y de Lanczos, una vez más, utilizan más términos polinómicos, pero son más lentos de calcular. En particular, en el caso de la interpolación de imágenes, las estrategias bicúbicas requieren más puntos de datos que el cuadrado de los valores de función más cercanos a un punto x; este gasto adicional puede ralentizar las herramientas gráficas, para las cuales cada búsqueda en la memoria implica un tiempo de cálculo adicional.

11.3 Teoría de la interpolación

Hasta ahora, nuestro tratamiento de la interpolación ha sido bastante heurístico. Si bien confiamos en nuestra intuición de lo que una interpolación "razonable" para un conjunto de valores de función es en su mayor parte una estrategia aceptable, pueden surgir problemas sutiles con diferentes métodos de interpolación que es importante reconocer.

11.3.1 Álgebra lineal de funciones

Comenzamos nuestra discusión planteando una variedad de estrategias de interpolación como bases diferentes para el conjunto de funciones f : R → R. Esta analogía con los espacios vectoriales se extiende a una teoría geométrica completa de las funciones y, de hecho, los primeros trabajos en el campo del análisis funcional esencialmente extiende la geometría de Rn a conjuntos de funciones. Aquí discutiremos funciones de una variable, aunque muchos aspectos de la extensión a funciones más generales son fáciles de llevar a cabo.

Así como podemos definir las nociones de amplitud y combinación lineal para funciones, para a, b R fijos podemos definir un producto interno de las funciones f(x) y g(x) de la siguiente manera:

$$\langle f, g \rangle \equiv \int_a^b f(x)g(x)\,dx.$$

Así como el producto interno A de vectores nos ayudó a derivar el algoritmo de gradientes conjugados y tenía mucho en común con el producto escalar, el producto interno funcional se puede usar para definir métodos de álgebra lineal para tratar con espacios de funciones y comprender su intervalo. También definimos una norma de una función como $\|f\| \equiv \sqrt{\langle f, f \rangle}$.

Ejemplo 11.6. Función producto interno Toma pn(x) = xb = 1 ser el n-ésimo monomio. Entonces, para a = 0 y tenemos:

$$\langle p_n, p_m \rangle = \int_0^1 x^n \cdot x^m\,dx$$
$$= \int_0^1 x^{n+m}\,dx$$
$$= \frac{1}{n+m+1}$$

Note que esto muestra:

$$\frac{pap}{pap}, \frac{p.m.}{p.m.} = \frac{pn,}{pm}$$

$$= \frac{pnpm\,(2n+1)}{(2m+1)\,n+m+1}$$

Este valor es aproximadamente 1 cuando n ≈ m pero n = m, lo que corrobora nuestra afirmación anterior de que los monomios se "superponen" considerablemente en [0, 1].

Dado este producto interno, podemos aplicar el algoritmo de Gram-Schmidt para encontrar una base ortonormal para el conjunto de polinomios. Si tomamos a = −1 y b = 1, obtenemos los polinomios de Legendre, representados en la Figura NÚMERO:

$$P0(x) = 1$$
$$P1(x) = x$$
$$P2(x) = \frac{1}{2}(3x^2 - 1)$$
$$P3(x) = (5\overline{x}\,2\,1^{3\text{ veces})}$$
$$P4(x) = \frac{}{8}(35x^4 - 30x^2 + 3)$$
$$\vdots \quad \vdots$$

Estos polinomios tienen muchas propiedades útiles gracias a su ortogonalidad. Por ejemplo, supongamos que deseamos aproximar f(x) con una suma ∑i aiPi(x). Si deseamos minimizar f − ∑i aiPi en la norma funcional, ¡este es un problema de mínimos cuadrados! Por ortogonalidad de la base de Legendre para R[x], una extensión simple de nuestros métodos de proyección muestra:

$$ai = \frac{f, pi}{pi, pi}$$

Por lo tanto, la aproximación de f usando polinomios se puede lograr simplemente integrando f contra los miembros de la base de Legendre; en el próximo capítulo aprenderemos cómo esta integral podría llevarse a cabo aproximadamente.

Dada una función positiva w(x), podemos definir un producto interno más general ·, ·w escribiendo

$$f, g_w = \int_a^b w(x)f(x)g(x)\,dx.$$

Si tomamos $w(x) = \frac{1}{\sqrt{1-x}}$ con a = −1 y b = 1, luego aplicando Gram-Schmidt produce el Chebyshev 2 polinomios:

$$T0(x) = 1$$
$$T1(x) = x$$
$$T2(x) = 2x^2 - 1$$
$$T3(x) = 4x^3 - 3x$$
$$T4(x) = 8x^4 - 8x^2 + 1$$
$$\vdots \quad \vdots$$

De hecho, se mantiene una identidad sorprendente para estos polinomios:

$$T_k(x) = \cos(k \arccos(x)).$$

Esta fórmula se puede comprobar comprobando explícitamente T0 y T1, y luego aplicando inductivamente la observación:

$$\begin{aligned} T_{k+1}(x) &= \cos((k+1)\arccos(x)) \\ &= 2x\cos(k\arccos(x)) - \cos((k-1)\arccos(x)) \text{ por la identidad} \\ &\qquad \cos((k+1)\theta) = 2\cos(k\theta)\cos(\theta) - \cos((k-1)\theta) \\ &= 2xT_k(x) - T_{k-1}(x) \end{aligned}$$

Esta fórmula de "recurrencia de tres términos" también brinda una manera fácil de generar los polinomios de Chebyshev.

Como se ilustra en la Figura NÚMERO, gracias a la fórmula trigonométrica de los polinomios de Chebyshev es fácil ver que los mínimos y máximos de Tk oscilan entre +1 y −1.
Además, estos extremos están ubicados en cos(iπ/k) (los llamados "puntos de Chebyshev") para i de 0 a k; esta buena distribución de extremos evita fenómenos oscilatorios como el que se muestra en la Figura NÚMERO cuando se usa un número finito de términos polinómicos para aproximar una función. De hecho, los tratamientos más técnicos de la interpolación de polinomios recomiendan colocar xi para la interpolación cerca de los puntos de Chebyshev para obtener un resultado uniforme.

11.3.2 Aproximación mediante polinomios por partes

Supongamos que deseamos aproximar una función f(x) con un polinomio de grado n en un intervalo [a, b]. Defina Δx como el espaciamiento b − a. Una medida del error de una aproximación es en función de Δx, que debería desaparecer cuando $\Delta x \to 0$. Entonces, si aproximamos f con polinomios por partes, este tipo de análisis nos dice qué tan separados debemos espaciar los polinomios para lograr un nivel deseado de aproximación.

Por ejemplo, supongamos que aproximamos f con una constante c = f(interpolación: $\frac{a+b}{2}$), como en constante por partes Si asumimos | f (x)| < M para todo x [a, b], tenemos:

$$\begin{aligned} \max_{x \in [a,b]} |f(x) - c| &\le \Delta x \text{ máx.} \quad M \text{ por el teorema del valor medio } x \in [a,b] \\ &\le M\Delta x \end{aligned}$$

Por lo tanto, esperamos un error $O(\Delta x)$ cuando se utiliza la interpolación constante por tramos.

Supongamos que, en cambio, aproximamos f usando interpolación lineal por partes, es decir, tomando

$$\tilde{f}(x) = f(a) + \text{segundo} - \text{un} \quad \frac{x - \text{un}}{\text{pensión completa). segundo - un}}$$

Por el teorema del valor medio, sabemos que f (x) = f (θ) para algún θ [a, b]. Escribir la expansión de Taylor sobre θ muestra $f(x) = f(\theta) + f(\theta)(x-\theta) + O(\Delta x^2)$ en [a, b], mientras que podemos reescribir nuestra La aproximación a lineal $f(x) = f(\theta) + f(\theta)(x-\theta)$. Por lo tanto, restar estas dos expresiones muestra que). No es difícil medida que el error de aproximación de f disminuye a $O(\Delta x^2$ predecir ese error de aproximación , aunque en la práctica la con un polinomio de grado n hace que $O(\Delta x^{n+1}$ de convergencia cuadrática aproximaciones lineales por partes sea suficiente para la mayoría de las aplicaciones.

11.4 Problemas

Ideas:

- Método de Horner para evaluar polinomios
- Estrategia recursiva para coeficientes de polinomios de Newton.
- Estrías, de Casteljeau
- Comprobar la interpolación del área del triángulo de la interpolación baricéntrica

Capítulo 12

Integración Numérica y Diferenciación

En el capítulo anterior, desarrollamos herramientas para completar valores razonables de una función f(x) dada una muestra de valores (xi , f(xi)) en el dominio de f. Obviamente, este problema de interpolación es útil en sí mismo para completar funciones que se sabe que son continuas o diferenciables pero cuyos valores solo se conocen en un conjunto de puntos aislados, pero en algunos casos luego deseamos estudiar las propiedades de estas funciones. En particular, si deseamos aplicar herramientas del cálculo a f , poder aproximar debemos sus integrales y derivadas.

De hecho, hay muchas aplicaciones en las que la integración y la diferenciación numérica juegan un papel clave en la computación. En el caso más sencillo, algunas funciones bien conocidas se definen como integrales. Por ejemplo, la "función de error" utilizada como distribución acumulativa de una curva de Gauss o de campana se escribe:

$$\operatorname{erf}(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \, dt$$

Se necesitan aproximaciones de erf(x) en muchos contextos estadísticos, y un enfoque razonable para encontrar estos valores es realizar la integral anterior numéricamente.

Otras veces, las aproximaciones numéricas de derivadas e integrales son parte de un sistema más grande. Por ejemplo, los métodos que desarrollaremos en capítulos futuros para aproximar soluciones a ecuaciones diferenciales dependerán en gran medida de estas aproximaciones. De manera similar, en electrodinámica computacional, las ecuaciones integrales que resuelven una función desconocida φ dado un núcleo K y una salida f aparecen en la relación:

$$f(x) = \int_{R^n} K(x,y)\varphi(y) \, dy.$$

Este tipo de ecuaciones deben resolverse para estimar los campos eléctricos y magnéticos, pero a menos que φ y K sean muy especiales, no podemos esperar encontrar tal integral en forma cerrada, y solo resolver esta ecuación para la función desconocida φ.

En este capítulo, desarrollaremos una variedad de métodos para la integración y diferenciación numérica dada una muestra de valores de función. Estos algoritmos suelen ser aproximaciones bastante sencillas, por lo que para compararlos también desarrollaremos algunas estrategias que evalúen qué tan bien esperamos que funcionen los diferentes métodos.

12.1 Motivación

No es difícil formular aplicaciones simples de integración y diferenciación numérica dada la frecuencia con la que las herramientas del cálculo aparecen en las fórmulas y técnicas básicas de la física, la estadística y otros campos. Aquí sugerimos algunos lugares menos obvios donde aparecen la integración y la diferenciación.

Ejemplo 12.1 (Muestreo de una distribución). Supongamos que nos dan una distribución de probabilidad p(t) en el intervalo [0, 1]; es decir, si muestreamos aleatoriamente valores de acuerdo con esta distribución, esperamos que p(t) sea proporcional al número de veces que dibujamos un valor cercano a t. Una tarea común es generar números aleatorios distribuidos como p(t).

En lugar de desarrollar un método especializado para hacerlo cada vez que recibimos un nuevo p(t), es posible hacer una observación útil. Definimos la función de distribución acumulada de p como

$$F(t) = \int_0^t p(x)\,dx.$$

Entonces, si X es un número aleatorio distribuido uniformemente en [0, 1], se puede demostrar que $F^{-1}(X)$ se distribuye como p, donde F^{-1} es el inverso de F. Por lo tanto, si podemos aproximar F o F^{-1} podemos generar números aleatorios según una distribución arbitraria p; esta aproximación equivale a integrar p, lo que puede tener que hacerse numéricamente cuando las integrales no se conocen en forma cerrada.

Ejemplo 12.2 (Optimización). Recuerda que la mayoría de nuestros métodos para minimizar y encontrar raíces de una función f dependían no solo de tener valores f(x) sino también su gradiente $\nabla f(x)$ e incluso Hessian H_f. Hemos visto que algoritmos como BFGS y el método de Broyden construyen aproximaciones aproximadas de las derivadas de f durante el proceso de optimización. Sin embargo, cuando f tiene frecuencias altas, puede ser mejor aproximar ∇f cerca de la iteración actual x_k en lugar de usar valores de puntos x potencialmente lejanos para $< k$.

Ejemplo 12.3 (Representación). La ecuación de renderizado del trazado de rayos y otros algoritmos para el renderizado de alta calidad es una integral que establece que la luz que sale de una superficie es igual a la integral de la luz que entra en la superficie en todas las direcciones entrantes posibles después de que se refleja y se difunde; esencialmente establece que la energía de la luz debe conservarse antes y después de que la luz interactúe con un objeto. Los algoritmos de representación deben aproximarse a esta integral para calcular la cantidad de luz emitida por una superficie que refleja la luz en una escena.

Ejemplo 12.4 (Procesamiento de imágenes). Supongamos que pensamos en una imagen como una función de dos variables I(x, y). Muchos filtros, incluidos los desenfoques gaussianos, se pueden considerar como convoluciones, dadas por

$$(f * g)(x, y) = \int f(u, v)\,g(x - u, y - v)\,du\,dv.$$

Por ejemplo, para desenfocar una imagen podríamos tomar g como Gaussiana; en este caso $(I * g)(x, y)$ puede considerarse como un promedio ponderado de los colores de I cerca del punto (x, y). En la práctica, las imágenes son cuadrículas discretas de píxeles, por lo que esta integral debe aproximarse.

Ejemplo 12.5 (Regla de Bayes). Supongamos que X e Y son variables aleatorias de valor continuo; podemos usar P(X) y P(Y) para expresar las probabilidades de que X e Y tomen valores particulares. A veces, conocer X puede afectar nuestro conocimiento de Y. Por ejemplo, si X es la presión arterial de un paciente e Y es el peso de un paciente,

entonces saber que un paciente tiene un peso alto puede sugerir que también tiene presión arterial alta. Por lo tanto, también podemos escribir distribuciones de probabilidad condicional P(X|Y) (léase “la probabilidad de X dado Y”) que expresen tales relaciones.

Una base de la teoría de la probabilidad moderna establece que P(X|Y) y P(Y|X) están relacionadas de la siguiente

$$P(X|Y) = \frac{\text{manera: } P(Y|X)P(X)}{P(Y|X)P(X)\,dY}$$

Estimar la integral en el denominador puede ser un problema serio en los algoritmos de aprendizaje automático donde las distribuciones de probabilidad toman formas complejas. Por lo tanto, se necesitan esquemas de integración aproximados y a menudo aleatorios para algoritmos en la selección de parámetros que utilizan este valor como parte de una técnica de optimización más amplia.

12.2 Cuadratura

Comenzaremos considerando el problema de la integración numérica o cuadratura. Este problema, en una sola variable, se puede expresar como "Dada una muestra de n puntos de alguna función f(x), f(x) dx". En la sección anterior, presentamos varias encontrar una aproximación de $\int_a^b$ situaciones que hervir hasta exactamente esta técnica.

Hay algunas variaciones del problema que requieren un tratamiento o adaptación ligeramente diferente. ción:

- Los extremos ayb pueden ser fijos, o podemos desear encontrar un esquema de cuadratura que pueda aproximar de manera eficiente las integrales para muchos pares (a, b) .
- Es posible que podamos consultar f(x) en cualquier x pero deseamos aproximar la integral usando relativamente pocas muestras, o podemos recibir una lista de pares precalculados (xi , f(xi)) y estamos obligados a usar estos datos puntos en nuestra aproximación.

Estas consideraciones deben tenerse en cuenta al diseñar algoritmos variados para el problema de la cuadratura.

12.2.1 Cuadratura interpoladora

Muchas de las estrategias de interpolación desarrolladas en el capítulo anterior pueden extenderse a métodos de cuadratura usando una observación muy simple. Supongamos que escribimos una función f(x) en términos de un conjunto de funciones base φi(x):

$$f(x) = \sum_i a_i \varphi_i(x).$$

Entonces, podemos encontrar la integral de f de la siguiente manera:

$$\int_a^b f(x)\,dx = \int_a^b \sum_i a_i \varphi_i(x)\,dx \text{ por definición de } f$$

$$= \sum_i a_i \int_a^b \varphi_i(x)\,dx$$

$$= \sum_i c_i a_i \text{ si hacemos la definición } c_i \equiv \int_a^b \varphi_i(x)\,dx$$

En otras palabras, integrar f simplemente implica combinar linealmente las integrales de las funciones básicas que forman f.

