Universidad Rey Juan Carlos

Práctica 4: Algoritmos voraces

Diseño y Análisis de Algoritmos

- Los códigos tendrán que probarse con el juez automático **DOMjudge**
  - gibson.escet.urjc.es
  - El nombre de usuario será "team-XXX", donde XXX es un número único por cada alumno. Podéis ver qué número os corresponde en un documento subido al aula virtual que describe la relación entre el nombre de usuario y el nombre de un alumno.
  - La contraseña es vuestro DNI (incluida la letra final en mayúsculas).
- Además de probar vuestros códigos con DOMjudge debéis subir el ficheros fuente al aula virtual
- No se entregará una memoria
- Fecha límite: Se especificará en el campus virtual
- 5 % de la nota final

Índice

1. Sistema de ecuaciones lineales - Descenso de gradiente (10 puntos)

1.1. Introducción

En este ejercicio tendréis que implementar un algoritmo de descenso de gradiente para hallar la solución, o una aproximación, de un sistema de ecuaciones lineales. El enunciado contiene una descripción matemática para aquellos alumnos que quieran comprender los detalles problema, y que se explicará en clase. No es necesario entender todos los detalles, pero sí será imprescindible comprender el algoritmo voraz, que es el que debéis implementar.

1.2. Enunciado del problema

Considérese el siguiente sistema de n ecuaciones lineales con m incógnitas (las variables x_j), donde $n \geqslant m$:

$$\left.\begin{array}{ccccccccc} a_{1,1}x_1 & + & a_{1,2}x_2 & + & \cdots & + & a_{1,m}x_m & = & b_1 \\ a_{2,1}x_1 & + & a_{2,2}x_2 & + & \cdots & + & a_{2,m}x_m & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n,1}x_1 & + & a_{n,2}x_2 & + & \cdots & + & a_{n,m}x_m & = & b_n \end{array}\right\} \tag{1}$$

El sistema se puede expresar de manera más concisa y compacta en notación matricial de la siguiente manera:

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{2}$$

En este caso, para $i = 1, \ldots, n$, y $j = 1, \ldots, m$ tenemos:

- $\mathbf{A}$ es una matriz de dimensiones $n \times m$, cuyos componentes son los elementos $a_{i,j}$. Se asumirá que el rango de la matriz $\mathbf{A}$ es igual a m (esto se va a cumplir en todos los casos de prueba, por lo que no es necesario comprobar este detalle técnico).
- $\mathbf{x}$ es un vector de dimensión m que representa las variables x_j.
- $\mathbf{b}$ es un vector de dimensión n, cuyos componentes son los elementos b_i.

El objetivo en este ejercicio es hallar los valores de $\mathbf{x}$ que satisfagan (2). Sin embargo, como puede haber más ecuaciones que incógnitas (si $n > m$), puede que el sistema no tenga solución. Por eso vamos a hallar una aproximación resolviendo el siguiente problema (convexo) de optimización:

$$\underset{\mathbf{x} \in \mathbb{R}^m}{\text{minimiza}} \quad \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2,$$

donde $\|\cdot\|^2$ es el módulo de un vector al cuadrado. Observad que si $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ es pequeño entonces el vector $\mathbf{A}\mathbf{x}$ será parecido al vector $\mathbf{b}$ (si el sistema tuviera

solución $\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2$ será igual a cero). Además, como $\|\mathbf{x}\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$, podemos reescribir el problema como:

$$\underset{\mathbf{x}\,\in\,\mathbb{R}^m}{\text{minimiza}} \quad \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} - 2\mathbf{b}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{b}. \tag{3}$$

En concreto, decimos que $f(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} - 2\mathbf{b}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{b}$ es la función objetivo del problema.

Hay varias formas de encontrar el mínimo de este problema de optimización. Una de ellas consiste en aplicar en algoritmo voraz de descenso de gradiente. Se trata de considerar el vector $\mathbf{x}_k$ como una posible solución en un cierto paso k, pero actualizarlo repetidas veces aplicando la fórmula:

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

En concreto, α es una constante positiva, y $\nabla f(\mathbf{x}_k)$ es el gradiente de la función objetivo para el vector $\mathbf{x}_k$. De esta manera, el método pretende disminuir el valor de la función objetivo en cada iteración k.

En general el gradiente de una función f para un vector $\mathbf{x}$ (que representa a j variables) se define de la siguiente manera:

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_j}\right)^{\mathrm{T}}. \tag{5}$$

Para el problema en (3) el gradiente en $\mathbf{x}$, expresado en notación matricial, es:

$$\nabla f(\mathbf{x}) = 2\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} - 2\mathbf{A}^{\mathrm{T}}\mathbf{b}. \tag{6}$$

Finalmente, el siguiente pseudocódigo describe el algoritmo voraz que debéis implementar:

```
1: function algoritmoVorazDescensoGradiente(A, b, x_0, α, ε)
2:     x ← x_0                                   ▷ Aproximación inicial
3:     g ← 2A^T Ax − 2A^T b                      ▷ Gradiente en x
4:     while ‖g‖ > ε do                          ▷ Iterar hasta que módulo del gradiente sea muy pequeño
5:         x ← x − αg                            ▷ Actualizar aproximación en dirección opuesta al gradiente
6:         g ← 2A^T Ax − 2A^T b                  ▷ Actualizar gradiente
7:     return x                                  ▷ Devolver aproximación final
```

