Computación de Altas Prestaciones-2022-23 Sesión 2

Computación de Altas Prestaciones

- Operaciones vectoriales y matriciales básicas:

1) Notación MATLAB
2) Repaso de operaciones básicas, matriciales y vectoriales.
3) Producto matriz por matriz; diferentes versiones.

Escalares, Vectores y Matrices: Notación

- Escalares: suelen representarse por letras griegas o con subíndices.
- Vectores: suelen representarse con letras latinas minúsculas

$$a \in \mathbb{R}^n \quad o \quad a \in \mathbb{R}^{n\times 1}$$

$$a = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}; \qquad b^T = \begin{pmatrix} b_1 & b_2 & \cdots & b_n \end{pmatrix}$$

Notación

- Notación para matrices:

$$A \in \Re^{m\times n} \leftrightarrow A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \ddots & \vdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,m} \end{pmatrix},\ a_{i,j} \in \Re,$$

- Notación para vectores:

$$v \in \Re^{m} \leftrightarrow v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}, v_i \in \Re$$

- Por defecto se supone que los vectores son COLUMNA

Notación ':', vectores(Matlab)

- ejemplo:

  1:5 es equivalente al vector [1 2 3 4 5].

- ejemplo:

  0.2:0.2:1.2 es [0.2 0.4 0.6 0.8 1.0 1.2].

- ejemplo:

  5:-1:1 es [5 4 3 2 1].

También vimos en el seminario 1 como usarlo en programación, para los bucles "for".

Notación ‘:’, Submatrices

- También sirve para referenciar submatrices o segmentos de vector:
  - ejemplo: A(1:4,3) Es el vector columna formado por las cuatro primeras entradas de la tercera columna de la matriz A.
  - ejemplo: A(:,3) Es el vector columna formado por la tercera columna de la matriz A.
  - ejemplo: A(:,[2 4]) Son las columna 2 y 4 de la matriz A.
- Estas expresiones pueden usarse en ambos lados de una asignación.

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Notación ':', Submatrices

- Sea la matriz A:

  `>> A = [1:4;5:8;9:12;13:16];`

La submatriz A([2 4],2:4) es la siguiente:

`>> A([ 2 4],2:4)`

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

```
ans =

     6     7     8
    14    15    16
```

Coste de operaciones matriciales

- Para el cálculo del coste de algoritmos matriciales se suele usar como unidad el número de FLOPS: (Floating Points operations) {+,-,*,/}

◆ Tiempo de ejecución de un Flop:

Tiempo que tarda en ejecutarse, por término medio, una operación elemental en un computador

- Depende de:
  - El computador
  - El algoritmo
  - La implementación
- Construyendo algoritmos de coste determinado (test Linpack)podemos evaluar la velocidad de un computador (número de flops/unidad de tiempo)

Operaciones vectoriales básicas

Operaciones Vectoriales Básicas: ($\alpha \in \Re$, $x,y,z \in \Re^n$), involucrando sólo vectores y escalares:

-Multiplicación escalar-vector: $z = \alpha\, x$ $\quad (z_i = \alpha\, x_i)$

-Suma de vectores: $z = y + x$ $(z_i = y_i + x_i)$

-Producto escalar (Producto interno): $\alpha = y^t x$

$$\rightarrow \alpha = \sum_{i=1}^{n} x_i y_i$$

Operaciones vectoriales básicas

Algoritmo para Producto interno: ($\alpha \in \Re$, $x,y \in \Re^n$)

```
α=0
for i=1:n
    α = x_i*y_i +α
end
```

Multiplicación vectorial o de Hadamard: z=y.*x

($z_i=y_i \cdot x_i$)

Operaciones vectoriales básicas

-Saxpy (scalar a x plus y): y= ax+y   ($y_i = ax_i + y_i$)

```
for i=1:n
    y_i = α*x_i+y_i
end
```

$$\begin{bmatrix} \\ y \\ \\ \end{bmatrix} = a \cdot \begin{bmatrix} \\ x \\ \\ \end{bmatrix} + \begin{bmatrix} \\ y \\ \\ \end{bmatrix}$$

¿Cuál es el coste de estas operaciones?