Ejemplo 12.6 (Monomiales). Supongamos que escribimos $f(x) = \sum_k a_k x^k$. Sabemos

$$\int_0^1 x^k \, dx = \frac{1}{k+1},$$

por lo que aplicando la derivación anterior sabemos

$$\int_0^1 f(x) \, dx = \sum_k \frac{a_k}{k+1}.$$

En otras palabras, en nuestra notación anterior hemos definido $c_k = \frac{1}{k+1}$.

Los esquemas en los que integramos una función interpolando muestras e integrando la función interpolada se conocen como reglas de interpolación en cuadratura; casi todos los métodos que presentaremos a continuación se pueden escribir de esta manera. Por supuesto, se nos puede presentar un problema del huevo y la gallina, si la integral $\int \varphi_i(x) \, dx$ no se conoce en forma cerrada. Ciertos métodos en elementos finitos de orden superior se ocupan de este problema dedicando tiempo computacional adicional para hacer una aproximación numérica de alta calidad de la integral de un solo φ_i, y luego, dado que todos los φ tienen una forma similar , aplican fórmulas de cambio de coordenadas para escribir integrales para las funciones de base restantes. Esta integral canónica se puede aproximar fuera de línea utilizando un esquema de alta precisión y luego reutilizarse.

12.2.2 Reglas de cuadratura

Si tenemos un conjunto de $(x_i , f(x_i))$ pares, nuestra discusión anterior sugiere la siguiente forma para una regla de cuadratura para aproximar la integral de f en algún intervalo:

$$Q[f] \equiv \sum_i w_i f(x_i).$$

Diferentes pesos producirán diferentes aproximaciones de la integral, que esperamos se vuelvan cada vez más similares a medida que muestreamos las x_i más densamente.

De hecho, incluso la teoría clásica de la integración sugiere que esta fórmula es un punto de partida razonable. Por ejemplo, la integral de Riemann presentada en muchas clases de introducción al cálculo toma la forma:

$$\int_a^b f(x) = \lim_{\Delta x_k \to 0} \sum_k f(\tilde{x}_k)(x_{k+1} - x_k)$$

Aquí, el intervalo [a, b] se divide en partes $a = x_1 < x_2 < \cdots < x_n = b$, donde $\Delta x_k = x_{k+1} - x_k$ y $\tilde{x}_k$ es cualquier punto en $[x_k , x_{k+1}]$. Para un conjunto fijo de x_k antes de tomar el límite, esta integral claramente se puede escribir en la forma $Q[f]$ anterior.

Desde esta perspectiva, las elecciones de $\{x_i\}$ y $\{w_i\}$ determinan completamente una estrategia para la cuadratura. Hay muchas formas de determinar estos valores, como veremos en la próxima sección y como ya hemos visto para la cuadratura interpoladora.

Ejemplo 12.7 (Método de coeficientes indeterminados). Supongamos que arreglamos $x_1, \ldots, x_n$ y deseamos encontrar un conjunto razonable de pesos acompañantes w_i de modo que $\sum_i w_i f(x_i)$ sea una aproximación adecuada de la integral

apagado . Una alternativa a la estrategia de función base listada arriba es usar el método de coeficientes indeterminados. En esta estrategia, elegimos n funciones f1(x), . . . , fn(x) cuyas integrales se conocen, y pedir que nuestra regla de cuadratura recupere exactamente las integrales de estas funciones:

$$\begin{aligned}
&\int_a^b f1(x)\,dx = w1\,f1(x1) + w2\,f1(x2) + \cdots + wn\,f1(xn)\\
&\int_a^b f2(x)\,dx = w1\,f2(x1) + w2\,f2(x2) + \cdots + wn\,f2(xn)\\
&\qquad\vdots \qquad\qquad \vdots\\
&\int_a^b fn(x)\,dx = w1\,fn(x1) + w2\,fn(x2) + \cdots + wn\,fn(xn)
\end{aligned}$$

Esto crea un sistema lineal de ecuaciones n × n para los wi .

Una opción común es tomar fk(x) = x las integrales^{k-1} , es decir, para asegurarse de que el esquema de cuadratura se recupera de los polinomios de bajo orden. Sabemos

$$\int_a^b k\,x\,dx = \frac{b\,k+1 - un^{k+1}}{k+1}.$$

Por lo tanto, obtenemos el siguiente sistema lineal de ecuaciones para los wi :

$$\begin{aligned}
&w1 + w2 + \cdots + wn = segundo - un\\
&b\,x1w1 + x2w2 + \cdots + xnwn = \frac{2\,2 - un}{2}\\
&^{2x}1w1 + x\,2^{2}w2 + \cdots + x\,^{2}nwn = \frac{^{segundo\,3} - un}{2}\\
&\qquad\vdots \qquad\qquad \vdots\\
&1x^{n-}{}^{1}\,w1 + x\,2\,w2 + \cdots + x^{n-1} = \frac{2\,2\,segundo - un}{2}
\end{aligned}$$

Este sistema es exactamente el sistema de Vandermonde discutido en §11.1.1.

12.2.3 Cuadratura de Newton-Cotes

La cuadratura gobierna cuando la x ¡s están espaciados uniformemente en [a, b] se conocen como cuadratura de Newton-Cotes gobierna. Como se ilustra en la Figura NÚMERO, hay dos opciones razonables de muestras espaciadas uniformemente:

- La cuadratura cerrada de Newton-Cotes coloca xi en ay b. En particular, para k {1, . . . , n} nosotros llevar

$$\equiv un + \frac{(k-1)(b-a)\,xk}{norte - 1}.$$

• La cuadratura abierta de Newton-Cotes no coloca un xi en a o b:

$$\text{un) } xk \equiv un + \frac{k(segundo -}{norte + 1}.$$

Después de hacer esta elección, las fórmulas de Newton-Cotes simplemente aplican la interpolación polinomial para aproximar la integral de a a b; el grado del polinomio obviamente debe ser n − 1 para mantener bien definida la regla de la cuadratura.

En general, mantendremos n relativamente pequeño. De esta forma evitamos los fenómenos de oscilación y ruido que ocurren cuando se ajustan polinomios de alto grado a un conjunto de puntos de datos. Al igual que en la interpolación de polinomios por partes, encadenaremos pequeñas partes en reglas compuestas al integrar en un intervalo grande [a, b].

Reglas cerradas. Las estrategias de cuadratura cerrada de Newton-Cotes requieren n ≥ 2 para evitar dividir por cero. Dos estrategias aparecen a menudo en la práctica:

• La regla trapezoidal se obtiene para n = 2 (entonces x1 = a y x2 = b) interpolando linealmente de f(a) a f(b). Se afirma que

$$\int_a^b dx \approx (b - a)\, 2 \qquad \frac{f(a) + f(b)\, f(x)}{}.$$

• La regla de Simpson se obtiene tomando n = 3, por lo que ahora tenemos

$$x1 = un$$

$$bx2 = \frac{a +}{-\ 2}$$

$$x3 = segundo$$

Integrando la parábola que pasa por estos tres puntos se obtiene

$$\int_a^b segundo - un\ f(x)\ dx \approx \frac{}{6} \quad f(a) + 4\, f \qquad \frac{a + b}{2} \quad + f(b)\ .$$

Reglas abiertas. Las reglas abiertas para la cuadratura permiten la posibilidad de n = 1, dando la regla simplista del punto medio:

$$\int_a^b f(x)\, dx \approx (b - a) f \qquad \frac{a + b}{2} \quad .$$

Los valores más grandes de n producen reglas similares a la regla de Simpson y la regla trapezoidal.

Integración compuesta. En general, es posible que deseemos integrar f(x) con más de uno, dos o tres valores xi . Es obvio cómo construir una regla compuesta a partir del punto medio o las reglas trapezoidales anteriores, como se ilustra en la Figura NÚMERO; simplemente sume los valores a lo largo de cada intervalo. Por ejemplo, si subdividimos [a, b] en k intervalos, entonces podemos tomar $\Delta x \equiv \frac{b-a}{k}$ y xi ≡ a + iΔx. Entonces, la regla compuesta del punto medio es:

$$\int_a^b f(x)\, dx \approx f \quad \sum_{yo=1}^{k} \quad \frac{xi+1 + xi\ 2}{} \quad \Delta x$$

Del mismo modo, la regla del trapecio compuesto es:

$$\int_a^b f(x)\,dx \approx \sum_{yo=1}^{k} \frac{f(xi) + f(xi+1)}{2}\,\Delta x$$

$$= \Delta x \left[\frac{1}{2} f(a) + f(x1) + f(x2) + \cdots + f(xk-1) + \frac{1}{2} f(b) \right]$$

separando los dos valores promedio de f en la primera línea y reindexando

Un tratamiento alternativo de la regla del punto medio compuesto es aplicar la fórmula de interpolación en cuadratura del §12.2.1 a la interpolación lineal por partes; De manera similar, la versión compuesta de la regla trapezoidal proviene de la interpolación lineal por partes.

La versión compuesta de la regla de Simpson, ilustrada en la Figura NÚMERO, encadena tres puntos a la vez para hacer aproximaciones parabólicas. Las parábolas adyacentes se encuentran en xi de índice par y pueden no compartir tangentes. Esta suma, que solo existe cuando n es par, se convierte en:

$$\int_a^b f(x)\,dx \approx \frac{\Delta x}{3} \left[f(a) + 2 \sum_{yo=1}^{n-2-1} f(x2i) + 4 \sum_{yo=1}^{n/2} f(x2i-1) + f(b) \right]$$

$$= \frac{\Delta x}{3} \left[f(a) + 4 f(x1) + 2 f(x2) + 4 f(x3) + 2 f(x4) + \cdots + 4 f(xn-1) + f(b) \right]$$

Exactitud. Hasta ahora, hemos desarrollado una serie de reglas de cuadratura que combinan efectivamente el mismo conjunto de f(xi) de diferentes maneras para obtener diferentes aproximaciones de la integral de f. Cada aproximación se basa en una suposición de ingeniería diferente, por lo que no está claro si alguna de estas reglas es mejor que otra. Por lo tanto, necesitamos desarrollar estimaciones de error que caractericen su comportamiento respectivo. Usaremos nuestros integradores de Newton-Cotes anteriores para mostrar cómo se pueden llevar a cabo tales comparaciones, como se presenta en CITE.

Primero, considere la regla de cuadratura del punto medio en un solo intervalo [a, b]. Definir $c \equiv \frac{1}{2}(a + b)$. El La serie de Taylor de f sobre c es:

$$f(x) = f(c) + f'(c)(x - c) + \frac{1}{2} f''(c)(x - c)^2 + \frac{1}{6} f'''(c)(x - c)^3 + \frac{1}{24} f''''(c)(x - c)^4 + \cdots$$

Así, por simetría alrededor de c, los términos impares desaparecen:

$$\int_a^b f(x)\,dx = (b - a) f(c) + \frac{1}{24} f''(c)(b - a)^3 + \frac{1}{1920} f''''(c)(b - a)^5 + \cdots$$

Observe que el primer término de esta suma es exactamente la estimación de $\int_a^b f(x)\,dx$ proporcionada por el punto medio la regla, por lo que esta regla es precisa hasta $O(\Delta x^3)$.

Ahora, reemplazando a y b en nuestra serie de Taylor para f sobre c muestra:

$$f(a) = f(c) + f'(c)(a - c) + \frac{1}{2} f''(c)(a - c)^2 + \frac{1}{6} f'''(c)(a - c)^3 + \cdots$$

$$f(b) = f(c) + f'(c)(b - c) + \frac{1}{2} f''(c)(b - c)^2 + \frac{1}{6} f'''(c)(b - c)^3 + \cdots$$

Sumar estos y multiplicar ambos lados por b−a/2 muestra:

$$\frac{f(a) + f(b)}{2}(b - a) = f(c)(b - a) + f''(c)\left((a - c)^2 + (b - c)^2\right) + \cdots$$

El término f (c) desaparece por definición de c. Observe que el lado izquierdo es la estimación integral de la regla trapezoidal, y el lado derecho concuerda con nuestra serie de Taylor para $\int_a^b f(x)\,dx$ hasta el término cúbico. En otras palabras, la regla trapezoidal también es $O(\Delta x^3)$ precisa en un solo intervalo.

Hacemos una pausa aquí para notar un resultado inicialmente sorprendente: ¡las reglas trapezoidal y del punto medio tienen el mismo orden de precisión! De hecho, el examen del término de tercer orden muestra que la regla del punto medio es aproximadamente dos veces más precisa que la regla trapezoidal. Este resultado parece contrario a la intuición, ya que la regla trapezoidal usa una aproximación lineal mientras que la regla del punto medio es constante.
Sin embargo, como se ilustra en la Figura NÚMERO, la regla del punto medio en realidad recupera la integral de las funciones lineales, lo que explica su grado adicional de precisión.

Se aplica un argumento similar para encontrar una estimación de error para la regla de Simpson. [ESCRIBA EXPLANACIÓN AQUÍ; OMITIR DE 205A]. Al final encontramos que la regla de Simpson tiene un error como $O(\Delta x^5)$.

Una advertencia importante se aplica a este tipo de análisis. En general, el teorema de Taylor solo se aplica cuando Δx es suficientemente pequeño. Si las muestras están muy separadas, entonces se aplican los inconvenientes de la interpolación polinomial y los fenómenos oscilatorios, como se analiza en la Sección NÚMERO, pueden causar resultados inestables para los esquemas de integración de alto orden.

Por lo tanto, volviendo al caso en que a y b están muy separados, ahora dividimos [a, b] en intervalos de ancho Δx y aplicamos cualquiera de nuestras reglas de cuadratura dentro de estos intervalos. Observe que nuestro número total de intervalos es $b-a/\Delta x$, por lo que debemos multiplicar nuestras estimaciones de error por $1/\Delta x$ en este caso. En particular, se cumplen los siguientes órdenes de precisión:

- Punto medio compuesto: $O(\Delta x^2)$
- Trapezoide compuesto: $O(\Delta x^2)$
- Simpson compuesta: $O(\Delta x^4)$

12.2.4 Cuadratura gaussiana

En algunas aplicaciones, podemos elegir las ubicaciones xi en las que se muestrea f. En este caso, podemos optimizar no solo los pesos para la regla de cuadratura sino también las ubicaciones xi para obtener la máxima calidad. Esta observación conduce a reglas de cuadratura desafiantes pero teóricamente atractivas.

Los detalles de esta técnica están fuera del alcance de nuestra discusión, pero proporcionamos un camino simple para su derivación. En particular, como en el ejemplo 12.7, suponga que deseamos optimizar x1, . . . , xn y w1, . . . , wn simultáneamente para aumentar el orden de un esquema de integración. Ahora tenemos 2n en lugar de n conocidos, por lo que podemos hacer cumplir la igualdad para 2n ejemplos:

$$\int_a^b f1(x)\,dx = w1\,f1(x1) + w2\,f1(x2) + \cdots + wn\,f1(xn)$$
$$\int_a^b f2(x)\,dx = w1\,f2(x1) + w2\,f2(x2) + \cdots + wn\,f2(xn)$$
$$\vdots \qquad\qquad \vdots$$
$$\int_a^b f2n(x)\,dx = w1\,fn(x1) + w2\,fn(x2) + \cdots + wn\,fn(xn)$$

Ahora tanto los xi como los wi son desconocidos, por lo que este sistema de ecuaciones ya no es lineal. Por ejemplo, si deseamos optimizar estos valores para polinomios en el intervalo $[-1, 1]$, haríamos

hay que resolver el siguiente sistema de polinomios (CITE):

$$w1 + w2 = \int_{-1}^{1} 1\,dx = 2$$

$$w1x1 + w2x2 = \int_{-1}^{1} x\,dx = 0$$

$$2\,w1x_1^{\,2} + w2x_2^{\,2} = \int_{-1}^{1} 2^2\,dx = x^3$$

$$3\,3 + w2x_2 = \int_{-1}^{1} {}^{3x}\_\,dx = 0$$

Puede darse el caso de que sistemas como este tengan raíces múltiples y otras degeneraciones que dependen no solo de la elección de fi (típicamente polinomios) sino también del intervalo sobre el cual estamos aproximando una integral. Además, estas reglas no son progresivas, en el sentido de que el conjunto de xi para n puntos de datos no tiene nada en común con los de k puntos de datos cuando k = n, por lo que es difícil reutilizar los datos para lograr una mejor estimación. Por otro lado, cuando son aplicables, la cuadratura gaussiana tiene el grado más alto posible para n fijo. Las reglas de cuadratura de Kronrod intentan evitar este problema optimizando la cuadratura con 2n + 1 puntos mientras se reutilizan los puntos gaussianos.

