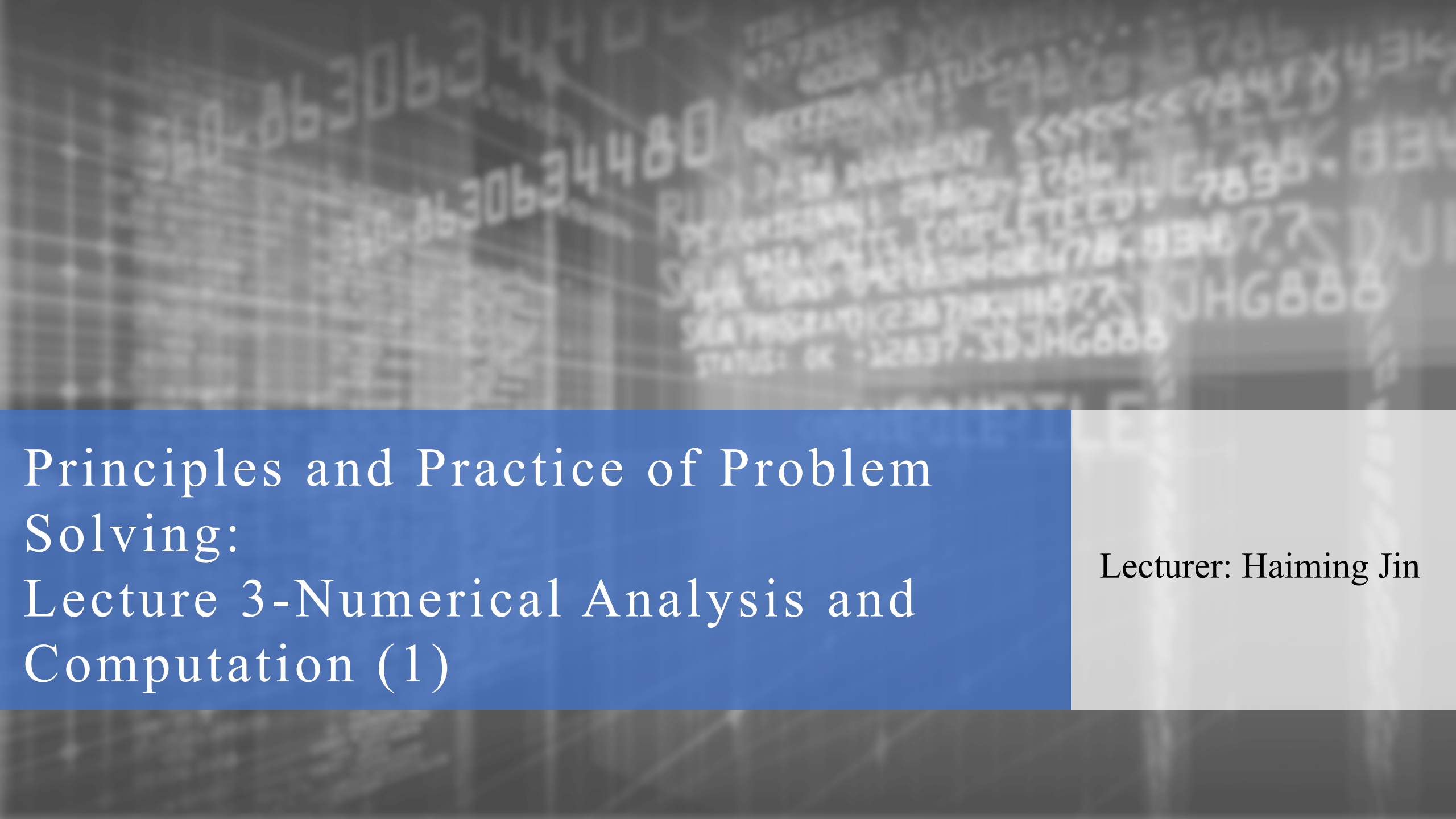


Principles and Practice of Problem Solving: Lecture 3-Numerical Analysis and Computation (1)

Lecturer: Haiming Jin

Overview

Topics covered

- Number-theoretic problems
- Polynomial arithmetic
- Polynomial interpolation
- Solving nonlinear equations
- Solving linear systems

Topics not covered

- Numerical differentiation and integration
- Solving partial/ordinary differential equations
- Many others (eigenvalue problems, evaluation of matrix functions, etc.)

