Lecture Notes in Advanced Matrix Computations

Lectures by Dr. Michael Tsatsomeros

Throughout these notes, □ signifies end proof, and ▲ signifies end of example.

Table of Contents

Notes by Jakob Streipel. Last updated April 27, 2018.

Lecture 0 Technicalities

0.1 Importance

The importance of computational/numerical linear algebra and matrix computations stems entirely from it's pervasive role in all applications in all of STEM. Solving linear systems of equations or resolving eigenvalue problems is a fundamental problem most questions in the sciences can be reduced to in some way, shape, or form.

In practice all of this is done by means of some computational package or program, raising the question of why, beyond mathematical curiosity, one needs to know how this works. Fundamentally, the answer to this question is that computer arithmetic isn't perfect. Floating point arithmetic by definition is somewhat imprecise, so it is important to understand where the errors are introduced and how to measure how large they are.[1]

0.2 Background

There will be almost no prerequisites for this course—knowledge of undergraduate linear algebra should suffice.

0.3 Material

The course will focus on three parts:

- Linear systems, from a numerical point of view. These are covered in sections 1.1–1.9, 2.1–2.9, and 7.1–7.6 in the text. The first of these deal with direct methods, the second set considers the numerical error and sensitivity of algorithms, and finally the last set concerns iterative methods.
- Least squares and singular value decomposition, covered by sections 3.1–3.6 and 4.1–4.4.
- Eigenvalue computations, in sections 5.1–5.9 and 6.1–6.7.

0.4 Matrix Multiplication

As a pretext for discussing notation we will introduce a way of looking at matrix multiplication that simplifies the understanding of many things.

Throughout this course we will let $x \in \mathbb{R}^n$ denote a column vector, i.e. a $n \times 1$ matrix. Moreover $A = [a_{ij}] \in \mathbb{R}^{m\times n} = M_{m,n}(\mathbb{R}) = M_{m,n}$ (the latter in case the underlying field is understood) will denote matrices with a_{ij} being a general element. For the most part we will deal with square matrices, the collection of which we will denote M_n.

We write

$$A = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \end{bmatrix} \in M_n$$

Date: January 8th, 2018.

[1] For instance, find some programming language (like JavaScript in your browser) that does pure floating point and have it execute $0.1 + 0.2$.

making a_j the column vectors of A, belonging to $\mathbb{R}^n$. Hence

$$Ax = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x_1 a_1 + x_2 a_2 + \ldots + x_n a_n \in \mathbb{R}^m$$

where we can therefore thing of the matrix multiplication as a linear combination of the rows of the matrix, with the weights being the elements of x.

Lecture 1 The Cost of Matrix Algebra

1.1 Block Matrices

In general, with $A \in M_{n,k}$ being an $n \times k$ matrix and $B \in M_{k,m}$ being a $k \times m$ matrix, we want to compute the product $AB = [c_{ij}]$, being a $n \times m$ matrix.

We know that

$$c_{ij} = \sum_{\ell=1}^{k} a_{i\ell} b_{\ell j}.$$

The fundamental question which we will ask ourselves, for this problem and others, is how long this computation takes; how many operations it takes. In particular we will count the number of ***floating point operations*** or ***FLOPS***.[2]

Typically we will reduce this problem, seemingly in a way which accomplishes nothing, but as we'll see makes life easier. The way we do this is to partition matrices into blocks, say

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

where A is a $m \times n$ matrix, and A_{11} is $m_1 \times n_1$, A_{12} is $m_1 \times n_2$, A_{21} is $m_2 \times n_1$, and finally A_{22} is $m_2 \times n_2$.

Hence $m_1 + m_2 = m$ and $n_1 + n_2 = n$.

Now similarly we partition

$$X = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} p_1 & p_2 \end{array} \\ \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} \end{array},$$

which we write in this way to indicate that X_{ij} is a $n_i \times p_j$ matrix, and by assumption X is $n \times p$ and $p_1 + p_2 = p$, thus.

We can therefore express AX as

$$AX = \begin{bmatrix} A_{11}X_{11} + A_{12}X_{21} & A_{11}X_{12} + A_{12}X_{22} \\ A_{21}X_{11} + A_{22}X_{21} & A_{21}X_{12} + A_{22}X_{22} \end{bmatrix}.$$

The reason this is advantageous is that each of the matrix multiplications in the new blocks can be computed independently of one another, meaning that we can distribute the separate computations to different processors and/or cores in parallel. It also results in less paging of memory.

When partitioned in such a way that the matrix multiplication works out, we say that A and X are partitioned ***conformally*** or ***conformably***.

Date: January 10th, 2018.

[2] See IEEE 754.

Example 1.1.1. As a practical example, consider, say,

$$\left[\begin{array}{c|cc} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot \end{array}\right] \left[\begin{array}{c|cc} \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{array}\right].$$

Most times, blocks will be square:

$$\left[\begin{array}{c|c} s \times s & * \\ \hline 0 & t \times t \end{array}\right] \left[\begin{array}{c|c} s \times s & * \\ \hline 0 & t \times t \end{array}\right]$$

for instance, is an example of the product of two ***block upper triangular matrices***, the product of which is again a block upper triangular matrix. ▲

Hence we are interested in studying and counting the number of FLOPS of various algorithms, as a way to compare the relative speed of different ways of doing things.

Example 1.1.2. For instance, consider the computation $d \coloneqq b + a \cdot c$. This performs one multiplication, counting as one FLOP, one addition, another FLOP, and one transfer to memory, which we'll ignore. In all this is a 2 FLOP computation.

Note that this is just one of many possible complexity measures; there is the issue of fetching and storing in memory which we ignore, to name one other. ▲

1.2 Systems of Linear Equations

The basic problem much of this course deals with is a system

$$Ax = b$$

with A being a given $m \times n$ matrix, b being a $m \times 1$ matrix of known values, and x being a $n \times 1$ matrix whose elements we are looking for.