12.2.5 Cuadratura adaptativa

Como ya hemos mostrado, hay ciertas funciones f cuyas integrales se aproximan mejor con una regla de cuadratura dada que con otras; por ejemplo, las reglas de punto medio y trapezoidal integran funciones lineales con total precisión, mientras que pueden ocurrir problemas de muestreo y otros problemas si f oscila rápidamente.

Recuerde que la regla de cuadratura de Gauss sugiere que la ubicación de los xi puede tener un efecto sobre la calidad de un esquema de cuadratura. Sin embargo, todavía hay una pieza de información que no hemos usado: los valores f(xi). Después de todo, estos determinan la calidad de nuestro esquema de cuadratura.

Con esto en mente, las estrategias de cuadratura adaptativa examinan la estimación actual y generan nuevos xi donde el integrando es más complicado. Las estrategias para la integración adaptativa a menudo comparan el resultado de múltiples técnicas de cuadratura, por ejemplo, trapezoidal y de punto medio, con la suposición de que coinciden donde el muestreo de f es suficiente (ver Figura NÚMERO). Si no están de acuerdo con alguna tolerancia en un intervalo dado, se genera un punto de muestra adicional y se actualizan las estimaciones integrales.

AGREGAR MÁS DETALLES O UN EJEMPLO; DISCUTA EL ALGORITMO RECURSIVO; VISTAZO Y GAUTSCHI

12.2.6 Variables Múltiples Muchas

veces deseamos integrar funciones f(x) donde x Rn . Por ejemplo, cuando n = 2 podemos integrar sobre un rectángulo calculando

$$\int_a^b \int_c^d f(x, y)\,dx\,dy.$$

De manera más general, como se ilustra en la figura NÚMERO¡, podríamos desear encontrar una Ω f(x) dx, integral donde Ω es un subconjunto de Rn

Una "maldición de la dimensionalidad" hace que la integración sea exponencialmente más difícil a medida que aumenta la dimensión. En particular, el número de muestras de f necesarias para lograr una precisión de cuadratura comparable para una integral en Rk aumenta como O(n).k Esta observación puede ser desalentadora pero es algo razonable: cuantas más dimensiones de entrada para f , más muestras se necesitan para comprender su comportamiento en todas las dimensiones.

La estrategia más simple para la integración en Rk es la integral integrada. Por ejemplo, si f es una función ción de dos variables, supongamos que deseamos encontrar $\int_a^b \int_c^d f(x, y)\, dx\, dy$. Para y fijo, podemos aproximar la integral interna sobre x usando una regla de cuadratura unidimensional; luego, integramos estos valores sobre y usando otra regla de cuadratura. Obviamente, ambos esquemas de integración inducen algún error, por lo que es posible que necesitemos muestrear xi más densamente que en una dimensión para lograr la calidad de salida deseada.

Alternativamente, así como subdividimos [a, b] en intervalos, podemos subdividir Ω en triángulos y rectángulos en 2D, poliedros o cajas en 3D, etc. y usar reglas de cuadratura de interpolación simples en cada pieza. Por ejemplo, una opción popular es integrar la salida de la interpolación baricéntrica dentro de los poliedros, ya que esta integral se conoce en forma cerrada.

Sin embargo, cuando n es alto, no es práctico dividir el dominio como se sugiere. En este caso, podemos utilizar el método aleatorio de Monte Carlo. En este caso, simplemente generamos k puntos aleatorios xi Ω con, por ejemplo, probabilidad uniforme. Promediar los valores f(xi) produce una aproximación de $\int_\Omega f(x)\, dx$ que converge como $1/\sqrt{k}$ – ¡independientemente de la dimensión de Ω! Así, en grandes dimensiones la estimación de Monte Carlo es preferible a los métodos de cuadratura deterministas anteriores.

MÁS DETALLES SOBRE LA CONVERGENCIA DE MONTE CARLO Y LA ELECCIÓN DE DISTRIBUCIONES SOBRE Ω

12.2.7 Acondicionamiento

Hasta ahora hemos considerado la calidad de un método de cuadratura usando valores de precisión $O(\Delta x^k)$; obviamente por esta métrica es preferible un conjunto de pesos de cuadratura con k grande.

Sin embargo, otra medida equilibra las medidas de precisión obtenidas con los argumentos de Taylor. En particular, recuerda que escribimos nuestra regla de cuadratura como $Q[f] \equiv \sum_i w_i f(x_i)$. Supongamos que perturbamos f a alguna otra f. Definir $\|f - \hat{f}\|_\infty \equiv \max_{x \in [a,b]} |f(x) - \hat{f}(x)|$. Entonces,

$$\frac{|Q[f] - Q[\hat{f}]|}{\|f - \hat{f}\|_\infty} = \frac{|\sum_i w_i (f(x_i) - \hat{f}(x_i))|}{\|f - \hat{f}\|_\infty}$$

$$\leq \frac{\sum_i |w_i|\, |f(x_i) - \hat{f}(x_i)|}{\text{ya que } |f(x_i) - \hat{f}(x_i)| \leq \|f - \hat{f}\|_\infty \text{ por definición.}} \quad \text{por la desigualdad triangular}$$

Así, la estabilidad o condicionamiento de una regla de cuadratura depende de la norma del conjunto de pesos w

En general, es fácil comprobar que a medida que aumentamos el orden de precisión de la cuadratura, el condicionamiento w empeora porque los wi toman valores negativos grandes; esto contrasta con el caso totalmente positivo, donde el condicionamiento está acotado por b − a porque $\sum_i w_i = b - a$ para el polinomio en los esquemas de terpolación y la mayoría de los métodos de orden bajo solo tienen coeficientes positivos (COMPROBAR). Este hecho es un reflejo de la misma intuición de que no debemos interpolar funciones utilizando polinomios de alto orden. Por lo tanto, en la práctica, generalmente preferimos la cuadratura compuesta a los métodos de alto orden, que pueden proporcionar mejores estimaciones pero pueden ser inestables bajo perturbaciones numéricas.

12.3 Diferenciación

La integración numérica es un problema relativamente estable. en que la influencia de cualquier valor único f(x) sobre $\int_a^b f(x)\,dx$ se reduce a cero a medida que a y b se distancian. Aproximar la derivada de una función f (x), por otro lado, no tiene tal propiedad de estabilidad. Desde la perspectiva del análisis de Fourier, se puede mostrar que la integral f(x) generalmente tiene frecuencias más bajas que f , mientras que la diferenciación para producir f amplifica las frecuencias altas de f , lo que hace que las restricciones de muestreo, el condicionamiento y la estabilidad sean particularmente desafiantes para aproximar f .

A pesar de las circunstancias desafiantes, las aproximaciones de derivadas suelen ser relativamente fáciles de calcular y pueden ser estables según la función en cuestión. De hecho, mientras desarrollábamos la regla de la secante, el método de Broyden, etc., usamos aproximaciones simples de derivadas y gradientes para ayudar a guiar las rutinas de optimización.

Aquí nos enfocaremos en aproximar f para f : R → R. Encontrar gradientes y jacobianos a menudo se logra diferenciando en una dimensión a la vez, reduciendo efectivamente al problema unidimensional que consideramos aquí.

12.3.1 Diferenciación de funciones de base

El caso más simple de diferenciación se da en las funciones que se construyen mediante rutinas de interpolación. Al igual que en §12.2.1, si podemos escribir $f(x) = \sum_i a_i \varphi_i(x)$ entonces por linealidad sabemos

$$f'(x) = \sum_i a_i \varphi_i'(x).$$

En otras palabras, ¡podemos pensar en las funciones φ_i como una base para las derivadas de funciones escritas en la base φ_i !

Un ejemplo de este procedimiento se muestra en la Figura NÚMERO. Este fenómeno a menudo conecta diferentes esquemas de interpolación. Por ejemplo, las funciones lineales por partes tienen derivadas constantes por partes, las funciones polinómicas tienen derivadas polinómicas de menor grado, y así sucesivamente; volveremos a esta estructura cuando consideremos discretizaciones de ecuaciones diferenciales parciales. Mientras tanto, es valioso saber en este caso que f se conoce con total certeza, aunque como en la Figura NÚMERO sus derivadas pueden exhibir discontinuidades indeseables.

12.3.2 Diferencias finitas

Un caso más común es que tenemos una función f(x) que podemos consultar pero cuyas derivadas son desconocidas. Esto sucede a menudo cuando f adopta una forma compleja o cuando un usuario proporciona f(x) como una subrutina sin información analítica sobre su estructura.

La definición de la derivada sugiere un enfoque razonable:

$$f'(x) \equiv \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}$$

Como cabría esperar, para un finito $h > 0$ con pequeños $|h|$ la expresión en el límite proporciona un valor posible que se aproxima a f (x).

Para corroborar esta intuición, podemos usar la serie de Taylor para escribir:

$$f(x+h) = f(x) + f'(x)h + f''(x)\frac{h^2}{2} + \cdots$$

Reordenando esta expresión se muestra:

$$(x) = \frac{f(x + h) - f(x)}{f} + O(h)\ h$$

Por lo tanto, la siguiente aproximación de diferencia directa de f tiene convergencia lineal:

$$\approx h \quad \frac{f(x + h) - f(x)}{f\ (x)}$$

De manera similar, invertir el signo de h muestra que las diferencias hacia atrás también tienen convergencia lineal:

$$(x) \approx h \quad \frac{f(x) - f(x - h)}{f}$$

De hecho, podemos mejorar la convergencia de nuestra aproximación usando un truco. por Taylor's teorema podemos escribir:

$$f(x + h) = f(x) + f\ (x)h + \frac{1}{2}f\ (x)h^{2} + \frac{1}{6}\frac{1}{}f\ (x)h^{3} + \cdots$$

$$f(x - h) = f(x) - f\ (x)h + \frac{1}{2}f\ (x)h^{2} - \frac{1}{6}f\ (x)h^{3} + \cdots$$

$$1 = f(x + h) - f(x - h) = 2\ f\ (x)h + f\ (x)h\ 3 \quad 3 + \cdots$$

$$= \frac{f(x + h) - f(x - h) =}{}f\ (x) + O(h\ 2h^{2})$$

Así, esta diferencia centrada da una aproximación de f (x) con convergencia cuadrática; este es el orden más alto de convergencia que podemos esperar lograr con una diferencia dividida. Sin embargo, podemos lograr una mayor precisión evaluando f en otros puntos, por ejemplo, x + 2h, aunque esta aproximación no se usa mucho en la práctica a favor de simplemente disminuir h.

La construcción de estimaciones de derivadas de orden superior puede realizarse mediante construcciones similares. Para ejemplo, si sumamos las expansiones de Taylor de f(x + h) y f(x − h) vemos

$$f(x + h) + f(x - h) = 2\ f(x) + f\ (x)h\ f(x + ^{2} + O(h^{3})$$

$$= \frac{h) - 2\ f(x) + f(x - h) = f\ (x)}{}+ O(h\ h\ 2 \quad ^{2})$$

Para predecir combinaciones similares para derivadas superiores, un truco es notar que nuestro segundo fórmula derivada se puede factorizar de manera diferente:

$$\frac{f(x + h) - 2\ f(x) + f(x - h)}{h\ 2} = \frac{\frac{f(x+h)-f(x)}{h} - \frac{f(x)-f(x-h)}{h}}{h}$$

Es decir, nuestra aproximación de la segunda derivada es una "diferencia finita de diferencias finitas". Una forma de interpretar esta fórmula se muestra en la Figura NÚMERO. Cuando calculamos la aproximación de la diferencia directa de f entre x y x + h, podemos pensar que esta pendiente vive en x + h/2; de manera similar, podemos usar diferencias hacia atrás para colocar una pendiente en x − h/2. Encontrar la pendiente entre estos valores vuelve a colocar la aproximación en x.

Una estrategia que puede mejorar la convergencia de las aproximaciones anteriores es la extrapolación de Richardson. Como ejemplo de un patrón más general, supongamos que deseamos usar diferencias directas para aproximar f. Definir

$$D(h) \equiv h \frac{f(x+h) - f(x)}{}.$$

Obviamente, D(h) tiende a f (x) cuando h → 0. Más específicamente, sin embargo, de nuestra discusión en §12.3.2 sabemos que D(h) toma la forma:

$$D(h) = f(x) + \frac{1}{2} f(x)h + O(h^2)$$

Supongamos que conocemos D(h) y D(αh) para algún 0 < α < 1. Sabemos:

$$D(\alpha h) = f(x) + \frac{1}{2} f(x)\alpha h + O(h^2)$$

Podemos escribir estas dos relaciones en una matriz:

$$\begin{matrix} 1 & \frac{1}{2}h \\ 1 & \frac{1}{2}\alpha h \end{matrix} \begin{matrix} f(x) \\ f(x) \end{matrix} = \begin{matrix} D(h) \\ D(\alpha h) \end{matrix} + O(h^2)$$

O equivalente,

$$\begin{matrix} f(x) \\ f(x) \end{matrix} = \begin{matrix} \frac{1}{2}h \\ 1\ 1 \quad \frac{1}{2}\alpha h \end{matrix}^{-1} \begin{matrix} D(h) \\ D(\alpha h) \end{matrix} + O(h^2)$$

Es decir, tomamos una aproximación O(h) de f (x) usando D(h) y la convertimos en una simulación ^{2}) aprox. O(h) . Esta ingeniosa técnica es un método para la aceleración de secuencias, ya que mejora el orden de convergencia de las aproximación D(h).El mismo truco es aplicable más generalmente a muchos otros problemas escribiendo una aproximación D(h) = a + bhn + O(h m) donde m > n, donde a es la cantidad que esperamos estimar y b es el siguiente término en la expansión de Taylor De hecho, la extrapolación de Richardson incluso se puede aplicar recursivamente para hacer aproximaciones de orden cada vez mayor.

12.3.3 Elección del tamaño del paso

A diferencia de la cuadratura, la diferenciación numérica tiene una propiedad curiosa. Parece que cualquier método que elijamos puede ser arbitrariamente preciso simplemente eligiendo una h suficientemente pequeña. Esta observación es atractiva desde la perspectiva de que podemos lograr aproximaciones de mayor calidad sin tiempo de cálculo adicional. El truco, sin embargo, es que debemos dividir por h y comparar más y más valores similares f(x) y f(x + h); en la aritmética de precisión finita, sumar y/o dividir por valores cercanos a cero provoca problemas numéricos e inestabilidades. Por lo tanto, hay un rango de valores de h que no son lo suficientemente grandes como para inducir un error de discretización significativo y no lo suficientemente pequeños como para generar problemas numéricos; La Figura NÚMERO muestra un ejemplo para diferenciar una función simple en la aritmética de punto flotante IEEE.

12.3.4 Cantidades integradas

No cubierto en CS 205A, otoño de 2013.

12.4 Problemas

- Cuadratura gaussiana: siempre contiene puntos medios, estrategia con polinomios ortogonales
- Cuadratura adaptativa
- Aplicaciones de la extrapolación de Richardson en otros lugares

Capítulo 13

Ecuaciones diferenciales ordinarias

Motivamos el problema de la interpolación en el Capítulo 11 pasando del análisis a la búsqueda de funciones. Es decir, en problemas como la interpolación y la regresión, la incógnita es una función f y el trabajo del algoritmo es completar los datos que faltan.

Continuamos esta discusión considerando problemas similares que involucran el llenado de valores de función. Aquí, nuestra incógnita sigue siendo una función f , pero en lugar de simplemente adivinar los valores que faltan, nos gustaría resolver problemas de diseño más complejos. Por ejemplo, considere los siguientes problemas:

- Hallar f que se aproxime a alguna otra función f0 pero que satisfaga criterios adicionales (suavidad, continuidad, baja frecuencia, etc.).
- Simular alguna relación dinámica o física como f(t) donde t es el tiempo.
- Encontrar f con valores similares a f0 pero ciertas propiedades en común con una función diferente g0.

En cada uno de estos casos, nuestra incógnita es una función f , pero nuestro criterio de éxito es más complicado que "coincide con un conjunto dado de puntos".

Las teorías de ecuaciones diferenciales ordinarias (EDO) y ecuaciones diferenciales parciales (EDP) estudian el caso en el que deseamos encontrar una función f(x) a partir de información o relaciones entre sus derivadas. Observe que ya hemos resuelto una versión simple de este problema en nuestra discusión sobre la cuadratura: Dada f (t), los métodos para la cuadratura brindan formas de aproximar f (t) usando la integración.