Number-Theoretic Problems

Number-theoretic problems include

- divisibility
- modular arithmetic
- even and odd number
- prime and composite number
- greatest common divisor (GCD)
- least common multiple (LCM)
- ...

Find the LCM of two positive integers.

Find all prime numbers smaller than a positive integer.

Find the LCM of Two Positive Integers

Example: Find the LCM of 8 and 20

Method 1: Enumeration

List all multiples of 8 in increasing order, check whether each of them is also a multiple of 20, find the smallest one in the list that is also a multiple of 20.

Check: 8, 16, 24, 32, …

Method 2: Prime Factorization

- List the prime factors of each number.
- Multiply each factor the greatest number of times it occurs in either number.

$8 = 2 \times 2 \times 2$

$20 = 2 \times 2 \times 5$

The LCM of 8 and 10 is: $2 \times 2 \times 2 \times 5 = 40$

Find the LCM of Two Positive Integers

Pseudocode of Method 1

```
A=8,  B=20
k=A
i=1
while  (  k%B  ! = 0  )
{
    k =i*A
    i=i+1
}
output k
```

Find the LCM of Two Positive Integers

Method 2: Prime Factorization

Find the prime factors in **an increasing order**, i.e., check whether A and B have factors 2, 3, 5, 7, …

A=6 =2×3
B=10=2×5

A=6/2=3
B=10/2=5

A=3/3 =1
B=5 (fixed)

A=1 (fixed)
B=5/5 =1

A&B have factor 2
Let P0=2

A has factor 3
Let P1=3

B has factor 5
Let P2=5

LCM = P0 × P1× P2 = 30

A=8 =2×2×2
B=20=2×2×5

A=8/2=2 ×2
B=10/2=2×5

A=2 ×2
B=10/5 =2

A&B have factor 2
Let P0=2

Neither has factor 3, but B has factor 5
Let P1=5

LCM ? = P0 × P1=10

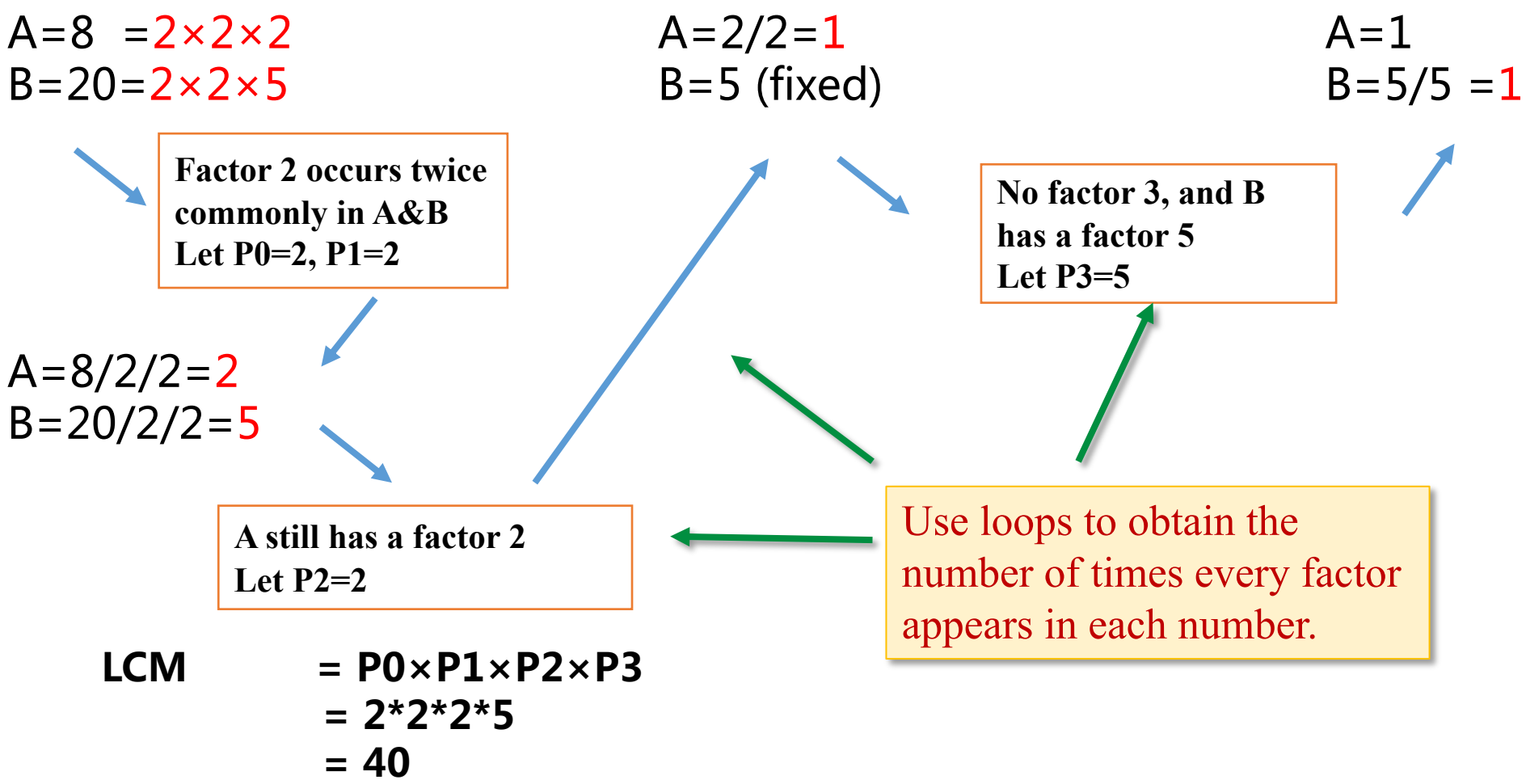
Reason for Incorrectness: The number of times every factor appears in each number is not correctly taken into account.