We know from basic linear algebra that there are three possibilities:

1. There is exactly one solution.
2. There are infinitely many solutions.
3. There are no solutions.

When $m = n$ and A is invertible (i.e. nonsingular), then clearly $x = A^{-1}b$. This seems great, but it has two severe drawbacks: A is not always invertible, and moreover even if it is, computing its inverse is *expensive*. This is a very bad idea in practice.

Hence we want to find alternative methods which are both faster, and crucially can handle all of the above possibilities well. As we have discussed before, there will be direct methods and iterative methods.

On the topic of inverting matrices, recall the following:

Theorem 1.2.1. *Let A be an $n \times n$ square matrix. Then the following are equivalent:*

(i) *A^{-1} exists.*

(*ii*) *If* $Ay = 0$*, then* $y = 0$*.*

(*iii*) *The columns of* A *are linearly independent.*

(*iv*) *The rows of* A *are linearly independent.*

(*v*) $\det A \neq 0$.

(*vi*) *For all* b*,* $Ax = b$ *has a (unique) solution.*

(*vii*) *Every eigenvalue of* A *is nonzero.*

1.3 Triangular Systems

Let $G = [g_{ij}]$, say an $n \times n$ matrix. We say that G is a ***lower triangular matrix*** if $g_{ij} = 0$ if $i < j$, meaning that it looks like

$$\begin{bmatrix} g_{11} & 0 & \cdots & 0 \\ * & g_{22} & & \vdots \\ \vdots & & \ddots & 0 \\ * & \cdots & * & g_{nn} \end{bmatrix}.$$

Importantly, if G is lower triangular, or if we have some good way of getting it into a lower triangular form, then solving a system for unknowns becomes easy. For example, if we're solving

$$\begin{bmatrix} g_{11} & 0 & 0 \\ g_{21} & g_{22} & 0 \\ g_{31} & g_{32} & g_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

for y_1, y_2, and y_3, we can see directly that $y_1 = b_1/g_{11}$, and knowing y_1 we can resolve

$$y_2 = \frac{b_2 - g_{21}y_1}{g_{22}},$$

and finally

$$y_3 = \frac{b_3 - g_{32}y_2 - g_{31}y_1}{g_{33}}.$$

If the system is larger than 3×3, the same sort of computations happen, just more of them.

This is known as a ***forward substitution*** or a ***forward elimination*** algorithm, the pseudo-code of which is

Algorithm 1.1: Forward substitution

```
input: matrix G, vector b
output: vector
begin
  for i = 1 : n
    for i = 1 : i − 1
      b_i ← b_i − g_ij b_j
    if g_ii = 0, stop (no solution)
    b_i ← b_i / g_ii
  return b
end
```

Lecture 2 LU Decomposition

We start this lecture by first counting the number of FLOPs required for the forward substitution algorithm from the end of last lecture.

The $b_i \leftarrow b_i - g_{ij}b_j$ operation is 2 FLOPs, and we do this

$$\sum_{i=1}^{n}\sum_{j=1}^{i-1} 2 = 2\sum_{i=1}^{n}\sum_{j=1}^{i-1} 1 = 2\sum_{i=1}^{n}(i-1) = 2\frac{n(n-1)}{2} = n(n-1)$$

times.

In addition we perform n comparisons with 0, and finally n divisions. Hence there are $n(n-1)+n+n = O(n^2)$ total FLOPs.

2.1 Gaussian Elimination and LU Decomposition

The idea is to exploit the method of solving $Ax = b$ when the coefficient matrix A is triangular. What, then, happens if A is not triangular?

We therefore want to conceive of a way to transform A into a triangular form whilst preserving solutions to $Ax = b$.

Definition 2.1.1. Two linear systems $Ax = b$ and $Cy = d$ are called ***equivalent*** if they have the same solutions.

There are three types of operations that result in equivalent systems:

1. Adding a multiple of one equation to another equation.
2. Interchanging two equations.
3. Multiplying an equation by a *nonzero* scalar.

Example 2.1.2. We will solve the following system of equations

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 2x_1 + 2x_2 - x_3 = 3, \\ 4x_1 - x_2 + 6x_3 = 4, \end{cases}$$

in three ways.

First, 'by hand.' We subtract row 1 from row 2, and subtract 2 times row 1 from row 3, yielding

$$\left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & 1 & -2 & 2 \\ 0 & -3 & 4 & 2 \end{array}\right].$$

Next adding 3 times row 2 to row 3 we get

$$\left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & 1 & -2 & 2 \\ 0 & 0 & -2 & 8 \end{array}\right].$$

The coefficient matrix is now upper triangular, so we can perform backward substitution; $-2x_3 = 8$ means that $x_3 = -4$. Using this $x_2 - 2x_3 = 2$ means

Date: January 12th, 2018.

$x_2 - 2(-4) = 2$, i.e. $x_2 = -6$. Finally $2x_1 + x_2 + x_3 = 1$, so $2x_1 - 6 - 4 = 1$ implies $x_1 = 11/2$.

A different point of view on the same computation is this: let us do the exact same operations, but express them in terms of matrix multiplication. In particular,

$$\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\0&3&1\end{bmatrix}}_{=L_3}\underbrace{\begin{bmatrix}1&0&0\\0&1&0\\-2&0&1\end{bmatrix}}_{=L_2}\underbrace{\begin{bmatrix}1&0&0\\-1&1&0\\0&0&1\end{bmatrix}}_{=L_1}\begin{bmatrix}2&1&1\\2&2&-1\\4&-1&6\end{bmatrix}=\begin{bmatrix}2&1&1\\0&1&-2\\0&0&-2\end{bmatrix}.$$

Hence $L_3L_2L_1A = U$, with U being upper triangular, and L_i all being lower triangular. Importantly L_i all have only 1 on the diagonals, meaning that they are invertible, so $A = L_1^{-1}L_2^{-1}L_3^{-1}U$.