En este capítulo, consideraremos el caso de ecuaciones diferenciales ordinarias y, en particular, problemas de valores iniciales. Aquí, la incógnita es una función $f(t) : R \to R^n$, y dada una ecuación satisfecha por f y sus derivadas así como por f(0); nuestro objetivo es predecir f(t) para t > 0. Proporcionaremos varios ejemplos de EDO que aparecen en la literatura informática y luego procederemos a describir técnicas de solución comunes.

Como nota, usaremos la notación f para denotar la derivada d f/dt de $f : [0, \infty) \to R^n$. Nuestro el objetivo será encontrar f(t) dadas las relaciones entre t, f(t), f (t), f (t), etc.

13.1 Motivación

Las ODE aparecen en casi cualquier parte del ejemplo científico, y no es difícil encontrar situaciones prácticas que requieran su solución. Por ejemplo, las leyes básicas del movimiento físico están dadas por una EDO:

Ejemplo 13.1 (Segunda Ley de Newton). Continuando con §5.1.2, recuerde que la Segunda ley del movimiento de Newton establece que F = ma, es decir, la fuerza total sobre un objeto es igual a su masa por su aceleración.
Si simulamos n partículas simultáneamente, entonces podemos pensar en combinar todas sus posiciones en un vector x R3n . De manera similar, podemos escribir una función F(t,x,x) R3n tomando el tiempo, las posiciones de las partículas y sus velocidades y devolviendo la fuerza total sobre cada partícula. Esta función puede tener en cuenta las interrelaciones entre partículas (por ejemplo, fuerzas gravitatorias o resortes), efectos externos como la resistencia del viento (que depende de x), fuerzas externas que varían con el tiempo t, etc.

Entonces, para encontrar las posiciones de todas las partículas como funciones del tiempo, deseamos resolver la ecuación x = F(t,x,x)/m. Por lo general, se nos dan las posiciones y velocidades de todas las partículas en el tiempo t = 0 como condición inicial.

Ejemplo 13.2 (Plegado de proteínas). En una escala más pequeña, las ecuaciones que gobiernan los movimientos de las moléculas también son ecuaciones diferenciales ordinarias. Un caso particularmente desafiante es el del plegamiento de proteínas, en el que la estructura geométrica de una proteína se predice mediante la simulación de fuerzas intermoleculares a lo largo del tiempo. Estas fuerzas toman muchas formas a menudo no lineales que continúan desafiando a los investigadores en biología computacional.

Ejemplo 13.3 (Descenso de gradiente). Supongamos que deseamos minimizar una función de energía E(x) sobre todo x. Aprendimos en el Capítulo 8 que − E(x) apunta en la dirección E decrece más en un x dado, así que hicimos una búsqueda lineal a lo largo de esa dirección desde x para minimizar E localmente. Una opción alternativa popular en ciertas teorías es resolver una EDO de la forma x = − E(x); en otras palabras, piense en x como una función del tiempo x(t) que intenta disminuir E caminando cuesta abajo.

Por ejemplo, supongamos que deseamos resolver Ax =b para A definida positiva simétrica. Sabemos por §10.1.1 x Ax −bx + c. que esto es equivalente a minimizar E(x) ≡ x = − f(x) $\frac{1}{2}$Por lo tanto, podríamos intentar resolver la ODE
=b − Ax. Como t → ∞, esperamos que x(t) satisfaga cada vez mejor el sistema lineal.

Ejemplo 13.4 (Simulación de multitudes). Supongamos que estamos escribiendo un software de videojuegos que requiere una simulación realista de multitudes virtuales de humanos, animales, naves espaciales y similares. Una estrategia para generar un movimiento plausible, ilustrada en la Figura NÚMERO, es usar ecuaciones diferenciales. Aquí, la velocidad de un miembro de la multitud se determina en función de su entorno; por ejemplo, en las multitudes humanas, la proximidad de otros humanos, la distancia a los obstáculos, etc., pueden afectar la dirección en la que se mueve un agente determinado. Estas reglas pueden ser simples, pero en conjunto su interacción es compleja. Los integradores estables para ecuaciones diferenciales son la base de esta maquinaria, ya que no deseamos tener un comportamiento notoriamente irreal o no físico.

13.2 Teoría de las EDO

Un tratamiento completo de la teoría de las ecuaciones diferenciales ordinarias está fuera del alcance de nuestra discusión, y remitimos al lector a CITE para obtener más detalles. Aparte de esto, mencionamos aquí algunos aspectos destacados que serán relevantes para nuestro desarrollo en secciones futuras.

13.2.1 Nociones básicas

El problema de valor inicial ODE más general toma la siguiente forma:

$$\text{Encuentre } f(t) : R \to \text{R}^{\text{norte}}$$

$$\text{R Satisfacer } F[t, f(t), f(t), f(t), \ldots, f^{(k)}(t)] = 0 \text{ Dado}$$

$$f(0), f(0), f(0), \ldots, f^{(k-1)}(0)$$

Aquí, F es alguna relación entre f y todas sus derivadas; usamos $f^{()}$ para denotar la -ésima derivada de f. Podemos pensar en las EDO como determinantes de la evolución de f en el tiempo t; conocemos f y sus derivadas en el tiempo cero y deseamos predecirlo en el futuro.

Las EDO toman muchas formas incluso en una sola variable. Por ejemplo, denote y = f(t) y suponga que y R1 . Luego, los ejemplos de ODE incluyen:

- y = 1 + cos t: esta EDO se puede resolver integrando ambos lados, por ejemplo, usando cuadratura métodos
- y = ay: Esta EDO es lineal en y
- $y = ay + e^{t}$: Esta ODE depende del tiempo y la posición
- y + 3y − y = t: Esta EDO involucra múltiples derivadas de y
- y sen y = e^{ty} : Esta ODE es no lineal en y y t.

Obviamente, las ODE más generales pueden ser difíciles de resolver. Restringiremos la mayor parte de nuestra discusión al caso de EDO explícitas, en las que se puede aislar la derivada de mayor orden:

Definición 13.1 (EDO Explícita). Una ODE es explícita si se puede escribir en la forma

$$f^{(k)}(t) = F[t, f(t), f(t), f(t), \ldots, f^{(k-1)}(t)].$$

Por ejemplo, una forma explícita de la segunda ley de Newton es $x(t) = \frac{1}{\text{metro}} a(t,x(t),x(t))$.

Sorprendentemente, generalizando el truco de §5.1.2, de hecho, cualquier EDO explícita puede convertirse en una ecuación de primer orden f (t) = F[t, f(t)], donde f tiene una salida multidimensional. Esta observación implica que no necesitamos más de una derivada en nuestro tratamiento de los algoritmos ODE. Para ver esta relación, simplemente recordamos que d 2y/dt2 = d/dt(dy/dt). Así, podemos definir una variable intermedia z ≡ dy/dt, y entender d 2y/dt2 como dz/dt con la restricción z = dy/dt. De manera más general, si deseamos resolver el problema explícito

$$f^{(k)}(t) = F[t, f(t), f(t), f(t), \ldots, f^{(k-1)}(t)],$$

donde f : R → Rn , luego definimos g(t) : R → Rkn usando la EDO de primer orden:

$$\frac{d}{dt}\begin{bmatrix} g_1(t) \\ g_2(t) \\ \vdots \\ g_{k-1}(t) \\ g_k(t) \end{bmatrix} = \begin{bmatrix} g_2(t) \\ g_3(t) \\ \vdots \\ g_k(t) \\ F[t, g_1(t), g_2(t), \ldots, g_{k-1}(t)] \end{bmatrix}$$

Aquí, denotamos gi(t) : R → Rn para contener n componentes de g. Entonces, g1(t) satisface la EDO original. Para verlo, comprobamos que nuestra ecuación anterior implica g2(t) = g 1 (t), g3(t) = g 2 (t) = g 1 (t), y así sucesivamente. Por lo tanto, hacer estas sustituciones muestra que la fila final codifica la EDO original.

El truco anterior simplificará nuestra notación, pero se debe tener cuidado para comprender que este enfoque no trivializa los cálculos. En particular, en muchos casos nuestra función f(t) solo tendrá una única salida, pero la ODE estará en varias derivadas. Reemplazamos este caso con una derivada y varias salidas.

Ejemplo 13.5 (expansión ODE). Supongamos que deseamos resolver y = 3y − 2y + y donde y(t) : R → R. Esta ecuación es equivalente a:

$$\frac{d}{dt}\begin{matrix} y \\ z \\ w \end{matrix} = \begin{matrix} 0\ 1\ 0 \\ 0\ 0\ 1 \\ 1\ -2\ 3 \end{matrix}\begin{matrix} y \\ z \\ w \end{matrix}$$

Así como nuestro truco anterior nos permite considerar solo ODE de primer orden, podemos restringir nuestra notación aún más a ODE autónomas. Estas ecuaciones son de la forma f (t) = F[f(t)], es decir, F ya no depende de t. Para ello, podríamos definir

$$g(t) \equiv \begin{matrix} f(t) \\ \bar{g}(t) \end{matrix} .$$

Entonces, podemos resolver la siguiente EDO para g en su lugar:

$$g\,(t) = \begin{matrix} f\,(t) \\ \bar{g}\,(t) \end{matrix} = \begin{matrix} F[\,f(t),\, \bar{g}(t)] \\ 1 \end{matrix} .$$

En particular, $\bar{g}(t) = t$ asumiendo que tomamos $\bar{g}(0) = 0$.

Es posible visualizar el comportamiento de las ODE de muchas maneras, ilustradas en la Figura NÚMERO. Por ejemplo, si la incógnita f(t) es una función de una sola variable, entonces podemos pensar que F[f(t)] proporciona la pendiente de f(t), como se muestra en la Figura NÚMERO. Alternativamente, si f(t) tiene salida en R2 , ya no podemos visualizar la dependencia del tiempo t, pero podemos dibujar el espacio de fase, que muestra la tangente de f(t) en cada (x, y) R2 .

13.2.2 Existencia y Unicidad

Antes de proceder a las discretizaciones del problema de valor inicial, debemos reconocer brevemente que no todos los problemas de ecuaciones diferenciales tienen solución. Además, algunas ecuaciones diferenciales admiten múltiples soluciones.

Ejemplo 13.6 (EDO no solucionable). Considere la ecuación y = 2y/t, con y(0) = 0 dado; observe que no estamos dividiendo por cero porque se prescribe y(0) . reescribiendo como

$$\frac{1}{y}\frac{dy}{dt} = \frac{2}{t}$$

e integrando con respecto a t en ambos lados muestra:

$$\text{en } |y| = 2 \ln t + c \text{ O}$$

equivalentemente, y = Ct2 para algún C R. Note que y(0) = 0 en esta expresión, contradiciendo nuestras condiciones iniciales. Por tanto, esta EDO no tiene solución con las condiciones iniciales dadas.

Ejemplo 13.7 (Soluciones no únicas). Ahora, considere la misma EDO con y(0) = 0. Considere y(t) dada por y(t) = Ct2 para cualquier C R. Entonces, y (t) = 2Ct. De este modo,

$$\frac{2}{\text{años}} = \frac{2Ct2}{t} = 2Ct = y\,(t),$$

mostrando que la ODE se resuelve mediante esta función independientemente de C. Por lo tanto, las soluciones de este problema no son únicas.

Afortunadamente, existe una rica teoría que caracteriza el comportamiento y la estabilidad de las soluciones de las ecuaciones diferenciales. Nuestro desarrollo en el próximo capítulo tendrá un conjunto más fuerte de condiciones necesarias para la existencia de una solución, pero de hecho bajo condiciones débiles en f es posible mostrar que una EDO f (t) = F[f(t)] tiene un solución. Por ejemplo, uno de esos teoremas garantiza la existencia local de una solución:

Teorema 13.1 (Existencia local y unicidad). Supongamos que F es continua y Lipschitz, es decir, F[y] − F[x]2 ≤ Ly −x2 para algún L. Entonces, la EDO f (t) = F[f(t)] admite exactamente una solución para todo t ≥ 0 independientemente de las condiciones iniciales.

En nuestro desarrollo posterior, supondremos que la EDO que intentamos resolver satisface las condiciones de tal teorema; esta suposición es bastante realista en el sentido de que, al menos localmente, tendría que haber un comportamiento bastante degenerado para romper suposiciones tan débiles.

13.2.3 Ecuaciones modelo

Una forma de ganar intuición sobre el comportamiento de las EDO es examinar el comportamiento de las soluciones de algunas ecuaciones modelo simples que se pueden resolver en forma cerrada. Estas ecuaciones representan linealizaciones de ecuaciones más prácticas y, por lo tanto, modelan localmente el tipo de comportamiento que podemos esperar.

Empezamos con ODEs en una sola variable. Dadas nuestras simplificaciones en §13.2.1, la ecuación más simple con la que podríamos esperar trabajar sería y = F[y], donde y(t) : R → R. Tomar una aproximación lineal produciría ecuaciones del tipo y = ay + b. Sustituyendo $y^{\bar{}} \equiv y + b/a$ muestra: $y^{\bar{}} = y = ay + b = a(y^{\bar{}} - b/a) + b = ay^{\bar{}}$. Por lo tanto, en las ecuaciones de nuestro modelo, la constante b simplemente induce un cambio, y para nuestro estudio fenomenológico en esta sección podemos suponer que b = 0.

Por el argumento anterior, localmente podemos entender el comportamiento de y = F[y] al estudiar el lineal ecuación y = ay. De hecho, la aplicación de argumentos estándar del cálculo muestra que

$$y(t) = Ceat\ .$$

Obviamente, hay tres casos, ilustrados en la Figura NÚMERO:

1. a > 0: En este caso, las soluciones se hacen cada vez más grandes; de hecho, si tanto y(t) como yˆ(t) satisfacen el ODE con condiciones de inicio ligeramente diferentes, ya que t → ∞ divergen.

2. a = 0: El sistema en este caso se resuelve por constantes; las soluciones con diferentes puntos de partida se mantienen separadas por la misma distancia.

3. a < 0: Entonces, todas las soluciones de la ODE se aproximan a 0 cuando t → ∞.

Decimos que los casos 2 y 3 son estables, en el sentido de que al perturbar y(0) se obtienen soluciones que se acercan cada vez más con el tiempo; el caso 1 es inestable, ya que un pequeño error al especificar el parámetro de entrada y(0) se amplificará a medida que avance el tiempo t. Las EDO inestables generan problemas computacionales mal planteados; sin una consideración cuidadosa, no podemos esperar que los métodos numéricos generen soluciones utilizables en este caso, ya que incluso las salidas teóricas son muy sensibles a las perturbaciones de la entrada.
Por otro lado, los problemas estables están bien planteados ya que los pequeños errores en y(0) se reducen con el tiempo.

Avanzando a múltiples dimensiones, podríamos estudiar la ecuación linealizada

$$y = Ay.$$

Como se explica en §5.1.2, si y1, · · · ,yk son vectores propios de A con valores propios λ1, . . . , λk y y(0) = entonces c1y1 + · · ckyk , · +

$$y(t) = c1e^{\lambda 1t}y1 + \cdots + cke^{\lambda k\, t}yk .$$

En otras palabras, los valores propios de A toman el lugar de a en nuestro ejemplo unidimensional. A partir de este resultado, no es difícil intuir que un sistema multivariable es estable exactamente cuando su radio espectral es menor que uno.

En realidad, deseamos resolver y = F[y] para las funciones generales F. Suponiendo que F es diferenciable, podemos escribir F[y] ≈ F[y0] + JF(y0)(y − y0), dando como resultado la ecuación del modelo anterior después de un turno. Por lo tanto, para períodos cortos de tiempo esperamos un comportamiento similar a la ecuación del modelo. Además, las condiciones del Teorema 13.1 pueden verse como un límite en el comportamiento de JF, proporcionando una conexión con teorías menos localizadas de ODE.

13.3 Esquemas de pasos de tiempo

Ahora procedemos a describir varios métodos para resolver la EDO no lineal y = F[y] para funciones potencialmente no lineales F. En general, dado un "paso de tiempo" h, nuestros métodos se utilizarán para generar estimaciones de y(t + h) dado y(t). La aplicación iterativa de estos métodos genera estimaciones de y0 ≡ y(t), y1 ≡ y(t + h), y2 ≡ y(t + 2h), y3 ≡ y(t + 3h), y así sucesivamente. Tenga en cuenta que dado que F no tiene dependencia t, el mecanismo para generar cada paso adicional es el mismo que el primero, por lo que en su mayor parte solo necesitaremos describir un solo paso de estos métodos. Llamamos integradores a los métodos para generar aproximaciones de y(t) , reflejando el hecho de que están integrando las derivadas en la ecuación de entrada.