We could use loops to obtain the number of times every factor appears in each number.

Find the LCM of Two Positive Integers

Method 2: Prime Factorization

Find the prime factors in **an increasing order**, i.e., check whether A and B have factors 2, 3, 5, 7, …



LCM = P0×P1×P2×P3
= 2*2*2*5
= 40

Find the LCM of Two Positive Integers

Method 2: Prime Factorization

```
i = 0 , C=max (A,B)        // p[i] records all factors
for (k=2;  k<=C;  k++) {
        while (k is a factor of A&B)  {
              p[i] = k;
             A=A/k;  B=B/k;  i++;
        }
        while (k is a factor of A)  {
              p[i] = k;  A=A/k ;
             i++;
        }
        while (k is a factor of B)  {
              p[i] = k;  B=B/k ;
             i++;
        }
}
```

Calculate p[0]*p[1]*…*p[i-1] to get the LCM

After class:

Better ways to find the LCM?

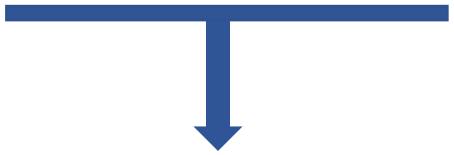
Find all Prime Numbers Less Than a Positive Integer N

Problem: Input a positive integer N, output all prime numbers less than N

Naive Method

for (k from 2 to N-1)

{Check whether k is a prime number}



```
for (i from 2 to k/2)
{
    if (k%i ==0)  then terminate the loop
}
if    the above loop terminates before i=k/2
      then k is a composite number
else
      k is a prime number
```

We check from 2 to k/2 whether there is a factor of k.

Find all Prime Numbers Less Than a Positive Integer N

Sieve of Eratosthenes

Feature: More efficiently find the prime numbers in [2, N] than the naive method.

Algorithm:

In the integer interval [2, N],

① Keep 2, and remove other multiples of 2,

② Keep 3, and remove other multiples of 3,

③ Keep 5, and remove other multiples of 5,

④ Carry out such operation until the multiples of prime numbers less than N have all been removed, and the remaining numbers are the prime number less than N

Find all Prime Numbers Less Than a Positive Integer N

Example of Sieve of Eratosthenes

Find all prime numbers in [2, 20]

Construct a list [2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20]

- Step 1: Find the 1st non-zero number 2 (prime number), set other multiples of 2 as 0 (i.e., remove from the list). Get

  [2 3 0 5 0 7 0 9 0 11 0 13 0 15 0 17 0 19 0]

- Step 2: Find the 2nd non-zero number 3 (prime number), set other multiples of 3 as 0 (i.e., remove from the list). Get

  [2 3 0 5 0 7 0 0 0 11 0 13 0 0 0 17 0 19 0]

- Step 3: Find the 3rd non-zero number 5 (prime number), set other multiples of 5 as 0 (i.e., remove from the list). Get

  [2 3 0 5 0 7 0 0 0 11 0 13 0 0 0 17 0 19 0]

- ...

