CS 510: NUMERICAL ANALYSIS, Fall 2019, 3 Credits

Instructor: Bahman Kalantari (kalantar@cs.rutgers.edu)
Lecture: Thursday 12:00-2:50 PM, TIL-226, LIV (Livingston).
Office Hours: Wed 1:00-2:00 PM, Hill Center 444 (also by appointment).
Prerequisites: Multivariate Calculus, Linear Algebra, Ability to program in a high level language, e.g., Matlab, Python, C, C++
Grading: Midterm 20%; Final 30%; Programming (Group) Project 50%
Teaching Assistant: Chun Leung Lau; larryl@cs.rutgers.edu; **Office:** Hill Center 427; **Office hours:** Tuesday 12:00-1:00PM
Objectives: Introduction to derivation, analysis, algorithms, and their computer implementation and application in order to solve fundamental numerical problems.
Course Outline:

- Floating point numbers and roundoff error
- Solution of nonlinear algebraic equations
  - bisection method, regula falsi, fixed point iteration, secant method, Newton's method
  - convergence rates (linear, quadratic)
  - systems of nonlinear equations - Newton's method
- Introduction to dynamical systems, fixed point iterations
  - basics of complex numbers, the geometric modulus principle, the fundamental theorem of algebra
  - Newton's method, Halley's method, the Basic Family of iteration functions
  - Taylor's theorem and generalized Taylor's theorem
  - basins of attraction, Voronoi diagrams of roots of polynomials
  - bounds on modulus of zeros of polynomials
  - Fatou and Julia sets, the Mandelbrot set
  - fractals, polynomiography
- Solution of linear algebraic systems
  - Gaussian elimination/ LU decomposition
  - special cases: symmetric, banded, sparse matrices
  - error analysis, norms, condition number
  - iterative methods (Jacobi, Gauss-Seidel, SOR, a geometric method, convergence rates)
  - overdetermined systems, least squares solutions
- Other numerical linear algebra topics + applications
  - QR decomposition
  - Singular value decomposition (SVD)
  - Web search, PageRank via power method, PageRank via a convex hull algorithm
  - some application of eigenvalue problems in optimization
- Interpolation, approximation of functions
  - the interpolating polynomial (its construction and error term)
  - piecewise polynomial interpolation, splines
  - Tchebycheff interpolation, minimax approximation
  - least squares approximation, orthogonal polynomials
- Numerical differentiation and integration
  - quadrature formulas, error terms
  - adaptive quadrature, Gaussian quadrature
  - numerical differentiation, error terms
- Numerical solution of ordinary differential equations
  - basic methods (Taylor methods, Runge-Kutta methods, multistep methods)
  - stability, consistency, convergence
  - higher order equations, systems

Some Reference:

M. T. Heath, Scientific Computing, An Introductory Survey, 2nd edition, McGraw-Hill, 2002.
G. Dahlquist & A. Bjorck, Numerical Methods, Prientice-Hall, 1974; SIAM, 2008.
G. H. Golub & C. F. Van Loan, Matrix Computations, 3rd edition, Johns Hopkins University Press, 1996.
Kendall Atkinson, An Introduction to Numerical Analysis, John Wiley & Sons, Inc., Second Edition, 1989.
Bahman Kalantari, Polynomial Root-Finding and Polynomiography, World Scientific, 2008.
Matlab tutorial + links to other references + some articles

Iteration Functions

1 Taylor Polynomial

Given a polynomial $p(z)$ of degree n and a complex number a we have

$$p(z) = p(a) + p'(a)(z-a) + \frac{p''(a)}{2!}(z-a)^2 + \cdots + \frac{p^{(n)}(a)}{n!}(z-a)^n. \tag{1}$$

Suppose θ is a root of $p(z)$, i.e. $p(\theta) = 0$. Let $z = \theta$, $a = z$ in the above we get

$$0 = p(\theta) = p(z) + p'(z)(\theta - z) + \frac{p''(z)}{2!}(\theta - z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^n. \tag{2}$$

The above holds for any root of θ and for any z. Adding z to both sides of the above, define

$$B_1(z) \equiv z - p(z) = z + p'(z)(\theta - z) + \frac{p''(z)}{2!}(\theta - z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^n. \tag{3}$$

Equivalently,

$$B_1(z) = z - p(z) = \theta + (p'(z) - 1)(\theta - z) + \frac{p''(z)}{2!}(\theta - z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^n. \tag{4}$$

Note that $B_1(\theta) = \theta$. So θ is a fixed point of $B_1(z)$.
Given z_0, define the fixed point iteration

$$z_{k+1} = B_1(z_k), \quad k \geq 0. \tag{5}$$

What can we say about the fixed point iteration? Will it converge when z_0 is close to θ?
From (4) we can write

$$B_1(z) - \theta = (p'(z) - 1)(\theta - z) + \frac{p''(z)}{2!}(\theta - z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^n. \tag{6}$$

Or,

$$B_1(z) - \theta = (\theta - z)\bigg((p'(z) - 1) + \frac{p''(z)}{2!}(\theta - z) + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^{n-1}\bigg). \tag{7}$$

Notice that we can write the above after factoring $(\theta - z)$ as

$$B_1(z) - \theta = (\theta - z)\bigg((p'(z) - 1) + (\theta - z)G(z)\bigg), \tag{8}$$

where $G(z)$ is a sum of terms. What is important is that when $(\theta - z)$ is small, $(\theta - z)G(z)$ is small. So if $|p'(\theta) - 1| < 1$. Then there is a neighborhood of the root θ so that for any z_0 in this neighborhood fixed point iteration converges to θ. A neighborhood of θ means the disc of some radius $r > 0$ centered at θ:

$$D_r(\theta) = \{z : |z - \theta| < r\}. \tag{9}$$

More formally, using the triangle inequality we can write,

$$|(p'(\theta)-1)+(\theta-z)G(z)| \le |(p'(\theta)-1)|+|(\theta-z)G(z)|. \tag{10}$$

So for example if $|(p'(\theta)-1)| < .9$, there will be a neighborhood where $|(p'(z)-1)| < .95$, and $|(\theta - z)G(z)| < .1$ for every z in this neighborhood.

2 Newton Method

From (2) we also get

$$zp'(z)-p(z) = \theta p'(z) + \frac{p''(z)}{2!}(\theta-z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta-z)^n. \tag{11}$$

Dividing by $p'(z)$ we get Newton's iteration function

$$B_2(z) \equiv z - \frac{p(z)}{p'(z)} = \theta + \frac{p''(z)}{p'(z)2!}(\theta-z)^2 + \cdots + \frac{p^{(n)}(z)}{p'(z)n!}(\theta-z)^n. \tag{12}$$

This means

$$B_2(z) \equiv z - \frac{p(z)}{p'(z)} - \theta \approx \frac{p''(z)}{p'(z)2!}(\theta-z)^2. \tag{13}$$

If $p'(\theta) \neq 0$, i.e. θ is a simple roots of $p(z)$, then $B_2(\theta) = \theta$, i.e. θ is a fixed point of $B_2(z)$. In fact multiple roots are also fixed points. And any fixed point of $B_2(z)$ is a root of $p(z)$.

Given z_0, define the fixed point iteration

$$z_{k+1} = B_2(z_k), \quad k \ge 0. \tag{14}$$

We have,

$$B_2(z) - \theta = \frac{p''(z)}{p'(z)2!}(\theta-z)^2 + \cdots + \frac{p^{(n)}(z)}{p'(z)n!}(\theta-z)^n. \tag{15}$$

This means if $p'(\theta) \neq 0$, there is a neighborhood around θ so that starting with any z_0 in this neighborhood the fixed point iteration converges. Furthermore,

$$\lim_{k\to\infty} \frac{z_{k+1}-\theta}{(\theta-z_k)^2} = \frac{p''(\theta)}{p'(\theta)2!}. \tag{16}$$

We say the rate of convergence is quadratic. This, roughly speaking, says when z_k is close enough to a root θ the error in each iteration doubles. Thus if it is 10^{-1}, you expect the next errors to be approximately 10^{-2}, 10^{-4}, 10^{-8}, etc.

3 Newton Method for Multiple Roots

Given a polynomial $p(z)$ and a root θ we say it is a root of multiplicity m if

$$p(z) = (z-\theta)^m q(z),$$

where $m \ge 1$ and $q(\theta) \neq 0$. When $m = 1$ we say θ is a simple root.

It is easy to show that in Newton's method,

$$B_2'(\theta) = 1 - \frac{1}{m}.$$

This implies

$$\lim_{k\to\infty} \frac{z_{k+1}-\theta}{(z_k-\theta)} = \frac{m-1}{m}. \tag{17}$$

4 Some Facts on Dynamics of Newton's Method

The basin of attraction of a root θ of a polynomial is the set of all input z_0 so that orbit of z_0, denoted by $O(z_0) = \{z_1, z_2, \dots, \}$, converges to θ. The basin of attraction is denoted by $A(\theta)$.

What kind of a set it it?

A subset O of the Euclidean plane is said to be open if given any point z_0 in O, there is an open disk $D_r(z_0) = \{z : |z - z_0| < r\}$ that is contained in O.

We claim $A(\theta)$ is an open set. First, there is a neighborhood, open disk at θ, say $D_r(\theta)$ for which any point in it converges to θ.

Fact: Newton's iteration function is continuous at θ.

Fact: Under continuity, inverse image of an open set is an open set.

$B_2^{-1}(D_r(\theta)) = \{z : B_2(z) \in D_r(\theta)\}$.

Now if $O(z_0)$ lies in $D_r(\theta)$, that is if the orbit at z_0 converges to θ, we claim that there is an open neighborhood of z_0, say $D_t(z_0)$, for some $t > 0$, such that any point z' in $D_t(z_0)$, the orbit of z' converges to θ. To prove the existence of $D_t(z_0)$, we use the fact that since z_0 converge to θ this means after so many Newton iterations, say N iterations, all the subsequent iterates stay in $D_r(\theta)$. Now take the inverse of $D_r(\theta)$. This is an open set, say O_1. Take the inverse image of O_1, say O_2. This is an open set. So after N inverse images we get an open set O_N which contains z_0 and every point in O_N converges to θ.

Immediate basin of attraction: The largest connected component of $A(\theta)$ that is contains θ.

5 Halley Method

We start again with the equation

$$0 = p(z) + p'(z)(\theta - z) + \frac{p''(z)}{2!}(\theta - z)^2 + \cdots + \frac{p^{(n)}(z)}{n!}(\theta - z)^n. \tag{18}$$

Using the above and a mixture of $B_1(z)$ and $B_2(z)$ we would like to make a new iteration function that its order of convergence is cubit.

First note

$$B_1(z) - B_2(z) = -p(z) + \frac{p(z)}{p'(z)} = (p'(z) - 1)(\theta - z) + \sum_{i=2}^{n} \frac{(p'(z) - 1)p^{(i)}(z)}{i!p'(z)}(\theta - z)^i. \tag{19}$$

Multiply the above by $p(z)$ and (18) by $-(p'(z) - 1)(\theta - z)$ and adding, we get

$$p(z)(B_1(z) - B_2(z)) = p^2(z)\frac{(1 - p'(z))}{p'(z)} = \sum_{i=2}^{n} u_i(z)(\theta - z)^i, \tag{20}$$

where

$$u_i(z) = (p'(z) - 1)\left(\frac{p(z)p^{(i)}(z)}{i!p'(z)} - \frac{p^{(i-1)}(z)}{(i-1)!}\right). \tag{21}$$

Multiplying (20) by

$$\frac{-p''(z)}{2p'(z)u_2(z)} \tag{22}$$

and adding it to the expansion of $B_2(z)$ we get

$$B_3(z) \equiv z - p(z)\frac{p'(z)}{p'(z)^2 - p(z)p''(z)/2} = \theta + \sum_{i=3}^{n} v_i(z)(\theta - z)^i, \tag{23}$$

where

$$v_i(z) = \left(\frac{p^{(i)}(z)}{i!p'(z)} - \frac{p''(z)}{2p'(z)}\frac{u_i(z)}{u_2(z)}\right). \tag{24}$$

The above iteration function is called Halley's method and has cubic order of convergence. This roughly means near a simple root if the current error is 10^{-1}, the next one roughly 10^{-3}, 10^{-6}, etc.