De importancia clave para nuestra consideración es la idea de estabilidad. Así como las ODE pueden ser estables o inestables, también lo pueden ser las discretizaciones. Por ejemplo, si h es demasiado grande, algunos esquemas acumularán errores a una tasa exponencial; por el contrario, otros métodos son estables en el sentido de que incluso si h es grande, las soluciones permanecerán acotadas. La estabilidad, sin embargo, puede competir con la precisión; a menudo, los esquemas estables en el tiempo son malas aproximaciones de y(t), incluso si se garantiza que no tendrán un comportamiento salvaje.

13.3.1 Euler directo

Nuestra primera estrategia ODE proviene de nuestra construcción del esquema de diferenciación directa en §12.3.2:

$$F[yk] = y (t) = \frac{yk+1 - yk}{h} + O(h)$$

Resolviendo esta relación para yk+1 muestra

$$y_{k+1} = y_k + hF[y_k] + O(h^2) \approx y_k + hF[y_k].$$

Por lo tanto, el esquema de Euler directo aplica la fórmula de la derecha para estimar yk+1 . Es una de las estrategias más eficientes para los pasos de tiempo, ya que simplemente evalúa F y suma un múltiplo del resultado a yk . Por esta razón, lo llamamos método explícito, es decir, hay una fórmula explícita para yk+1 en términos de yk y F.

Analizar la precisión de este método es bastante sencillo. Observe que nuestra aproximación), por lo que cada paso de yk+1 es O(h ^{2}induce un error cuadrático. Llamamos a este error error de truncamiento localizado porque es el error inducido por un solo paso; la palabra “truncamiento” se refiere al hecho de que truncamos una serie de Taylor para obtener esta fórmula. Por supuesto, nuestro iterateyk ya puede ser inexacto gracias a errores de truncamiento acumulados de iteraciones anteriores. Si integramos de t0 a t con pasos O(1/h) , entonces nuestro error total parece O(h); esta estimación representa un error de truncamiento global y, por lo tanto, generalmente escribimos que el esquema de Euler directo es "exacto de primer orden".

La estabilidad de este método requiere algo más de consideración. En nuestra discusión, calcularemos la estabilidad de los métodos en el caso de una variable y = ay, con la intuición de que enunciados similares se trasladan a ecuaciones multidimensionales reemplazando a con el radio espectral. En este caso, sabemos

$$y_{k+1} = y_k + ahy_k = (1 + ah)y_k .$$

En otras palabras, yk = (1 + ah) ky0. Así, el integrador es estable cuando |1 + ah| ≤ 1, ya que de lo contrario |yk | → ∞ exponencialmente. Suponiendo a < 0 (de lo contrario, el problema está mal planteado), podemos simplificar:

$$|1 + ah| \le 1 \quad -1 \le 1 + ah \le 1 \quad -2 \le ah \le 0$$

$$0 \le h \le \frac{2}{|a|}, \text{ ya que un} < 0$$

Por lo tanto, Euler adelantado admite una restricción de paso de tiempo para la estabilidad dada por nuestra condición final en h. En otras palabras, la salida de Euler directo puede explotar incluso cuando y = ay es estable si h no es lo suficientemente pequeño. La figura NÚMERO ilustra lo que sucede cuando se obedece o se viola esta condición.
En múltiples dimensiones, podemos reemplazar esta restricción con una análoga usando el radio espectral de A. Para EDO no lineales, esta fórmula brinda una guía para la estabilidad al menos localmente en el tiempo; globalmente h puede tener que ajustarse si el jacobiano de F se vuelve peor condicionado.

13.3.2 Euler hacia atrás

De manera similar, podríamos haber aplicado el esquema de diferenciación hacia atrás en yk+1 para diseñar un integrador ODE:

$$F[y_{k+1}] = y(t) = h \frac{y_{k+1} - y_k}{} + O(h)$$

Por lo tanto, resolvemos el siguiente sistema de ecuaciones potencialmente no lineal para yk+1 :

$$y_k = y_{k+1} - hF[y_{k+1}].$$

Debido a que tenemos que resolver esta ecuación para yk+1 , Euler hacia atrás es un integrador implícito.

Este método es preciso de primer orden como Euler directo por una prueba idéntica. La estabilidad de este método, sin embargo, contrasta considerablemente con Euler avanzado. Una vez más considerando la ecuación del modelo y = ay, escribimos:

$$y_k \quad y_k = y_{k+1} - hay_{k+1} = \quad y_{k+1} = \frac{}{1 - ha}.$$

Paralelamente a nuestro argumento anterior, Euler hacia atrás es estable bajo la siguiente condición:

$$\frac{1}{} \leq 1 \qquad |1 - ha| \geq 1 \; |1 - ha|$$

$$1 - ha \leq -1 \text{ o } 1 - ha \geq 1$$

$$2 \qquad h \leq \text{o } \frac{h}{a} \geq 0, \text{ para } a < 0$$

Obviamente, siempre tomamos h ≥ 0, por lo que Euler hacia atrás es incondicionalmente estable.

Por supuesto, incluso si Euler hacia atrás es estable, no es necesariamente exacto. Si h es demasiado grande, y_k se aproximará a cero demasiado rápido. Al simular telas y otros materiales físicos que requieren muchos detalles de alta frecuencia para ser realistas, Euler hacia atrás puede no ser una opción efectiva.
Además, tenemos que invertir F[·] para resolver y_{k+1} .

Ejemplo 13.8 (Euler al revés). Supongamos que deseamos resolver y = Ay para A Rn×n . Luego, para encontrar y_{k+1} resolvemos el siguiente sistema:

$$y_k = y_{k+1} - hAy_{k+1} = \quad y_{k+1} = (In\times n - hA) \qquad {}^{-1}_{\text{año}\_}$$

13.3.3 Método trapezoidal

Suponga que y_k se conoce en el tiempo t_k y que y_{k+1} representa el valor en el tiempo $t_{k+1} = t_k + h$. Supongamos que también conocemos $y_{k+1/2}$ a mitad de camino entre estos dos pasos. Entonces, por nuestra derivación de la diferenciación centrada sabemos:

$$y_{k+1} = y_k + hF[y_{k+1/2}] + O(h^3)$$

De nuestra derivación de la regla trapezoidal:

$$\frac{F[y_{k+1}] + F[y_k]}{2} = F[y_{k+1/2}] + O(h^2)$$

Sustituyendo esta relación se obtiene nuestro primer esquema de integración de segundo orden, el método trapezoidal para integrar ODE:

$$y_{k+1} = y_k + h\frac{F[y_{k+1}] + F[y_k]}{2}$$

Al igual que Euler hacia atrás, este método es implícito ya que debemos resolver esta ecuación para y_{k+1} .

Una vez más realizando análisis de estabilidad en y = ay, encontramos en este caso pasos de tiempo del método trapezoidal resolver

$$y_{k+1} = y_k + \frac{1}{2}{}^{-}ja(y_{k+1} + y_k)$$

En otras palabras,

$$y_k = \frac{1 + \frac{1}{2}ha}{1 - \frac{1}{2}hak}\, y_0.$$

Por lo tanto, el método es estable cuando

$$\frac{ha\,1 - 1}{1 - \frac{2}{ha}\,1} < 1.$$

Es fácil ver que esta desigualdad se cumple siempre que a < 0 y h > 0, lo que demuestra que el método trapezoidal es incondicionalmente estable.

A pesar de su mayor orden de precisión con estabilidad mantenida, el método trapezoidal, sin embargo, tiene algunos inconvenientes que lo hacen menos popular que el de Euler inverso. En particular, considere la relación

$$R \equiv \frac{yk+1}{si} = \frac{1 + \frac{1}{2}Ja}{2\,1\,1 - \frac{1}{2}Ja}$$

Cuando a < 0, para h lo suficientemente grande, esta relación eventualmente se vuelve negativa; de hecho, como h → ∞, tenemos R → −1. Por lo tanto, como se ilustra en la figura NÚMERO, si los pasos de tiempo son demasiado grandes, el método trapezoidal de integración tiende a exhibir un comportamiento oscilatorio indeseable que no se parece en nada a lo que podríamos esperar para las soluciones de y = ay .

13.3.4 Métodos de Runge-Kutta

Se puede derivar una clase de integradores haciendo la siguiente observación:

$$yk+1 = yk + \int_{tk}^{tk+\Delta t} y\,(t)\,dt \quad \text{por el Teorema Fundamental del Cálculo}$$

$$= yk + \int_{tk}^{tk+\Delta t} F[y(t)]\,dt$$

Por supuesto, usar esta fórmula directamente no funciona para formular un método de pasos de tiempo, ya que no conocemos y(t), pero la aplicación cuidadosa de nuestras fórmulas de cuadratura del capítulo anterior puede generar estrategias factibles.

Por ejemplo, supongamos que aplicamos el método trapezoidal para la integración. Entonces, encontramos:

$$h\,yk+1 = yk + \frac{}{2}(F[yk\,] + F[yk+1\,]) + O(h^{3})$$

Esta es la fórmula que escribimos para el método trapezoidal en §13.3.3.

Sin embargo , si no deseamos resolver para yk+1 implícitamente, debemos encontrar una expresión para approxi). Haciendo compañero F[yk+1]. Sin embargo, utilizando el método de Euler, sabemos que yk+1 = yk + hF[yk] + O(h esta sustitución de yk+1 no afecta el orden de aproximación del paso de tiempo trapezoidal anterior, por lo que podemos escribir:

$$yk+1 = yk + 2\frac{h}{}(F[yk\,] + F[yk + hF[yk\,]]) + O(h^{3})$$

Ignorando el O(h de) términos produce una nueva estrategia de integración conocida como el método de Heun, que es segundo orden preciso y explícito.

Si estudiamos el comportamiento de estabilidad del método de Heun para y = ay para a < 0, sabemos:

$$h\ y_{k+1} = y_k + \frac{}{} (ay_k + a(y_k + hay_k))$$

$$+ 2 \qquad \frac{2}{} h\ a(2 + ha)\ y_k = 1$$

$$1 = 1 + ha + h\ 2\ \frac{}{}^{2\ 2\ un} \quad si$$

Por lo tanto, el método es estable cuando

$$1 - 1 \le 1 + ha + h\ 2\ 2 - 2^{2\ 2\ un} \le 1$$

$$-4 \le 2ha + h \qquad un \le 0$$

La desigualdad de la derecha muestra $h \le a < 0$, $\frac{-2}{}$ y el de la izquierda siempre es cierto para $h > 0$ y $|a|$, por lo que la condición de estabilidad es $h \le \frac{}{2\ |a|}$.

método de Heun es un ejemplo de un método de Runge-Kutta derivado aplicando métodos de cuadratura a la integral anterior y sustituyendo los pasos de Euler en F[·]. Forward Euler es un método de Runge-Kutta preciso de primer orden, y el método de Heun es de segundo orden. Un método popular de Runge Kutta de cuarto orden (abreviado “RK4”) viene dado por:

$$= y_k + 6 \qquad \frac{h}{} (k1 + 2k2 + 2k3 + k4)\ y_{k+1}$$

donde $k1 = F[y_k]$

$$k2 = F\ y_k + \frac{1}{2} hk1$$

$$k3 = F\ y_k + \frac{1}{2} hk2$$

$$k4 = F\ y_k + hk3$$

Este método se puede derivar aplicando la regla de Simpson para la cuadratura.

Los métodos de Runge-Kutta son populares porque son explícitos y, por lo tanto, fáciles de evaluar al tiempo que proporcionan altos grados de precisión. Sin embargo, el costo de esta precisión es que F[·] debe evaluarse más veces. Además, las estrategias de Runge-Kutta se pueden extender a métodos implícitos que pueden resolver ecuaciones rígidas.

13.3.5 Integradores exponenciales Una clase

de integradores que logra una gran precisión cuando F[·] es aproximadamente lineal es usar nuestra solución a la ecuación modelo explícitamente. En particular, si estuviéramos resolviendo la EDO y = Ay, usando vectores propios de A (o cualquier otro método) podríamos encontrar una solución explícita y(t) como se explica en §13.2.3. Usualmente escribimos $y_{k+1} = e$ $Ahyk$, codifica nuestra exponenciación de los valores propios (de hecho podemos encontrar una matriz e Ah de esta expresión que resuelve la EDO al tiempo h). Ahora, si escribimos donde e_{ah}

$$y = Ay + G[y],$$

donde G es una función no lineal pero pequeña, podemos lograr una precisión bastante alta integrando la parte A explícitamente y luego aproximando la parte G no lineal por separado. Por ejemplo, el integrador exponencial de primer orden aplica Euler directo al término G no lineal:

$$Ah\ y_{k+1} = e\ y_k - A^{-1}(1 - e\ Ah)G[y_k]$$

El análisis que revela las ventajas de este método es más complejo que lo que hemos escrito, pero intuitivamente está claro que estos métodos se comportarán particularmente bien cuando G es pequeño.

13.4 Métodos multivalor

Las transformaciones en §13.2.1 nos permitieron simplificar considerablemente la notación en la sección anterior al reducir todas las EDO explícitas a la forma y = F[y]. De hecho, aunque todas las EDO explícitas pueden escribirse de esta manera, no está claro que siempre deban hacerlo.

En particular, cuando redujimos las EDO de k-ésimo orden a EDO de primer orden, introdujimos una serie de variables que representaban desde la primera hasta la k- 1 -ésima derivada de la salida deseada. De hecho, en nuestra solución final solo nos importa la derivada cero, es decir, la función en sí misma, por lo que los órdenes de precisión en las variables temporales son menos importantes.

Desde esta perspectiva, considere la serie de Taylor

$$2\ y(t_k + h) = y(t_k) + hy\ (t_k) + y\ (t_k) + \Theta(h^3)$$

Si solo conocemos y hasta $O(h^2)$, esto no afecta nuestra aproximación, ya que y se multiplica por h. De manera similar, si solo conocemos y hasta O(h), esta aproximación no afectará los términos de la serie de Taylor anteriores porque se multiplicará por h 2/2. Por lo tanto, ahora consideramos meth (k) (t) = F[t,y (t),y (t), "multivalor" . . . ,y (k−1) (t)] de la función y. con precisión de diferente orden para ods, diseñado para integrar y diferentes derivadas

Dada la importancia de la segunda ley de Newton F = ma, nos limitaremos al caso y = F[t,y,y]; existen muchas extensiones para el caso de orden k-ésimo menos común. Introducimos un vector de "velocidad" v(t) = y (t) y un vector de "aceleración"a. Por nuestra reducción anterior, deseamos resolver el siguiente sistema de primer orden:

$$\begin{aligned} y\ (t) &= v(t)\ v\ (t) \\ =a(t)\ a(t) &= \\ F[t,y(t),v(t)] & \end{aligned}$$

Nuestro objetivo es derivar un integrador diseñado específicamente para este sistema.

13.4.1 Esquemas Newmark

Comenzaremos derivando la famosa clase de integradores de Newmark.1 Denotemos y_k , v_k y a_k como los vectores de posición, velocidad y aceleración en el tiempo t_k ; nuestro objetivo es avanzar al tiempo $t_{k+1} \equiv t_k + h$.

1Seguimos el desarrollo en http://www.stanford.edu/group/frg/course_work/AA242B/CA-AA242B-Ch7.pdf.