There are three important facts about triangular matrices we need:

1. Inverses of lower (upper) triangular matrices are also lower (upper) triangular.
2. Products of lower (upper) triangular matrices are also lower (upper) triangular.
3. The types of lower triangular matrices used above are easy to invert. Namely, L_1 subtracts row 1 from row 2, so its inverse is simply adding row 1 back to row 2, and similarly for L_2 and L_3, or such matrices in general.

Hence

$$L_1^{-1} = \begin{bmatrix}1&0&0\\1&1&0\\0&0&1\end{bmatrix}, \quad L_2^{-1} = \begin{bmatrix}1&0&0\\0&1&0\\2&0&1\end{bmatrix}, \quad \text{and} \quad L_3^{-1} = \begin{bmatrix}1&0&0\\0&1&0\\0&-3&1\end{bmatrix}$$

and moreover

$$L_1^{-1}L_2^{-1}L_3^{-1} = \begin{bmatrix}1&0&0\\1&1&0\\2&-3&1\end{bmatrix} = L.$$

Therefore $A = LU$, and we have successfully LU-decomposed A.

Hence to solve $Ax = b$, we use $A = LU$ and solve $LUx = b$, which is easy since it's equivalent with $Ux = L^{-1}b = c$, and then we use U being triangular to solve this in $O(n^2)$ operations. ▲

Lecture 3 Partial Pivoting

3.1 LU Decomposition continued

Last time we solved a linear system by Gaussian elimination, arriving at $A = LU$, with L being lower triangular and U being upper triangular. This meant that $Ax = b$ could be rewritten as $LUx = b$, and so if we let $y = L^{-1}b$, then $Ux = y$ gives us the solution to the system, and this latter system can be solved with backward substitution since U is upper triangular.

Date: January 17th, 2018.

What allowed us to proceed this way was the following:

First of all, $Ax = b$ had a unique solution, meaning that A is invertible, and so did $Ux = L^{-1}b$.

Secondly, we were able to proceed with Gaussian elimination using only row operations of type 1. Specifically we did not need row exchanges because all pivotal positions were nonzero.

When does this happen, we ask, the answer to which is when all ***leading principal minors*** are nonzero.

A ***principal submatrix*** of a matrix A is a submatrix of A contained in the same set of rows and columnns of A, a ***principal minor*** is the determinant of such a submatrix, and a ***leading principal minor*** is the determinant of such a matrix which is taken from the rows $\{1, 2, \ldots, k\}$ for $k \leq n$, with $A \in M_n$.

When performing Gaussian elimination, the first pivot is a_{11}, which is the first leading principal minor. The first step adds a multiple of the first row to all rows, and the second pivot becomes

$$a_{22}^{(2)} = a_2 2 - \frac{a_{21}a_{12}}{a_{11}},$$

which is nonzero if and only if $a_{11}a_{22} - a_{21}a_{12} \neq 0$, which is the second leading principal minor, and so forth.

Note, for the record, that a variant of LU decomposition is LDU decomposition, wherein one simply decomposes U into a diagonal part to ensure U has 1's along the diagonal. For example,

$$\begin{bmatrix} 2 & 1 & 1 \\ 2 & 2 & -1 \\ 4 & -1 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & -3 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 \\ 0 & 1 & -2 \\ 0 & 0 & -2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & -3 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{bmatrix} \begin{bmatrix} 1 & 1/2 & 1/2 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}.$$

3.2 Gaussian Elimination with (Partial) Pivoting

What happens if a pivotal position is zero? The answer is quite simple; we permute columns!

For example, starting with

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

we switch row 1 and row 3, then proceed with ordinary elimination, yielding

$$\begin{bmatrix} 2 & -2 & 1 \\ 0 & 2 & 5/2 \\ 0 & 4 & 1 \end{bmatrix}.$$

The reason why we chose row 3 is to maximise the size of the pivot—this guarantees that entries don't grow too much when dividing through by the pivot. One can show that in doing so, no entry will grow by more than double.

For this reason we again permute, putting row 3 in place of row 2, and then proceed, finally yielding

$$\begin{bmatrix} 2 & -2 & 1 \\ 0 & 4 & 1 \\ 0 & 0 & 2 \end{bmatrix}.$$

Summarised this looks like

$$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & -1/2 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ -1/2 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}0 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}0 & 4 & 1\\ 1 & 1 & 3\\ 2 & -2 & 1\end{bmatrix} = \begin{bmatrix}2 & -2 & 1\\ 0 & 4 & 1\\ 0 & 0 & 2\end{bmatrix}.$$

Calling the elementary matrices L_2, P_2, L_1, and P_1 in the order shown, we have

$$A = P_1^{-1}L_1^{-1}P_2^{-1}L_2^{-1}U,$$

but this is not quite nice enough. However, if we first let $\hat{A} = P_2P_1A$, then we can apply Gaussian elimination to $\hat{A}$ (maybe with a different L) with no row exchanges required, and so $\hat{A} = L'U$, and $PA = L'U$, with $P = P_2P_1$. This is called ***LU factorisation with (partial) pivoting***.

Lecture 4 Complete Pivoting

4.1 LU Decomposition continued

The strategy we used of moving the largest pivot in a column (via row operations) is called partial pivoting, as discussed. This is to distinguish it from ***complete pivoting***, wherein we look not only for the largest element in the column, but in the entire (sub)matrix!

Consider, for example,

$$\begin{bmatrix}0 & 1 & 3\\ 2 & 3 & -4\\ 1 & 1 & 1\end{bmatrix}.$$

The last element in the second row, -4, is the largest in modulus, so we permute row 1 and 2, and then columns 1 and 3, yielding

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

a process we can describe by

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

which we'll write, once finished, as $PAQ = LU$.

Let us consider the flop count of LU factorisation. Without pivoting, we're performing an addition and a multiplication, then subtracting from each element, so we have

$$2n^2+2(n-1)^2+\ldots+2\cdot1^2 = 2\sum_{k=1}^{n}k^2 = 2\frac{n(n+1)(2n+1)}{6} = \frac{2}{3}n^3+n^2+\frac{n}{3} = O(n^3).$$

Date: January 17th, 2018.