6 Horner's Method

Consider a polynomial

$$p_n(z) = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0.$$

To efficiently evaluate the polynomial at a point z_0 we use nested multiplication:

$$p(z_0) = (\cdots((a_n z_0 + a_{n-1})z_0 + a_{n-2})z_0 + \cdots + a_1)z_0 + a_0.$$

Let $b_n = a_n$ and recursively define

$$b_{n-m} = b_{n-m+1} z_0 + a_{n-m}, \quad m = 1, \ldots, n.$$

Performing n multiplications and n additions we evaluate $p(z)$ at z_0 to get

$$p(z_0) = b_0.$$

Let

$$p_1(z) = b_n z^{n-1} + b_{n-2} z^{n-2} + \cdots + b_2 z + b_1.$$

We have

$$p_1(z)(z - z_0) + b_0 = p(z).$$

From the above, differentiating we get

$$p'(z) = p_1'(z)(z - z_0) + p_1(z). \tag{25}$$

Thus

$$p'(z_0) = p_1(z_0).$$

So from Horner's recursion we get $p(z_0)$ and $p'(z_0)$. By repeating this process we can compute all the normalized derivatives of $p(z)$ at z_0. By induction from (25) we get

$$p^{(i)}(z) = p_1^{(i)}(z)(z - z_0) + i p_1^{(i-1)}(z). \tag{26}$$

Substituting $z = z_0$ gives

$$p^{(i)}(z_0) = i p_1^{(i-1)}(z_0), \quad i = 1, \ldots, n. \tag{27}$$

To summarize, given a polynomial

$$p_n(z) = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0,$$

and a particular z_0, we can compute all the normalized derivatives of $p(z)$ at z_0:

$$b_i^{(i)} = \frac{p^i(z_0)}{i!}, \quad i = 0, \ldots, n,$$

Set

$$b_m^{(-1)} = a_m, \quad m = 0, \ldots, n.$$

For $i = 0, \ldots, n$, do

$$b_n^{(i)} = b_n^{(i-1)}.$$

For $m = 1, \ldots, n - i$, do

$$b_{n-m}^{(i)} = z_0 b_{n-m+1}^{(i)} + b_{n-m}^{(i-1)}.$$

Robust Newton Method for Polynomials

Bahman Kalantari

1 Introduction

Consider a complex polynomial

$$p(z) = a_n z^n + \cdots + a_1 z + a_0, \tag{1}$$

with coefficients $a_j \in \mathbb{C}$, $z = x+iy$, $i = \sqrt{-1}$, and $x, y \in \mathbb{R}$. The Newton iterations are defined recursively by the formula

$$z_{j+1} = z_j - \frac{p(z_j)}{p'(z_j)}, \quad j = 0, 1, ..., \tag{2}$$

where $z_0 \in \mathbb{C}$ is the starting point, or *seed*, and $p'(z)$ is the derivative of $p(z)$. The sequence $\{z_j\}_{j=0}^{\infty}$ is called the *orbit* of z_0. The *basin of attraction* of a root θ of $p(z)$ is the set of all seeds z_0 whose orbit converges to θ. The *immediate* basin of attraction of a root is the *maximal connected component* of the basin of attraction containing the root. An interpretation of Newton iterations is the application of *fixed point* iterations to the rational function $N_p(z) = z - p(z)/p'(z)$. Notably, the iterate z_{j+1} is undefined if z_j is a *critical point* of $p(z)$, *i.e.,* if $p'(z_j) = 0$. When the iterate z_{j+1} is defined, it can be interpreted as the root of linearized approximation to $p(z)$.

The *modulus* of a complex number $z = x + iy$ is $|z| = \sqrt{x^2 + y^2}$. Equivalently, $|z| = \sqrt{z\overline{z}}$, where $\overline{z} = x - iy$ is the *conjugate* of z. In general, Newton iterates do not necessarily monotonically decrease the modulus of the polynomial after each iteration: at an arbitrary step j, we may obtain $p(z_{j+1})$ such that $|p(z_{j+1})| > |p(z_j)|$. For example, if $p(z) = z^2 - 1$, then $|p(z_{j+1})| >> |p(z_j)|$ for small $|z_j|$. However, near *simple* roots θ (i.e. $p'(\theta) \neq 0$), the rate of convergence is quadratic, thus requiring very few iterations to get highly accurate approximations. Another drawback of the Newton Method is that its orbits need not even converge; some cycle. For instance, in the case of $p(z) = z^3 - 2z + 2$, the Newton iterate at $z_0 = 0$ is $z_1 = 1$ and the iterate at z_1 is z_0, resulting in a cycle between 0 and 1. Finally, an orbit may converge yet be complex to the point of practical unwieldiness, even when $p(z)$ is a cubic polynomial.

In this forgoing approach we consider minimization of the modulus of a complex polynomial $p(z)$. It is more convenient to consider square of the modulus,

$$F(z) = |p(z)|^2 = p(z)\overline{p(z)}. \tag{3}$$

Clearly, θ is a root of $p(z)$ if and only if $F(\theta) = 0$.

2 The Robust Newton Method

Given $z_0 \in \mathbb{C}$ with $p(z_0) \neq 0$, define the following quantities:

$$\left.\begin{array}{l} k = \min\{j \in \{1,\dots,n\} : p^{(j)}(z_0) \neq 0\} \\ u_k = \dfrac{1}{k!} p(z_0)\overline{p^{(k)}(z_0)} \\ \gamma = \ \ 2 \cdot Re(u_k^{k-1}) \\ \delta = -2 \cdot Im(u_k^{k-1}) \\ c_k = \max\{|\gamma|, |\delta|\} \end{array}\right| \quad \theta = \begin{cases} 0, & \text{if } c_k = |\gamma|, \gamma < 0 \\ \pi/k, & \text{if } c_k = |\gamma|, \gamma > 0 \\ \pi/(2k), & \text{if } c_k = |\delta|, \delta < 0 \\ 3\pi/(2k), & \text{if } c_k = |\delta|, \delta > 0 \end{cases} \quad \left| \begin{array}{l} \\ \\ \\ \\ \\ A = \max_{j\geq 0}\{\frac{|p^{(j)}(z_0)|}{j!}\}. \end{array}\right. \tag{4}$$

In particular, note that A is equal to the maximum of the modulus taken over the coefficients of the Taylor expansion of $p(z)$ at z_0.

Definition 1. The *robust Newton iterate* at z_0 is

$$\widehat{N}_p(z_0) = z_0 + \frac{C_k}{3}\frac{u_k}{|u_k|}e^{i\theta}, \quad C_k = \frac{c_k|u_k|^{2-k}}{6A^2}. \tag{5}$$

We call $(u_k/|u_k|)e^{i\theta}$ the *normalized robust Newton direction* at z_0. We call $C_k/3$ the *step-size*. In particular, when $k = 1$, we have $c_1 = 2$ and $\theta = \pi$. Thus $e^{i\theta} = e^{i\pi} = -1$, and $C_1 = |u_1|/3A^2$ so that

$$\widehat{N}_p(z_0) = z_0 - \frac{p(z_0)\overline{p'(z_0)}}{9A^2} = z_0 - \frac{|p'(z_0)|^2}{9A^2}\left(\frac{p(z_0)}{p'(z_0)}\right). \tag{6}$$

Definition 1 defines the Robust Newton Method everywhere, including at critical points. In particular, when $k = 1$ the normalized robust Newton direction is simply a positive scalar multiple of the standard Newton direction. Also, by the definition of A, $|p'(z_0)|^2/9A^2 \leq 1/9$. Thus the robust Newton iterate always lies on the line segment between z_0 and the standard Newton iterate, $z_0 - p(z_0)/p'(z_0)$. This seemingly simple modification when $k = 1$, together with the ability to define the iterates when $k > 1$, will guarantee that the polynomial modulus at the new point, $z_1 = \widehat{N}_p(z_0)$, will necessarily decrease by a computable estimate.

3 A Generic Robust Newton Method

Algorithm 1 Robust Newton Method

Pick $z_0 \in \mathbb{C}$
$t \leftarrow 0$
while $|p(z_t)p'(z_t)| \neq 0$, **do**
  $z_{t+1} \leftarrow \widehat{N}_p(z_t)$, $t \leftarrow t + 1$
end while

Except at most $(n-1)$ critical points, the index k equals 1 so that the iterate is defined according to (6). This simple modification of Newton Method assures global convergence to a root or a critical point of $p(z)$, while reducing $|p(z)|$ at each iteration.

The orbit of z_0 will necessarily converge to a root of $p(z)$ if $|p(z_0)|$ is below a critical threshold. This threshold is the minimum of $|p(z)|$ taken over critical points that are not roots of $p(z)$.

Now let $\varepsilon \in (0, 1)$ be a selected tolerance. We wish to iterate the algorithm until z_t satisfies $|p(z_t)| \leq \varepsilon$. However, the algorithm may produce instead a point with $|p(z_t)p'(z_t)| \leq \varepsilon$. To turn the generic algorithm into a practical one we consider a modification.

It can be shown that as long as $|p(z_t)p'(z_t)| \geq \varepsilon$, each iteration of the Robust Newton Method decreases $F(z)$ by at least $\varepsilon^2/9A^2$. When $|p(z_t)| > \varepsilon$, but $|p(z_t)p'(z_t)| < \varepsilon$, the decrement could be small. In such a case $p'(z_t)$ is small, giving an indication that the subsequent iterates may be converging to a critical point. To avoid this, we treat z_t as if it were a critical point and redefine its index as the smallest k such that $|p(z_t)p^{(k)}(z_t)|/k! \geq \varepsilon$. Then we proceed to define the next iterate as if z_t were a critical point with index k. Such an index is well defined for any ε less than $|p^{(n)}(z)|/n!$, the modulus of the coefficient of z^n in $p(z)$. Since we have adjusted the next iterate, z_{t+1}, the inequality $|p(z_{t+1})| < |p(z_t)|$ may not hold. If the inequality holds, we have succeeded to reduce the modulus and proceed as usual. However, if $|p(z_{t+1})| \geq |p(z_t)|$, we return to z_t and proceed to compute the next robust Newton iterate, repeating this process. Eventually, using this scheme, either we avoid convergence to a critical point while monotonically reducing $|p(z)|$, or the sequence of iterates will near a critical point. However, by continuity and the formula for robust Newton iterate at a critical point, we can be assured that the scheme explained here will escape the critical point and from that point on we proceed as usual.

3.1 Examples

Example 1. Consider the case where $p(z) = z^2 - 1$. As proven by Cayley, for any seed z_0 not on the y-axis, the orbit of z_0 under Newton iteration defined by $N_p(z) = z - (z^2 - 1)/2z$ converges to the root closest to z_0. No point on the y-axis converges. However, the orbits are different for the Robust Newton Method. We consider robust Newton iterations for $z_0 = 0$ and $z_0 = \varepsilon i$, $\varepsilon > 0$. Consider $z_0 = 0$. From Definition 1 and the values $p(0) = -1$, $p'(0) = 0$, and $p''(0)/2 = 1$, we get $A = 1$, $k = 2$, $u_2 = -1$, $u_2/|u_2| = -1$, $\gamma = -2$, $c_2 = 2$, $\theta = 0$, $e^{i\theta} = 1$ and $C_2 = 1/3$. It follows from (5) that the robust Newton iterate is $z_1 = -1/9$. The decrement is $F(z_1) - F(z_0) \approx -2/81$. Next, let $z_0 = \varepsilon i$, $\varepsilon > 0$. Then $p(z_0) = -(1 + \varepsilon^2)$, $p'(z_0) = 2\varepsilon i$. Thus $k = 1$, $A = \max\{1 + \varepsilon^2, 2\varepsilon, 1\} = 1 + \varepsilon^2$. Substituting these into (6) we get, $z_1 = \widehat{N}_p(z_0) = \varepsilon i - 2\varepsilon i/9(1 + \varepsilon^2)$. We see that z_1 is closer to the origin than z_0 by a factor that improves iteratively. Thus, starting with any $\varepsilon \in (0, \infty)$, the sequence $z_{k+1} = \widehat{N}_p(z_k)$ monotonically converges to the origin, a critical point. By virtue of the fact the robust Newton iterate is defined at the origin, we adjust the iterates so as to avoid convergence to it. We treat a near-critical point as if it is critical point and compute the next iterate accordingly. Thus for ε small we treat $z_0 = \varepsilon i$ as if it is a critical point with index $k = 2$. We get $u_2 = p(z_0)p''(z_0)/2 = -(1 + \varepsilon^2)$. Proceeding to define the robust Newton iterate with $k = 2$, $u_2/|u_2| = -1$, $\gamma = 2u_2 = -2(1 + \varepsilon^2)$. Thus $c_2 = 2(1 + \varepsilon^2)$ and $\theta = 0$ so that $e^{i\theta} = 1$ and $C_2 = 1/3(1 + \varepsilon^2)$. Thus the robust Newton iterate becomes $z_1 = -1/9(1 + \varepsilon^2) + \varepsilon i$. It is easy to see that for ε small enough $|p(z_1)| < 1 = |p(0)| < |p(z_0)|$. This together with the fact that in each iteration the Robust Newton Method decreases the current polynomial modulus implies the subsequent iterates will never get closer to the origin. In summary, by treating a near-critical point as a critical point and using the robust Newton iterates we have bypassed a critical point for good.

