MSBD 5007 HW1

RONG Shuo

February 28, 2025

Question1

Given two vectors $\boldsymbol{a} = [a_0, a_1, \cdots, a_{N-1}]$ and $\boldsymbol{b} = [b_0, b_1, \cdots, b_{N-1}]^T$, their *circular convolution* is defined by

$$(\boldsymbol{a} \circledast \boldsymbol{b})_k = \sum_{j=0}^{N-1} a_j b_{k-j}, k = 0, 1, \cdots, N-1,$$

where $\boldsymbol{b}$ is extended periodically, i.e., $b_{k-j} = b_{(k-j)+N}$ if $-N \le k-j \le -1$. Let $\boldsymbol{f}, \boldsymbol{g}$, and $\boldsymbol{h}$ be vectors in $\mathbb{R}^N$. Prove that the circular convolution satisfies:
(a) $\boldsymbol{f} \circledast \boldsymbol{g} = \boldsymbol{g} \circledast \boldsymbol{f}$.
(b) $\boldsymbol{f} \circledast (\boldsymbol{g} \circledast \boldsymbol{h}) = (\boldsymbol{f} \circledast \boldsymbol{g}) \circledast \boldsymbol{h}$.

(a)

$$\begin{aligned}
&(\boldsymbol{f} \circledast \boldsymbol{g})_k \\
&= \sum_{j=0}^{N-1} f_j g_{k-j} \\
&= \sum_{j=k+1}^{N-1} f_j g_{k-j} + \sum_{j=0}^{k} f_j g_{k-j} \\
&= \sum_{j=k+1}^{N-1} f_j g_{N+k-j} + \sum_{j=0}^{k} f_j g_{k-j} \\
&= \sum_{i=k+1}^{N-1} f_{N+k-i} g_i + \sum_{j=0}^{k} f_j g_{k-j} \\
&= \sum_{i=k+1}^{N-1} g_i f_{k-i} + \sum_{i=0}^{k} f_{k-i} g_i \\
&= \sum_{i=k+1}^{N-1} g_i f_{k-i} + \sum_{i=0}^{k} g_i f_{k-i} \\
&= \sum_{i=0}^{N-1} g_i f_{k-i} \\
&= (\boldsymbol{g} \circledast \boldsymbol{f})_k
\end{aligned}$$

Therefore, we can conclude that: $\boldsymbol{f} \circledast \boldsymbol{g} = \boldsymbol{g} \circledast \boldsymbol{f}$.

(b)

$$
\begin{aligned}
(\boldsymbol{f} \circledast (\boldsymbol{g} \circledast \boldsymbol{h}))_k &= (\boldsymbol{f} \circledast (\boldsymbol{h} \circledast \boldsymbol{g}))_k \\
&= \sum_{j=0}^{N-1} f_j (\boldsymbol{h} \circledast \boldsymbol{g})_{k-j} \\
&= \sum_{j=0}^{N-1} f_j \sum_{i=0}^{N-1} h_i g_{k-j-i} \\
&= \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} f_j h_i g_{k-j-i} \\
&= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f_j h_i g_{k-j-i} \\
&= \sum_{i=0}^{N-1} h_i \sum_{j=0}^{N-1} f_j g_{k-j-i} \\
&= \sum_{i=0}^{N-1} h_i (\boldsymbol{f} \circledast \boldsymbol{g})_{k-i} \\
&= (\boldsymbol{h} \circledast (\boldsymbol{f} \circledast \boldsymbol{g}))_k \\
&= ((\boldsymbol{f} \circledast \boldsymbol{g}) \circledast \boldsymbol{h})_k
\end{aligned}
$$

Therefore, we can conclude that: $\boldsymbol{f} \circledast (\boldsymbol{g} \circledast \boldsymbol{h}) = (\boldsymbol{f} \circledast \boldsymbol{g}) \circledast \boldsymbol{h}$.

Question2

Let $\boldsymbol{A} = \begin{bmatrix} 2 & -1 & 3 \\ 1 & 2 & 1 \\ -3 & -1 & 2 \end{bmatrix}$ be a 3×3 matrix.