Find all Prime Numbers Less Than a Positive Integer N

C++ Implementation of Sieve of Eratosthenes

```
int prime[N];        // define a length-N array, where N is a constant
                             defined somewhere before
void PrimeTable( int N ) {
    int i, j;
    for(i = 2; i <= N; i++)       // construct list [ 2 3 4 5 6 7 8 9 10 …]
        prime[i] = i;
    for(i = 2;  i< N; i++)
    {
        if( prime[i]!=0 )
            for(j = i+i; j <= N; j+=i)    // set 2*i, 3*i, 4*i … as 0
                prime[j] = 0;
    }
}
```

Polynomial Arithmetic

- We discuss the multiplication and division of polynomials with a single variable.
- Multiplication

  Multiply polynomials $A(x) = a_0 + a_1 x + \cdots + a_n x^n$ and $B(x) = b_0 + b_1 x + \cdots + b_m x^m$

  , and store the result in polynomial $C(x) = c_0 + c_1 x + \cdots + c_k x^k$.
- Division

  Divide $\mathrm{P}(x) = p_0 + p_1 x + \cdots + p_{\mathrm{m}} x^m$ by $\mathrm{Q}(x) = q_0 + q_1 x + \cdots + q_n x^n$, and obtain quotient $\mathrm{T}(x) = t_0 + t_1 x + \cdots + t_{m-n} x^{m-n}$ and remainder $\mathrm{R}(x) = r_0 + r_1 x + \cdots + r_k x^k$.

Polynomial Arithmetic

- Above all, we have to address the problem of storing polynomials in a computer.
- **Intuitive way:** Store the coefficients of $x^0, x^1, \cdots, x^n$ in the positions of an array with index 0,1,…,n respectively. Then, the **elements** in the array are the **coefficients** and the **indices** correspond to the **powers** .

$1 + 2x + 0x^2 + x^3$ 

0	1	2	3
1	2	0	1

- By such method of storing polynomials, we transform polynomial addition and subtraction into operations on the corresponding arrays.
- How about polynomial multiplication and division?

Polynomial Multiplication

Given $A(x) = a_0 + a_1 x + \cdots + a_n x^n$ and $B(x) = b_0 + b_1 x + \cdots + b_m x^m$

$$C(x) = A(x) \times B(x) = c_0 + c_1 x + \cdots + c_k x^k$$

We need to multiple all coefficients of $A(x)$ with all coefficients of $B(x)$. Thus, we have $(m+1)\times(n+1)$ terms in total, i.e., $a_i \times b_j$ for each $i = 0,1,\ldots, n$, $j = 0,1,\ldots, m$.

```
for(i = 0;  i <= n;  i++)
    for( j = 0;  j <= m;  j++)
        get a[i]*b[j] to calculate the coefficients of C(x);
```

Polynomial Multiplication

As a part of the coefficient of x^{i+j}, $a_i \times b_j$ should be summed up to get the coefficient c_{i+j}.

Code for polynomial multiplication:

```
…
for(i = 0; i<=m+n; ++i)   c[i]=0;        //The highest-order term of C(x) is x^(m+n)
for(i = 0; i<=m; ++i)
{
        for(j = 0; j <= n; ++j)
        {
            c[i+j]=a[i]*b[j]+c[i+j];
        }
}
```