Adding partial pivoting into the mixture produces an extra $n-k$ comparisons at step k, so

$$\sum_{k=1}^{n-1}(n-k) = \frac{n^2-n}{2} = O(n^2),$$

so the algorithm is still $O(n^3)$.

The same is true with complete pivoting; we get $(n-k)^2$ comparisons each step, which means

$$\sum_{k=1}^{n-1}(n-k)^2 = \frac{2n^3-3n^2+n}{6} = O(n^3),$$

so the final, complete pivoting algorithm, is still $O(n^3)$.

4.2 Positive Definite Systems

A special kind of matrix arising in applications is this:

Definition 4.2.1 (Symmetric matrix)**.** A matrix $A \in M_n(\mathbb{R})$ is called ***symmetric*** if $A = A^T$.

Definition 4.2.2 (Positive definite)**.** A symmetric matrix A which satisfies $x^TAx > 0$ for all $x \neq 0$ in $\mathbb{R}^n$ is called ***positive definite***.

Example 4.2.3. The matrix

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

is positive definite since

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 2x_1^2 - 2x_1x_2 + x_2^2 = x_1^2 + (x_1-x_2)^2 > 0$$

since $x \neq 0$. ▲

Theorem 4.2.4. *If $A \in M_n$ is positive definite, then A is nonsingular (i.e. invertible).*

Proof. Consider the contrapositive; let A be singular. Then there exists some $x \in \mathbb{R}^n$ with $x \neq 0$ such that $Ax = 0$. Then $x^TAx = x^T0 = 0$, so A is not positive definite. □

Corollary 4.2.5. *If A is positive definite, then $Ax = b$ has a unique solution for all b, namely $x = A^{-1}b$.*

Theorem 4.2.6. *Let M be an $n \times n$ nonsingular matrix. Then $A = M^TM$ (as well as MM^T) is a positive definite matrix.*

Proof. First, since $A = M^TM$ we have $A^T = M^T(M^T)^T = M^TM = A$, whence A is symmetric. Next, let $x \in \mathbb{R}^n \setminus \{0\}$. Then

$$x^TAx = x^TM^TMx = (Mx)^T(Mx)$$

which is therefore

$$y_1^2 + y_2^2 + \ldots + y_n^2$$

if we call $Mx = [y_i]$. Since $x \neq 0$ and M is invertible, $y = Mx \neq 0$, and so the above is positive. Hence $x^TAx > 0$, and A is positive definite. □

Lecture 5 Cholesky Decomposition

5.1 Positive Definiteness and Cholesky Decomposition

The converse of the theorem at the end of last lecture is also true.

Theorem 5.1.1 (Cholesky decomposition theorem)**.** *Let $A \in M_n(\mathbb{R})$ be positive definite. Then A can be (uniquely) decomposed into a product $A = R^T R$, called the* **Cholesky decomposition***, where $R = [r_{ij}] \in M_n(\mathbb{R})$ is upper triangular with diagonal entries $r_{ii} > 0$.*

We will prove three basic properties of positive definite matrices in order to arrive at the above theorem:

Proposition 5.1.2. *If $A = [a_{ij}] \in M_n(\mathbb{R})$ is positive definite, then $a_{ii} > 0$ for all $i = 1, 2, \ldots, n$.*

Proof. Let $x = e_j$ be the jth column of the identity matrix. Then $0 < x^T Ax$ since A is positive definite, and moreover

$$0 < x^T Ax = e_j^T Ae_j = a_{jj}$$

for all $j = 1, 2, \ldots, n$. □

Proposition 5.1.3. *If*

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

if positive definite, where A_{11} and A_{22} are square (principal) submatrices of A, then A_{11} and A_{22} are positive definite.

Proof. To prove A_{11} is positive definite, let $y \in \mathbb{R}^k$ with $A_{11} \in M_k(\mathbb{R})$. Then

$$y^T A_{11} y = \begin{bmatrix} y^T & 0 \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} y \\ 0 \end{bmatrix} > 0$$

where we extend the y-vector with appropriately many zeros. To show that A_{22} we do the same thing, except with leading zeros instead of trailing zeros. □

Proposition 5.1.4. *If $A, X \in M_n(\mathbb{R})$, with A positive definite and X nonsingular, then $B = X^T AX$ is positive definite as well.*

Proof. For every $y \in \mathbb{R}^n$ with $y \neq 0$, we have

$$y^T By = y^T X^T AXy = (Xy)^T A(Xy) > 0$$

since Xy is a nonzero vector since X is nonsingular. □

Proof of theorem. We prove the result by induction on n, the size of the matrix.

For $n = 1$ the matrix $A = \begin{bmatrix} a_{11} \end{bmatrix}$ has a_{11} by Proposition 5.1.2 and

$$A = \begin{bmatrix} \sqrt{a_{11}}\sqrt{a_{11}} \end{bmatrix} = \begin{bmatrix} \sqrt{a_{11}} \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} \end{bmatrix} = R^T R$$

which is vacuously upper triangular.

Date: January 22nd, 2018.

Suppose now that every $(n-1)\times(n-1)$ positive definite matrix has a unique Cholesky decomposition with positive diagonal entries.

