Horner’ Algorithm, Precision and Errors

CS3010

Numerical Methods

Dr. Amar Raheja

Section 1.1

Lecture 2

Nested Multiplication

- Compute the following polynomial
  - $p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0$
- assume existence of array a containing *n*+1 values indexed from 0 to *n*, the value x, and the integer *n*
- Suppose existence of subroutine pow(x,i) that computes x^i, how would one compute the sum of all terms?

```
sum = a[0];
for (i=1; i<=n; i++)
    sum=sum + a[i]*pow(x,i);
```

- What's the complexity? O(n)= ?

Iteration vs. Recursion for power function

- Power can be implemented as iterative or recursive algorithm. How many arithmetic ops in both cases?

iterative

```
term = 1.0;
        for (j=1; j<=i; j++)
                term = term*x;
         return term;
```

recursive

```
if (i==0)
        return 1.0;
else if (i%2 == 1)             // i is odd
         return x*pow(x, i-1);
else {
        term = pow(x, i/2);
        return term*term;
    }
```

Horner’s Nested Multiplication Algorithm

- Optimal complexity for general polynomials using nested multiplication
- Factor using nested multiplication - Horner’s algorithm or synthetic division

$p(x) = 4x^4 + 3x^3 – 2x^2 + 6x – 9$

$= (((((4)x+3)x–2)x+6)x–9)$

$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0$

$p(x) = a_0 + x(a_1+x(a_2+\ldots+x(a_{n-1}+ x(a_n))\ldots))$

$$\boldsymbol{p(x) = \sum_{i=0}^{n} a_i x^i = \sum_{i=0}^{n} \left(a_i \prod_{j=1}^{i} x \right)}$$

remember $\boldsymbol{\sum_{k=n}^{m} x_k = x_n + x_{n+1} + \cdots + x_m}$ and $\boldsymbol{\sum_{k=n}^{m} x_k = x_n x_{n+1} \ldots x_m}$

```
sum = a[n];
for (i=n-1;  i>=0; i--)
        sum=sum*x + a[i];
```

Horner’s Nested Multiplication Example

- Show how $p(x) = 5 + 3x - 7x^2 + 2x^3$ should be computed (write it in Horner’s Nested Form)

$$p(x) = 5 + x\left(3 + x\big((-7) + x(2)\big)\right)$$

What happens in a function like this?

$$p(x) = -6 + 3x - 9x^3 + 4x^5$$

- Some terms are missing, so the coefficients will be 0 for this and don’t have to be written

$$p(x) = -6 + 3x + 0x^2 - 9x^3 + 0x^4 + 4x^5$$

$$p(x) = -6 + x\left(3 + x^2\big((-9) + x^2(4)\big)\right)$$

- Notice that the highest x powers are removed from terms as we progress to the right

Deflation of a Polynomial

For any given polynomial, $p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x^1 + a_0$

- In the deflation process, if *r* is a zero of the polynomial *p*, then *x* − *r* is a factor of *p*

$$\Rightarrow p(x) = (x - r)Q(x) + R$$

Q(*x*) is quotient polynomial of degree *n*-1 and *R* is a remainder term

$$\Rightarrow Q(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \cdots + b_1 x^1 + b_0$$

$$a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x^1 + a_0 = (x - r)(b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \cdots + b_1 x^1 + b_0) + b_{-1}$$

- Note that if r is an exact root, $b_{-1} = p(r) = 0$
- Equating like coefficients from both sides, one gets

$$a_n = b_{n-1};\ a_{n-1} + rb_{n-1} = b_{n-2};\ a_{n-2} + rb_{n-2} = b_{n-3}$$

$$a_1 + rb_1 = b_0; a_0 + rb_0 = b_{-1}$$

Deflation process of a polynomial: Example

- Following arrangement is often used to carry out Horner's Algorithm

$$
\begin{array}{r|rrrrrr}
 & a_n & a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \\
r\,) & & rb_{n-1} & rb_{n-2} & \dots & rb_1 & rb_0 \\
\hline
 & b_{n-1} & b_{n-2} & b_{n-3} & \dots & b_0 & b_{-1}
\end{array}
$$

- Use Horner's algorithm to evaluate $p(x) = x^4 - 4x^3 + 7x^2 - 5x - 2$

$$
\begin{array}{r|rrrrr}
 & 1 & -4 & 7 & -5 & -2 \\
3\,) & & 3 & -3 & 12 & 21 \\
\hline
 & 1 & -1 & 4 & 7 & 19
\end{array}
$$