Polynomial Division

- Use subtractions to implement division

dividend $x^3 - 3x^2 - x - 1$

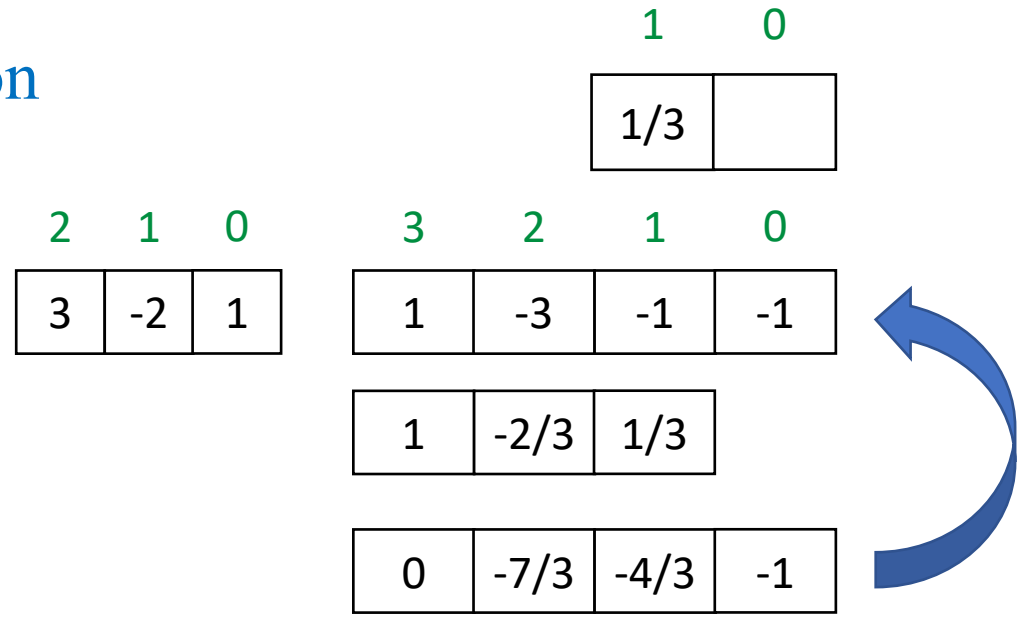
divisor $3x^2 - 2x + 1$

$$
\begin{array}{r}
\frac{1}{3}x - \frac{7}{9} \\
3x^2 - 2x + 1 \overline{)\, x^3 - 3x^2 - x - 1} \\
\underline{x^3 - \frac{2}{3}x^2 + \frac{1}{3}x} \\
-\frac{7}{3}x^2 - \frac{4}{3}x - 1 \\
\underline{-\frac{7}{3}x^2 + \frac{14}{9}x - \frac{7}{9}} \\
-\frac{26}{9}x - \frac{2}{9}
\end{array}
$$

Polynomial Division

- Array operations in polynomial division

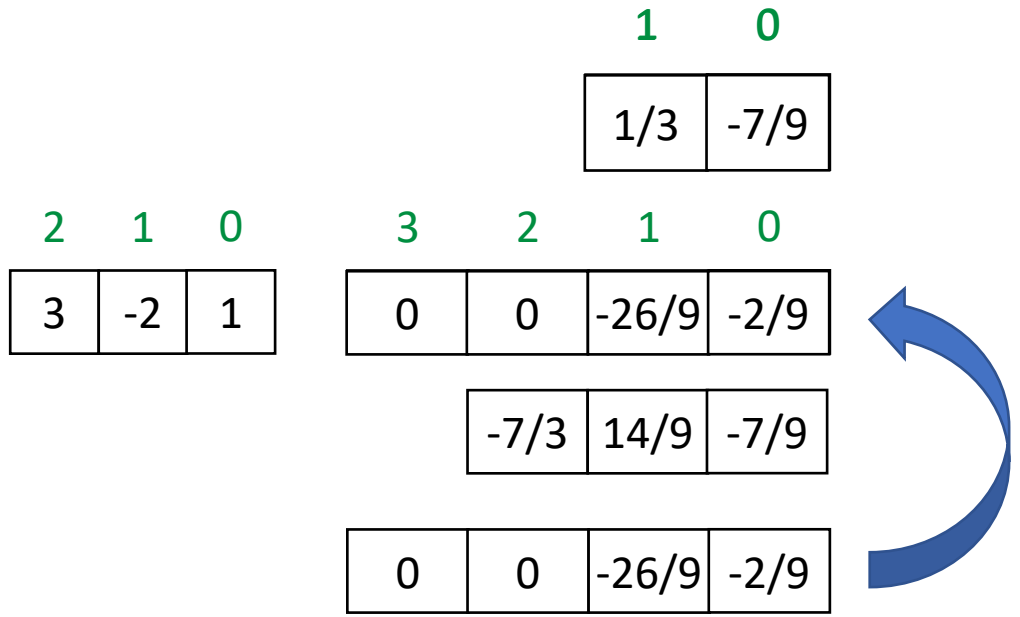
$$
\begin{array}{r}
\frac{1}{3}x \quad -\frac{7}{9} \\
3x^2-2x+1 \overline{)\, x^3-3x^2-x \quad -1} \\
\underline{x^3-\frac{2}{3}x^2+\frac{1}{3}x} \\
-\frac{7}{3}x^2-\frac{4}{3}x \quad -1 \\
\underline{-\frac{7}{3}x^2+\frac{14}{9}x-\frac{7}{9}} \\
-\frac{26}{9}x-\frac{2}{9}
\end{array}
$$