Operaciones Matriciales Básicas

Multiplicación Matriz-vector (Gaxpy): $x \in \Re^n$,$y \in \Re^m$,$A \in \Re^{m \times n}$

y=A*x;

En formato de Actualización: y=Ax+y

$$y_i = \sum_{j=1}^{n} A_{i,j} x_j + y_i \,; \qquad i = 1, \cdots, m$$

Se puede considerar como “saxpy” generalizado o “gaxpy”.

$$\begin{pmatrix} y \end{pmatrix} = \begin{pmatrix} A \end{pmatrix} \cdot \begin{pmatrix} x \end{pmatrix} + \begin{pmatrix} y \end{pmatrix}$$

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Operaciones Matriciales Básicas

Dos versiones:

1) Componente a componente o “Gaxpy” por filas

```
for i=1:m
    for j=1:n
        y_i=y_i+A_i,j*x_j
    end
end
```

$$y_i = y_i + A_{i,j} * x_j$$

Operaciones Matriciales Básicas

Segunda versión: por columnas

for j=1:n
    for i=1:m
        $y_i=y_i+A_{i,j}*x_j$
    end
end

Operaciones Matriciales Básicas: Producto Matriz-Vector

$$\begin{pmatrix} 1 & 2 \\ 3 & 5 \\ 4 & 6 \end{pmatrix} \begin{pmatrix} 7 \\ 8 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 \cdot 7 + 2 \cdot 8 \\ 3 \cdot 7 + 5 \cdot 8 \\ 4 \cdot 7 + 6 \cdot 8 \end{pmatrix}}_{\text{Por filas}} = \underbrace{\begin{pmatrix} 1 \\ 3 \\ 4 \end{pmatrix} \cdot 7 + \begin{pmatrix} 2 \\ 5 \\ 6 \end{pmatrix} \cdot 8}_{\text{Por columnas}} = \begin{pmatrix} 23 \\ 61 \\ 76 \end{pmatrix}$$

Por filas

Por columnas

¿Cuál sería más eficiente?

Operaciones Matriciales Básicas

Aplicación de la notación “Dos Puntos” al Gaxpy

Por filas

```
for i=1:m
        y(i)=y(i)+A(:,i)*x
end
```

Producto interno

Por columnas

```
for j=1:m
        y=y+x(j)*A(j,:)
end
```

Saxpy

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Operaciones Matriciales Básicas

Producto Externo: $x \cdot y^T$

$$\begin{pmatrix}1\\2\\3\end{pmatrix}\begin{pmatrix}4 & 5\end{pmatrix} = \begin{pmatrix}4 & 5\\8 & 10\\12 & 15\end{pmatrix}$$

En forma de Update $A=A+xy^T$:

```
for i=1:m
        for j=1:n
                A(i,j)=A(i,j)+x(i)*y(j)
        end
end
```

Tambien tiene versión por columnas

Operaciones Matriciales Básicas

Producto de Matrices: C=A·B+C

$$A \in \mathbb{R}^{m,p},\ \mathrm{B} \in \mathbb{R}^{p,n}, C \in \mathbb{R}^{m,n};$$

$$\mathrm{C{=}C{+}A{\cdot}B} \leftrightarrow\ c_{i,j} = c_{i,j} + \sum_{k=1}^{p} a_{i,k}\, b_{k,j}$$

$$\begin{pmatrix} \\ \mathrm{C} \\ \\ \end{pmatrix} = \begin{pmatrix} \\ \mathrm{A} \\ \\ \end{pmatrix} \begin{pmatrix} \\ \mathrm{B} \\ \\ \end{pmatrix} + \begin{pmatrix} \\ \mathrm{C} \\ \\ \end{pmatrix}$$

Recordemos que para poder hacer el producto de dos matrices, el número de columnas de la primera matriz debe ser igual al número de filas de la segunda matriz

Operaciones Matriciales Básicas

- El producto escalar de dos vectores es también un producto matricial
- El Gaxpy o producto matriz por vector también es un producto de matrices
- El producto externo (vector por vector, pero con resultado una matriz) también es un producto de matrices

¿Cuál es el coste en flops de esta operación?