- Thus, we obtain $p(3) = 19$, and we can write

$$p(x) = (x - 3)(x^3 - x^2 + 4x + 7) + 19$$

Pseudocode for Horner's Algorithm

$$
\begin{array}{l}
\textbf{integer } i, n; \quad \textbf{real } p, r; \quad \textbf{real array } (a_i)_{0:n}, (b_i)_{0:n-1} \\
b_{n-1} \leftarrow a_n \\
\textbf{for } i = n - 1 \textbf{ to } 0 \textbf{ do} \\
\qquad b_{i-1} \leftarrow a_i + r b_i \\
\textbf{end for}
\end{array}
$$

- Notice that $b_{-1} = p(r)$ in this pseudocode. If f is an exact root, then $b_{-1} = p(r) = 0$.
- The array $a[]$ stores all the polynomial coefficients

Deflating Polynomial multiple times

- With one step of deflation

$$\Rightarrow p(x) = (x - r)q(x) + p(r)$$

- Differentiate both sides to get

$$p'(x) = q(x) + (x - r)q'(x)$$

- At $x \;=\; r$, we get $p'(r) = q(r)$
- Above implies that deflate $q(x)$ at $x = r$ to get $p'(r)$

$$\begin{array}{r|ccccccl}
 & a_n & a_{n-1} & a_{n-2} & \cdots & a_2 & a_1 & a_0 \\
r\,) & & rb_{n-1} & rb_{n-2} & \cdots & rb_2 & rb_1 & rb_0 \\
\hline
 & b_{n-1} & b_{n-2} & b_{n-3} & \cdots & b_1 & b_0 & \boxed{b_{-1}} = p(r) \\
 & & rc_{n-2} & rc_{n-3} & \cdots & c_2 & rc_1 & \\
\hline
 & c_{n-2} & c_{n-3} & c_{n-4} & \cdots & c_0 & \boxed{c_{-1}} = p'(r) &
\end{array}$$

Example

- Use Horner's algorithm to evaluate $p'(3)$ for

$$p(x) = x^4 - 4x^3 + 7x^2 - 5x - 2$$

$$\begin{array}{r|rrrrl}
 & 1 & -4 & 7 & -5 & -2 \\
3) & & 3 & -3 & 12 & 21 \\
\hline
 & 1 & -1 & 4 & 7 & \boxed{19} = p(3) \\
 & & 3 & 6 & 30 & \\
\hline
 & 1 & 2 & 10 & \boxed{37} = p'(3) &
\end{array}$$

$$p(x) = (x-3)(x^3 - x^2 + 4x + 7) + 19$$

$$p(x) = (x-3)q(x) + 19 \; and \; p'(x) = q(x) + (x-3)q'(x)$$

$$q(x) = x^3 - x^2 + 4x + 7$$

Hence, we have $p(3) = 19, and\ p'(3) = q(3) = 37$

Pseudocode for Horner's Algorithm for calculating differential

integer i, n; **real** p, r
real array $(a_i)_{0:n}, (b_i)_{0:n-1}$
$\alpha \leftarrow a_n; \beta \leftarrow 0$
for $i = n - 1$ **to** 0
  $\beta \leftarrow \alpha + r\beta$
  $\alpha \leftarrow a_i + r\alpha$
end for

- Notice that $\alpha = p(r)$ and $\beta = p'(r)$ in this pseudocode
- The array $a[]$ stores all the polynomial coefficients

Concept of Error

- Numerical methods yield approximate results, results that are close to the exact analytical solution. We cannot exactly compute the exact errors associated with numerical methods.
  - Only rarely given data are exact, since they originate from measurements. Therefore there is probably error in the input information.
  - Algorithm itself usually introduces errors as well, e.g., unavoidable round-off errors
  - The output information will then contain error from both these sources.
- How confident are we in our approximate result?
- The question is "*how much error is present in our calculation and is it tolerable?*"

Absolute and Relative Error

- Suppose that α and β are two numbers, of which one is regarded as an approximation to the other.
- The **error** of β as an approximation to α is $\alpha - \beta$; that is, the error equals the exact value minus the approximate value.
- The **absolute error** of β as an approximation to α is $|\alpha - \beta|$. The **relative error** of β as an approximation to α is $|\alpha - \beta|/|\alpha|$
- Notice that in computing the absolute error, the roles of α and β are the same, whereas in computing the relative error, it is essential to distinguish one of the two numbers as *correct*.
- Observe that the relative error is undefined in the case $\alpha = 0$
- For practical reasons, the relative error is usually more meaningful than the absolute error.