Polynomial Division

- Array operations in polynomial division

$$
\begin{array}{r}
\frac{1}{3}x \quad -\frac{7}{9} \\
3x^2-2x+1 \overline{)\, x^3-3x^2-x-1} \\
\underline{x^3-\frac{2}{3}x^2+\frac{1}{3}x} \\
-\frac{7}{3}x^2-\frac{4}{3}x-1 \\
\underline{-\frac{7}{3}x^2+\frac{14}{9}x-\frac{7}{9}} \\
-\frac{26}{9}x-\frac{2}{9}
\end{array}
$$



result :

dividend x^3-3x^2-x-1 → remainder $-\frac{26}{9}x-\frac{2}{9}$

divisor $3x^2-2x+1$

quotient $\frac{1}{3}x-\frac{7}{9}$

Polynomial Division

Code for polynomial division

dividend $x^3 - 3x^2 - x - 1$ m=3

divisor $3x^2 - 2x + 1$ n=2

quotient $\frac{1}{3}x - \frac{7}{9}$ m-n=1

remainder $-\frac{26}{9}x - \frac{2}{9}$

```
……
//The highest order of the quotient T(x) is x^(m-n)
for( i = m-n;  i>=0;  i-- )
{
    T[i] = P[i+n]/Q[n];  // coefficient of the quotient
    for(j = n; j >= 0; j--)
    {
        P[i+j]=P[i+j]-Q[j]*T[i];
    }

}
```

Correctness: For i=m-n, P[i+j] in the inner loop is P[m], P[m-1],…, P[m-n]

Polynomial Interpolation

- **Definition of the polynomial interpolation problem:**

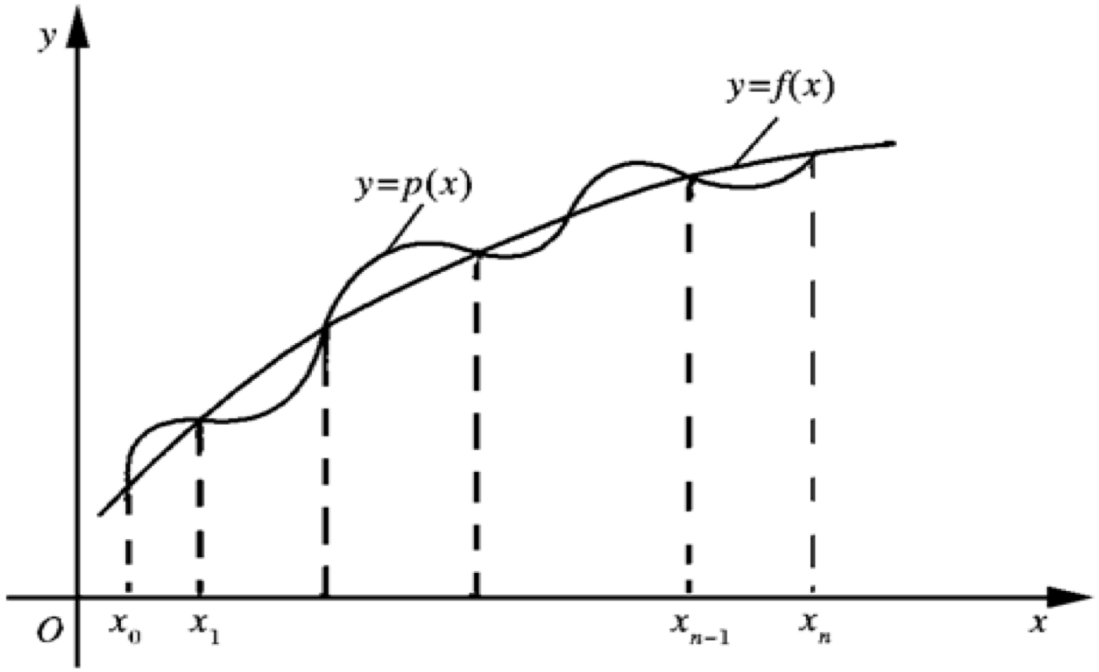
  Let $y = f(x)$ be a real function over $[a, b]$, x_i $(i = 0,1, \cdots, n)$ be $n + 1$ distinct real numbers in $[a, b]$. Given that $f(x_i) = y_i$, find a polynomial $P_n(x)$ with order at most n such that $P_n(x_i) = y_i$.

$P_n(x)$ is the degree-n interpolation polynomial of $f(x)$.

(x_i, y_i) is an interpolation point.