Let

$$A = \begin{bmatrix} a_{11} & b^T \\ b & \hat{A} \end{bmatrix}$$

with $\hat{A}$ a $(n-1)\times(n-1)$ matrix, $b \in \mathbb{R}^{n-1}$, and A being symmetric since we assume it positive definite. By Proposition 5.1.3 $\hat{A}$ is positive definite and so $a_{11} > 0$. Hence let $r_1 = +\sqrt{a_{11}} > 0$ and $s = r_1^{-1}b \in \mathbb{R}^{n-1}$. Let $\tilde{A} = \hat{A} - ss^T$, the Schur complement. Then, by straight forward computation,

$$A = \begin{bmatrix} r_1 & 0 \\ s & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \tilde{A} \end{bmatrix} \begin{bmatrix} r_1 & s^T \\ 0 & I \end{bmatrix}.$$

Let

$$X = \begin{bmatrix} r_1 & s^T \\ 0 & I \end{bmatrix}^{-1}$$

whence by Proposition 5.1.4 $B = X^T A X$ is positive definite. But

$$B = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{A} \end{bmatrix}$$

so by induction $\tilde{A} = \tilde{R}^T\tilde{R}$ has unique Cholesky decomposition. Finally observe

$$A = \begin{bmatrix} r_1 & 0 \\ s & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \tilde{R}^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \tilde{R} \end{bmatrix} \begin{bmatrix} r_1 & s^T \\ 0 & I \end{bmatrix} = \begin{bmatrix} r_1 & 0 \\ s & \tilde{R}^T \end{bmatrix} \begin{bmatrix} r_1 & s^T \\ 0 & \tilde{R} \end{bmatrix} = R^T R$$

as claimed. □

Between the two theorems, we have essentially characterised all positive definite matrices. They are all matrices of the form $M^T M$ with M invertible.

We make two notes in closing. First, the notion and results on positive definite matrices generalises easily to ***Hermitian*** matrices in $M_n(\mathbb{C})$, these being matrices such that $A = A^*$, where A^* is the conjugate transpose. Note, for the record, that in fact a positive definite matrix in this sense is necessarily Hermitian, so one needn't assume it, unlike the real case where we must assume symmetry.

Secondly, there are other proofs of the Cholesky decomposition theorem. In particular, $A = LU$ can be LU decomposed without pivoting by Proposition 5.1.3, since it tells us that all principal submatrices, and in particular the leading ones, are nonsingular. Then in fact $R = U$ and $L = R^T$.

Lecture 6 Vector and Matrix Norms

6.1 Vector Norms

Definition 6.1.1 (Vector norm)**.** A ***norm*** (or ***vector norm***) on $\mathbb{R}^n$ is a function $\|\cdot\|\colon \mathbb{R}^n \to \mathbb{R}$ that assigns a real number to every vector in $\mathbb{R}^n$ such that for every $x, y \in \mathbb{R}^n$ and every $a \in \mathbb{R}$

Date: January 24th, 2018.

(*i*) $\|x\| > 0$ if $x \neq 0$ (and $\|0\| = 0$);

(*ii*) $\|ax\| - |a|\|x\|$, called homogeneity; and

(*iii*) $\|x + y\| \leq \|x\| + \|y\|$, the triangle inequality.

Note that by homogeneity

$$\|0\| = \|0 \cdot x\| = |0|\|x\| = 0\|x\| = 0,$$

so it is redundant to include $\|0\| = 0$ in the definition. Secondly, note that every norm induces a distance $\|x - y\|$ between x and y.

Example 6.1.2. Maybe the most familiar vector norm is the ***Euclidean norm***, namely

$$\|x\|_2 = \Big(\sum_{i=1}^{n} |x_i|^2\Big)^{1/2}.$$

Clearly $\|x\|_2 > 0$ for $x \neq 0$, $\|0\|_2 = 0$, and $\|ax\|_2 = |a|\|x\|_2$. To show that the triangle inequality is satisfied, let us first prove the ***Cauchy-Schwarz inequality***: for every $x, y \in \mathbb{R}^n$,

$$\Big|\sum_{i=1}^{n} x_i y_i\Big| \leq \|x\|_2 \|y\|_2.$$

To prove this, take any $t \in \mathbb{R}$ and note that

$$0 \leq \sum_{i=1}^{n} (x_i + t y_i)^2 = \sum_{i=1}^{n} x_i^2 + 2t \sum_{i=1}^{n} x_i y_i + t^2 \sum_{i=1}^{n} y_i^2 = c + bt + at^2.$$

For whic quadratic to always be nonnegative, its roots must not be two distinct real numbers, meaning that $b^2 - 4ac \leq 0$. Hence solving for ac we have

$$\Big(\frac{b}{2}\Big)^2 \leq ac$$

and since a and c are sums of nonnegative numbers they are nonnegative, so their square roots are real, and so

$$\frac{b}{2} \leq \sqrt{a}\sqrt{c},$$

which is the Cauchy-Schwarz ineguality proved.

With this in hand the triangle inequality follows, since

$$\|x + y\|_2^2 = \sum_{i=1}^{n} (x_i + y_i)^2 = \sum_{i=1}^{n} x_i^2 + 2\sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n} y_i^2$$

which by the Cauchy-Schwarz inequality means that

$$\|x + y\|_2^2 \leq \|x\|_2^2 + 2\|x\|_2\|y\|_2 + \|y\|_2^2 = (\|x\|_2 + \|y\|_2)^2.$$

Taking square roots, which we can do since both sides are nonnegative, we have

$$\|x + y\|_2 \leq \|x\|_2 + \|y\|_2,$$

as desired. ▲

Example 6.1.3. A similar norm to the Euclidean one is

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

and a related but seemingly different norm is

$$\|x\|_\infty = \max_{1 \le i \le n} |x_i|.$$

Finally, a very large class of vector norms are A-norms, generated by a positive definite matrix $A \in M_n(\mathbb{R})$, defined as

$$\|x\|_A = (x^T A x)^{1/2}. \qquad \blacktriangle$$

6.2 Matrix Norms

Definition 6.2.1 (Matrix norm)**.** A ***matrix norm*** is a function $\|\cdot\| \colon M_n(\mathbb{R}) \to \mathbb{R}$ is a function that assigns a real number to any $n \times n$ matrix such that if $A, B \in M_n(\mathbb{R})$ and every $a \in \mathbb{R}$, we have

(*i*) $\|A\| > 0$ if $A \neq 0$;

(*ii*) $\|aA\| = |a|\|A\|$;

(*iii*) $\|A + B\| \le \|A\| + \|B\|$; and

(*iv*) $\|AB\| \le \|A\|\|B\|$.