Use y(t), v(t) y a(t) para denotar las funciones del tiempo suponiendo que empezamos en tk . Entonces, obviamente podemos escribir

$$v_{k+1} = v_k + \int_{t_k}^{t_{k+1}} a(t)\,dt$$

También podemos escribir yk+1 como una integral que involucra a(t), siguiendo algunos pasos:

$$y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} v(t)\,dt$$

$$= y_k + [tv(t)]_{t_k}^{t_{k+1}} - \int_{t_k}^{t_{k+1}} ta(t)\,dt \text{ después de la integración por partes}$$

$$= y_k + t_{k+1}v_{k+1} - t_k v_k - \int_{t_k}^{t_{k+1}} ta(t)\,dt \text{ expandiendo el término de diferencia}$$

$$= y_k + hv_k + t_{k+1}v_{k+1} - t_{k+1}v_k - \int_{t_k}^{t_{k+1}} ta(t)\,dt \text{ sumando y restando } hv_k$$

$$= y_k + hv_k + t_{k+1}(v_{k+1} - v_k) - \int_{t_k}^{t_{k+1}} ta(t)\,dt \text{ después de factorizar}$$

$$= y_k + hv_k + t_{k+1}\int_{t_k}^{t_{k+1}} a(t)\,dt - \int_{t_k}^{t_{k+1}} ta(t)\,dt \text{ ya que } v(t) = a(t)$$

$$= y_k + hv_k + \int_{t_k}^{t_{k+1}} (t_{k+1} - t)a(t)\,dt$$

Supongamos que elegimos $\tau \in [t_k, t_{k+1}]$. Entonces, podemos escribir expresiones para a_k y a_{k+1} usando la serie de Taylor sobre τ:

$$a_k = a(\tau) + a'(\tau)(t_k - \tau) + O(h^2)$$
$$a_{k+1} = a(\tau) + a'(\tau)(t_{k+1} - \tau) + O(h^2)$$

Para cualquier constante $\gamma \in \mathbb{R}$, si escalamos la primera ecuación por $1 - \gamma$ y la segunda por γ y sumamos los resultados, encontramos:

$$a(\tau) = (1-\gamma)a_k + \gamma a_{k+1} + a'(\tau)((\gamma - 1)(t_k - \tau) - \gamma(t_{k+1} - \tau)) + O(h^2)$$
$$= (1-\gamma)a_k + \gamma a_{k+1} + a'(\tau)(\tau - h\gamma - t_k) + O(h^2) \text{ después de sustituir } t_{k+1} = t_k + h$$

Al final, deseamos integrar a de tk a tk+1 para obtener el cambio de velocidad. Así, calculamos:

$$\int_{t_k}^{t_{k+1}} a(\tau)\,d\tau = (1-\gamma)ha_k + \gamma h a_{k+1} + \int_{t_k}^{t_{k+1}} a'(\tau)(\tau - h\gamma - t_k)\,d\tau + O(h^3)$$

$$= (1-\gamma)ha_k + \gamma h a_{k+1} + O(h^2),$$

donde el segundo paso se cumple porque el integrando es O(h) y el intervalo de integración tiene un ancho h. En otras palabras, ahora sabemos:

$$v_{k+1} = v_k + (1-\gamma)ha_k + \gamma h a_{k+1} + O(h^2)$$

Para hacer una aproximación similar para yk+1 , podemos escribir

$$\int_{t_k}^{t_{k+1}} (t_{k+1} - t)a(t)\,dt = \int_{t_k}^{t_{k+1}} (t_{k+1} - \tau)((1-\gamma)a_k + \gamma a_{k+1} + a'(\tau)(\tau - h\gamma - t_k))\,d\tau + O(h^3)$$

$$= \frac{TK\,1}{2}(1-\gamma)h^2 a_k + \frac{1}{2}\gamma h^2 a_{k+1} + O(h^2) \text{ por un argumento similar.}$$

Por lo tanto, podemos usar nuestra relación anterior para mostrar:

$$y_{k+1} = y_k + hv_k + \int_{t_k}^{t_{k+1}} (t_{k+1} - t)a(t)\,dt \text{ de antes}$$

$$= y_k + hv_k + \left(\frac{1}{2} - \beta\right)h^2 a_k + \beta h^2 a_{k+1} + O(h^2)$$

Aquí, usamos β en lugar de γ (y absorbimos un factor de dos en el proceso) porque el γ que elegimos para aproximar yk+1 no tiene que ser el mismo que elegimos para aproximar vk+1 .

Después de toda esta integración, hemos derivado la clase de esquemas de Newmark, con dos entradas parámetros γ y β, que tiene una precisión de primer orden según la prueba anterior:

$$y_{k+1} = y_k + hv_k + \left(\frac{1}{2} - \beta\right)h^2 a_k + \beta h^2 a_{2k+1}$$

$$v_{k+1} = v_k + (1 - \gamma)ha_k + \gamma ha_{k+1}$$

$$a_k = F[t_k, y_k, v_k]$$

Diferentes opciones de β y γ conducen a diferentes esquemas. Por ejemplo, considere los siguientes ejemplos:

- β = γ = 0 da el integrador de aceleración constante:

$$y_{k+1} = y_k + hv_k + h\,a_k\,\frac{1}{2}$$

$$v_{k+1} = v_k + ha_k$$

  Este integrador es explícito y se cumple exactamente cuando la aceleración es una función constante.

- β = 1/2, γ = 1 da el integrador de aceleración implícito constante:

$$y_{k+1} = y_k + hv_k + h\,a_{k+1}\,\frac{1}{2}$$

$$v_{k+1} = v_k + ha_{k+1}$$

  Aquí, la velocidad se escalona implícitamente utilizando Euler hacia atrás, lo que proporciona una precisión de primer orden. La actualización de y, sin embargo, se puede escribir

$$y_{k+1} = y_k + \frac{1}{2}h(v_k + v_{k+1}),$$

  mostrando que se actualiza localmente por la regla del punto medio; este es nuestro primer ejemplo de un esquema donde las actualizaciones de velocidad y posición tienen diferentes órdenes de precisión. Aun así, es posible demostrar que esta técnica, sin embargo, es globalmente precisa de primer orden en y.

- β = 1/4, γ = 1/2 da el siguiente esquema trapezoidal de segundo orden después de un poco de álgebra:

$$x_{k+1} = x_k + \frac{1}{2}h(v_k + v_{k+1})$$

$$v_{k+1} = v_k + \frac{1}{2}h(a_k + a_{k+1})$$

- β = 0, γ = 1/2 da un esquema de diferenciación central preciso de segundo orden. en el canónico
  forma, tenemos

$$x_{k+1} = x_k + hv_k + h\,a_k\,\frac{1}{2}$$

$$v_{k+1} = v_k + \frac{1}{2}h(a_k + a_{k+1})$$

El método gana su nombre porque la simplificación de las ecuaciones anteriores conduce a la forma alternativa:

$$v_{k+1} = \frac{y_{k+2} - y_k}{2h}$$

$$y_{k+1} - \frac{2y_{k+1} + y_k}{a_{k+1}} = h^2$$

Es posible demostrar que los métodos de Newmark son incondicionalmente estables cuando $4\beta > 2\gamma > 1$ y que la precisión de segundo orden ocurre exactamente cuando $\gamma = 1/2$ (COMPROBAR).

13.4.2 Cuadrícula escalonada

Una forma diferente de lograr una precisión de segundo orden iny es usar diferencias centradas sobre el tiempo $t_{k+1/2} \equiv t_k + h/2$:

$$y_{k+1} = y_k + hv_{k+1/2}$$ En

lugar de usar los argumentos de Taylor para tratar de mover $v_{k+1/2}$, simplemente podemos almacenar velocidades v en puntos medios en la cuadrícula de pasos de tiempo.

Luego, podemos usar una actualización similar para avanzar las velocidades:

$$v_{k+3/2} = v_{k+1/2} + ha_{k+1}\ .$$

Tenga en cuenta que esta actualización también tiene una precisión de segundo orden para x, ya que si sustituimos nuestras expresiones por $v_{k+1/2}$ y $v_{k+3/2}$ podemos escribir:

$$a_{k+1} = \frac{1}{h^2}(y_{k+2} - 2y_{k+1} + y_k)$$

Finalmente, una simple aproximación es suficiente para el término de aceleración ya que es un término de orden superior:

$$a_{k+1} = F\left[t_{k+1}, x_{k+1}, \frac{1}{2}(v_{k+1/2} + v_{k+3/2})\right]$$

Esta expresión se puede sustituir en la expresión de $v_{k+3/2}$.

Cuando F[·] no depende de v, el método es completamente explícito:

$$y_{k+1} = y_k + hv_{k+1/2}$$

$$a_{k+1} = F[t_{k+1}, y_{k+1}]$$

$$v_{k+3/2} = v_{k+1/2} + ha_{k+1}$$

Esto se conoce como el método de integración de salto, gracias a la cuadrícula escalonada de tiempos y al hecho de que cada punto medio se usa para actualizar la siguiente velocidad o posición.

De lo contrario, si la actualización de velocidad depende de v, entonces el método se vuelve implícito. A menudo, la dependencia de la velocidad es simétrica; por ejemplo, la resistencia del viento simplemente cambia de signo si inviertes la dirección en la que te mueves. Esta propiedad puede dar lugar a matrices simétricas en el paso implícito para actualizar las velocidades, lo que permite utilizar gradientes conjugados y métodos iterativos rápidos relacionados para resolver.

13.5 Tareas pendientes

- Definir ODE rígido
- Proporcionar una tabla de métodos de paso de tiempo para F[t;y]
- Usar la notación y de manera más consistente

13.6 Problemas

- TVD RK
- Métodos multipaso/multivalor a la Heath
- Integración Verlet
- Integradores simplécticos

capitulo 14

Ecuaciones diferenciales parciales

Nuestra intuición para las ecuaciones diferenciales ordinarias generalmente proviene de la evolución temporal de los sistemas físicos. Ecuaciones como la segunda ley de Newton, que determina el movimiento de los objetos físicos a lo largo del tiempo, dominan la literatura sobre tales problemas de valores iniciales; ejemplos adicionales provienen de concentraciones químicas que reaccionan con el tiempo, poblaciones de depredadores y presas que interactúan de una estación a otra, y así sucesivamente. En cada caso, se conoce la configuración inicial, por ejemplo, las posiciones y velocidades de las partículas en un sistema en el tiempo cero, y la tarea es predecir el comportamiento a medida que pasa el tiempo.

En este capítulo, sin embargo, consideramos la posibilidad de relaciones de acoplamiento entre diferentes derivadas de una función. No es difícil encontrar ejemplos en los que este acoplamiento sea necesario. Por ejemplo, al simular cantidades de humo o gases como "gradientes de presión", la derivada de la presión de un gas en el espacio, calcule cómo se mueve el gas con el tiempo; esta estructura es razonable ya que el gas se difunde naturalmente desde las regiones de alta presión hacia las regiones de baja presión. En el procesamiento de imágenes, las derivadas se acoplan aún más naturalmente, ya que las mediciones sobre las imágenes tienden a ocurrir en las direcciones x e y simultáneamente.

Las ecuaciones que acoplan derivadas de funciones se conocen como ecuaciones diferenciales parciales. Son el tema de una teoría rica pero fuertemente matizada que merece un tratamiento a mayor escala, por lo que nuestro objetivo aquí será resumir las ideas clave y proporcionar suficiente material para resolver los problemas que comúnmente aparecen en la práctica.

14.1 Motivación

Las ecuaciones diferenciales parciales (EDP) surgen cuando la incógnita es alguna función f : Rn → Rm. Nos dan una o más relaciones entre las derivadas parciales de f, y el objetivo es encontrar una f que satisfaga los criterios. Las PDE aparecen en casi cualquier rama de las matemáticas aplicadas, y enumeramos solo algunas a continuación.

Aparte, antes de introducir PDE específicas, deberíamos introducir alguna notación. En particular, hay algunas combinaciones de derivadas parciales que aparecen con frecuencia en el mundo de las PDE. Si f : R3 → R y v : R3 → R3 , entonces vale la pena recordar los siguientes operadores:

$$\text{Gradiente:} \quad \nabla f \equiv \frac{\partial f}{\partial x_1}\hat{\imath} + \frac{\partial f}{\partial x_2}\hat{\jmath} + \frac{\partial f}{\partial x_3}\hat{k}$$

$$\text{Divergencia:} \quad \cdot v \equiv + + \frac{\partial v1}{\partial x1} \frac{\partial v2}{\partial x2} \frac{\partial v3}{\partial x3} \frac{\partial v3}{\partial 2} \frac{\partial}{\partial 2}$$

$$\text{Rizo:} \quad \times v \equiv \frac{2\,fff}{} - , \frac{\partial v1}{\partial x3} - \frac{\partial v3}{\partial x1}, \frac{\partial v2}{\partial x1} - \frac{\partial v1}{\partial x2}$$

$$\text{Laplaciano:} \quad 2\,f \equiv + + \frac{\partial}{\partial x\,2} \frac{\partial}{\partial x\,2} \frac{1}{\partial x\,2}$$

Ejemplo 14.1 (Simulación de fluidos). El flujo de fluidos como el agua y el humo se rige por las ecuaciones de Navier Stokes, un sistema de PDE en muchas variables. En particular, suponga que un fluido se mueve en alguna región $\Omega \subseteq R3$. Definimos las siguientes variables, ilustradas en la Figura NÚMERO:

- $t \in [0, \infty)$: Tiempo
- $v(t) : \Omega \to R3$: La velocidad del fluido
- $\rho(t) : \Omega \to R$: La densidad del fluido
- $p(t) : \Omega \to R$: La presión del fluido
- $f(t) : \Omega \to R3$: Fuerzas externas como la gravedad sobre el fluido

Si el fluido tiene una viscosidad μ, entonces, si asumimos que es incompresible, las ecuaciones de Navier-Stokes establecen:

$$\rho \frac{\partial v}{\partial t} + v \cdot \nabla v = -\nabla p + \mu \nabla^2 v + f$$

Aquí, $\nabla^2 v = \partial 2v1/\partial x\,2 + \partial 2v2/\partial x\,2 + \partial 2v3/\partial x\,2$; pensamos en el gradiente ∇ como un gradiente en el espacio en lugar de $\partial f\,\partial f\,\partial x1$, $\partial x2$, tiempo, es decir, $\nabla f = (\frac{\partial f}{\partial x1}, \frac{\partial f}{\partial x2}, \frac{\partial f}{\partial x3})$. Este sistema de ecuaciones determina la dinámica temporal del movimiento de fluidos y en realidad se puede construir aplicando la segunda ley de Newton al seguimiento de "partículas" de fluido. Su declaración, sin embargo, involucra no solo derivadas en el tiempo $\frac{\partial}{\partial t}$ pero también derivadas en el espacio ∇, por lo que es una PDE.

Ejemplo 14.2 (ecuaciones de Maxwell). Las ecuaciones de Maxwell determinan la interacción de los campos eléctricos E y los campos magnéticos B a lo largo del tiempo. Al igual que con las ecuaciones de Navier-Stokes, consideramos que el gradiente, la divergencia y el rotacional toman derivadas parciales en el espacio (y no en el tiempo t). Entonces, el sistema de Maxwell (en forma "fuerte") se puede escribir:

$$\text{Ley de Gauss para campos eléctricos:} \quad \cdot E = \frac{\rho}{\varepsilon 0}$$

$$\text{Ley de Gauss para el magnetismo:} \quad \cdot B = 0$$

$$\text{Ley de Faraday:} \quad \times E = -\frac{\partial B}{\partial t}$$

$$\text{Ley de Amp`ere:} \quad \times B = \mu 0\, J + \varepsilon 0 \frac{\partial E}{\partial t}$$

Aquí, $\varepsilon 0$ y $\mu 0$ son constantes físicas y J codifica la densidad de corriente eléctrica. Al igual que las ecuaciones de Navier Stokes, las ecuaciones de Maxwell relacionan las derivadas de las cantidades físicas en el tiempo t con sus derivadas en el espacio (dadas por los términos rotacional y de divergencia).

Ejemplo 14.3 (ecuación de Laplace). Supongamos que Ω es un dominio en R2 con límite ∂Ω y que tenemos una función g : ∂Ω → R, ilustrada en la Figura NÚMERO. Podemos desear interpolar g al interior de Ω. Sin embargo, cuando Ω tiene una forma irregular, nuestras estrategias de interpolación del Capítulo 11 pueden fallar.

Supongamos que definimos f(x) : Ω → R como la función de interpolación. Entonces, una estrategia inspirada en nuestro enfoque de mínimos cuadrados es definir un funcional de energía:

$$E[f] = \int_\Omega \|\nabla f(x)\|_2^2 \, dx$$

Es decir, E[f] mide la "derivada total" de f medida tomando la norma de su gradiente e integrando esta cantidad sobre todo Ω. Las funciones f que fluctúan salvajemente tendrán valores altos de E[f] ya que la pendiente f será grande en muchos lugares; Las funciones suaves y de baja frecuencia f , por otro lado, tendrán E[f] pequeña ya que su pendiente será pequeña en todas partes.1 Entonces, podríamos pedir que f interpole g siendo lo más suave posible en el interior de Ω usando la siguiente optimización:

$$\begin{aligned} &\text{minimizar}_f \; E[f] \\ &\text{tal que } f(x) = g(x) \quad x \quad \partial\Omega \end{aligned}$$

Esta configuración se parece a las optimizaciones que hemos resuelto en ejemplos anteriores, ¡pero ahora nuestra incógnita es una función f en lugar de un punto en Rn !