(a) Find the LU decomposition of the matrix $\boldsymbol{A}$. The final result will look like this:

$$
\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}
$$

(b) Use the result in (a) to solve the system:

$$
\begin{aligned}
2x_1 - x_2 + 3x_3 &= 3 \\
x_1 + 2x_2 + x_3 &= 4 \\
-3x_1 - x_2 + 2x_3 &= 5
\end{aligned}
$$

(a)

$$
\begin{bmatrix} 2 & -1 & 3 \\ 1/2 & 5/2 & -1/2 \\ -3/2 & -5/2 & 13/2 \end{bmatrix}_{k=1}
$$

$$
\begin{bmatrix} 2 & -1 & 3 \\ 1/2 & 5/2 & -1/2 \\ -3/2 & -1 & 6 \end{bmatrix}_{k=2}
$$

From this, we can get:

$$
A = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -3/2 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & -1 & 3 \\ 0 & 5/2 & -1/2 \\ 0 & 0 & 6 \end{bmatrix}
$$

(b)

From (a), we know:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -3/2 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & -1 & 3 \\ 0 & 5/2 & -1/2 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

Assume $\boldsymbol{LUx} = \boldsymbol{b}$ as $\boldsymbol{Ly} = \boldsymbol{b}$, s.t.: From (a), we know:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -3/2 & -1 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}$$

From these, we can solve:

$$\begin{aligned} y_1 &= 3 \\ y_2 &= 5/2 \\ y_3 &= 12 \end{aligned}$$

s.t.

$$\begin{bmatrix} 2 & -1 & 3 \\ 0 & 5/2 & -1/2 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 5/2 \\ 12 \end{bmatrix}$$

From these, we can solve:

$$\begin{aligned} x_1 &= -4/5 \\ x_2 &= 7/5 \\ x_3 &= 2 \end{aligned}$$

Question3

Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ be a tri-diagonal matrix(i.e. $a_{ij} = 0$ if $|i - j| > 1$). The pattern of nonzero entries is illustrated below:

$$\begin{bmatrix} \times & \times & & & \\ \times & \times & \times & & \\ & \ddots & \ddots & \ddots & \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix}$$

Develop an algorithm with complexity $O(n)$ to compute the LU decomposition of $\boldsymbol{A}$, assuming all the pivots are non-zero.

Algorithm 1 LU Decomposition

for k = 1 to n **do**
  $a_{k,k+1} \leftarrow \frac{a_{k,k+1}}{a_{k,k}}$
  $a_{k+1,k+1} \leftarrow a_{k+1,k+1} - a_{k,k+1} \cdot a_{k+1,k}$
end for

According to the LU decomposition algorithm, we can get the updated matrix $\boldsymbol{A}^*$, and we can get the matrix $\boldsymbol{L}$ and $\boldsymbol{U}$ as follows:

$$\boldsymbol{L} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ a_{21}^* & 1 & 0 & 0 & 0 & \cdots & 0 \\ a_{31}^* & a_{32}^* & 1 & 0 & 0 & \cdots & 0 \\ a_{41}^* & a_{42}^* & a_{43}^* & 1 & 0 & \cdots & 0 \\ a_{51}^* & a_{52}^* & a_{53}^* & a_{54}^* & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}^* & a_{n2}^* & a_{n3}^* & a_{n4}^* & a_{n5}^* & \cdots & 1 \end{bmatrix}$$

and

$$\boldsymbol{U} = \begin{bmatrix} a_{11}^* & a_{12}^* & a_{13}^* & a_{14}^* & a_{15}^* & \cdots & a_{1n}^* \\ 0 & a_{22}^* & a_{23}^* & a_{24}^* & a_{25}^* & \cdots & a_{2n}^* \\ 0 & 0 & a_{33}^* & a_{34}^* & a_{35}^* & \cdots & a_{3n}^* \\ 0 & 0 & 0 & a_{44}^* & a_{45}^* & \cdots & a_{4n}^* \\ 0 & 0 & 0 & 0 & a_{55}^* & \cdots & a_{5n}^* \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & a_{nn}^* \end{bmatrix}$$

Therefore, the algorithm has a complexity of $O(n)$.