Polynomial Interpolation Problem

◆ Geometric explanation of interpolation problems



$P_n(x)$ is a polynomial curve that traverses the $n+1$ points (x_i, y_i), for $i = 0,1,\cdots,n$.

Polynomial Interpolation Problem-Lagrange Interpolation

- **Basis function method:**

  If we can find a series of polynomials $l_k(x)$ such that $l_k(x_k) = 1$ and $l_k(x_j) = 0$ $(\forall j \neq k)$, then the value of $P(x) = \sum_{j=0}^{n} l_j(x)$ **at x_k is determined only by $l_k(x)$**, and $P(x_k) = 1$.

- Then, function

$$P_n(x) = \sum_{j=0}^{n} y_j l_j(x)$$

is the interpolation polynomial we aim to obtain.

$$\begin{aligned} P_n(x_k) &= \sum_{j=0}^{n} y_j l_j(x_k) \\ &= y_k l_k(x_k) \\ &= y_k \end{aligned}$$

Polynomial Interpolation Problem-Lagrange Interpolation

Construction of the interpolation polynomial:

Firstly, construct:

$$l_k(x) = \begin{cases} 1 & x = x_k \\ 0 & x = x_j \quad j \neq k \end{cases}$$

$l_k(x)$ is a polynomial with order at most n

Next, construct:

$$P_n(x) = \sum_{j=0}^{n} y_j l_j(x)$$

Polynomial Interpolation Problem-Lagrange Interpolation

- How to construct $l_k(x)$?

$l_k(x)$ has n zero points $x_0, x_1, \cdots, x_{k-1}, x_{k+1}, \cdots, x_n$.

The order of $l_k(x)$ is at most n .

$$l_k(x) = c(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)$$

where c is a constant yet to be determined.

Next, as $l_k(x_k) = 1$, we have $c = \dfrac{1}{(x_k-x_0)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$

Finally, $l_k(x) = \dfrac{(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$

Polynomial Interpolation Problem-Lagrange Interpolation

Using Lagrange interpolation in practice

Typically, we do not consider to obtain the following **closed-form expression** of the interpolation polynomial, as obtaining the coefficients is overly complicated.

$$P_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

A more common way is to calculate the approximate value at some point directly using $l_k(x)$.

Use Lagrange interpolation to solve the following problem

Given: $x[0], x[1], \ldots, x[n]$ on the x-axis, and the corresponding values of the function $y[0], y[1], \ldots, y[n]$

Objective: Use interpolation polynomial to obtain an approximate value of the function at the point xx.

Polynomial Interpolation Problem-Lagrange Interpolation

■ **How to calculate $l_k(xx)$ using a piece of code?**

```
S=1;
for(j=0;j<=n;j++)
    if(j!=k) S =S * (xx-x[j])/(x[k]-x[j]);
```

Note here ! ! !

To calculate $\boldsymbol{y[k]l_k(xx)}$, then

```
S= y[k];
for(j=0; j<=n; j++)
    if(j!=k) S =S * (xx-x[j])/(x[k]-x[j]);
```

S obtained in this way is one of the multiple values that need to be added together.

Polynomial Interpolation Problem-Newton Interpolation

- Theoretically, a degree-n polynomial can be written as the linear combination of any degree-0, degree-1, degree-2, …, and degree-n polynomial.

- Then, the interpolation polynomial can be written as the linear combination of

  $1, x\text{-}x_0, (x\text{-}x_0)(x\text{-}x_1), \ldots, (x\text{-}x_0)(x\text{-}x_1)(x\text{-}x_{n\text{-}1})$.

- Newton interpolation polynomial can be written as:

$$N_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

Polynomial Interpolation Problem-Newton Interpolation

$$N_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

We consider a simplified scenario, i.e., $x_k = x_0 + kh \ (k = 0,1,\cdots,n)$

$$N_n(x_0) = a_0 = y_0 \quad \rightarrow \quad a_0 = y_0$$

$$N_n(x_1) = a_0 + a_1(x_1 - x_0) = y_1 \quad \rightarrow \quad a_1 = (y_1 - y_0)/h = \frac{\Delta y_0}{h}$$

