Algorithms in Systems Engineering
IE170

Lecture 24

Dr. Ted Ralphs

References for Today's Lecture

- Required reading
  - CLRS Chapter 28

Continuous Mathematics

- So far, all the algorithms we have studied have been for problems in *discrete mathematics*.
- Discrete mathematics is the study of problems for which there are only a finite number of possible outcomes.
- Algorithms for discrete problems typically only involve integer values.
- *Continuous mathematics* is the study of problems for which the number of possible outcomes is infinite.
- Algorithms for these problems typically involve real numbers.
- The difficulty with such algorithms is that a computer is not capable of representing most real numbers exactly.
- To represent a real number in a computer, the value must be rounded.
- This can create serious problems for some algorithms.

Common Problems in Continuous Mathematics

- Solving systems of equations
- Finding the inverse of a matrix
- Finding the eigenvalues or eigenvectors of a matrix
- Least-squares problems
- Solving systems of inequalities
- Factoring a matrix

Floating-point Numbers

- There are basically three schemes by which real numbers can be approximated in a computer.
  - As a rational number.
  - Using fixed point representation.
  - Using floating point representation.
- In modern computers, real numbers are approximated using a *floating point* representation.
- The floating-point numbers F are a subset of the real numbers.
- A floating point number consists of three parts:
  - The sign
  - The exponent
  - The mantissa

Floating-point Arithmetic

- Arithmetic with floating point numbers is different than regular arithmetic.
- This is because after each operation, the answer must be rounded off.
- The biggest problems occur when numbers on different scales appear in the same calculation.
- Example
  - Assume 10 digit precision
  - $(10^{-10} + 1) - 1 = 0$
  - $10^{-10} + (1 - 1) = 10^{-10}$
- The example show that floating point operations do not have the same properties as familiar arithmetic operations.

Numerical Analysis

- *Numerical analysis* is the study of algorithms for problems from continuous mathematics.
- For the purposes of studying the problems of continuous mathematics, we will define a *problem* as a map $f : X \to Y$, where X and Y are subsets of the real numbers.
- A *numerical algorithm* is a procedure which calculates $F(x) \in Y$, an approximation of $f(x)$.
- Because we have to use *floating point* arithmetic and other approximations, our answers will not be exact.

Conditioning

- Problem conditioning has to do with how much the true answer to a problem changes if the input is changed slightly.
- A problem is *well-conditioned* if $x' \approx x \Rightarrow f(x') \approx f(x)$.
- Otherwise, it is *ill-conditioned*.
- Notice that well-conditioned requires all small perturbations to have a small effect.
- Ill-conditioned only requires some small perturbations to have a large effect.

Stability

- Algorithm stability has to do with whether the algorithm give an answer that is close to the true answer for some input close to the true input.
- An algorithm is *stable* if $F(x) \approx f(x')$ for some $x' \approx x$.
- This says that a stable algorithm computes "nearly the right answer" to "nearly the right question."
- Notice the contrast between conditioning and stability:
  - Conditioning applies to problems.
  - Stability applies to algorithms.

Numerical Accuracy

- Stability plus good conditioning implies *accuracy*.
- If a stable algorithm is applied to a well-conditioned problem, then $F(x) \approx f(x)$.
- Conversely, if a problem is ill-conditioned, an accurate solution may not be possible or even meaningful.
- We cannot ask more of an algorithm than stability.

Examples

- Addition, subtraction, multiplication, division.
  - Addition, multiplication, division with positive numbers are well-conditioned problems.
  - Subtraction is not.
- Zeros of a quadratic equation.
  - The problem of computing the two roots is well-conditioned.
  - However, the quadratic formula is not a stable algorithm.
- Solving systems of linear equations $Ax = b$.
  - Conditioning depends on the matrix A.

More Examples

- Calculating e^{-a} with $a > 0$ by Taylor Series.
  - The round-off error is approximately u times the largest partial sum.
  - Calculating e^{a} and then taking its inverse gives a full-precision answer.
- Roots of a quadratic $(ax^2 + bx + c)$
  - If $x_1 \approx 0$ and $x_2 >> 0$, then the quadratic formula is unstable.
  - Computing x_2 by the quadratic formula and then setting $x_1 = cx_2/a$ is stable.

More Examples

- Matrix factorization.
  - Generally ill-conditioned.
  - There are stable algorithms, however.
- Zeros of a polynomial.
  - Generally ill-conditioned.
- Eigenvalues of a matrix.
  - For a symmetric matrix, finding eigenvalues is well-conditioned, finding eigenvectors is ill-conditioned.
  - For non-symmetric matrices, both are ill-conditioned.
  - In all cases, there are stable algorithms.