Example 2. Consider $p(z) = z^3 - 2z + 2$ at $z_0 = 0$ and $z_0 = 1$, the points in a cycle. If we pick $z_0 = \sqrt{2/3}$, a critical point, Newton's method is not defined. The only way to decrease the modulus here is to move into the complex plane; doing so is possible with the Robust Newton Method, and the orbit of z_0 will converge to a root as expected.

4 Programming Project

A number of programming projects can be described based on the Robust Newton Method. Some of them are described below.

1. Implement the Robust Newton Method and test it on some polynomial. You can also combine it with the usual Newton Method in this way:

Given an iterate z_t, generate Newton's iterate $z_{t+1} = z_t - p(z_t)/p'(z_t)$. If $|p(z_{t+1}| < |p(z_t)|$, pick z_{t+1} to be the next iterate. Otherwise, using z_t select z_{t+1} to be based on Robust Newton Method. Give a

thorough description of the performance of these on some generated polynomials.

2. Produce a polynomiography of the performance of Robust Newton Method, and the combination of Robust Newton Method and Newton Method. This means pick a square, divide it into pixels and for each pixel iterate the Robust Newton Method and do color coding.

3. Use the Robust Newton Method to compute all roots of $p(z)$. This will be described later.

Chapter 16

Bounds on Roots of Polynomials and Analytic Functions

In this chapter we make use of the Basic Family to derive an infinite family of lower bounds on the gap between two distinct zeros of a given analytic function $f(z)$. We then use the bounds to compute lower bounds on the distance from an arbitrary complex point to the nearest root of $f(z)$. In particular, when $f(z)$ is a polynomial, for each $m \geq 2$ we give explicit upper and lower bounds, U_m and L_m on the modulus of zeros. These bounds are efficiently computable and have many theoretical and practical applications, for instance in Weyl's classical quad-tree algorithm for computing all roots of a complex polynomial. McNamee and Olhovsky (2005) computational comparison shows even U_4 is superior to more than 45 existing bounds in the literature. Even for $m = 2$ our estimate of lower bound is more than twice as good as Smale's bound, Smale (1986), or its improved version given in . Blum *et al.* (1998). A significant property of these bounds, as proved by Jin (2006), is their asymptotic convergence to the radii of tightest annulus containing the zeros. Jin has also given an efficient, $O(mn)$-time algorithm, for the computation of the first m bounds for a polynomial of degree n.

16.1 Introduction

Computing apriori bound on the zeros of polynomials is an interesting and important problem with many theoretical and practical applications. There is a vast literature on this topic, as see the recent book McNamee (2007). As a consequence of the main results in this chapter we will show that if $f(z) = a_n z^n + \cdots + a_1 z + a_0$ is a polynomial with $a_n a_0 \neq 0$, for each $m \geq 2$ we can state upper (and lower) bounds on its zeros the first few of which are given below.

Let $r_m \in [1/2, 1)$ be the unique positive root of the polynomial $q(t) =$

$t^{m-1}+t-1$. Assume ξ is any root of $f(z)$. Then For $m=2$, $r_2=0.5$ and we have

$$|\xi| \leq \frac{1}{r_2} \max\left\{ \left| \frac{a_{n-k+1}}{a_n} \right|^{\frac{1}{k-1}} : k=2,\ldots,n+1 \right\}.$$

For $m=3$, $r_3=0.618034$ and we have

$$|\xi| \leq \frac{1}{r_3} \max\left\{ \left| \frac{1}{{a_n}^2} \det \begin{pmatrix} a_{n-1} & a_{n-k+1} \\ a_n & a_{n-k+2} \end{pmatrix} \right|^{\frac{1}{k-1}} : k=3,\ldots,n+2 \right\},$$

where $a_{-1}=0$.

For $m=4$, $r_4=0.682328$ and we have

$$|\xi| \leq \frac{1}{r_4} \max\left\{ \left| \frac{1}{{a_n}^3} \det \begin{pmatrix} a_{n-1} & a_{n-2} & a_{n-k+1} \\ a_n & a_{n-1} & a_{n-k+2} \\ 0 & a_n & a_{n-k+3} \end{pmatrix} \right|^{\frac{1}{k-1}} : k=4,\ldots,n+3 \right\},$$

where $a_{-1}=a_{-2}=0$.

For $m=5$, $r_5=0.724492$ and we have

$$|\xi| \leq \frac{1}{r_5} \max\left\{ \left| \frac{1}{{a_n}^4} \det \begin{pmatrix} a_{n-1} & a_{n-2} & a_{n-3} & a_{n-k+1} \\ a_n & a_{n-1} & a_{n-2} & a_{n-k+2} \\ 0 & a_n & a_{n-1} & a_{n-k+3} \\ 0 & 0 & a_n & a_{n-k+4} \end{pmatrix} \right|^{\frac{1}{k-1}} : k=5,\ldots,n+4 \right\},$$

where $a_{-1}=a_{-2}=a_{-3}=0$.

16.2 Estimate to Zeros of Analytic Functions

Let $f(z)$ be a complex-valued function analytic everywhere on the complex plane. Consider Newton's iteration function

$$N(z) = z - \frac{f(z)}{f'(z)}. \tag{16.1}$$

Define

$$\gamma(z) = \sup \left\{ \left| \frac{f^{(k)}(z)}{f'(z)k!} \right|^{1/(k-1)}, k \geq 2 \right\}. \tag{16.2}$$

From Smale's analysis of the one-point theory for Newton's method the following theorem is deducible:

Theorem 16.1 (Smale (1986)). *If ξ, ξ' are distinct zeros of f, ξ a simple zero, then they are separated by a distance according to*

$$|\xi - \xi'| \geq \frac{3 - \sqrt{7}}{2\gamma(\xi)} \approx \frac{.177}{\gamma(\xi)}. \tag{16.3}$$

The following stronger lower bound is given in Blum *et al.* (1998) (Corollary 1, page 158):

$$|\xi - \xi'| \geq \frac{5 - \sqrt{17}}{4\gamma(\xi)} \approx \frac{.219}{\gamma(\xi)}. \tag{16.4}$$

Such theorems are referred as *separation theorems*. Dediue (1997) gives separation theorems for system of complex polynomials and in particular polynomials in one complex variable.

In this chapter we will derive a family of lower bounds indexed by an integer $m \geq 2$ on the gap of Theorem 16.1 which in particular when $m = 2$ improves (16.3) as well as (16.4) by replacing their lower bounds with $1/(2\gamma(\xi))$ which is more than twice as good. Our results make use of the Basic Family, $\{B_m(z), m = 2, \dots\}$.

The chapter is organized as follows: In Section 2, we describe the Basic Family and its significant relevant properties for complex polynomials. We then extend these to the case of analytic functions. In Section 3, we make use of the Basic Family to derive lower bounds on the distance from a simple zero of f to its nearest distinct zero. In Section 4, we make use of the preceding lower bounds to derive lower bounds on the distance between an arbitrary point and the nearest root of f. In particular using the latter result we show that given a complex polynomial f, for each $m \geq 2$ we can compute an annulus containing the roots. In Section 5, we consider the application of the bounds on the modulus of roots within Weyl's algorithm. We conclude the chapter in Section 6.

16.3 The Basic Family for General Analytic Functions

Assume that $f(z)$ is a complex polynomial of degree n. Consider the Basic Family:

$$B_m(z) \equiv z - f(z) \ \frac{D_{m-2}(z)}{D_{m-1}(z)}, \tag{16.5}$$

where for each $m \geq 2$, $D_0(z) = 1$, and for each $m \geq 1$

$$D_m(z) = \det \begin{pmatrix} f'(z) & \frac{f''(z)}{2!} & \dots & \frac{f^{(m-1)}(x)}{(m-1)!} & \frac{f^{(m)}(z)}{(m)!} \\ f(z) & f'(z) & \ddots & \ddots & \frac{f^{(m-1)}(z)}{(m-1)!} \\ 0 & f(z) & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \frac{f''(z)}{2!} \\ 0 & 0 & \dots & f(z) & f'(z) \end{pmatrix}. \tag{16.6}$$

If ξ is a root of f, $D_m(\xi) = f'(\xi)^m$. Thus, whether or not ξ is a simple root of f it is a fixed-point of B_m since we have

$$B_m(\xi) = \xi - f(\xi)\frac{f'(\xi)^{m-2}}{f'(\xi)^{m-1}} = \xi - \frac{f(\xi)}{f'(\xi)} = \xi.$$

With

$$\widehat{D}_{m,k}(z) = \det \begin{pmatrix} \frac{f''(z)}{2!} & \frac{f'''(z)}{3!} & \dots & \frac{f^{(m)}(z)}{(m)!} & \frac{f^{(k)}(z)}{k!} \\ f'(z) & \frac{f''(z)}{2!} & \ddots & \frac{f^{(m-1)}(z)}{(m-1)!} & \frac{f^{(k-1)}(z)}{(k-1)!} \\ f(z) & f'(z) & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \frac{f''(z)}{2!} & \frac{f^{(k-m+2)}(z)}{(k-m+2)!} \\ 0 & 0 & \dots & f'(z) & \frac{f^{(k-m+1)}(z)}{(k-m+1)!} \end{pmatrix} \tag{16.7}$$

where $m \geq 1$, and $k \geq (m+1)$, the following theorem is already proved in Chapter 11 (Corollary 11.2, Chapter 11) is a consequence of the main determinantal theorem.

Theorem 16.2. *Assume that $f(z)$ is a complex polynomial of degree n. Let ξ be a root of $f(z)$. Then, except for finitely many values of $z \in C$, $B_m(z) \in C$, and*

$$B_m(z) = \xi + \sum_{k=m}^{m+n-2} (-1)^m \frac{\widehat{D}_{m-1,k}(z)}{D_{m-1}(z)}(\xi - z)^k. \tag{16.8}$$

We will now proceed by proving a more general version of Theorem 16.2

Theorem 16.3. *Let $f(z)$ be a complex-valued function analytic over the entire complex plane. For each $m \geq 2$, define $B_m(z)$ as in (16.5). Then $B_m(z)$ satisfies (16.8). Then,*

$$B_m(z) = \xi + \sum_{k=m}^{\infty} (-1)^m \frac{\widehat{D}_{m-1,k}(z)}{D_{m-1}(z)}(\xi - z)^k. \tag{16.9}$$

Lecture 4

CS 510

NORM of vectors & matrices

Consider $\mathbb{R}^n$ & $\mathbb{C}^n$

$x \in \mathbb{R}^n$

$\|x\|_2 = \left(\sum x_i^2\right)^{1/2}$

$\|x\|_p = \left(\sum x_i^p\right)^{1/p}$, $1 \le p \le \infty$

In general a function N from $\mathbb{R}^n \longrightarrow \mathbb{R}_+$ is a norm if

$N(x) \geq 0, \forall x \in \mathbb{R}^n$

$N(x) = 0 \iff x = 0$

$N(\alpha x) = |\alpha| N(x), \forall \alpha \in \mathbb{R}, x \in \mathbb{R}^n$

$N(x+y) \leq N(x) + N(y)$

The Euclidean norm $\|x\|_2$ satisfies this.