This last property is called ***submultiplicativity*** and is not always taken as an axiom of matrix norms.

Lecture 7 Vector and Matrix Norms continued

7.1 Matrix Norms

Example 7.1.1. Maybe the simplest matrix norm is the ***Frobenius norm***, defined for $A \in M_n(\mathbb{R})$ as

$$\|A\|_F = \Big(\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|^2 \Big)^{1/2}.$$

That this is a norm is almost obvious; the three first conditions are almost identical to that of the Euclidean norm, and for submultiplicativity note that

$$\|AB\|_F = \sum_{i=1}^{n} \sum_{j=1}^{n} \Big| \sum_{k=1}^{n} a_{ik} b_{kj} \Big|^2,$$

which allows us to use the Cauchy-Schwarz inequality. $\blacktriangle$

Date: January 26th, 2018.

Example 7.1.2. Every vector norm $\|\cdot\|_v$ can be used to define a matrix norm $\|\cdot\|_M$ by

$$\|A\|_M = \max_{x\neq 0} \frac{\|Ax\|_v}{\|x\|_v}$$

called the ***induced matrix norm*** (by $\|\cdot\|_v$). ▲

We'll prove that this is a matrix norm in a moment.

Theorem 7.1.3. *For a vector norm and its induced matrix norm we have*

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

In fact there exists $x \in \mathbb{R}^n \setminus \{0\}$ such that $\|Ax\| = \|A\|\|x\|$.

Proof. If $x = 0$, then the result is trivially true, so let us take $x \neq 0$. Then

$$\frac{\|Ax\|}{\|x\|} \leq \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \|A\|.$$

Multiply both sides by $\|x\|$, then $\|Ax\| \leq \|A\|\|x\|$.

To show that equality is attained, first we note that for $x \neq 0$,

$$\frac{\|Ax\|}{\|x\|} = \frac{\|Ax\|/\|x\|}{\|x\|/\|x\|} = \left\|A\frac{x}{\|x\|}\right\|,$$

where $\|x/\|x\|\| = 1$. Hence we can restrict the maximum over unit vectors, so

$$\|A\| = \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|,$$

and since the unit ball is closed and bounded (i.e. compact) the maximum must be attained. □

So the induced matrix norm is a norm since $A \neq 0$ means that

$$\|A\| = \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} \geq \frac{\|A\hat{x}\|}{\|\hat{x}\|}$$

for all nonzero $\hat{x} \in \mathbb{R}^n$, and since the latter is in terms of vector norms we know to be vector norms, this is positive.

For homogeneity, the constant simply factors out of the maximum since it is independent of x and A.

The triangle inequality follows from the triangle inequality of the vector norm, since

$$\|A + B\| = \max_{x\neq 0} \frac{\|(A+B)x\|}{\|x\|} \leq \max_{x\neq 0} \frac{\|Ax\| + \|Bx\|}{\|x\|} = \|A\| + \|B\|.$$

Finally we use the previous theorem to establish submultiplicativity:

$$\|AB\| = \max_{x\neq 0} \frac{\|ABx\|}{\|x\|} \leq \max_{x\neq 0} \frac{\|A\|\|Bx\|}{\|x\|} \leq \max_{x\neq 0} \frac{\|A\|\|B\|\|x\|}{\|x\|} = \|A\|\|B\|$$

where for both inequalities the theorem was invoked.

Oftentimes we will denote the induced matrix norm using the same subscript as the vector norm from which it stems. Say

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p\Big)^{1/p}$$

with p a real number greater than or equal to 1 (though for our purposes pretty much always a positive integer), then we write

$$\|A\|_p = \max_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p}.$$

In particular, $\|A\|_2$ is called the ***spectral norm***. Note, for the record, that despite visual similarities $\|\cdot\|_2 \neq \|\cdot\|_F$.

We make three basic observations about induced matrix norms:

- Say $\|\cdot\|$ is an induced matrix norm. Then

$$\|I\| = \max_{x \neq 0} \frac{\|Ix\|}{\|x\|} = \max_{x \neq 0} \frac{\|x\|}{\|x\|} = 1.$$

- By contrast,

$$\|I\|_F = \sum_{i=1}^{n} \sum_{j=1}^{n} |I_{ij}|^2 = n,$$

  meaning that $\|\cdot\|_F$ is *not* an induced norm, i.e. there is no vector norm inducing the Frobenius norm.

- Finally, using Cauchy-Schwarz we have that $\|A\|_2 \leq \|A\|_F$.

Theorem 7.1.4. *Let $A \in M_n(\mathbb{R})$. Then*

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}|,$$

i.e. the maximum (in absolute value) column sum, and

$$\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|$$

meaning the maximum row sum.

7.2 Condition Numbers

Our goal is to come up with a useful measure of sensitivity in solving the linear system $Ax = b$. For the most simple situation, let us take $b \neq 0$ and A invertible, so that $x = A^{-1}b$ is the unique solution.

Now consider the linear system

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

where δb is a "small" vector relative to b, e.g. an error in measurements. Ideally we would like $\hat{x}$ to be close to x in the sense that they are approximately equal. Define $\delta x = \hat{x} - x$, and then we want

$$\frac{\|\delta x\|}{\|x\|}$$

to be small whenever

$$\frac{\|\delta b\|}{\|b\|}.$$

Lecture 8 Condition Numbers

8.1 Solving Perturbed Systems

Notice that $Ax = b$ implies that $A(x + \delta x) = b + \delta b$ which in turn means that $A\delta x = \delta b$, whence $\delta x = A^{-1}\delta b$.

