Advanced Mathematical Programming
IE417

Lecture 15

Dr. Ted Ralphs

Reading for This Lecture

- Sections 8.1-8.5

Numerical Analysis

Numerical Analysis

- *Numerical analysis* is the study of algorithms for problems from continuous mathematics.
- A *problem* is a map from $f : X \to Y$, where X and Y are normal vector spaces.
- A *numerical algorithm* is a procedure which calculates $F(x) \in Y$, an approximation of $f(x)$.
- As we have already discussed, we can define these algorithms in terms of an algorithmic map.
- Because we have to use floating point arithmetic and other approximations, our answers will not be exact.

Conditioning

- 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.
- Condition number of a problem
  - Absolute
  - Relative

Stability

- 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.

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.

Floating-point Arithmetic

- The floating-point numbers F are a subset of the real numbers.
- For a real number x, let $fl(x) \in F$ denote the floating point approximation to x.
- Let $\odot$ and $\cdot$ represent the four floating point and exact arithmetic operations.
- Typically, there is a number $u << l$ called *machine epsilon* such that
  - $fl(x) = x(1 + \varepsilon)$ for some ε with $|\varepsilon| \leq u$.
  - $\forall a, b \in F, a \odot b = (a \cdot b)(1 + \varepsilon)$ for some $|\varepsilon|$ with $\varepsilon \leq u$.

Stability of Floating Point Arithmetic

- Floating point arithmetic is stable for computing sums, products, quotients, and differences of two numbers.
- Sequences of these operation can be unstable however.
- Example
  - Assume 10 digit precision
  - $(10^{-10} + 1) - 1 = 0$
  - $10^{-10} + (1 - 1) = 10^{-10}$
- Floating point operations are not always associative.

More Bad 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.

Backward Error Analysis

- *Backward error analysis* is a method of analyzing round-off error and assessing stability.
- We want to show that the result of a floating-point operation has the same effect as if the original data had been perturbed by an amount in $O(u)$.
- If we can show this, then the algorithm is stable.

More Examples

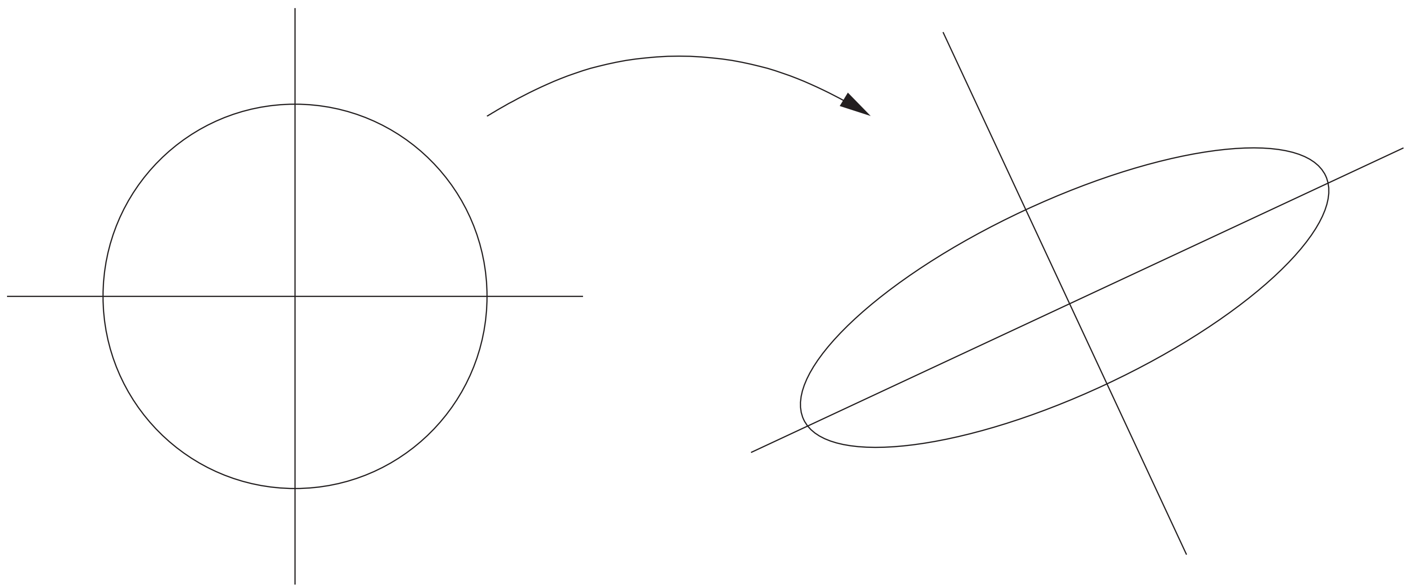
- Matrix factorization.
  - Generally ill-conditioned.
  - There are stable algorithms, however.
- Zeros of a polynomial.
  - Generally ill-conditioned.
- Eigenvalues of a matrix.
  - For a symmetric matrix, finding eigenvalues is well-conditioned, finding eigenvectors is ill-conditioned.
  - For non-symmetric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