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

$$\|x\|_\infty = \max\{|x_i| : i = 1, \cdots, n\}$$

A subset C of $\mathbb{R}^n$ is convex if whenever $x, y \in C$

$$\alpha x + (1-\alpha) y \in C$$

$$\forall \ \alpha \in [0,1]$$

The unit ball wrt
respect to a norm $\|\cdot\|$

is

$$B = \{x : \|x\| \leq 1\}$$

claim: B is convex

$x, y \in B$, $\|\alpha x + (1-\alpha)y\| \leq$

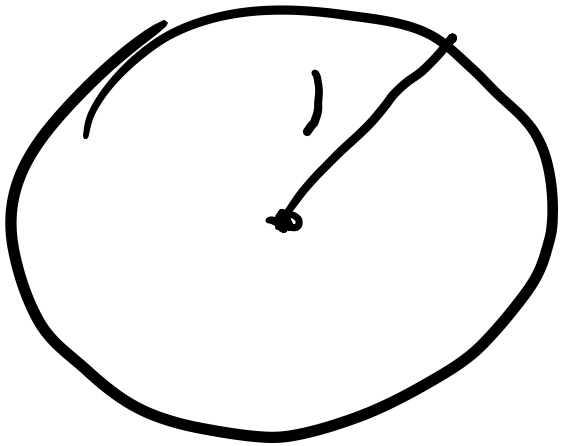
$\|\alpha x\| + \|(1-\alpha)y\| = \alpha\|x\| +$
$(1-\alpha)\|y\| \leq 1$

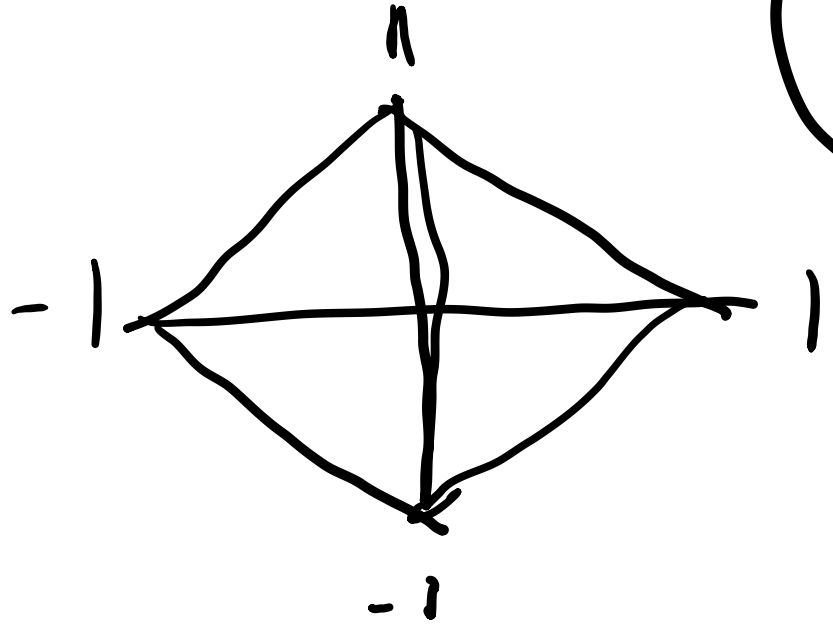
$\|\cdot\|_2$

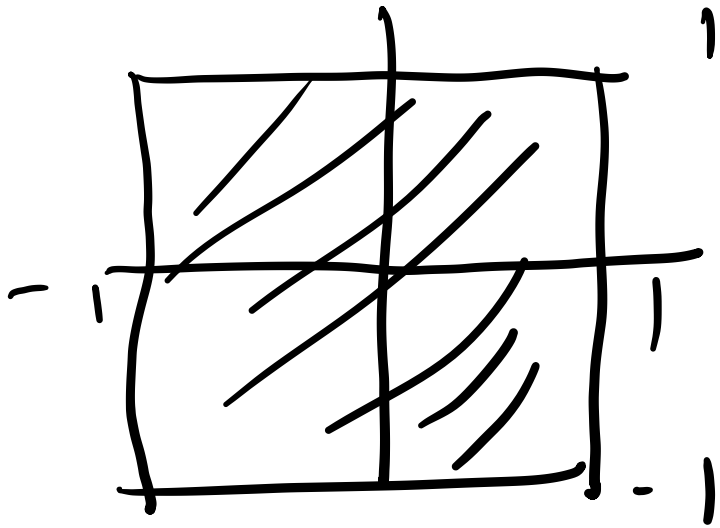
$\|\cdot\|_1$

$\|\cdot\|_\infty$

B







If $x \in \mathbb{C}^n$,

$$\|x\|_2 = \left(\sum_{i=1}^{n} x_i \bar{x}_i \right)^{1/2}$$

$\bar{x}_i$ = Conjugate of x

$$\|x\|_1 = \sum |x_i| \quad , \quad |x_i| = \sqrt{x_i \bar{x}_i}$$

$$\|x\|_\infty = \max \{ |x_i| : i = 1, \ldots, n \}.$$

Given a sequence of pts

$\{x^0, x^1, \ldots, x^k, \} \subseteq \mathbb{R}^n$

we write $x^k \longrightarrow x_*$

if $\| x^k - x_* \| \longrightarrow 0$

Consider the set of $n \times n$ real or complex matrices.

They can be viewed as vectors in $\mathbb{R}^{n^2}$ or $\mathbb{C}^{n^2}$.

$\therefore$ we can speak of norm for matrices.

Given $n \times n$ matrices we consider a function defined over them as a norm if it satisfies:

1. $N(A) \geq 0$, $\forall A$, $N(A) = 0 \iff A = 0$
2. $N(\alpha A) = |\alpha| N(A)$
3. $N(A + B) \leq N(A) + N(B)$

plus two additional properties

4. $N(AB) \leq N(A)\,N(B)$

5. $N(Ax) \leq N(A)\,\|x\|_2$

We write for $N(A) = \|A\|$
But will specify what the norm is..

Given any vector norm on $x \in \mathbb{R}^n$ or $\mathbb{C}^n$, it induces a norm on A.

Example will be given with $\|\cdot\|_1$, $\|\cdot\|_2$, $\|\cdot\|_\infty$.

Matrix norm

Let A be $m \times n$ matrix over $\mathbb{R}$ or $\mathbb{C}$.

Given any norm $\|\cdot\|$ on $\mathbb{R}^n$ or $\mathbb{C}^n$, we define

$$\|A\| = \sup_{\|x\| \leq 1} \|Ax\|, \quad \|x\| \leq 1 \text{ can be replaced w. } \|x\| = 1$$

What is $\|A\|_1$?

Pick $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$$Ax = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix}$$

$$\|Ax\|_1 = |a_{11}x_1 + a_{12}x_2| + |a_{21}x_1 + a_{22}x_2|$$

When is this sum maximized if $|x|_1 \leq 1$

If we choose $x_1 = \pm 1$, then

$$\|Ax\|_1 = |a_{11}| + |a_{21}|$$

If we choose $x_2 = \pm 1$, then

$$\|Ax\|_1 = |a_{12}| + |a_{22}|$$

So $\|A\|_1 = \max\{|a_{11}| + |a_{21}|, |a_{12}| + |a_{22}|\}$

$\therefore \|A\|_1 =$ max of 1-norm of columns of A

Works for general A.

What is $\|A\|_\infty$?

Pick $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$$Ax = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix}$$

$$\|Ax\|_\infty = \max_{\|x\|_\infty = 1} \left\{ \begin{matrix} |a_{11}x_{11} + a_{12}x_2| \\ |a_{21}x_1 + a_{22}x_2| \end{matrix} \right\}$$

If $\|x\|_\infty = 1$ it means $|x_1| = 1 = |x_2|$ we can have

So

$\|A\|_\infty$ = maximum of $\|\ \|_1$ norm of rows of A.

What is $\|A\|_2$?

$$\|A\|_2 = \max_{\|x\|_2 \leq 1} \|Ax\|_2$$

How to compute this ?

Let A be an $n \times n$ real matrix.

Eigenvalues of A are defined by $Ax = \lambda x$ where $x \neq 0$

We say λ is an eigenvalue of A if $Ax = \lambda x$ for some $x \neq 0$.

Ex. $A = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$

$$Ax = \begin{bmatrix} x_1 - x_2 \\ -x_1 + 2x_2 \end{bmatrix}$$

When is $Ax = \lambda x$?

This happens if

$$\det(A - \lambda I) = 0$$

$$A - \lambda I = \begin{bmatrix} 1-\lambda & -1 \\ -1 & 2-\lambda \end{bmatrix}$$

$$\det(A - \lambda I) = (1-\lambda)(2-\lambda) + 1$$

$$= \lambda^2 - 3\lambda + 3 = 0$$

$$\lambda = \frac{+3 \pm \sqrt{9-12}}{2} = \frac{3 \pm \sqrt{3}\, i}{2}$$

So eigenvalues of real matrix could be complex numbers.

And the eigenvectors of A then complex vectors.

Given $A = (a_{ij})$,

$A^T = (a_{ji})$, transpose of A.

$A^* = (\overline{a_{ji}})$, conjugate transpose

Given $x \in \mathbb{R}^n$, $x^T x = \|x\|_2^2$.

Given $x \in \mathbb{C}^n$, $x^* x = \|x\|_2^2$

Given $x, y \in \mathbb{R}^n$,

$$x^T y = \sum x_i y_i = \langle x, y \rangle$$

inner product

Given $x, y \in \mathbb{C}^n$

$$x^* y = \sum \bar{x}_i y_i = \langle x, y \rangle$$

inner product

$$\langle x, y \rangle \leq \|x\|_2 \|y\|_2$$

Cauchy-Schwartz inequality

Given an $n \times n$ matrix A, Eigenvalues of A are solutions to

$$f_A(\lambda) = \det(A - \lambda I) = 0$$

$$\det(A - \lambda I) = \det \begin{bmatrix} a_{11} - \lambda & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} - \lambda & & \\ \vdots & & \ddots & \\ a_{2n} & \cdots & & a_{nn} - \lambda \end{bmatrix}$$

$$f_A(\lambda) = (a_{11}-\lambda)(a_{22}-\lambda)\cdots(a_{nn}-\lambda)$$
$$+ \text{ polynomial of degree} \leq n-2$$

$f_A(\lambda)$ has n roots.

$$f_A(\lambda) = (-1)^n \lambda^n +$$
$$(-1)^{n-1}(a_{11}+\cdots+a_{nn})\lambda^{n-1} + \cdots$$

Note that sum of the roots of $f_A(\lambda) = 0$ is

$$a_{11} + \cdots + a_{nn}$$

called Trace of A

$$TR(A)$$

What is $\|A\|_2$?

Suppose A is real.

$$\|A\|_2 = \max_{\|x\|_2 \le 1} \|Ax\|_2$$

What $(\|Ax\|_2)^2 = (Ax)^T(Ax)$

$$= x^T A^T A x$$

Clearly, maximizing this over $\|x\|_2 \le 1$

This is equivalent to

$$\max \; x^T A^T A x$$

$$\|x\|_2 = 1 \quad , \quad x^T x = 1$$

From Lagrange multiplier optimality condition: (gradient of function is proportional to gradient of constraint)

$$A^T A x = \lambda x$$

This in particular means λ is eigenvalue

From $A^TAx = \lambda x$

and $\|x\|_2 = 1$ we get

$$x^TA^TAx = \lambda x^Tx = \lambda$$

This means the maximum occurs when λ is the largest eigenvalue of A^TA. So $\|A\|_2^2 = \sqrt{r(A^TA)}$

$r(A^TA)$ = (largest eigenvalue of A^TA)

Largest eigenvalue of A^TA is a real number because eigenvalues of real symmetric matrix are real.

So for $n \times n$ real matrix A

$$\|A\|_2 = \sqrt{r(A^TA)}$$

If A is symmetric,

$$A^T = A$$

So $A^T A = A^2$

Then $\|A\|_2 =$ largest eigenvalue of A in absolute value

What is $\|A\|_2$ if A is complex?

Fact:

If $\lambda_1, \ldots, \lambda_n$ are eigenvalues & $u_1, \ldots, u_n$ corresponding set of orthogonal eigenvectors, then any $x \in \mathbb{C}^n$ $\rightarrow$

$$x = \sum \alpha_i u_i$$

Now

$$\|A\|_2 = \max_{\|x\|_2 = 1} \|Ax\|_2$$

$$Ax = A\left(\sum \alpha_i u_i\right) =$$

$$\sum \alpha_i A u_i = \sum \alpha_i \lambda_i u_i$$

$$\|Ax\|_2 = \|\sum \alpha_i \lambda_i u_i\|$$

$$= \sqrt{\left(\sum \bar{\alpha}_i \bar{\lambda}_i u_i^*\right)\left(\sum \alpha_i \lambda_i u_i\right)}$$

$$= \sqrt{\sum |\alpha_i|^2 |\lambda_i|^2}$$

$$\|x\|_2 = 1 \Rightarrow \sum |\alpha_i|^2 = 1$$

So $\max \|Ax\|_2 = \max |\lambda_i|$

∴ For general A

$$\|A\|_2 = \sqrt{r(A^* A)}$$

Thm. Let $\rho(A)$ be max $|\lambda|$ such that λ is an eigenvalue of A.

Then for any matrix norm $\|A\|$, $\rho(A) \leq \|A\|$

Pf. $\rho(A) = |\lambda| = \|\lambda x\|$ for some x, $\|x\|=1$

$= \|Ax\| \leq \|A\| \|x\| = \|A\|$.

Frobenius norm of a matrix

$$F(A) = \left(\sum_{1 \le i,j \le n} |a_{ij}|^2 \right)^{1/2}$$

i.e. we look at A as a vector in $\mathbb{C}^{n^2}$.

we can show it is a matrix norm

The only part we need to prove

$$F(AB) \leq F(A)F(B)$$

Thm. Let A be $n \times n$.

Suppose $\|A\| < 1$ for some norm.

Then $I - A$ is invertible

& $(I-A)^{-1} = I + A + A^2 + \cdots + A^n + \cdots$

Pf:

$$(I - A)(I + A \cdots + A^m) =$$

$$I - A^{m+1}$$

so

$$1 + A + A^2 + \cdots + A^m = (I - A)^{-1}(I - A^{m+1})$$

As $m \to \infty$ $A^{m+1} \longrightarrow 0$

Thm. For any $n \times n$ matrix

$$I + A + \frac{A^2}{2!} + \cdots + \frac{A^n}{n!} + \cdots$$

converges to a matrix we denote by e^A

Linear systems

$$Ax = b \text{ , } A \; n \times n$$

If A is invertible then is a unique solution

$$x = A^{-1}b$$

Suppose

$$Ax = b$$

$$A\hat{x} = \hat{b}$$

where $\hat{x}$ is the unique solution to a system with perturbed b. How much does $\hat{x}$ change?

