HW8 - Math 5610

Andrew Sheridan

November 2016

Problem 1

Explain why accumulation of roundoff error is inevitable when arithmetic operations are performed in a floating point system. When is this accumulation of roundoff errors tolerable in numerical calculations.

Not all numerical values can be accurately represented in a floating point system. For every numerical computation we are likely to produce roundoff error because the results cannot be perfectly represented. These perturbations accumulate throughout calculations, and with every expression the total error is likely to increase in magnitude. The accumulation of roundoff error is tolerable when the perturbations to the results are smaller than the significant magnitude of the problem, or when the roundoff errors are dominated by discretization errors.

Problem 2

Explain the importance of the machine precision of a computer.

It is important to take into account machine precision when performing numerical calculations because computers cannot perfectly represent all possible numerical values. There exists a finite amount of precision where values can become too small or too large in magnitude for a computer to represent, or where its mantissa contains too many digits to be stored in completion. When performing a large number of sequential calculations, a small error in the beginning can perpetuate and create larger errors later on.

Problem 3

What is the basic ingredient for convergence of a functional iteration algorithm in the location of the roots of a nonlinear function? What controls the speed of convergence in such a method?

In functional iteration, the main factor in regards to convergence is the function $g(x)$ which we select for our iteration. If its derivative, $g'(x)$, has an absolute value which is less than ρ (where $\rho < 1$) for all x in the interval, then we can be assured functional iteration will converge. The speed of convergence is controlled by ρ. The smaller our value ρ, the faster the convergence.

Problem 4

State the disadvantages of computing the inverse of a square matrix in solving a system of linear equations $Ax = b$.

Computing the inverse of a matrix generally takes more time than solving the system using Gaussian Elimination and Back Substitution. Using the inverse is also more prone to introduce errors.

Problem 5

Give an example/explanation that distinguishes between problem conditioning and algorithm stability

A problem is well conditioned if small changes to its input results in only small perturbations to its output.

$$f(x) \approx f(\bar{x}) \text{ for } x \approx \bar{x}$$

A problem is ill-conditioned if small input changes result in large output changes.

$$f(x) \not\approx f(\bar{x}) \text{ for } x \approx \bar{x}$$

An algorithm $f(x)$ is stable if a small change in input, $\bar{x}$, results is approximately the exact result $F(x)$.

$$F(x) \approx f(\bar{x})$$

Problem 6

Show that the following difference is a second order approximation of the derivative of a function, f, at x = x0.

$$f'(x_0) \approx \frac{f(x_0 + \frac{h}{2}) - f(x_0 + \frac{h}{2})}{h}$$

Use Taylor Series expansions

$$f(x_0 + \frac{h}{2}) \approx f(x_0) + \frac{h}{2} f'(x_0) + \frac{h^2}{4} f''(\xi_1))$$

$$f(x_0 - \frac{h}{2}) \approx f(x_0) - \frac{h}{2} f'(x_0) + \frac{h^2}{4} f''(\xi_2))$$

$$f(x_0 + \frac{h}{2}) - f(x_0 - \frac{h}{2}) \approx f'(x_0) + \frac{h^2}{4} f''(\xi_1) - \frac{h^2}{4} f''(\xi_2)$$

$$\frac{f(x_0 + \frac{h}{2}) - f(x_0 - \frac{h}{2})}{h} \approx f'(x_0) + \frac{h}{4} f''(\xi_1) - \frac{h}{4} f''(\xi_2)$$

$$f'(x_0) \approx f'(x_0) + \frac{h}{4} f''(\xi_1) - \frac{h}{4} f''(\xi_2)$$