Singular Value Decomposition

- Problem: Given a matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$, we wish to find $x \in \mathbb{R}^n$ such that $Ax = b$.

- Diagonal form of a matrix
  - An orthogonal matrix U has the property $U^T U = UU^T = I$.
  - Given $A \in \mathbb{R}^{m \times n}$, there exist orthogonal matrices U, V such that
    * $U^T AV = D$ where D is a diagonal matrix where
    * diagonal elements of D are $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_r > \mu_{r+1} = \ldots = \mu_n = 0$, and
    * r is the rank of A
    * μ_i is the non-negative square root of the i^{th} eigenvalue of $A^T A$.
  - This is called the *singular value decomposition*.

Importance of the SVD



Effect of multiplying by a matrix

Implications

- Multiplying by (square) A represents a *rotation* and a *scaling* of axes to get from one space to another.
- μ_i is the non-negative square root of the i^{th} eigenvalue.
- Notice that $\|A\| = \|D\| = \mu_1$.
- So the norm of A is the maximum amount any axis gets magnified by A.
- If $r = n$, then we can easily derive the inverse of A.
- Also, $\|A^{-1}\| = \|A\|^{-1} = 1/\mu_n$.

Condition of a Linear System

- Consider the problem of solving $Ax = b$.
- If we perturb b, how much does the x change?
- $x + \delta x = A^{-1}(b + \delta b) \Rightarrow \delta x = A^{-1}\delta b$
- $\|\delta x\| \leq \|A^{-1}\| \cdot \|\delta b\|$
- $\|\delta x\| \cdot \|b\| \leq \|A\| \cdot \|A^{-1}\| \cdot \|x\| \cdot \|\delta b\|$
- $\|\delta x\|/\|x\| \leq \|A\| \cdot \|A^{-1}\| \cdot (\|\delta b\|/\|b\|)$
- The *condition number* of a matrix is the quantity $cond(A) = \|A\| \cdot \|A^{-1}\|$

Condition Number

- Note that $cond(A) = \mu_1/\mu_n$.
- Hence it is a relative measure of how much distortion A causes to its input.
- It is also a measure of how much the inaccuracies in b get multiplied in x when solving systems $Ax = b$.
- If b is the result of a previous calculation, then $\|\delta b\|/\|b\|$ is at best equal to u machine epsilon.
- The inaccuracies in x will then be at best $u \cdot cond(A)$.

Interpretation

- Orthogonal matrices have a norm of 1 and hence don't cause any scaling or distortion.
- Singular matrices have at least one singular value equal to 0 and hence have a norm of "infinity."
- "Nearly singular" matrices are the ones that cause problems.
- These are ones that have singular values "relatively close" to zero.