Let $\delta x = x - \hat{x}$, $\delta b = b - \hat{b}$

we get

$$A \delta x = \delta b$$

or $$\delta x = A^{-1} \delta b$$

$$\|\delta x\| \leq \|A^{-1}\| \|\delta b\| \quad \text{— (1)}$$

Also from $Ax = b$

$$\|b\| \leq \|A\| \|x\| \quad \text{— (2)}$$

So

$$\frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|} \quad \text{---} \quad (3)$$

From (1) & (3) we get

$$\frac{\|\delta x\|}{\|x\|} \leq \underbrace{\|A\| \|A^{-1}\|}_{\text{Cond}(A)} \frac{\|\delta b\|}{\|b\|}$$

Lecture 5

CS 510

Linear systems

Want to solve

$$Ax = b$$

A $n \times n$ invertible

b $n \times 1$

x $n \times 1$ unknown

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

$$A = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & & & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{pmatrix}$$

assume $a_{11}^{(1)} \neq 0$

$$A = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & & & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} \end{pmatrix}$$

Set

$$m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}} \quad , i = 2, \ldots, n$$

multiply row 1 by $-m_{i1}$ and add to row i. This gives:

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & & \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} \end{pmatrix}$$

$$a_{ij}^{(2)} = a_{ij}^{(1)} - m_{i1}\, a_{1j}^{(1)}$$

Set

$$m_{i2} = \frac{a_{i2}^{(2)}}{a_{22}^{(2)}} \quad , i = 3, \ldots, n$$

multiply row 2 by $-m_{i2}$ and add to row i. This gives:

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & 0 & a_{33}^{(3)} \cdots & a_{3n}^{(3)} \\ \vdots & \vdots & & \\ 0 & 0 & \cdots & a_{nn}^{(2)} \end{pmatrix}$$

$$a_{ij}^{(3)} = a_{ij}^{(2)} - m_{i2}\, a_{2j}^{(2)}$$

Eventually we get

$$\begin{pmatrix} a_{11}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & & \ddots & \\ 0 & & & a_{nn}^{(n)} \end{pmatrix} = U$$

Let

$$L = \begin{pmatrix} 1 & & & 0 \\ m_{21} & 1 & & \\ m_{31} & m_{32} & 1 & \\ \vdots & \vdots & & \ddots \\ m_{n1} & m_{n2} & \cdots & m_{n,n-1} \end{pmatrix}$$

Thm. If $a_{ii}^{(i)} \neq 0$, $i = 1, \cdots n-1$ then process L, U s.t.

$$A = LU$$

Operation counts:

Will count multiplication/Div

First step:

$(n-1)^2$

second step

$(n-2)^2$ etc

Overall

$$1^2 + 2^2 + \dots + (n-1)^2 = \frac{(n-1)(n)(2(n+1)+3)}{6}$$

$$\sim \frac{n^3}{3}$$

Solving $Ax = b$:

$LUx = b$

Let $y = Ux$, $Ly = b$

Solve for y, then solve for x.

These are done by forward & backward substitution respectively.

Each costs

$$1+2+\cdots+(n-1)=\frac{n(n-1)}{2}\approx\frac{n}{2}$$

$\therefore$ t_0 solve

$$Ax = b$$

Costs

$$\frac{n^3}{3} + \frac{n^2}{2} + \frac{n^2}{2}$$ operations

LU factorization with partial pivoting.

Pick pivot row to be row with largest $a_{i1}^{(1)}$ entry, $i=1,\dots,n$

This makes absolute value of multipliers to be ≤ 1

Do this for each column.

This results in

$$LU = PA$$

where P is a permutation matrix

EX.

$$x_1 - x_2 + 2x_3 = 2$$

$$-x_1 + 2x_2 + x_3 = 2$$

$$2x_1 - 4x_2 + x_3 = -1$$

$$A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & 2 & 1 \\ 2 & -4 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$$P = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 2 & -4 & 1 \\ -1 & 2 & 1 \\ 1 & -1 & 2 \end{bmatrix}$$

$$m_{21} = -\frac{1}{2}$$

$$m_{31} = \frac{1}{2}$$

$$\begin{bmatrix} 2 & -4 & 1 \\ 0 & 0 & \frac{3}{2} \\ 0 & 1 & \frac{3}{2} \end{bmatrix}, P = \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -4 & 1 \\ 0 & 1 & \frac{3}{2} \\ 0 & 0 & \frac{3}{2} \end{bmatrix} = U, P = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -4 & 1 \\ m_{21} & 0 & \frac{3}{2} \\ m_{31} & 1 & \frac{3}{2} \end{bmatrix}$$

$$\begin{bmatrix} 2 & -4 & 1 \\ m_{31} & 1 & \frac{3}{2} \\ m_{21} & 0 & \frac{3}{2} \end{bmatrix} = U$$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & & \\ -\frac{1}{2} & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & -1 & 1 \\ 0 & 1 & \frac{3}{2} \\ 0 & 0 & 7 \end{bmatrix} = \begin{bmatrix} 2 & -4 & 1 \\ 1 & -1 & 2 \\ -1 & 2 & 1 \end{bmatrix}$$

To solve $Ax = b$, we solve

$$Ly = Pb = \begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}$$

This gives $y_1 = -1$

$$\frac{1}{2} y_1 + y_2 = 2$$

$$y_2 = 2 + \frac{1}{2} = \frac{5}{2}$$

$$-\frac{1}{2} y_1 + y_3 = 2, \quad y_3 = \frac{3}{2}$$

Next we solve

$$Ux = y$$

$$\begin{bmatrix} 2 & -4 & 1 \\ 0 & 1 & \frac{3}{2} \\ 0 & 0 & \frac{3}{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ \frac{5}{2} \\ \frac{3}{2} \end{bmatrix}$$

$x_3 = 1$, $x_2 + \frac{3}{2}x_3 = \frac{5}{2}$, $x_2 = 1$

$2x_1 - 4x_2 + x_3 = -1$, $x_1 = 1$

In summary:
in LU factorization with partial
pivoting we use the matrix
A to store L & U and when
necessary we permute rows
but keep track using an array
p.

Computing A^{-1}:

First compute LU

Then compute ith column of A^{-1}, say $x^{(i)}$

by solving

$$LUx^{(i)} = e^{(i)}$$

$$e^{(i)} = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow 1 \text{ in } i\text{-th location}$$

Complexity :

$$\frac{n^3}{3} + n \cdot n^2$$

$$= \frac{4}{3} n^3$$

Iterative Method to Solve

$$Ax = b$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & a_{2n} & \cdots & a_{nn} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

$$A = D - L - U$$

$$D = \begin{bmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{bmatrix}, \quad L = \begin{bmatrix} 0 & & 0 \\ -a_{21} & \ddots & \vdots \\ \vdots & & \\ -a_{n1} & \cdots & 0 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & -a_{12} & \cdots \\ & \ddots & -a_{ij} \\ & & \ddots \\ & & 0 \end{bmatrix}$$

Jacobi Method

$$Ax = Dx - (L+U)x = b$$

so

$$Dx = (L+U)x + b$$

Now given $x^{(k)}$ as an approximation to solution of $Ax = b$, we compute $x^{(k+1)}$ from

$$D x^{(k+1)} = (L+U) x^{(k)} + b$$

or

$$x^{(k+1)} = D^{-1}(L+U) x^{(k)} + D^{-1} b$$

Let $T = D^{-1}(L+U)$, $c = D^{-1} b$

so $x^{(k+1)} = T x^{(k)} + c$

This is Jacobi method

Clearly a_{ii} must be nonzero.

$$D^{-1} = \begin{bmatrix} \frac{1}{a_{11}} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{a_{nn}} \end{bmatrix}$$

Let's write

$$T_j = D^{-1}(L+U), \quad C_j = D^{-1}b$$

Each iteration takes $O(n^2)$ operations so if we don't need to iterate many times we gain over Gaussian method

Will it converge ?

Suppose x_s is solution to

$$Ax = b$$

Then

$$x_s = T_j x_s + C \quad (1)$$

Also

$$x^{(k)} = T_j x^{(k-1)} + C \quad (2)$$

Subtracting (1) from (2)

$$x^{(k)} - x_s = T_j(x^{(k-1)} - x_s)$$

If we let

$$e^{(k)} = x^{(k)} - x^* \quad \text{(error)}$$

then

$$e^{(k)} = T_j e^{(k-1)}$$

so

$$\|e^{(k)}\| \leq \|T_j\| \cdot \|e^{(k-1)}\|$$

or

$$\|e^{(k)}\| \leq \|T_j\|^2 \cdot \|e^{(k-2)}\|$$

$$\|e^{(k)}\| \leq \|T_j\|^k \|e^{(0)}\|$$

If $\rho(T_j) < 1$, then

we can choose a matrix norm

so that $\|T_j\| < 1 \Rightarrow$ convergence

We say A is diagonally dominant if

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|.$$

Thm. If A is diagonally dominant then Jacobi method converges.

Pf.

$$T_j = D^{-1}(L+U)$$

$$\|T_j\|_\infty < 1 .$$

Recall $\|\ \|_\infty$ norm of a matrix is max $\|\ \|_1$ norm of its rows.

Gauss-Seidel

We start with

$$Ax = b$$

$$A = D - L - U$$

$$(D - L - U)x = b$$

We write

$$(D - L)x = Ux + b$$

=iteratively:

$$(D - L)x^{(k)} = Ux^{(k-1)} + b$$

$$x^{(k)} = (D - L)^{-1}Ux^{(k-1)} + (D - L)^{-1}b$$

$$x^{(k)} = T_g x^{(k-1)} + C_g$$

$$T_g = (D - L)^{-1} U$$

$$C_g = (D - L)^{-1} b$$

As in Jacobi, if

x_* is solution to $Ax = b$

$$x^{(k)} - x_* = T_g \left(x^{(k-1)} - x_* \right)$$

$$e^{(k)} = T_g e^{(k-1)} = T_g^k e^{(0)}$$

Successive over-relaxation method

SOR

$$Ax = b$$

$$A = D - L - U$$

We have

$$Lx = (-D + U)x + b$$

Let w be any scalar

$$wLx = w(-D + U)x + wb$$

add

$$Dx = Dx$$

to both sides

we set

$$(D - wL)x = [(1-w)D + wU]x + wb$$

Write

$$(D - wL)x^{(k)} = [(1-w)D + wU]x^{(k-1)} + wb$$

Then

$$x^{(k)} = T_w x^{(k-1)} + c_w$$

where

$$T_w = (D - wL)^{-1} [(1-w)D + wU]$$

$$c_w = (D - wL)^{-1} wb$$

SOR : ω

$$x^{(k)} = T_\omega x^{(k-1)} + C_\omega$$

We remark that to solve

$(D - \omega L)^{-1} u = b'$ time.

can be done in $O(n^2)$

In general Convergence of SOR & the selection of w is not so easy and may not Converse.

However, some results are known.

Thm (Kahan) If $a_{ii} \neq 0$ $\forall i$, then $\rho(T_\omega) \geq |\omega - 1|$. So SOR cannot converge if $\omega \notin (0, 2)$.

Thm (Ostrowski-Reich)

If A is PD (positive definite)

$w \in (0, 2)$, then

SOR converges for

any $x^{(0)}$.

Thm. If A is PD & tridiagonal, then $\rho(T_g) = \rho(T_j)^2 < 1$ and optimal w is

$$w = \frac{2}{1 + \sqrt{1 - \rho(T_j)^2}}$$

and with this choice of w

$$P(T_w) = w - 1.$$

Example.

$$A = \begin{bmatrix} 4 & 3 & 0 \\ 3 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}$$

Claim:

A is Positive definite,

i.e. $x^T A x > 0$

$\forall x$

$$x^T A x =$$

$$4x_1^2 + 4x_2^2 + 4x_3^2$$

$$+ 3x_1x_2 + 3x_2x_1$$

$$- x_2x_3 - x_2x_3$$

$$= 3(x_1 + x_2)^2 + (x_2 - x_3)^2$$

$$+ x_1^2 \qquad + 3x_3^2$$

If $x \neq 0$

$x^T A x > 0$

$\therefore$ Positive definite

This is > 0 $\forall x$

So A is Positive definite.

So SOR Converges for any $w \in (0, 2)$

and any initial $x^{(0)}$.