Hence

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

and similarly

$$\|b\| \leq \|A\|\|x\|.$$

Hence

$$\frac{\|\delta x\|}{\|x\|} \leq \|A^{-1}\|\|\delta b\|\|A\|\frac{1}{\|b\|} = \|A^{-1}\|\|A\|\frac{\|\delta b\|}{\|b\|}.$$

Definition 8.1.1 (Condition number)**.** For an invertible matrix A, we define the ***condition number*** by

$$k(A) = \|A^{-1}\|\|A\|.$$

Hence

$$\frac{\|\delta x\|}{\|x\|} \leq k(A)\frac{\|\delta b\|}{\|b\|}.$$

Remark 8.1.2. Note that the inequality above is sharp since $Ax = b$ and $A(x + \delta x) = b + \delta b$ have (unique) solutions.

Moreover $k(A)$ provides a measure of how large the relative error is in solving the perturbed system.

If $k(A)$ is not too large, we call A ***well-conditioned***, and if $k(A)$ is large, we call A ***ill-conditioned***. We can quantify "large" based on out standards, tolerances, etc. in the problem at hand.

Proposition 8.1.3. *For every induced norm $\|\cdot\|$,*

(*i*) $\|I\| = 1$,

(*ii*) $k(A) \geq 1$.

Date: January 29th, 2018.

Proof. For the first one, note that

$$\|I\| = \max_{x \neq 0} \frac{\|Ix\|}{\|x\|} = 1.$$

The second one follows from

$$\|AA^{-1}\| = \|I\| = 1$$

and, by submultiplicativity,

$$\|AA^{-1}\| \leq \|A\|\|A^{-1}\| = k(A),$$

so $k(A) \geq 1$. □

Hence the best-conditioned matrix has $k(A) = 1$.

Note that obviously the condition number depends on the norm at hand, so often we'll decorate our condition number with the same subscript as the norm, say

$$k_2(A) = \|A\|_p \|A^{-1}\|_p,$$

typically with $p = 1, 2, \infty$.

The first two are easy to compute, except for the caveat of having to find the inverse of A, and k_2 is based on $\|\cdot\|_2$ which we'll learn to compute effectively later.

Example 8.1.4. The following is a fairly ill-conditioned matrix:

$$A = \begin{bmatrix} 1000 & 999 \\ 999 & 998 \end{bmatrix}.$$

Then

$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} -998 & 999 \\ 999 & -1000 \end{bmatrix}$$

whereby

$$k_1(A) = k_\infty(A) = 1999 \cdot 1999 = 3.996 \cdot 10^6.$$

Using a computer we also find that

$$k_2(A) = 3.992 \cdot 10^6.$$

These are large condition numbers, which makes sense; the solution to a system $Ax = b$ is the intersection between two *almost* parallel lines, so perturbing the lines just slightly moves the intersection a lot. ▲

Of course the cut-off between well- and ill-conditioned is subjective.

Example 8.1.5. Suppose $k(A) \leq 10^2$. Then if $\|\delta b\|/\|b\| \approx 10^{-4}$, then

$$\frac{\|\delta x\|}{\|x\|} \leq 10^2 \cdot 10^{-4} = 10^{-2},$$

so we have accuracy up to the second decimal, which depending on context could be good enough. ▲

Example 8.1.6. Sometimes the accuracy of our floating point arithmetic is the deciding factor. Suppose we're solving $Ax = b + \delta b$ with $\|\delta b\|/\|b\| \approx 10^{-7}$, i.e. our arithmetic is accurate to seven decimals. If $k(A) = 10^7$, then $\|\delta x\|/\|x\| \approx 1$, which is bad.

But if, say, our floating point precision was better than 10^{-7}, then the same sort of conditioned matrix would be acceptable. ▲

Example 8.1.7 (Hilbert matrix)**.** A famous ill-conditioned matrix is the Hilbert matrix, defined by $H_n = [h_{ij}] \in M_n(\mathbb{R})$, with $h_{ij} = 1/(i+j-1)$. So, for instance,

$$H_4 = \begin{bmatrix} 1 & 1/2 & 1/3 & 1/4 \\ 1/2 & 1/3 & 1/4 & 1/5 \\ 1/3 & 1/4 & 1/5 & 1/6 \\ 1/4 & 1/5 & 1/6 & 1/7 \end{bmatrix}.$$

We have $k_2(H_4) = 1.6 \cdot 10^4$, and notably the condition number grows quickly. For instance, $k_2(H_8) = 1.5 \cdot 10^8$. ▲

Definition 8.1.8. Given $A \in M_n(\mathbb{R})$, define

$$\max \operatorname{mag}(A) = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|},$$

the maximum magnification, and also the induced norm $\|A\|$, and similarly

$$\min \operatorname{mag}(A) = \min_{x \neq 0} \frac{\|Ax\|}{\|x\|},$$

the minimum magnification.

Theorem 8.1.9. *Let A be nonsingular. Then*

(*i*) *first*

$$\max \operatorname{mag}(A) = \frac{1}{\min \operatorname{mag}(A^{-1})} \qquad \textit{and} \qquad \min \operatorname{mag}(A) = \frac{1}{\max \operatorname{mag}(A^{-1})},$$

(*ii*) *and*

$$k(A) = \frac{\max \operatorname{mag}(A)}{\min \operatorname{mag}(A)}.$$

Lecture 9 Estimating the Condition Number

9.1 More on Condition Numbers

We start by proving the theorem stated at the end of last lecture.

Proof. For the first part, consider

$$\begin{aligned} \max \operatorname{mag}(A) &= \max_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{y \neq 0} \frac{\|y\|}{\|A^{-1}y\|} \\ &= \frac{1}{\min_{y \neq 0} \frac{\|A^{-1}y\|}{\|y\|}} = \frac{1}{\min \operatorname{mag}(A^{-1})} \end{aligned}$$

Date: January 31th, 2018.

where we took $y = Ax$.

In almost identical fashion,

$$\min \operatorname{mag}(A) = \frac{1}{\max \operatorname{mag}(A^{-1})}.$$

Hence

$$k(A) = \|A\|\|A^{-1}\| = \max \operatorname{mag}(A) \max \operatorname{mag}(A^{-1}) = \frac{\max \operatorname{mag}(A)}{\min \operatorname{mag}(A)}. \qquad \square$$

This means that ill-conditioned matrices are ones for which the maximum magnification is much larger than the minimum magnification.