Si f minimiza E, entonces E[f + h] ≥ E[f] para todas las funciones h(x). Esta afirmación es cierta incluso para pequeñas perturbaciones E[f + εh] cuando ε → 0. Dividiendo por ε y tomando el límite como ε → 0, debemos tener $\frac{d}{d\varepsilon} E[f + \varepsilon h]|_{\varepsilon=0} = 0$; esto es como igualar a cero las derivadas direccionales de una función para encontrar sus mínimos. Podemos simplificar:

$$\begin{aligned} E[f + \varepsilon h] &= \int_\Omega \|f(x) + \varepsilon \quad h(x)\|_2^2 \, días \, x \\ &= \int_\Omega \left(\quad f(x) \quad {}_2^2 + 2\varepsilon \quad f(x) \cdot \quad h(x) + \varepsilon^2 \quad h(x) \quad {}_2^2 \right) dx \end{aligned}$$

Espectáculos diferenciadores:

$$\begin{aligned} \frac{d}{} \text{—} \, mi[f + \varepsilon h] &= d\varepsilon \int_\Omega (2 \quad f(x) \cdot \quad h(x) + 2\varepsilon \quad h(x) \quad {}_2^2) \, dx \\ \frac{d}{d\varepsilon} mi[f + \varepsilon h]|_{\varepsilon=0} &= 2 \int_\Omega [\quad f(x) \cdot \quad h(x)] \, dx \end{aligned}$$

Esta derivada debe ser igual a cero para todo h, por lo que en particular podemos elegir h(x) = 0 para todo x ∂Ω. Entonces, aplicando integración por partes, tenemos:

$$\frac{d}{d\varepsilon} mi[f + \varepsilon h]|_{\varepsilon=0} = -2 \int_\Omega h(x) \quad 2 f(x) \, dx$$

Esta expresión debe ser igual a cero para todas (¡todas!) las perturbaciones h, por lo que debemos tener 2 f(x) = 0 para todo x Ω\∂Ω (una prueba formal está fuera del alcance de nuestra discusión). Es decir, el problema de interpolación anterior

1La notación E[·] que se usa aquí no significa "expectativa" como podría ser en la teoría de la probabilidad, sino que simplemente es un funcional de "energía"; es notación estándar en áreas de análisis funcional.

se puede resolver usando la siguiente PDE: 2

$$f(x) = 0 \quad f(x) = g(x) \quad x \quad \partial\Omega$$

Esta ecuación se conoce como la ecuación de Laplace y se puede resolver usando métodos lineales definidos positivos dispersos como los que cubrimos en el Capítulo 10. Como hemos visto, se puede aplicar a problemas de interpolación para dominios irregulares Ω; además, E[f] se puede aumentar para medir otras propiedades de f , por ejemplo, qué tan bien f se aproxima a alguna función ruidosa f0, para derivar EDP relacionadas haciendo un paralelismo con el argumento anterior.

Ejemplo 14.4 (Ecuación de Eikonal). Supongamos que Ω Rn es una región cerrada del espacio. Entonces, podríamos tomar d(x) como una función que mide la distancia desde algún punto x0 hasta x completamente dentro de Ω. Cuando Ω es convexo, podemos escribir d en forma cerrada:

$$d(x) = x - x0.$$

Sin embargo, como se ilustra en la Figura NÚMERO, si Ω no es convexo o es un dominio más complicado como una superficie, las distancias se vuelven más complicadas de calcular. En este caso, las funciones de distancia d satisfacen la condición localizada conocida como ecuación eikonal:

$$d2 = 1.$$

Si podemos calcularlo, d puede usarse para tareas como planificar rutas de robots minimizando la distancia que tienen que viajar con la restricción de que solo pueden moverse en Ω.

Se utilizan algoritmos especializados conocidos como métodos de marcha rápida para encontrar estimaciones de d dados x0 y Ω mediante la integración de la ecuación eikonal. Esta ecuación no es lineal en la derivada d, por lo que los métodos de integración para esta ecuación son algo especializados y la prueba de su efectividad es compleja. Curiosamente, pero como era de esperar, muchos algoritmos para resolver la ecuación eikonal tienen una estructura similar al algoritmo de Dijkstra para calcular las rutas más cortas a lo largo de los gráficos.

Ejemplo 14.5 (Análisis armónico). Diferentes objetos responden de manera diferente a las vibraciones y, en gran parte, estas respuestas son funciones de la geometría de los objetos. Por ejemplo, los violonchelos y los pianos pueden tocar la misma nota, pero incluso un músico inexperto puede distinguir fácilmente entre los sonidos que hacen. Desde un punto de vista matemático, podemos tomar Ω R3 como una forma representada como una superficie o como un volumen.
Si sujetamos los bordes de la forma, entonces su espectro de frecuencia viene dado por las soluciones del siguiente problema de valores propios diferenciales:

$$2 f = \lambda f$$

$$f(x) = 0 \quad x \quad \partial\Omega, \text{ donde}$$

2 es el laplaciano de Ω y ∂Ω es la frontera de Ω. La Figura NÚMERO muestra ejemplos de estas funciones en diferentes dominios Ω.

Es fácil comprobar que sen kx resuelve este problema cuando Ω es el intervalo [0, 2π], para k Z. En particular, el laplaciano en una dimensión es ∂ 2/∂x 2 , y así podemos comprobar:

$$\frac{\partial^2}{\partial x^2} \operatorname{sen} kx = k \frac{\partial}{\partial x} \cos kx$$

$$2 = -k \operatorname{sen} kx$$

$$\operatorname{sen} k \cdot 0 = 0$$

$$\operatorname{sen} k \cdot 2\pi = 0$$

Por lo tanto, las funciones propias son sen kx con valores propios −k 2·

14.2 Definiciones básicas

Usando la notación de CITE, supondremos que nuestra incógnita es alguna función f : Rn → R. Para ecuaciones de hasta tres variables, usaremos la notación de subíndices para denotar derivadas parciales:

$$f_x \equiv \frac{\partial f}{\partial x},$$
$$f_y \equiv \frac{\partial f}{\partial y\,\partial},$$
$$f_{xy} \equiv \frac{2\,f}{\partial x \partial y},$$

etcétera.

Las derivadas parciales por lo general se expresan como relaciones entre dos o más derivadas de f, de como la siguiente manera:

- Lineal, homogéneo: $f_{xx} + f_{xy} - f_y = 0$
- Lineal: $f_{xx} - y\,f_{yy} + f = xy^2$
- No lineal: $f_{xx}^2 = f_{xy}$

En general, lo que realmente deseamos es encontrar f : Ω → R para algún Ω Rn . Así como las EDO se establecieron como problemas con valores iniciales, la mayoría de las EDO se establecerán como problemas con valores en la frontera. Es decir, nuestro trabajo será llenar f en el interior de Ω dados los valores en su frontera. De hecho, podemos pensar en el problema del valor inicial de ODE de esta manera: el dominio es Ω = [0, ∞), con límite ∂Ω = {0}, que es donde proporcionamos datos de entrada. La figura NÚMERO ilustra ejemplos más complejos. Las condiciones de contorno para estos problemas toman muchas formas:

- Las condiciones de Dirichlet simplemente especifican el valor de f(x) en ∂Ω
- Las condiciones de Neumann especifican las derivadas de f(x) en ∂Ω
- Las condiciones mixtas o Robin combinan estos dos

14.3 Ecuaciones modelo

Recuerde del capítulo anterior que pudimos comprender muchas propiedades de las EDO al examinar una ecuación modelo y = ay. Podemos intentar seguir un enfoque similar para las PDE, aunque encontraremos que la historia tiene más matices cuando las derivadas se vinculan entre sí.

Al igual que con la ecuación modelo para ODE, estudiaremos el caso lineal de una sola variable. También nos limitaremos a sistemas de segundo orden, es decir, sistemas que contengan como máximo la segunda derivada de u; el modelo ODE era de primer orden, pero aquí necesitamos al menos dos órdenes para estudiar cómo interactúan las derivadas de una manera no trivial.

Una PDE lineal de segundo orden tiene la siguiente forma general:

$$\text{yo} + C = 0 \overline{\sum_{ij} a_{ij} \frac{\partial F}{\partial x_i \partial x_j}} + \sum_i b_i \frac{\partial F}{\partial x_i}$$

Formalmente, podríamos definir el "operador de gradiente" como:

$$\equiv \quad \frac{}{\partial \partial x1}, \frac{}{\partial \partial x2}, \ldots, \frac{}{\partial \partial xn} \quad .$$

Debe comprobar que este operador es una notación razonable en el sentido de que expresiones como f , · v y ×v proporcionan las expresiones adecuadas. En esta notación, se puede pensar que la PDE adopta la forma de una matriz:

$$(\quad A \quad + \quad \cdot b + c)f = 0.$$

Esta forma tiene mucho en común con nuestro estudio de formas cuadráticas en gradientes conjugados y, de hecho, generalmente caracterizamos las PDE por la estructura de A:

- Si A es definida positiva o negativa, el sistema es elíptico.
- Si A es semidefinido positivo o negativo, el sistema es parabólico.
- Si A tiene un solo valor propio de signo diferente al resto, el sistema es hiperbólico.
- Si A no cumple ninguno de los criterios, el sistema es ultrahiperbólico.

Estos criterios se enumeran aproximadamente en orden del nivel de dificultad para resolver cada tipo de ecuación. Consideramos los primeros tres casos a continuación y brindamos ejemplos del comportamiento correspondiente; Las ecuaciones ultrahiperbólicas no aparecen con tanta frecuencia en la práctica y requieren técnicas altamente especializadas para su solución.

TODO: Reducción a forma canónica a través de cosas propias de A (no en 205A)

14.3.1 PDE elípticas

Así como las matrices definidas positivas permiten algoritmos especializados como la descomposición de Cholesky y gradientes conjugados que simplifican su inversión, las PDE elípticas tienen una estructura particularmente fuerte que conduce a técnicas de solución efectivas.

La PDE elíptica modelo es la ecuación de Laplace, dada por 2 f = g para alguna función g dada como en el ejemplo 14.3. Por ejemplo, en dos variables la ecuación de Laplace se convierte en

$$f_{xx} + f_{yy} = g.$$

La figura NÚMERO ilustra algunas soluciones de la ecuación de Laplace para diferentes opciones de u y f en un dominio bidimensional.

Las ecuaciones elípticas tienen muchas propiedades importantes. De particular importancia teórica y práctica es la idea de la regularidad elíptica, que las soluciones de las PDE elípticas automáticamente son funciones suaves en C ∞(Ω). Esta propiedad no es inmediatamente obvia: una EDP de segundo orden en f solo requiere que f sea dos veces diferenciable para que tenga sentido, pero de hecho, bajo condiciones débiles, automáticamente son infinitamente diferenciables. Esta propiedad se presta a la intuición física de que las ecuaciones elípticas representan equilibrios físicos estables como la pose de reposo de una lámina de goma estirada.
También se garantiza que las ecuaciones elípticas de segundo orden en la forma anterior admiten soluciones, a diferencia de las PDE en algunas otras formas.

Ejemplo 14.6 (Poisson en una variable). La ecuación de Laplace con g = 0 recibe el nombre especial de ecuación de Poisson. En una variable, se puede escribir f (x) = 0, que trivialmente se resuelve por f (x) = αx + β. Esta ecuación es suficiente para examinar posibles condiciones de contorno en [a, b]:

- Las condiciones de Dirichlet para esta ecuación simplemente especifican f(a) yf (b); obviamente hay una única recta que pasa por (a, f(a)) y (b, f(b)), que proporciona la solución a la ecuación.

- Las condiciones de Neumann especificarían f (a) y f (b). Pero, f (a) = f (b) = α para f(x) = αx + β. De esta forma, los valores límite de los problemas de Neumann pueden estar sujetos a las condiciones de compatibilidad necesarias para admitir una solución. Además, la elección de β no afecta las condiciones de contorno, por lo que cuando se cumplen, la solución no es única.

14.3.2 PDE parabólicas

Continuando con la estructura paralela del álgebra lineal, los sistemas de ecuaciones semidefinidos positivos son solo un poco más difíciles de manejar que los definidos positivos. En particular, las matrices semidefinidas positivas admiten un espacio nulo que debe ser tratado con cuidado, pero en el resto de direcciones las matrices se comportan igual que en el caso definido.

La ecuación del calor es el modelo parabólico PDE. Supongamos que f(0; x, y) es una distribución de calor en alguna región Ω R2 en el tiempo t = 0. Entonces, la ecuación del calor determina cómo se difunde el calor en el tiempo t como una función f(t; x, y) :

$$\frac{\partial f}{} = \alpha \quad 2 f , \partial t$$

donde α > 0 y volvemos a pensar en 2 como el laplaciano en las variables espaciales x e y, es decir, 2 = ∂ 2/∂x 2 + ∂ 2/∂y 2 . Esta ecuación debe ser parabólica, ya que existe el mismo coeficiente α delante de fxx y fyy, pero ftt no figura en la ecuación.

La figura NÚMERO ilustra una interpretación fenomenológica de la ecuación del calor. Podemos pensar que 2 f mide la convexidad de f , como en la Figura NÚMERO(a). Por lo tanto, la ecuación del calor aumenta u con el tiempo cuando su valor se "ahueca" hacia arriba, y disminuye f en caso contrario. Esta retroalimentación negativa es estable y conduce al equilibrio cuando t → ∞.

Hay dos condiciones de contorno necesarias para la ecuación del calor, las cuales vienen con interpretaciones físicas directas:

- La distribución de calor f(0; x, y) en el tiempo t = 0 en todos los puntos (x, y) Ω

- Comportamiento de f cuando t > 0 en los puntos (x, y) ∂Ω. Estas condiciones de frontera describen el comportamiento en la frontera del dominio. Las condiciones de Dirichlet proporcionan aquí f(t; x, y) para todo t ≥ 0 y (x, y) ∂Ω, correspondientes a la situación en la que un agente externo fija las temperaturas en la frontera del dominio. Estas condiciones pueden ocurrir si Ω es un trozo de papel de aluminio junto a una fuente de calor cuya temperatura no se ve afectada significativamente por el papel de aluminio, como un refrigerador grande o un horno. Por el contrario, las condiciones de Neumann especifican la derivada de f en la dirección normal a la frontera ∂Ω, como en la Figura NÚMERO; corresponden a fijar el flujo de calor fuera de Ω provocado por diferentes tipos de aislamiento.

14.3.3 PDE hiperbólicas

La ecuación del modelo final es la ecuación de onda, correspondiente al caso de matriz indefinida:

$$\frac{\partial 2}{\partial t 2} F - C 2 \quad 2 F = 0$$

La ecuación de onda es hiperbólica porque la segunda derivada en el tiempo tiene signo opuesto a las dos derivadas espaciales. Esta ecuación determina el movimiento de las ondas a través de un medio elástico como una lámina de goma; por ejemplo, se puede derivar aplicando la segunda ley de Newton a puntos en una pieza de elástico, donde x e y son posiciones en la hoja y f(t; x, y) es la altura de la pieza de elástico en el tiempo t.

La figura NÚMERO ilustra una solución unidimensional de la ecuación de onda. El comportamiento de las olas contrasta considerablemente con la difusión de calor en que, como t → ∞, la energía puede no difundirse. En particular, las ondas pueden rebotar de un lado a otro indefinidamente en un dominio. Por esta razón, veremos que las estrategias de integración implícita pueden no ser apropiadas para integrar PDE hiperbólicas porque tienden a amortiguar el movimiento.

Las condiciones de contorno para la ecuación de onda son similares a las de la ecuación de calor, pero ahora debemos especificar tanto f(0; x, y) como ft(0; x, y) en el tiempo cero:

- Las condiciones en t = 0 especifican la posición y la velocidad de la onda en el tiempo inicial.
- Las condiciones de contorno en Ω determinan lo que sucede en los extremos del material. Las condiciones de Dirichlet corresponden a la fijación de los lados de la onda, por ejemplo, puntear una cuerda de violonchelo, que se mantiene plana en sus dos extremos sobre el instrumento. Las condiciones de Neumann corresponden a dejar intactos los extremos de la onda como el extremo de un látigo.

14.4 Derivados como Operadores

En las PDE y en otros lugares, podemos pensar en las derivadas como operadores que actúan sobre funciones de la misma manera que las matrices actúan sobre vectores. Nuestra elección de notación a menudo refleja este paralelo: la derivada d f/dx parece el producto de un operador d/dx y una función f .De hecho, la diferenciación es un operador lineal como la multiplicación de matrices, ya que para todo f , g : R → R y a, b R

$$\frac{d}{dx}(a\,f(x) + b\,g(x)) = a\,\frac{d}{dx}f(x) + b\,\frac{d}{dx}g(x).$$

De hecho, cuando discretizamos las EDP para solución numérica, podemos llevar a cabo esta analogía por completo. Por ejemplo, considere una función f en [0, 1] discretizada usando n + 1 muestras espaciadas uniformemente, como en la Figura NÚMERO. Recuerda que el espacio entre dos muestras es h = 1/n. En el Capítulo 12, desarrollamos una aproximación para la segunda derivada f (x) :

$$f(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(h)$$

Suponga que nuestras n muestras de f(x) en [0, 1] son y0 ≡ f(0), y1 ≡ f(h), y2 ≡ f(2h), . . . , yn = f(nh).
Luego, aplicar nuestra fórmula anterior da una estrategia para aproximar f en cada punto de la cuadrícula:

$$y_k \equiv \frac{y_{k+1} - 2y_k + y_{k-1}}{h^2}$$

Es decir, la segunda derivada de una función en una cuadrícula de puntos se puede calcular usando la plantilla 1— − 2 —1 ilustrada en la Figura NÚMERO(a).