What is optimal w?

$$T_j = D^{-1}(-L-U)$$

$$= \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 0 & -3 & 0 \\ -3 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$T_J = \begin{bmatrix} 0 & -\frac{3}{4} & 0 \\ -\frac{3}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & 0 \end{bmatrix}$$

What is $P(T_j) = ?$

We need to find eigenvalues of T

$$\det(T_j - \lambda I)$$

$$\begin{vmatrix} -\lambda & -\frac{3}{4} & 0 \\ -\frac{3}{4} & -\lambda & \frac{1}{4} \\ 0 & \frac{1}{4} & -\lambda \end{vmatrix}$$

$$= -\lambda\left(\lambda^2 - \frac{1}{16}\right)$$

So $\lambda = \frac{1}{4}$ is largest

$\therefore P(T_j) = \frac{1}{4}$

opt w is

$$w = \frac{2}{1+\sqrt{1-\frac{1}{2}}} = \frac{2}{1+\frac{\sqrt{3}}{2}} \approx 1.25$$

Accelerated SOR

AOR

$$x^{(k)} = T_{\sigma,\omega} x^{(k-1)} + C_{\sigma,\omega}$$

where

$$T_{\sigma,w} =$$

$$(D-\sigma L)^{-1}\left[(1-w)D + (w-\sigma)L + wU\right]$$

$$C_{\sigma,w} = w(D-\sigma L)^{-1}b$$

If $\sigma = 0$, $w = 1$, Jacobi

If $\sigma = w = 1$, Gauss-Seidel

If $\sigma = w$, SOR

Lecture 6

CS 510

The Householder matrix and application in QR method

Householder Matrix

Let $w \in \mathbb{R}^n$ be a nonzero vector of norm $= 1$,

$$w^T w = 1$$

Define

$$H = I - 2ww^T$$

, $ww^T = [\,][\,]$

$= [n \times n]$

$$H^T = I - 2ww^T = H$$

Also note

$$HH = \left(I - 2ww^T\right)\left(I - 2ww^T\right)$$

$$= I - 4ww^T + 4(ww^T)(ww^T)$$

But $(ww^T)ww^T = w(w^Tw)w^T = ww^T$

So $H \cdot H = H^2 = I$

Suppose x is a given non-zero vector. Is there a w s.t.

$$H = I - 2ww^T$$

satisfies

$$Hx = \pm \|x\|_2 e_1 \; ?$$

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Answer: $x = (x_1, \cdots, x_n)$ given

Let $v = x + \text{sign}(x_1)\|x\| e_1$,

where $\text{sign}(x_1) = \begin{cases} 1, & \text{if } x_1 \geq 0 \\ -1, & \text{if } x < 0 \end{cases}$

- $w = \frac{v}{\|v\|}$

Then if

$$H = I - 2ww^T,$$

$$Hx = \operatorname{sign}(x_1)\|x\| e_1$$

We can in general find w so that

$$H = I - 2ww^T$$

satisfies

$$Hx = \text{sign}(x_k)\|x\| e_k$$

$$e_k = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \leftarrow 1 \text{ in } k\text{-th position}$$

Ex.

$$x = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \|x\| = \sqrt{2}$$

$$v = x - \sqrt{2}\, e_1 = \begin{bmatrix} -1-\sqrt{2} \\ 1 \end{bmatrix}$$

$$\|v\|^2 = 1 + 2 + 2\sqrt{2} + 1 = (4 + 2\sqrt{2})$$

$$w = \frac{\begin{bmatrix} -1-\sqrt{2} \\ 1 \end{bmatrix}}{(4+2\sqrt{2})^{1/2}}$$

So

$$H_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} -$$

$$\frac{2}{(4+2\sqrt{2})} \begin{bmatrix} (-1-\sqrt{2})^2 & -1-\sqrt{2} \\ -1-\sqrt{2} & 1 \end{bmatrix}$$

$$HX = \begin{bmatrix} -1 \\ 1 \end{bmatrix} - \frac{2}{4+2\sqrt{2}} \begin{bmatrix} (-1-\sqrt{2})(1+\sqrt{2}+1) \\ 1+\sqrt{2}+1 \end{bmatrix}$$

$$= \begin{bmatrix} x \\ 0 \end{bmatrix}$$

✓

QR Method

Suppose A is $n \times n$ real matrix

we can use Householder matrices

to get

$$A = QR$$

where $Q^{-1} = Q^T$, (Thus $Q^T Q = I$)

and R is upper triangular

$A = [c_1, c_2, \ldots, c_n]$, $c_i = i\text{-th}$ column of A

We can find H_1 so that

$$H_1 A = \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a'_{21} & \cdots & a'_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & a'_{n1} & \cdots & a'_{nn} \end{bmatrix}$$

Now we find H_2 so that

$H_2 H_1 A =$

$$\begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a_{22}{}^{2} & \cdots & a^{2}_{2n} \\ \vdots & 0 & & \\ \vdots & \vdots & \ddots & \\ 0 & 0 & & \end{bmatrix}$$

In order to find such H_2 imagine

$$H_1 A = \begin{bmatrix} a_{11}^1 & a_{12}^1 & \cdots & a_{1n}^1 \\ 0 & & & \\ \vdots & & A_2 & \\ 0 & & & \end{bmatrix}$$

A_2 is $(n-1) \times (n-1)$ matrix.

Now we pick

$$H_2 = \begin{bmatrix} 1 & 0 \cdots 0 \\ 0 & \\ \vdots & \hat{H}_2 \\ 0 & \end{bmatrix}$$

In other words $H_2 H_1 A$ does not touch the first column of $H_1 A$.

This way in n steps we can compute

$$H_n H_{n-1} \cdots H_1 A = R$$

R upper triangular.

Let $Q^T = H_n H_{n-1} \cdots H_1$

Then

$$Q = H_1^T \cdots H_n^T = H_1 \cdots H_n$$

Also

$$Q Q^T = H_1 \cdots \underbrace{H_n \times H_n}_{I} \cdots \times H_1$$

$$= \cdots I$$

So

$$A = QR$$

We can show the # of operations is $O(n^3)$ (like LU factorization)

Applications of QR factorization

1. Solving $Ax = b$

If we compute $QR = A$

then we solve

$$QRx = b$$

Let

$$y = Rx$$

Then we set

$$Qy = b$$

$$\Rightarrow y = Q^T b$$

. Then we solve for x

in $RX = Y$

It is somewhat like LU factorization but more stable. However, it takes twice as many operations

2. Application in Computing all Eigenvalues of an $n \times n$ matrix A.

It works as follows

Compute QR factorization of A.

Let $A_1 = A$. Compute Q_1, R_1 factorization of A_1.

Let $A_2 = R_1 Q_1$

Since $A_1 = Q_1 R_1 \Rightarrow Q_1^T A_1 = R_1$

So $A_2 = Q_1^T A_1 Q_1$

Next Compute QR factorization of A_2

$$A_2 = Q_2 R_2$$

Let $A_3 = R_2 Q_2$

It can be shown that under some conditions

A_m converges to an upper triangular matrix whose diagonal entries are eigenvalues.

Convergence could be slow.

QR factorization is also valid if A is a matrix with complex entries.

In such case we define

$$H = I - 2ww^*$$

where w^* is conjugate transpose

In this general case

$$H^* = H$$

3. Finding all roots of a polynomial

suppose we want to compute roots of

$$p(z) = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0$$

Consider $n \times n$ matrix

$$A = \begin{bmatrix} -\frac{a_{n-1}}{a_n} & -\frac{a_{n-2}}{a_n} & \cdots & -\frac{a_1}{a_n} & -\frac{a_0}{a_n} \\ 1 & 0 & \cdots & & 0 \\ 0 & 1 & 0 & \cdots & \\ \vdots & & \ddots & & \\ 0 & \cdots & & 1 & 0 \end{bmatrix}$$

Let

$$v = \begin{bmatrix} \lambda^{n-1} \\ \lambda^{n-2} \\ \vdots \\ \lambda^{i} \\ \vdots \\ \lambda^{1} \\ \lambda^{0} \end{bmatrix}$$

, where λ is a root of $p(z)$

We claim $Av = \lambda v$

$$AV = \begin{bmatrix} -a_{n-1}\lambda^{n-1} - \cdots - a_1\lambda - a_0 \\ a_n \\ \lambda^{n-1} \\ \lambda^{n-2} \\ \\ \\ \lambda \end{bmatrix}$$

But $\lambda V = \begin{bmatrix} \lambda^n \\ \lambda^{n-1} \\ \vdots \\ \lambda \end{bmatrix}$, + these are the same.

We can thus apply QR method to find all roots of $p(z)$.

However this is not necessarily the preferred method.

Power Method to compute dominant eigenvalue of a matrix.

Dominant means eigenvalue with largest modulus.

Power Method:

Pick $v_0 \in \mathbb{R}^n$

Inductively define

$$\mu_k = v_k^T A v_k$$

$$v_{k+1} = A v_k / \|A v_k\| .$$

Suppose the eigen values of A are

$$\lambda_1, \ldots, \lambda_n$$

& suppose

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$$

That is to say the

We claim:

μ_k Converges to λ_1.

pf:

Since v_k is obtained from v_0 by repeated multiplication by A, k times

$$v_1 = \frac{Av_0}{\|Av_0\|} = A\frac{v_0}{\|Av_0\|}$$

$$= A\frac{v_0}{\|Av_0\|} = A($$

suppose $v_k = c_1 v_1 \lambda^k +$

$$v_k = \alpha_1 \lambda_1^k u_1 + \alpha_2 \lambda_2^k u_2 + \cdots + \alpha_n \lambda_n^k u_n$$

for some constants $\alpha_1, \ldots, \alpha_n$

Then

$$A v_k = \alpha_1 \lambda_1^{k+1} u_1 + \alpha_2 \lambda_2^{k+1} + \cdots + \alpha_n \lambda_n^{k+1} u_n$$

So

$$v_k^T v_k = \alpha_1^2 \lambda_1^{2k} + \alpha_2^2 \lambda_2^{2k} + \cdots \alpha_n^2 \lambda_n^{2k}$$

$$= 1$$

$$v_k^T A v_k = \alpha_1^2 \lambda_1^{2k+1} + \alpha_2^2 \lambda_2^{2k+1} + \cdots$$

$$\text{So } \frac{v_k^T A v_k}{v_k^T v_k} = \frac{\lambda_1^{2k+1}}{\lambda_1^{2k}} \left[\frac{\alpha_1^2 + \alpha_2^2 \left(\frac{\lambda_2}{\lambda_1}\right)^{2k+1} \cdots}{\alpha_1^2 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^{2k} \cdots} \right]$$

Now

$$\left[\cdots \right] \longrightarrow 1$$

$$\frac{\lambda_1^{2k+1}}{\lambda_1^{2k}} = \lambda_1$$

So $\mu_k \longrightarrow \lambda_1$

Triangle Algorithm

Given a set of point

$$S = \{v_1, \dots, v_n\} \subseteq \mathbb{R}^m$$

& $p \in \mathbb{R}^m$

we want to know if $p \in \text{Conv}(S)$

i.e. if

$$p = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

where $\alpha_1 + \alpha_2 + \dots + \alpha_n = 1$, $\alpha_i \geq 0$

If we let

$$A = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}, \quad m \times n$$

$$\alpha = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}$$

then we want to see

if $A\alpha = P$, $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$

$e^T \alpha = 1$

$\alpha \geq 0$

Given $p' \in \text{Conv}(S)$, say

$$p' = \alpha_1 v_1 + \cdots + \alpha_n v_n, \quad \sum \alpha_i = 1, \alpha_i \geq 0$$

we say $v_j \in S$ is a pivot

if $\|p' - v_j\| \geq \|p - v_j\|$

Equivalently, if

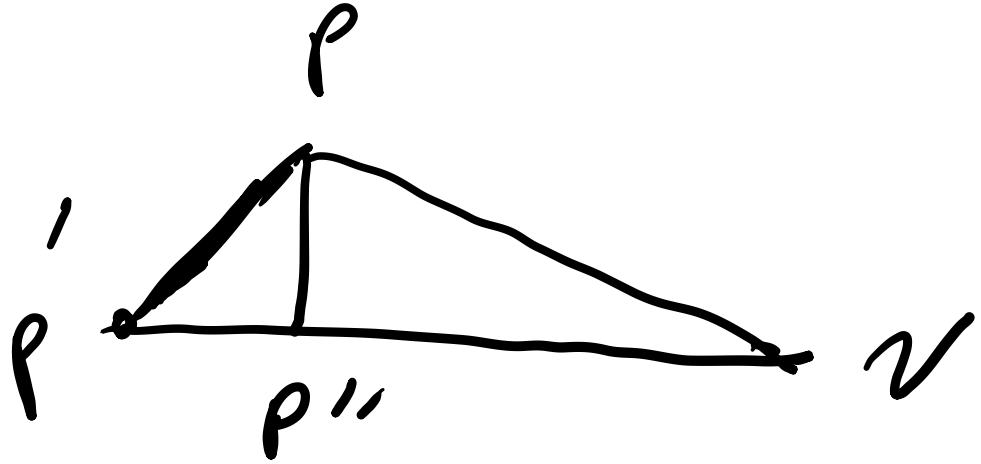
$$(p - p')^T v_j \geq \frac{1}{2}\left(\|p\|^2 - \|p'\|^2\right).$$

We say p' is a witness if no pivot exists, i.e.

$$(p - p')^T v_i < \frac{1}{2}(\|p\| - \|p'\|^2)$$

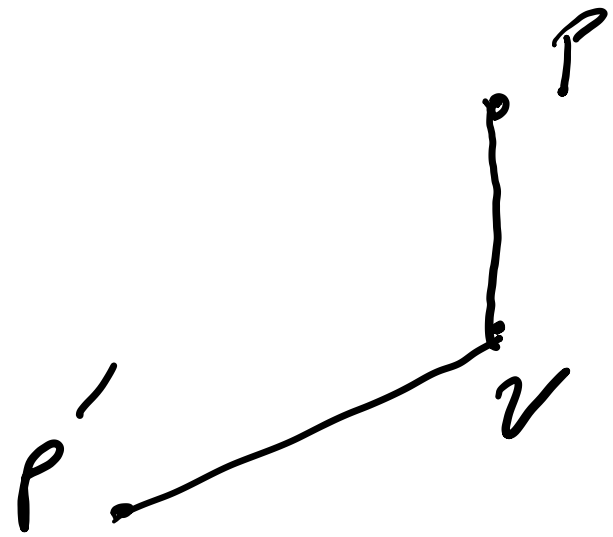
for $i = 1, \ldots, n$

If V is a pivot
then we can get closer to P:



we project P
on $P'V$.