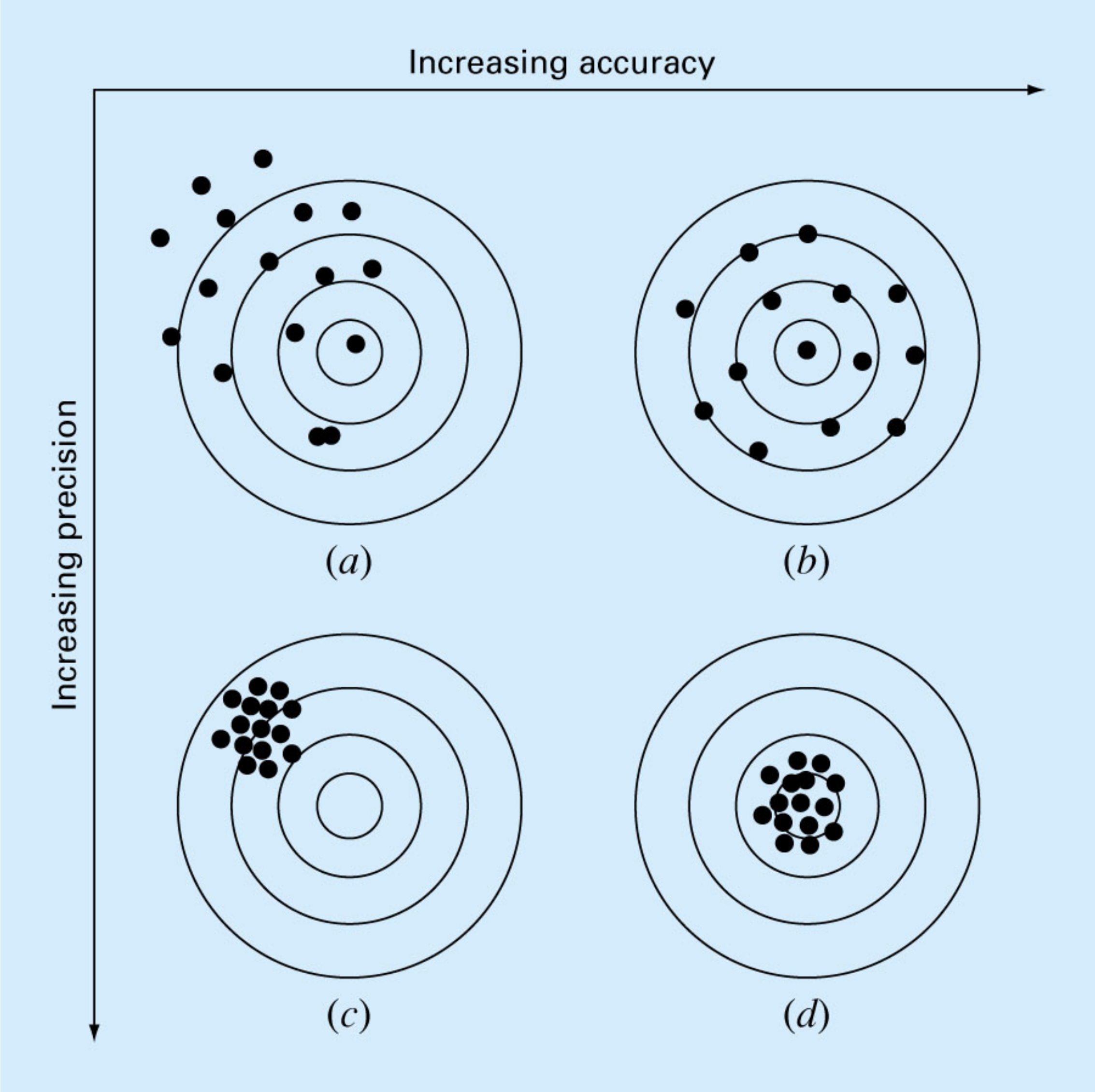
Example of absolute and relative errors

- For example, if $\alpha_1 = 1.333$, $\beta_1 = 1.334$, and $\alpha_2 = 0.001$, $\beta_2 = 0.002$, then the absolute error of β_i as an approximation to α_i is the same in both cases—namely, 10^{-3}.
- However, the relative errors are $3/4 \times 10^{-3}$ and 1, respectively.
- The relative error clearly indicates that β_1 is a good approximation to α_1 but that β_2 is a poor approximation to α_2.
- $absolute\ error\ =\ |exact\ value\ -\ approximate\ value|$
- $relative\ error\ =\ \dfrac{|exact\ value - approximate\ value|}{|exact\ value|}$
- Consider $x = 0.00347$ rounded to $\tilde{x} = 0.0035$ and $y = 30.158$ rounded to $\tilde{y} = 30.16$. Find absolute and relative errors
- For x, absolute error is 0.3×10^{-4}, and relative error is 0.865×10^{-2}
- For y, absolute error is 0.2×10^{-2}, and relative error is 0.66×10^{-4}

Accuracy vs. Precision

- Accuracy: How close is a computed or measured value to the true value
  - Absolute error deals with accuracy
- Precision (or *reproducibility*): How close is a computed or measured value to previously computed or measured values.
  - Relative error deals with precision
  - Relates to significance
- Inaccuracy (or *bias*): A systematic deviation from the actual value.
- Imprecision (or *uncertainty*): Magnitude of scatter.

Accuracy and Precision



Precision and Significance

- Precision: Number of digits carried by the computer
  - **floating point data type: 6-7 decimal digits**
  - **double data type: about 15 decimal digits**
- Significance: Number of digits where the computer answer and true answer agree
  - Leading zeros are never significant.
  - Embedded zeros are always significant.
  - Trailing zeros are significant only if the decimal point is specified.
  - Hint: Change the number to scientific notation. It is easier to see.

Significant Digits

- **Significant digits** are digits beginning with the leftmost *nonzero* digit and ending with the rightmost *correct* digit, including final zeros that are exact.

Number	# of significant digits	Normalized Scientific Notation	Comment
