

Lecture # 1 Introduction

First go over introductory handout at www.cse.psu.edu/~barlow/cse455/intro.pdf.

Numerical analysis is the science of computing the solutions of problems that are posed mathematically in the field of real or complex numbers.

Today I will just give a couple of examples.

Example 1 *Computation of* $\pi \approx 3.14159265358979$

π is known as the ratio of the circumference of a circle and its diameter.

Archimedes Method

Inscribe a regular n -gon in a circle of radius 1. Compute the perimeter of its upper half. This is easiest to do if $n = 2^k$. Using geometric reasoning (given in class),

$$p(n) = 2n \sin \frac{\pi}{2n}.$$

Even though this refers to π , we can compute its value for $n = 1, 2, \dots$, without computing π and without punching the sin button on a calculator. (Computing the sin function requires you to know π !)

From geometric reasoning, we know that

$$\sin \frac{\pi}{2} = 1, \quad \sin \frac{\pi}{4} = \sqrt{2}/2.$$

So

$$p(1) = 2 \cdot \sin \frac{\pi}{2} = 2, \quad p(2) = 4 \cdot \sin \frac{\pi}{4} = 2\sqrt{2} = 2.828427125$$

but what is $p(4) = 8 \cdot \sin \frac{\pi}{8}$? Use half angle formulas!

$$\sin \frac{\theta}{2} = \sqrt{\frac{1 - \cos \theta}{2}},$$
$$\cos \theta = \sqrt{1 - \sin^2 \theta}.$$

That yields

$$\sin \frac{\pi}{8} = 0.382683432.$$

Thus

$$p(4) = 8 \cdot \sin \frac{\pi}{8} = 3.061467459.$$

Continuing this process we get

n	$\sin \pi/2n$	$p(n)$
1	1	2
2	$\sqrt{0.5}$	2.82842712
4	0.382683432	3.061467459
8	0.195090322	3.121445152
16	0.09801714	3.136548491
32	0.049067674	3.140331157

This method is slow, but “sure.” Later, we will give a modern enhancement that is faster. Letting $h = 1/(2n)$, we have that

$$p(n) = \frac{\sin \pi h}{h} = \pi - a_2 h^2 + a_4 h^4 - \dots$$

where $a_k = \pi^{k+1}/(k+1)!$. Thus this converges to π at roughly the rate of $O(h^2)$. This is essentially an approximation problem. There is no finite algorithm to compute π , since it is a transcendental number (irrational and not the root of any polynomial with integer coefficients). However, we can approximate it arbitrarily well.

The following problem poses completely different issues.

Example 2 *Roots of*

$$p(x) = ax^2 + bx + c = 0.$$

for constants a, b, c .

The solution is just

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

How do we go about writing a code to compute this.

Take care of special cases.

Case I $a = 0, b \neq 0$. It is no longer a quadratic, it is linear. Only solution is

$$x_1 = -c/b.$$

Case II $a = b = 0$ If $c \neq 0$, no solution. If $c = 0$, all x are solutions.

The cases you spent time on in high school had to do with the discriminant $b^2 - 4ac$.

Case III $b^2 - 4ac < 0$. Two Complex Roots (not real).

$$x_{1,2} = -\frac{b}{2a} \pm \mathbf{i} \frac{\sqrt{4ac - b^2}}{2a}, \quad \mathbf{i}^2 = -1.$$

Case IV $b^2 - 4ac = 0$. One Double Root (Real)

$$x_1 = x_2 = -\frac{b}{2a}.$$

Case V $b^2 - 4ac > 0$. Two Distinct Real Roots. Use Formula??

Show code in www.cse.psu.edu/~barlow/cse455/quadroots0.m that implements this algorithm.

Instead of computing $b^2 - 4ac$, we compute

$$\text{test_val} = \sqrt{2a}\sqrt{2c}.$$

Then Cases III, IV, and V are $b < \text{test_val}$, $b == \text{test_val}$, and $b > \text{test_val}$. In Case V, the discriminant is computed from

$$b^2 - 4ac = \sqrt{b - \text{test_val}}\sqrt{b + \text{test_val}}.$$

This is more accurate in floating point arithmetic and guards against overflow and underflow.

Try the case $a = 1$, $b = 2$, $c = 10^{-17}$. The two real roots are (to about 17 digits)

$$x_1 = -2, \quad x_2 = -5 \cdot 10^{-18}.$$

The above algorithm in MATLAB gets x_1 right, but $x_2 = 0$. MATLAB double precision stores about 15 digits and in those 15 digits $\sqrt{b^2 - 4ac} - b = 0$. That is because we are subtracting two close numbers and one of these is approximate, so this difference is “all rounding error.” A simple observation gets around this problem.

The “large” root of the quadratic in Case V is

$$x_1 = \frac{-b - \text{sign}(b)\sqrt{b^2 - 4ac}}{2a}$$

and the two roots satisfy

$$x_1x_2 = c/a.$$

Notice that except inside the square root, we are adding numbers of the same sign! After some algebra, we get a formula for the small root

$$x_2 = c/(ax_1) = \frac{-2c}{b + \text{sign}(b)\sqrt{b^2 - 4ac}}.$$

Using this formula we design the code in www.cse.psu.edu/~barlow/cse455/quadroots.m and that code compute both roots to machine precision.

In this example, we have an exact formula, but in floating point arithmetic the formula yields results that are significantly different from what it yields in real arithmetic.