We could also have this situation:



In this case we simply take $p'' = V$.

p'' can be computed as:

Let $\alpha_x = \dfrac{(p-p')^T (v_j - p')}{\|v_j - p'\|^2}$

If we have the first situation

then

$$p'' = (1-\alpha_x)\, p' + \alpha_x\, v = \sum_{i=1}^{n} \alpha_i' v_i$$

$$\alpha_j' = (1-\alpha_x)\,\alpha_j + \alpha_x, \quad \alpha_i' = (1-\alpha_x)\,\alpha_i, \quad i \neq j$$

So given $p' = \sum \alpha_i v_i$ we

Can compute a representation of

p'' in terms of v_i's

$$p'' = \sum \alpha_i' v_i$$

Triangle Algorithm

Input S, P, ϵ

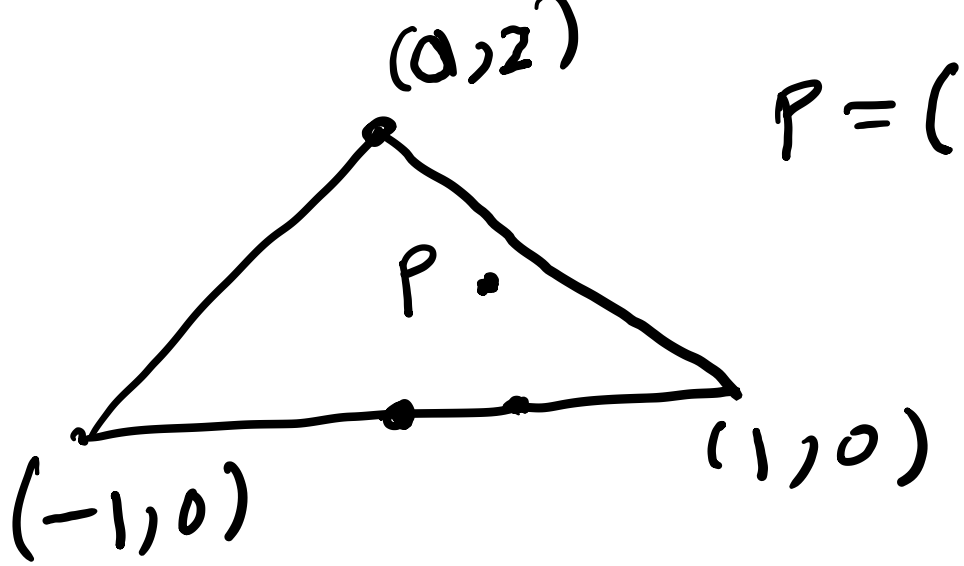
step 0. Let $P' = V = \text{argmin}\{\|P - V_i\|\}$
i.e. the closest V_i to P.

step 1. If $\|P - P'\| \leq \epsilon \|P - V\|$,
output P' as an approximate
solution, stop.

Step 2. Replace $\mathcal{V}$ with a pivot $\mathcal{V}_j$, compute P''. Replace P' with P'' and Goto step 1.

Try one or two iterations of the algorithm both geometrically and algebraically.

Ex.



$P = (\frac{1}{2}, 1)$

Solving $Ax = b$

Assume A is $n \times n$, invertible.

Thm. Suppose $x_* = A^{-1}b \geq 0$

There exist $\alpha > 0$, $x \geq 0$

such that

$$Ax = \alpha b$$

$$\sum x_i + \alpha = 1 \quad , \quad x_i \geq 0, \alpha \geq 0$$

In other words

$0 = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$ is in $\mathrm{Conv}(A, -b)$.

We can solve $Ax = b$ with the assumption that $x_* = A^{-1}b \geq 0$ via the trivial algorithm:

In each iteration we have a pair (x, α) such that

$$p' = Ax - \alpha b \,, \quad \sum x_i + \alpha = 1$$

$$x_i \,, \alpha \geq 0$$

we check if

$$\| A\frac{x}{\alpha} - b \| \leq \epsilon \max\{\|a_1\|, \ldots, \|a_n\|, \|b\|\}.$$

a_i = i-th column of A

If so we stop.

otherwise, we iterate.

What if $A^{-1}b \neq 0$?

If $Ax = b$

then $Ax + w = b + w$

suppose we choose $w = tAe$

where $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$, t a scalar.

Then $A(x + te) = b + tAe$

$\exists\, t_*$ such that $x + t_* e \geq 0$

and t_* is the smallest such t such that $x + te \geq 0$.

If we knew such t_*, we add $t_* Ae$ to both sides and do as before.

Since we don't know t_x we try to pick a t & incrementally change it if necessary.

Incremental Triangle Alg.

Assume for a given $t_0 \geq 0$ we have tried to use triangle algorithm to find x_0 s.t.

$$\|A(x_0 - t_0 e) - b\| \leq$$
$$\in \max\{\|a_1\|, \ldots, \|a_n\|, \|b\|\}$$

where a_i = i-th column of A

If this is possible, we are done.

Otherwise, the simplest strategy is to increase t_0 to t_1 & repeat.

There are better ideas. (later)

Lecture 7

CS 510

Triangle Algorithm Review

Input S, p, ϵ $\quad S = \{v_1, \cdots, v_n\}$

step 0. Let $p' = v = \operatorname{argmin}\{\|p - v_i\|\}$
i.e. the closest v_i to p.

step 1. If $\|p - p'\| \leq \epsilon \|p - v\|$,
output p' as an approximate
solution, stop.

Step 2. Replace $\mathcal{V}$ with a pivot $\mathcal{V}_j$, compute P''. Replace P' with P'' and Goto step 1.

Solving $Ax = b$

Assume A is $n \times n$, invertible.

Thm. Suppose $x_* = A^{-1}b \geq 0$

There exist $\alpha > 0$, $x \geq 0$

such that

$$Ax = \alpha b$$

$$\sum x_i + \alpha = 1 \quad , \quad x_i \geq 0, \alpha \geq 0$$

In other words

$0 = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$ is in $\mathrm{Conv}(A, -b)$.

We can solve $Ax = b$ with the assumption that $x_* = A^{-1}b \geq 0$ via the trivial algorithm:

In each iteration we have a pair (x, α) such that

$$p' = Ax - \alpha b \,, \quad \sum x_i + \alpha = 1$$

$$x_i \,, \alpha \geq 0$$

we check if

$$\| A\frac{x}{\alpha} - b \| \leq \epsilon \max\{\|a_1\|, \ldots, \|a_n\|, \|b\|\}.$$

a_i = i-th column of A

If so we stop.

otherwise, we iterate.

What if $A^{-1}b \neq 0$?

If $Ax = b$

then $Ax + w = b + w$

suppose we choose $w = tAe$

where $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$, t a scalar.

Then $A(x + te) = b + tAe$

$\exists\, t_*$ such that $x + t_* e \geq 0$

and t_* is the smallest such t such that $x + te \geq 0$.

If we knew such t_*, we add $t_* Ae$ to both sides and do as before.

Since we don't know t_x we try to pick a t & incrementally change it if necessary.

Incremental Triangle Alg.

Assume for a given $t_0 \geq 0$ we have tried to use triangle algorithm to find x_0 s.t.

$$\|A(x_0 - t_0 e) - b\| \leq$$
$$\in \max\{\|a_1\|, \ldots, \|a_n\|, \|b\|\}$$

where a_i = i-th column of A

If this is possible, we are done.

Otherwise, the simplest strategy is to increase t_0 to t_1 & repeat.

There are better ideas. (later)

Eigenvalue - Eigenvector Remarks

Suppose A is $n \times n$ Hermitian i.e. $A = A^*$ conjugate transpose.

We claim all eigenvalues are real (not eigenvectors).

How to prove this?

Theorem (Schur Normal Form)

Let A be an $n \times n$ matrix with entries in $\mathbb{C}$.

There exists Unitary matrix U, i.e. $U^* = U^{-1}$ so that

$$A = UTU^*$$

with T triangular

Proof is by induction on n.

The following is a important consequence of the thm.

Theorem (Principal Axes Thm)

Let A be Hermitian ($A = A^*$) $n \times n$

Then all its eigenvalues are real, say $\lambda_1, \dots, \lambda_n$, not necessarily distinct.

Additionally, there exist corresponding eigenvectors $u_1, \dots, u_n$ such that they are orthonormal, i.e

$$u_i^* u_j = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}$$

If A is real $u_1, \ldots, u_n$ can be
then to be real vector.

Finally,

$$A = U \Lambda U^*, \text{ where}$$

$$\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$$

$$A = \begin{bmatrix} | & | & & | \\ u_1 & u_2 & \cdots & u_n \\ | & | & & | \end{bmatrix} \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix} \begin{bmatrix} - & u_1^* & - \\ - & u_2^* & - \\ & \vdots & \\ - & u_n^* & - \end{bmatrix}.$$

Pf. From Schur Normal Form

$$A = UTU^*, \quad UU^* = I, \quad T \text{ Triangular}$$

Then

$$A^* = UT^*U^*$$

Since $A = A^*$,

it follows that $T = T^*$.

But $T = T^*$ $\Rightarrow$ T is diagonal & $t_{ii} = \overline{t_{ii}}$ $\Rightarrow$ t_{ii} is real. Let $\lambda_i = t_{ii}$.

So $AU = UT$

$U = [u_1, \ldots, u_n]$

So $Au_i = \lambda_i u_i$.

If A is real then we can take U to be real.

This can be seen as follows:

For each λ_i we solve $Ax = \lambda_i x$

But this gives u_i that is real.

Suppose $\operatorname{rank}(A - \lambda_i I) = n - 1$.

Then if $\lambda_i \neq \lambda_j$

u_i & u_j are orthogonal $\rightarrow$

This is because on the

on hand we have

$$A u_i = \lambda_i u_i$$

$$A u_j = \lambda_j u_j$$

But $u_j^T A u_i = \lambda_i u_i^T u_j$

$u_i^T A u_j = \lambda_j u_i^T u_j$

But $u_j^T A u_i = u_i^T A u_j$

So if $\lambda_i \neq \lambda_j$, $u_i^T u_j = 0$.