0.00682	3	0.682×10^{-2}	Leading zeros are not significant
1.072	4	$0.1072\ (\times 10^{1})$	Embedded zeros are always significant
300	1	0.3×10^{3}	Trailing zeros are significant only if the decimal point is specified
300.0	4	0.3000×10^{3}	
300.00	5	0.30000×10^{3}	

Rounding and Chopping numbers

- A number x is ***chopped* to *n digits*** or figures when all digits that follow the nth digit are discarded and none of the remaining n digits are changed.
- Conversely, x is ***rounded* to *n digits*** or figures when x is replaced by an n-digit number that approximates x with minimum error. The question of whether to round up or down an $(n + 1)$-digit decimal number that ends with a 5 is best handled by always selecting the rounded n-digit number with an *even* nth digit.
  - If digits beyond nth digit are > 5000…, then round up, i.e. drop all digits beyond n and increase the nth digit by 1
  - If digits beyond nth digit are < 5000…, then round down, i.e. discard all digits beyond n
  - If equal to 5000…, then round the nth digit so that it is even (causes round up or down equally often)

Chop and Round these numbers

- What are following truncated(chopped) and rounded to 2 decimal digits ?

0.217, 0.256, 0.25, 0.2499, 0.259, 0.475, 0.365

Rounding and Chopping Error

- What is the maximum error in rounding and chopping to *n* significant digits?
- Rounding:

$$|x - \tilde{x}| \leq \frac{1}{2} \times 10^{-n}$$

- Chopping:

$$|x - \tilde{x}| \leq 10^{-n}$$

- Why/How ?

Example: Chop 0.256 or round 0.256

Significant digits of precision class exercise

$$0.1036\,x + 0.2122\,y = 0.7381$$

$$0.2081\,x + 0.4247\,y = 0.9327$$

- Solve the following simple set of linear equations
  - with 4 digits rounding (means round after each calculation to 4 significant digits)
  - with 3 digits rounding
- Value of y will change from -547 to 343.9
- If done with 10 significant digits in computer, it will be 356.2907199
- The lesson learned in this example is that data thought to be accurate should be carried with full precision and *not* be rounded off prior to each of the calculations.