Una sutileza que no abordamos es lo que sucede en y 0 e y n , ya que la fórmula anterior requeriría y−1 e yn+1.
De hecho, esta decisión codifica las condiciones de contorno introducidas en §14.2.
Teniendo en cuenta que y0 = f(0) y yn = f(1), los ejemplos de posibles condiciones de contorno para f incluyen:

- Condiciones de contorno de Dirichlet: y−1 = yn+1 = 0, es decir, simplemente fije el valor de y más allá del puntos finales

- Condiciones de contorno de Neumann: y−1 = y0 y yn+1 = yn, codificando la condición de contorno f (0) = f (1) = 0.

- Condiciones de frontera periódicas: y−1 = yn y yn+1 = y0, haciendo la identificación f(0) = f(1)

Supongamos que apilamos las muestras yk en un vector y Rn+1 y las muestras y k en el en un segundo vector w Rn+1 . Entonces, nuestra construcción anterior es fácil de ver que h 2w = L1y, donde L1 es una de las siguientes opciones:

$$\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -2 \end{bmatrix} \quad \begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ & & & 1 & -1 \end{bmatrix} \quad \begin{bmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \ddots & \ddots & \ddots \\ 1 & & & 1 & -2 \end{bmatrix}$$

Dirichlet Neumann Periódico

Es decir, la matriz L puede considerarse como una versión discretizada del operador y Rn+1 $\frac{d}{2\,dx2}$ actuando en lugar de funciones f : [0, 1] → R.

Podemos escribir una aproximación similar para 2 f cuando muestreamos f : [0, 1] × [0, 1] → R con una cuadrícula de valores, como en la Figura NÚMERO. En particular, recuerde que en este caso 2 f = fxx + fyy, por lo que en particular podemos sumar las segundas derivadas de xey como lo hicimos en el ejemplo unidimensional anterior. Esto conduce a una plantilla de 1— − 2—1 doblada , como en la figura NÚMERO. Si numeramos nuestras muestras como yk, ≡ f(kh, h), entonces nuestra fórmula para el Laplaciano de f es en este caso:

$$(\ 2\ y)k, \equiv h^{-2}\ (y(k-1), + yk,(-1) + y(k+1), + yk,(+1) - 4yk,)$$

Si una vez más combinamos nuestras muestras de y e y en y y w, entonces, usando una construcción similar y una elección de condiciones de contorno, podemos escribir una vez más h 2w = L2y. Esta cuadrícula bidimensional Laplaciana L2 aparece en muchas aplicaciones de procesamiento de imágenes, donde (k,) se usa para indexar píxeles en una imagen.

Una pregunta natural después de la discusión anterior es por qué saltamos a la segunda derivada laplaciana en nuestra discusión anterior en lugar de discretizar la primera derivada f (x). En principio, no hay ninguna razón por la que no podamos hacer matrices D similares implementando aproximaciones de f hacia adelante, hacia atrás o en diferencias simétricas. Sin embargo, algunos tecnicismos hacen que esta tarea sea un poco más difícil, como se detalla a continuación.

Lo más importante es que debemos decidir qué aproximación de la primera derivada usar. Si escribimos y k (yk+1 − yk), por ejemplo, estaremos en la posición anormalmente asimétrica de necesitar una condición de contorno en yn pero no en y0. Por el contrario, podríamos usar la diferencia simétrica (yk+1 − yk−1), pero esta discretización adolece de un poste de cerca más sutil problema ilustrado en la figura NÚMERO. En particular, esta versión de y k ignora el valor de yk y solo mira a sus vecinos yk−1 y yk+1 , lo que puede crear artefactos ya que cada fila de D solo involucra yk para k par o impar pero no para ambos.

1 como la diferencia directa h − $\frac{1}{2h}$

Si usamos derivadas hacia adelante o hacia atrás para evitar los problemas del poste de la cerca, perdemos un orden de precisión y también sufrimos las asimetrías descritas anteriormente. Al igual que con la integración de salto

algoritmo de §13.4.2, una forma de evitar estos problemas es pensar en las derivadas como si estuvieran en la mitad de los puntos de la cuadrícula, como se ilustra en la Figura NÚMERO. En el caso unidimensional, este derivadas para etiquetar la $\frac{1}{\text{hora}}$ cambio corresponde $(y_{k+1} - y_k)$ a $y_{k+1/2}$. Esta técnica de colocar diferentes diferencia en vértices, bordes y centros de celdas de cuadrícula es particularmente común en la simulación de fluidos, que mantiene presiones, velocidades de fluidos, etc. en ubicaciones que simplifican los cálculos.

Dejando a un lado estas sutilezas, nuestra conclusión principal de esta discusión es que si discretizamos una función $f(x)$ al hacer un seguimiento de las muestras (x_i , y_i) , entonces las aproximaciones más razonables de las derivadas de f serán computables como un producto Lx para alguna matriz L. Esta observación completa la analogía: "Las derivadas actúan sobre las funciones como las matrices actúan sobre los vectores". O en notación de examen estandarizada: Derivadas: Funciones :: Matrices: Vectores

14.5 Resolviendo PDE Numéricamente

Queda mucho por decir sobre la teoría de las PDE. Las cuestiones de existencia y unicidad, así como la posibilidad de caracterizar soluciones para una variedad de PDE, conducen a debates matizados que utilizan aspectos avanzados del análisis real. Si bien se necesita una comprensión completa de estas propiedades para demostrar rigurosamente la efectividad de las discretizaciones de PDE, ya tenemos suficiente para sugerir algunas técnicas que se usan en la práctica.

14.5.1 Resolución de ecuaciones elípticas

Ya hemos hecho la mayor parte del trabajo para resolver PDE elípticas en §14.4. En particular, supongamos que deseamos resolver una PDE elíptica lineal de la forma $Lf = g$. Aquí L es un operador diferencial; por ejemplo, para resolver la ecuación de Laplace tomaríamos $L \equiv \nabla^2$ laplaciana. Luego, en §14.4 mostramos que si discretizamos f tomando un conjunto de muestras en un vector y con $y_i = f(x_i)$, entonces una aproximación correspondiente de Lf puede escribirse Ly para alguna matriz L. Si también discretizamos g usando muestras en un vectorb, luego resolviendo la PDE elíptica $Lf = g$ se aproxima resolviendo el sistema lineal $Ly = b$.

Ejemplo 14.7 (Discretización de PDE elíptica). Supongamos que deseamos aproximar soluciones a $f''(x) = g(x)$ en $[0, 1]$ con condiciones de frontera $f(0) = f(1) = 0$. Aproximaremos $f(x)$ con un vector $y \in \mathbb{R}^n$ muestreo f de la siguiente manera:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \approx \begin{bmatrix} f(h) \\ f(2h) \\ \vdots \\ f(nh) \end{bmatrix}$$

donde $h = 1/n+1$. No agregamos muestras en $x = 0$ o $x = 1$ ya que las condiciones de contorno determinan los valores allí. Usaremosb para contener un conjunto análogo de valores para $g(x)$.

Dadas nuestras condiciones de contorno, discretizamos f (x) como h^{-2} Ly, donde L viene dada por:

$$L \equiv \begin{pmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{pmatrix}$$

Por lo tanto, nuestra solución aproximada de la EDP está dada por y = h 2L −1b.

Así como las PDE elípticas son las PDE más sencillas de resolver, sus discretizaciones usando matrices como en el ejemplo anterior son las más sencillas de resolver. De hecho, generalmente la discretización de un operador elíptico L es una matriz definida positiva y dispersa perfectamente adecuada para las técnicas de solución derivadas en el Capítulo 10.

Ejemplo 14.8 (Operadores elípticos como matrices). Considere la matriz L del ejemplo 14.7. Podemos demostrar que L es definido negativo (y por lo tanto el sistema definido positivo −Ly = −h 2b puede resolverse usando gradientes conjugados) observando que −L = DD para la matriz D R(n+1)×n dada por:

$$re = \begin{pmatrix} 1 & & & & \\ -1 & 1 & -1 & & \\ & 1 & & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & -1 \end{pmatrix}$$

Esta matriz no es más que la primera derivada en diferencias finitas, por lo que esta observación es paralela al hecho de que d 2 f/dx2 = d/dx(d f/dx). Por lo tanto, x Lx = −x DDx = −Dx ≤ 0, mostrando que L es semidefinido negativo. Es fácil ver Dx = 0 exactamente cuando x = 0, completando la demostración de que L es definida negativa.

14.5.2 Resolución de ecuaciones parabólicas e hiperbólicas

Las ecuaciones parabólicas e hiperbólicas generalmente introducen una variable de tiempo en la formulación, que también es diferenciada pero potencialmente de menor orden. Dado que las soluciones de las ecuaciones parabólicas admiten muchas propiedades de estabilidad, las técnicas numéricas para manejar esta variable temporal suelen ser estables y están bien condicionadas; por el contrario, se debe tener más cuidado para tratar el comportamiento hiperbólico y evitar la amortiguación del movimiento con el tiempo.

Métodos semidiscretos Probablemente el método más sencillo para resolver ecuaciones simples dependientes del tiempo es usar un método semidiscreto. Aquí, discretizamos el dominio pero no la variable de tiempo, lo que lleva a una EDO que se puede resolver usando los métodos del Capítulo 13.

Ejemplo 14.9 (Ecuación de calor semidiscreta). Considere la ecuación del calor en una variable, dada por ft = fxx, donde f(t; x) representa el calor en la posición x y el tiempo t. Como datos de límite, el usuario proporciona una

función f0(x) tal que f(0; x) ≡ f0(x); también adjuntamos el límite x {0, 1} a un refrigerador y así hacemos cumplir f(t; 0) = f(t; 1) = 0.

Supongamos que discretizamos la variable x definiendo:

$$f1(t) \equiv f(h; t)$$
$$f2(t) \equiv f(2h; t)$$
$$\vdots \quad \vdots$$
$$fn(t) \equiv f(nh; t),$$

donde, como en el ejemplo 14.7, tomamos h = 1/n+1 y omitimos las muestras en x {0, 1} ya que las proporcionan las condiciones de contorno.

Combinando estos fi , podemos definir f(t) : R → Rn como la versión semidiscreta de f donde hemos muestreado en el espacio pero no en el tiempo. Por nuestra construcción, la aproximación PDE semidiscreta es la ODE dada por f (t) = L f (t).

El ejemplo anterior muestra una instancia de un patrón muy general para ecuaciones parabólicas. Cuando simulamos fenómenos continuos como el calor que se mueve a través de un dominio o los productos químicos que se difunden a través de una membrana, generalmente hay una variable de tiempo y luego varias variables espaciales que se diferencian de forma elíptica. Cuando discretizamos este sistema de forma semidiscreta, podemos usar estrategias de integración ODE para su solución. De hecho, de la misma manera que la matriz utilizada para resolver una ecuación elíptica lineal como en §14.5.1 generalmente es definida positiva o negativa, cuando escribimos una EDP parabólica semidiscreta f = L f , la matriz L suele ser definida negativa . Esta observación implica que f resolver esta EDO continua es incondicionalmente estable, ya que los valores propios negativos se amortiguan con el tiempo.

Como se señaló en el capítulo anterior, tenemos muchas opciones para resolver la EDO en el tiempo que resulta de una discretización espacial. Si los pasos de tiempo son pequeños y limitados, los métodos explícitos pueden ser aceptables. Los solucionadores implícitos a menudo se aplican para resolver PDE parabólicas; el comportamiento difusivo de Euler implícito puede generar inexactitud, pero en términos de comportamiento parece similar a la difusión proporcionada por la ecuación del calor y puede ser aceptable incluso con pasos de tiempo bastante grandes. Las PDE hiperbólicas pueden requerir pasos implícitos para la estabilidad, pero los integradores avanzados, como los "integradores simplécticos", pueden evitar el suavizado excesivo causado por este tipo de pasos.

Un enfoque contrastante es escribir soluciones de sistemas semidiscretos f = L f en términos de vectores propios de L. Suponga que v1, . . . ,vn son vectores propios de L con valores propios λ1, . . . , λn y que sabemos f(0) = c1v1 + · · · + cnvn. Entonces, recuerda que la solución de f = L f está dada por:

$$f(t) = \sum_i c_i e^{\lambda_i t} v_i$$

Esta fórmula no es nada nuevo más allá de §5.1.2, que presentamos durante nuestra discusión sobre vectores propios y valores propios. Los vectores propios de L, sin embargo, pueden tener significado físico en el caso de una PDE semidiscreta, como en el ejemplo 14.5, que mostró que los vectores propios de los laplacianos L corresponden a diferentes vibraciones resonantes del dominio. Por lo tanto, este enfoque de vector propio se puede aplicar para desarrollar, por ejemplo, "aproximaciones de baja frecuencia" de los datos de valor inicial al truncar la suma anterior sobre i, con la ventaja de que la dependencia t se conoce exactamente sin saltos de tiempo.

Ejemplo 14.10 (Funciones propias del laplaciano). La figura NÚMERO muestra los vectores propios de la matriz L del ejemplo 14.7. Los vectores propios con valores propios bajos corresponden a funciones de baja frecuencia en [0, 1] con valores fijos en los extremos y pueden ser buenas aproximaciones de f(x) cuando es relativamente uniforme.

Métodos completamente discretos Alternativamente, podríamos tratar las variables de espacio y tiempo de manera más democrática y discretizarlas simultáneamente. Esta estrategia produce un sistema de ecuaciones para resolver más como §14.5.1. Este método es fácil de formular en paralelo con el caso elíptico, pero los sistemas de ecuaciones lineales resultantes pueden ser grandes si la dependencia entre los pasos de tiempo tiene un alcance global.

Ejemplo 14.11 (Difusión de calor completamente discreta). Explícito, implícito, Crank-Nicolson. No cubierto en CS 205A.

Es importante tener en cuenta que, al final, incluso los métodos semidiscretos pueden considerarse completamente discretos en el sentido de que el método ODE de pasos temporales aún discretiza la variable t; la diferencia radica principalmente en la clasificación de cómo se derivaron los métodos. Sin embargo, una ventaja de las técnicas semidiscretas es que pueden ajustar el paso de tiempo para t dependiendo de la iteración actual, por ejemplo, si los objetos se mueven rápidamente en una simulación física, podría tener sentido tomar más pasos de tiempo y resolver este movimiento. Algunos métodos incluso ajustan la discretización del dominio de los valores de x en caso de que se necesite más resolución cerca de discontinuidades locales u otros artefactos.

14.6 Método de los Elementos Finitos

No cubierto en 205A.

14.7 Ejemplos en la práctica

En lugar de un tratamiento riguroso de todas las técnicas PDE de uso común, en esta sección proporcionamos ejemplos de dónde aparecen en la práctica en informática.

14.7.1 Procesamiento de imágenes de dominio de gradiente

14.7.2 Filtrado que conserva los bordes

14.7.3 Fluidos basados en rejilla

14.8 Tareas pendientes

- Más sobre existencia/singularidad
- Condiciones de lámparas fluorescentes compactas
- Teorema de equivalencia de Lax
- Consistencia, estabilidad y amigos

14.9 Problemas

- Mostrar que 1d Laplaciano se puede factorizar como DD para la matriz de primera derivada D
- Resolver PDE de primer orden

Expresiones de gratitud

[La discusión va aquí]

Agradezco mucho a los estudiantes de CS 205A de Stanford, otoño de 2013, por detectar numerosos errores tipográficos y errores en el desarrollo de este libro. La siguiente es una lista sin duda incompleta de los estudiantes que contribuyeron a este esfuerzo: Tao Du, Lennart Jansson, Miles Johnson, Luke Knepper, Minjae Lee, Nisha Masharani, John Reyna, William Song, Ben-Han Sung, Martina Troesch, Ozhan Turgut, Patrick Ward, Joongyeub Yeo y Yang Zhao.

Un agradecimiento especial a Jan Heiland y Tao Du por ayudar a aclarar la derivación del algoritmo BFGS.

[Más discusión aquí]