Operaciones Matriciales Básicas

Producto de Matrices: C=A·B+C

$$A \in \Re^{m\times n}, B \in \Re^{n\times p}, C \in \Re^{m\times p}; \quad C = C + A \cdot B \leftrightarrow c_{i,j} = c_{i,j} + \sum_{k=1}^{n} a_{i,k} b_{k,j}$$

```
for i=1:m
    for j=1:p
        for k=1:n
            C(i,j)=C(i,j)+A(i,k)*B(k,j)
        end
    end
end
```

Producto Escalar

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1\cdot 5 + 2\cdot 7 & 1\cdot 6 + 2\cdot 8 \\ 3\cdot 5 + 4\cdot 7 & 3\cdot 6 + 4\cdot 8 \end{pmatrix} = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

Operaciones Matriciales Básicas

```
for j=1:p
      for i=1:m
            for k=1:n
                  C(i,j)=C(i,j)+A(i,k)*B(k,j)
            end
      end
end
```

Producto Matriz vector

$$C(:,1) = C(:,1) + A * B(:,1)$$

$$C(:,2) = C(:,2) + A * B(:,2)$$

$$\ldots$$

La columna i-ésima de la matriz C se puede obtener como producto de la matriz A por la columna i-ésima de B

Operaciones Matriciales Básicas

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \left[\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 5 \\ 7 \end{pmatrix}, \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 6 \\ 8 \end{pmatrix}\right] = \cdots$$

El producto matriz-vector (gaxpy) se puede expresar por filas (basado en producto escalar) o por columnas (basado en saxpy):

Operaciones Matriciales Básicas

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \left(5\begin{bmatrix} 1 \\ 3 \end{bmatrix} + 7\begin{bmatrix} 2 \\ 4 \end{bmatrix},\quad 6\begin{bmatrix} 1 \\ 3 \end{bmatrix} + 8\begin{bmatrix} 2 \\ 4 \end{bmatrix} \right) = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

```
for j=1:p
        for k=1:n
                for i=1:m
                        C(i,j)=C(i,j)+A(i,k)*B(k,j)
                end
        end
end
```

saxpy

Operaciones Matriciales Básicas

Pregunta 1: ¿Cuántas versiones del producto matriz por matriz podemos construir ? ¿Tienen todas el mismo coste?

Pregunta 2: ¿Cuál es la operación interna que se realiza en cada versión?

Pregunta 3: ¿Cómo se accede a cada matriz en el bucle mas interno de cada versión? Si la matriz se almacena internamente por filas, ¿Cuál o cuales serán las versiones más eficientes?

Operaciones Matriciales Básicas

Versión **kij**:

```
for k=1:n
        C=C+A(:,k)*B(k,:)
end
```

El producto A·B se puede considerar como la suma de **n** productos externos

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix} \begin{pmatrix} 5 & 6 \end{pmatrix} + \begin{pmatrix} 2 \\ 4 \end{pmatrix} \begin{pmatrix} 7 & 8 \end{pmatrix} = \begin{pmatrix} 5 & 6 \\ 15 & 18 \end{pmatrix} + \begin{pmatrix} 14 & 16 \\ 28 & 32 \end{pmatrix}$$

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Operaciones Matriciales Básicas “Nivel” de cada operación

	Cuantos datos?	Cuantos flops?	Nivel
Saxpy o Prod. int	O(n)	O(n)	Nivel 1
Gaxpy	O(n^2)	O(n^2)	Nivel 2
Producto de Matrices	O(n^2)	O(n^3)	Nivel 3

Operaciones Matriciales Básicas “OverWriting”

Overwriting: Devolver el resultado de un cálculo (o subrutina) sobreescribiendo alguno de los argumentos de entrada

-Ventajas:
1) Menos Memoria
2) Posiblemente, menos accesos a memoria: Mas rápido

-Desventajas: Puede ser necesario espacio “extra” de trabajo

Ejemplo: Producto Matriz-Matriz C=A·B+C, versión **jki,** suponiendo que las tres matrices son del mismo tamaño, n*n; Deseamos que la matriz de entrada **B** sea sobre-escrita con la de salida, **C.**

Operaciones Matriciales Básicas "OverWriting"

```
C(1:n,1:n)=0
for j=1:n
        for k=1:n
                C(:,j)=C(:,j)+A(:,k)*B(k,j)
        end
end
```

↓ ?

```
for j=1:n
        for k=1:n
                B(:,j)=B(:,j)+A(:,k)*B(k,j)
        end
end
```

NO

Operaciones Matriciales Básicas "OverWriting"

Hace falta "espacio de trabajo" (workspace) para guardar la j-ésima columna de B hasta que sea "seguro" sobreescribirla

```
for j=1:n
        w(1:n)=0
        for k=1:n
                w(:)=w(:)+A(:,k)*B(k,j)
        end
        B(:,j)=w(:)
end
```

El espacio de trabajo en este caso es de dimensión **n**, mucho más pequeño que la matriz C.