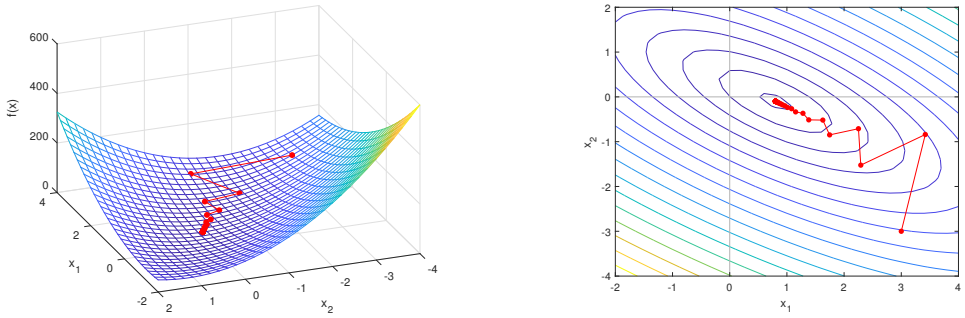
Los parámetros de entrada a la función son la matriz $\mathbf{A}$, el vector $\mathbf{b}$, un vector inicial $\mathbf{x}_0$, una constante positiva α, y otro valor positivo ε. Por ejemplo, para:

$$\mathbf{A} = \begin{pmatrix} 1 & -2 \\ 3 & 4 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \mathbf{x}_0 = \begin{pmatrix} 3 \\ -3 \end{pmatrix}, \quad \alpha = 0{,}03, \quad \varepsilon = 10^{-10},$$

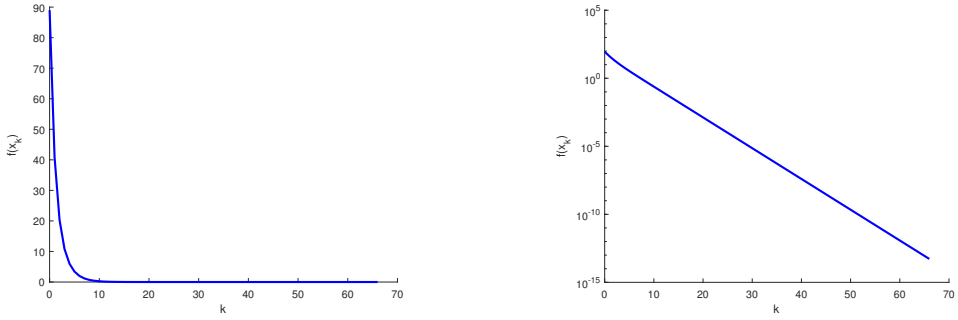
la solución es:

$$\mathbf{x} = \begin{pmatrix} \frac{4}{5} \\ -\frac{1}{10} \end{pmatrix}$$

Las siguientes gráficas ilustran este enfoque. Se empieza en el punto $\mathbf{x}_0 = (3, -3)$ que no es la solución del problema. Pero aplicando (4) repetidas veces se obtienen mejores soluciones progresivamente, hasta que el proceso converge al mínimo de la función objetivo $f(\mathbf{x})$. En este caso, como el sistema de ecuaciones tiene solución, la función objetivo llega a ser prácticamente 0. Esto se aprecia en las dos gráficas inferiores, en la que se ha usado también una escala logarítmica en el eje de las Y para apreciar mejor cómo desciende $f(\mathbf{x}_k)$.



Evolución de las aproximaciones, en 3d, y en 2d con curvas de nivel (isolíneas) de $f(\mathbf{x})$.



Evolución de la función objetivo con escala Y lineal y logarítmica. En este caso hay solución exacta, por lo que $f(\mathbf{x}_k)$ tiende a 0 a medida que aumenta k.

1.2.1. Descripción de la entrada

La primera línea de la entrada contendrá el número de ecuaciones $0 < n \leqslant 10$ e incógnitas $0 < m \leqslant 10$, donde $n \geqslant m$ (no es necesario comprobar que se cumplan las precondiciones). A continuación se especifica la matriz $\mathbf{A}$ mediante n filas, cada una con m enteros, separados por espacios. Después se especifica el vector $\mathbf{b}$, con sus n elementos en una sola fila, los cuales también serán enteros. A continuación la entrada contiene m valores reales del vector inicial $\mathbf{x}_0$. La siguiente línea contiene el valor real de $\alpha > 0$, y la última línea el valor de $\varepsilon > 0$.

1.2.2. Descripción de la salida

La salida contendrá los m valores reales del vector solución $\mathbf{x}$, separados por espacios. Cada componente de $\mathbf{x}$ se escribirá con 4 cifras decimales exactamente (se

redondearán según el valor de la quinta cifra decimal). Por último, se escribirá un salto de línea.

Ejemplo de entrada 1

```
2 2↵
1 -2↵
3 4↵
1 2↵
3 -3↵
0.03↵
0.0000000001↵
```

Salida para el ejemplo de entrada 1

```
0.8000 -0.1000↵
```

Ejemplo de entrada 2

```
3 2↵
1 2↵
3 4↵
5 6↵
1 -1 0↵
3.5 1↵
0.01↵
0.0000000001↵
```

Salida para el ejemplo de entrada 1

```
-1.0000 0.7500↵
```