Question4

To accelerate matrix multiplications, the *Coppersmith Winograd* algorithm reduces the number of scalar multiplications by cleverly reformulating the inner product. Assume that n is even and define, for any vector $x \in \mathbb{R}^n$,

$$f(\boldsymbol{x}) = \sum_{i=1}^{n/2} x_{2i-1}x_{2i}$$

.

(a) Prove that for all vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, the inner product can be re-expressed as

$$\boldsymbol{x}^T\boldsymbol{y} = \sum_{i=1}^{n/2}((x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1})) - f(\boldsymbol{x}) - f(\boldsymbol{y}).$$

(b) Now consider the matrix product $\boldsymbol{C} = \boldsymbol{AB}$, where $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}$. Devise an algorithm to compute $\boldsymbol{C}$ using $\frac{n^3}{2} + O(n^2)$ scalar multiplications.

Note: A standard matrix multiplication requires n^3 scalar multiplications. By combining this method with other techniques, one can obtain the Coppersmith Winograd algorithm, which has an asymptotic complexity of $O(n^{2.375})$.

(a)

$$\begin{aligned}
&\sum_{i=1}^{n/2}((x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1})) \\
&= \sum_{i=1}^{n/2} x_{2i-1}x_{2i} + y_{2i}x_{2i} + x_{2i-1}y_{2i-1} + y_{2i}y_{2i-1} \\
&= \sum_{i=1}^{n/2} x_{2i-1}x_{2i} + \sum_{i=1}^{n/2} y_{2i}x_{2i} + \sum_{i=1}^{n/2} x_{2i-1}y_{2i-1} + \sum_{i=1}^{n/2} y_{2i}y_{2i-1} \\
&= \boldsymbol{x}^T\boldsymbol{y} + f(\boldsymbol{x}) + f(\boldsymbol{y})
\end{aligned}$$

So we can conclude that:

$$\boldsymbol{x}^T\boldsymbol{y} = \sum_{i=1}^{n/2}((x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1})) - f(\boldsymbol{x}) - f(\boldsymbol{y}).$$

(b)

For each row $\boldsymbol{a}_i \in \boldsymbol{A}$, calculating $f(\boldsymbol{a}_i)$ costs $\frac{n}{2}$ scalar multiplications. Therefore, the total cost of calculating $f(\boldsymbol{a}_i), \forall \boldsymbol{a}_i \in \boldsymbol{A}$ is $n \times \frac{n}{2} = \frac{n^2}{2}$ scalar multiplications. Similarly, the total cost of calculating $f(\boldsymbol{b}_i), \forall \boldsymbol{b}_i \in \boldsymbol{B}$ is $\frac{n^2}{2}$ scalar multiplications, too.
For each multiplication between row in $\boldsymbol{A}$ and column in $\boldsymbol{B}$, we need to calculate $\frac{n}{2}$ times multiplication, if the $f(\boldsymbol{a}_i)$ and $f(\boldsymbol{b}_i)$ is given. Therefore, the total cost of calculating $\boldsymbol{A} \times \boldsymbol{B}$ is $n \times n \times \frac{n}{2} + O(n^2) = \frac{n^3}{2} + O(n^2)$ scalar multiplications.
Therefore, the algorithm to compute $\boldsymbol{C} = \boldsymbol{AB}$ is as follows:

Algorithm 2 Matrix Multiplication

Input: Matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}$
Output: Matrix $\boldsymbol{C} = \boldsymbol{AB}$
Compute $f(\boldsymbol{a}_i)$ for each row $\boldsymbol{a}_i$ in $\boldsymbol{A}$
Compute $f(\boldsymbol{b}_i)$ for each column $\boldsymbol{b}_i$ in $\boldsymbol{B}$
for each row $\boldsymbol{a}_i$ in $\boldsymbol{A}$ **do**
  for each column $\boldsymbol{b}_j$ in $\boldsymbol{B}$ **do**
    Compute $\sum_{k=1}^{n/2}((a_{i,2k-1} + b_{2k,j})(a_{i,2k} + b_{2k-1,j}))$
    Compute $c_{i,j} = \sum_{k=1}^{n/2}((a_{i,2k-1} + b_{2k,j})(a_{i,2k} + b_{2k-1,j})) - f(\boldsymbol{a}_i) - f(\boldsymbol{b}_j)$
  end for
end for
return $\boldsymbol{C}$