Ex. $A = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 2 \end{bmatrix}$

Eigenvalues are $\lambda_1 = 1$, $\lambda_2 = 2$, $\lambda_3 = 4$

$$u_1 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}, \quad u_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix},$$

$$u_3 = \frac{1}{\sqrt{6}} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

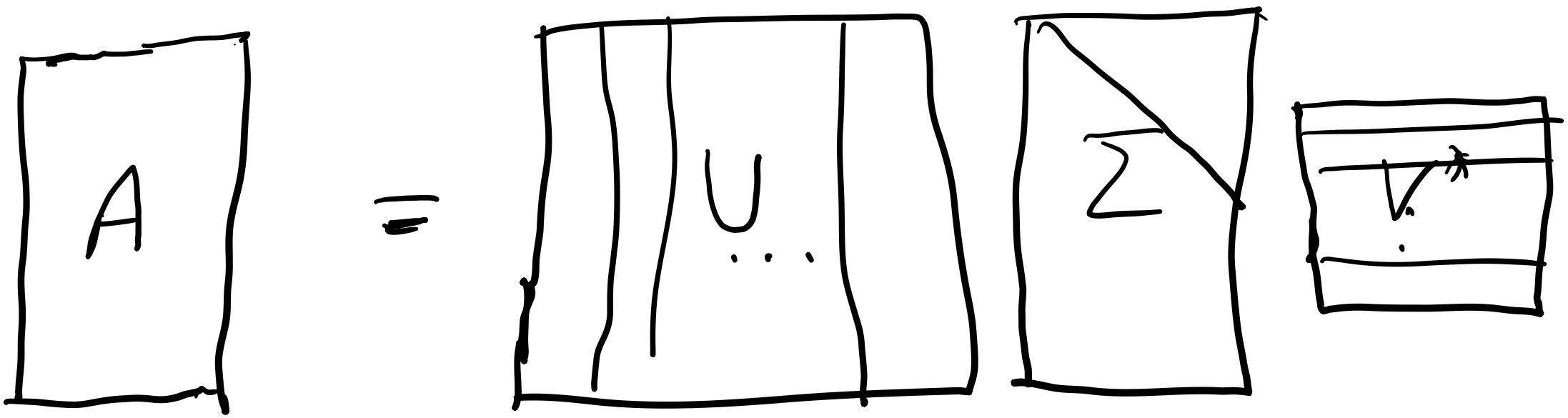
Singular value Decomposition

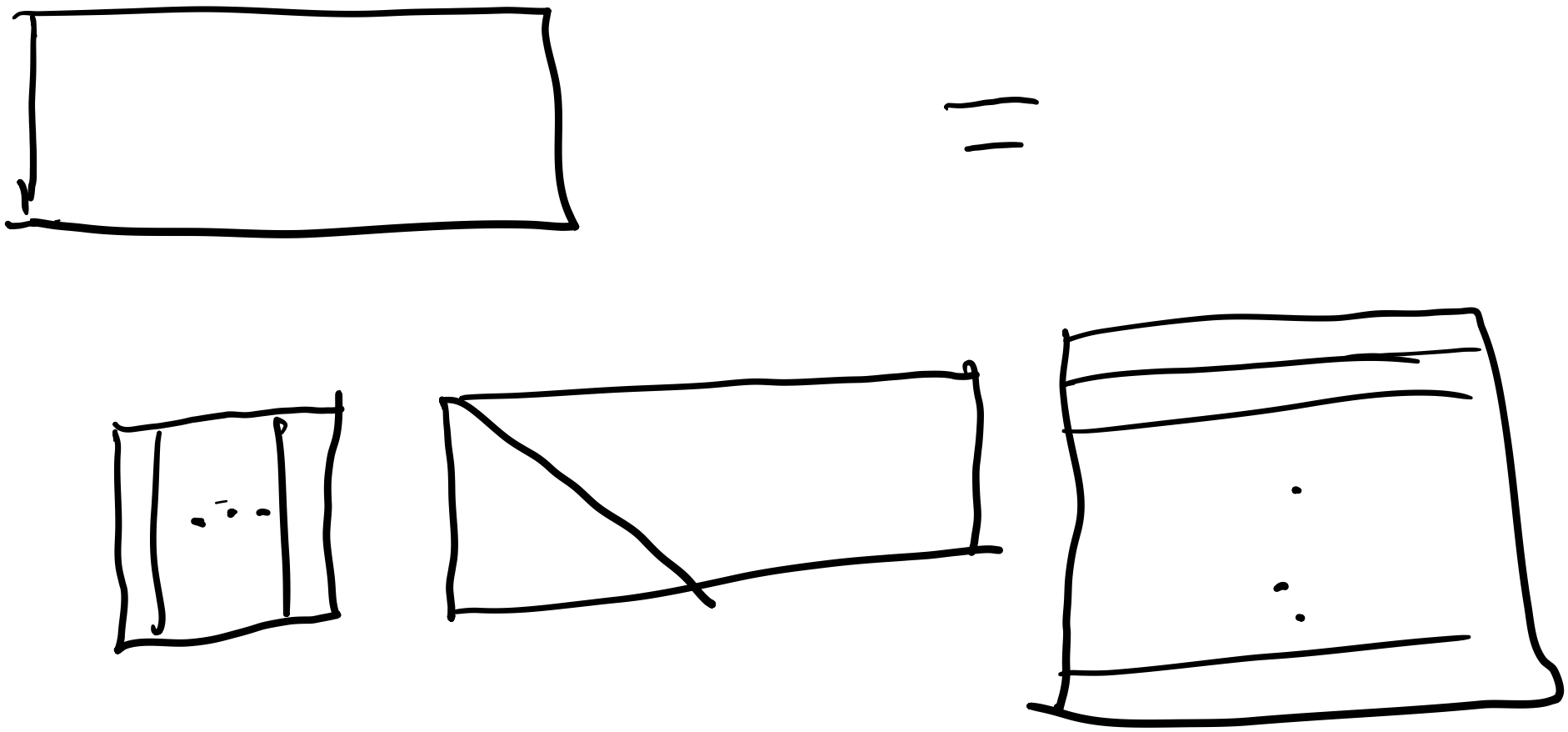
Thm (SVD) Let A be $m \times n$ complex matrix.

Then there exist unitary matrices U, $m \times m$, V $n \times n$, & $m \times n$ matrix Σ with at most $\min\{m, n\}$ positive diagonal entries & all others zero

$$A = U \Sigma V^*$$

$m \times n$ $\quad$ $m \times m$ $\quad$ $m \times n$ $\quad$ $n \times n$





Σ diagonal entries ≥ 0 and

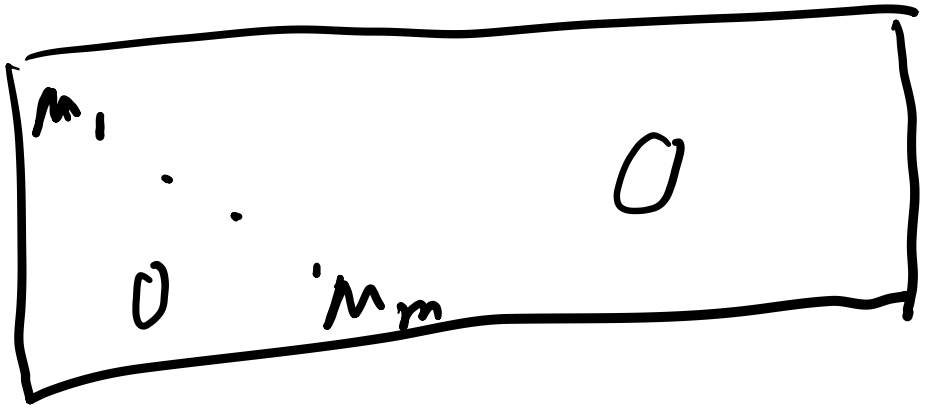
Can be written as

$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$

$$\begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \\ & 0 & \end{bmatrix}$$

The σ_i's are called singular values of A

If Σ is like



$$n_1 \geq n_2 \cdots \geq n_m$$

Pf. Let $M = A^* A$ .. M is $n \times n$

Then M is Hermitian.

By Principal axis thm

$$M = V \Lambda V^* \quad , \quad V V^* = V^* V = I$$

so

$$V^* M V = \Lambda \quad , \quad \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$$

$$V^* A^* A V = \Lambda$$

$\lambda_i \geq 0$ since M is positive semidefinite.

Let $W = AV = [w_1, w_2, \cdots, w_n]$

$$W^{\lambda} = \begin{bmatrix} w_1^{\lambda} \\ w_2^{\lambda} \\ \vdots \\ w_n^{\lambda} \end{bmatrix}$$

suppose $r \leq \min\{m, n\}$ of the λ_i's are positive.

We can assume U is such that $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_r > 0$.

Then $w_i^* w_j = \begin{cases} 0, & i \neq j, \\ \lambda_i, & i = j \end{cases}$

For $i = 1, \ldots, r$ let $u_i = \frac{1}{\sqrt{\lambda_i}} w_i$

For $i = r+1, \ldots, n$, choose u_i so that $u_1, \ldots, u_n$ forms an orthonormal set

Let $U = [u_1, \ldots, u_n]$.

This can be done by solving equation: For instance to compute u_{r+1} we solve r equation

$$u_i^T x = 0, \quad i = 1, \ldots r$$

for a nontrivial solution.

So in this fashion we can extend $u_1, \ldots, u_r$ to an orthonormal basis of $\mathbb{R}^n$

To extend: we can do this one at a time: Let u_{r+1} be a solution to

$$u_i^T x = 0, \quad i = 1, \ldots, r$$

$$\|u_{r+1}\| = 1, \text{ etc.}$$ Then

$$U^*U = I$$

and if we let

$$\Sigma = \begin{bmatrix} \mu_1 & & 0 \\ & \ddots & \\ 0 & & \mu_r \\ & 0 & \end{bmatrix}, \qquad \mu_i = \sqrt{\lambda_i}$$

then

$$U\Sigma = AV$$

or

$$U\Sigma V^* = A .$$

Remark. Suppose $n > m$.
We can then let $M = AA^*$.
Then we get
$A^2 = U \Sigma V^*$ for appropriate U, Σ, V^*
Taking * of both sides we get
$A = V \Sigma U^*$
.

Corollary:

If A is real then

$$A = U \Sigma V^T$$

where $VV^T = I$, $UU^T = I$,

U, V real.

EX. Suppose

$$A = \begin{bmatrix} 2 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}, \quad A \text{ is } 2 \times 3$$

Try $M = AA^T$

$$M = \begin{bmatrix} 2 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix} \quad , \quad \begin{bmatrix} 5-\lambda & 2 \\ 2 & 2-\lambda \end{bmatrix}$$

Eigenvalues:

$$(5-\lambda)(2-\lambda) - 4 =$$

$$\lambda^2 - 7\lambda + 10 - 4 = 0$$

$$\lambda^2 - 7\lambda + 6 = 0$$

$$\lambda = 1, \quad \lambda = 6$$

eigenvector

$$\begin{bmatrix} 5-\lambda & 2 \\ 2 & 2-\lambda \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 2 \\ 2 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} -x_1 + 2x_2 = 0 \\ 2x_1 - 4x_2 = 0 \end{cases}$$

$$x_1 = 2x_2$$

$$v_1 = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$\lambda = 1$

$$\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$2x_1 + x_2 = 0$$

$$2x_1 = -x_2$$

$$v_2 = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$V = [v_1, v_2] = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}$$

$$V^T = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}.$$

Next we need to find

$$U = [u_1, u_2, u_3]$$

$$A^T V = \frac{1}{\sqrt{5}} \begin{bmatrix} 2 & 1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix}$$

$$= \frac{1}{\sqrt{5}} \begin{bmatrix} 5 & 0 \\ 1 & -2 \\ -2 & -1 \end{bmatrix} = [w_1, w_2]$$

Then

$$u_1 = \frac{w_1}{\|w_1\|}, \quad u_2 = \frac{w_2}{\|w_2\|}$$

$$u_1 = \frac{1}{\sqrt{30}}\begin{bmatrix} 5 \\ 1 \\ -2 \end{bmatrix}, \quad u_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$$

We Compute u_3 as a vector of norm one s.t. $u_3 \perp u_1 = 0$, $u_3^T u_2 = 0$

or we solve

$$5x_1 + x_2 - 2x_3 = 0$$
$$-x_2 - x_3 = 0$$

Let $x_3 = 1$, $x_2 = -1$, $x_1 = \frac{3}{5}$

or $\frac{1}{\sqrt{59}} \begin{bmatrix} 3 \\ -5 \\ 5 \end{bmatrix}$

Note $\mu_1 = \sqrt{6}$, $\mu_2 = 1$

So we have

$$A^T = U \Sigma V^T,$$

$$\Sigma = \begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_2 \\ 0 & 0 \end{bmatrix}$$

$$A = V \Sigma^T U^T.$$

Thm. Gerschgorin Thm.

Let A be $n \times n$, $A = (a_{ij})$

Let $r_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$, $i = 1, \dots, n$

Let $Z_i = \{ z \in \mathbb{C} : |z - a_{ii}| \leq r_i \}$

$i = 1, \dots, n$.

If λ is an eigenvalue of A, then λ must belong to one of the circles Z_i, $i=1,\dots,n$.

Pf. We have $Ax = \lambda x$ for some $x \neq 0$.

Assume x is s.t.

$$\|x\|_\infty = 1$$

Suppose $|x_k| = 1$, for some k.

We have

$$\sum_{j=1}^{n} a_{kj} x_j = \lambda x_k$$

So $$x_k(\lambda - a_{kk}) = \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj} x_j.$$

Or $$|\lambda - a_{kk}| \leq \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| = r_k.$$