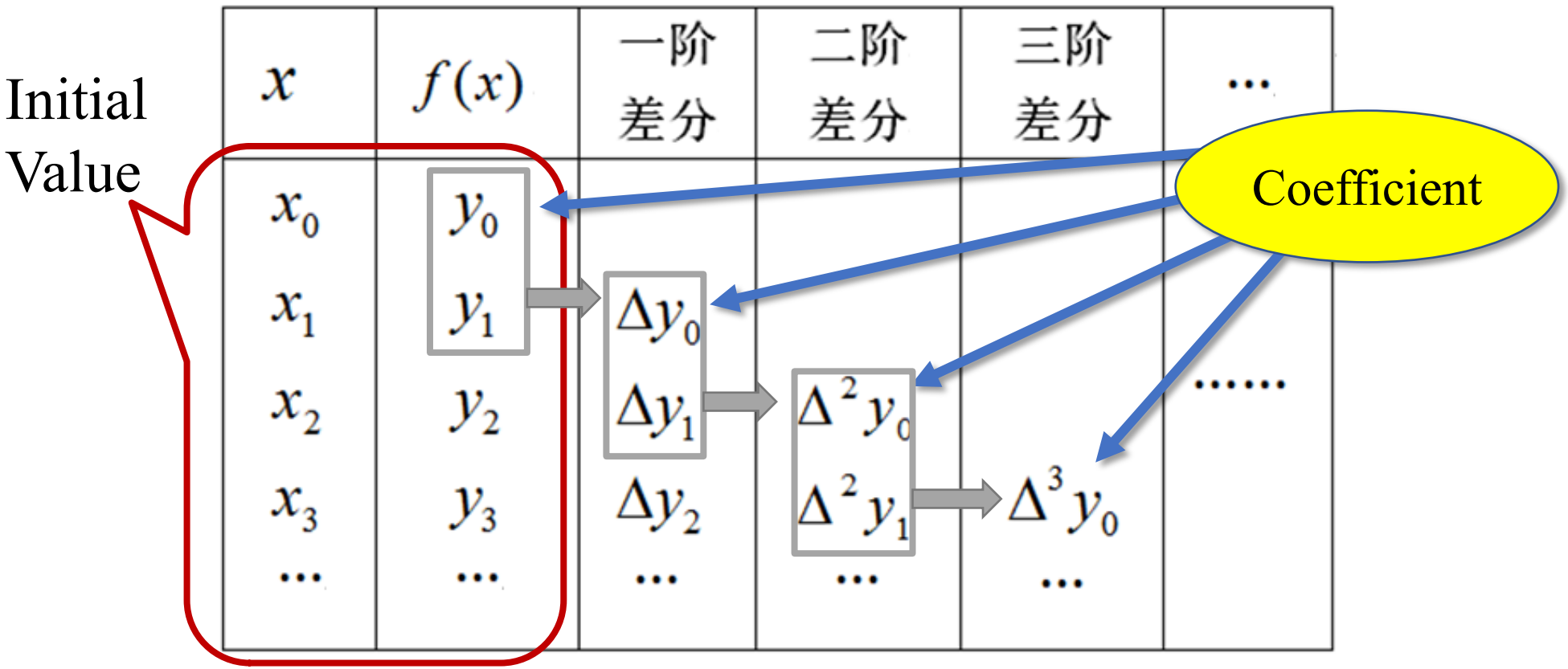
$$N_n(x_2) = \cdots\cdots = y_2 \quad \rightarrow \quad a_2 = \frac{\Delta y_1 - \Delta y_0}{2h^2} = \frac{\Delta^2 y_0}{2!h^2}$$

$$\cdots\cdots \quad \cdots\cdots \quad a_k = \frac{\Delta^k f(x_0)}{k!h^k}$$

Polynomial Interpolation Problem-Newton Interpolation

$$N_n(x_0 + th) = y_0 + \sum_{k=1}^{n}\left(\frac{\Delta^k y_0}{k!h^k} h^k \prod_{i=0}^{k-1}(t-i)\right) = y_0 + \sum_{k=1}^{n}\left(\frac{\Delta^k y_0}{k!} \prod_{i=0}^{k-1}(t-i)\right)$$

Difference Table



After Class

- Better ways to find the LCM?
- Compare Lagrange interpolation with Newton interpolation.
- Other polynomial interpolation methods?
  - Neville’s method
- Newton interpolation polynomial at non-equally spaced points ?

References and Readings

- **Kenneth H. Rosen, "Discrete Mathematics and Its Applications" (6th Edition)**
  - Chapter 3.5 Primes and Greatest Common Divisors

- **Richard L. Burden, J. Douglas Faires, "Numerical Analysis" (9th Edition)**
  - Chapter 3 Interpolation and Polynomial Approximation