Note also that for singular matrices, $\min \operatorname{mag}(A) = 0$, but it is wrong to make the association between $k(A)$ and $\det(A)$.

If A is close to being singular, then $k(A)$ will be large, but a singular matrix *can* be well-conditioned (though obviously not in the sense of condition number; they're not defined for singular matrices).

9.2 Ill-conditioning by Scaling

Example 9.2.1. Consider the system

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

with $\varepsilon > 0$ small. The solutions to this system is obviously $x_1 = x_2 = 1$, and also $k_1(A) = k_2(A) = k_\infty(A) = 1/\varepsilon$. Hence A can be made arbitrarily ill-conditioned by choosing a small enough ε.

Now take

$$b + \delta b = \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix} + \begin{bmatrix} 0 \\ \varepsilon \end{bmatrix} = \begin{bmatrix} 1 \\ 2\varepsilon \end{bmatrix}.$$

This is naturally a very small perturbation, since ε is small. Solving $A\hat{x} = b + \delta b$ we get

$$\hat{x} = x + \delta x = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

which is quite different.

However, this ill-conditioning is in some sense artificial, since by rescaling the second row we get

$$\begin{bmatrix} 1 & 0 \\ 0 & 1/\varepsilon \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \varepsilon \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1/\varepsilon \end{bmatrix} \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix}$$

which results in

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

which is as well-conditioned as a matrix can be. ▲

Therefore we ask how we can identify such bad scaling, and how we can remove it.

Theorem 9.2.2. *Let $A \in M_n(\mathbb{R})$ be a nonsingular matrix with columns*

$$a_1, a_2, \ldots, a_n \in \mathbb{R}^n.$$

Then for every i and j we have

$$k_p(A) \geq \frac{\|a_i\|_p}{\|a_j\|_p}$$

for $1 \leq p \leq \infty$.

Proof. Let $a_i = Ae_i$, with e_i the ith column of the identity matrix. Then

$$\max \operatorname{mag}(A) = \max_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p} \geq \frac{\|Ae_i\|_p}{\|e_i\|_p} = \|a_i\|_p$$

for every $i = 1, 2, \ldots, n$. Similarly $\min \operatorname{mag}(A) \leq \|a_j\|_p$ for $j = 1, 2, \ldots, n$. Hence

$$k_p(A) = \frac{\max \operatorname{mag}(A)}{\min \operatorname{mag}(A)} \geq \frac{\|a_i\|_p}{\|a_j\|_p}.$$ □

So if the columns of A differs in norm by several orders of magnitude, then $k_p(A)$ is large. The advice in this situation is to scale columns (i.e. by postmultiplying by an appropriate diagonal matrix) to make them more balanced.

Note, however, that this is a necessary but not sufficient condition for well-conditioning.

Note also that $k_\infty(A) = k_1(A^T)$, so badly scaled rows and badly scaled columns relate to each other.

9.3 Estimating the Condition Number

In theory, $k(A) = \|A\|\|A^{-1}\|$, but finding A^{-1} is in general very computationally expensive. Much like how the best form of research is asking someone who knows, we could ask Matlab by executing `cond(A, p)`. But this just computes an approximate estimate.

The trick is to find good bounds.

Recall that

$$\frac{\|A^{-1}w\|_1}{\|w\|_1} \leq \|A^{-1}\|_1$$

so in solving $Ax = b$, take $w = b$ and so $x = A^{-1}w$, whence

$$\frac{\|x\|_1}{\|b\|_1} \leq \|A^{-1}\|_1$$

and so

$$k_1(A) = \|A\|_1\|A^{-1}\|_1 \geq \|A\|_1 \frac{\|x\|_1}{\|b\|_1} \geq \frac{\|A\|_1\|A^{-1}w\|_1}{\|w\|_1},$$

where $A^{-1}w$, for a given w, is easy to compute without finding the inverse by simply solving $Ax = w$, since $x = A^{-1}w$.

If we choose w in the direction of maximum magnification, i.e. so that this lower bound grows the fastest,

$$k_1(A) \approx \frac{\|A\|_1\|A^{-1}w\|_1}{\|w\|_1}.$$

Lecture 10 Perturbing Not Only the Vector

10.1 Perturbing the Coefficient Matrix

We considered perturbing b to $b + \delta b$, and solving $A\hat{x} = b + \delta b$, compared to $Ax = b$.

Now let us instead perturb A to $A + \delta A$, thinking about $\|\delta A\|/\|A\|$ as being very small. So the system $Ax = b$ becomes $(A + \delta A)\hat{x} = b$. We want x and $\hat{x}$ to be close under certain conditions.

First, however, we need to preserve nonsingularity.

Theorem 10.1.1. *Let A be nonsingular. If*

$$\frac{\|\delta A\|}{\|A\|} < \frac{1}{k(A)},$$

then $A + \delta A$ is also nonsingular.

Proof. We prove the contrapositive. Supopose A is nonsingular and $A + \delta A$ is singular, meaning that there exists some $y \neq 0$ such that $(A + \delta A)y = 0$. Then

$$y = -A^{-1}\delta A y$$

meaning, by taking norms and using submultiplicativity, that

$$\|y\| = \|-A^{-1}\delta A y\| \leq \|A^{-1}\|\|\delta A\|\|y\|$$

which, since $\|y\| > 0$ since $y \neq 0$, means that $1 \leq \|A^{-1}\|\|A\|$ and so

$$\frac{\|\delta A\|}{\|A\|} \geq \frac{1}{\|A^{-1}\|\|A\|} = \frac{1}{k(A)}. \qquad \square$$

Remark 10.1.2. It is maybe of interest to note that

$$\frac{\|\delta A\|}{\|A\|} - \frac{1}{k(A)}$$

can be used as a distance to singularity for nonsingular matrices.

Consider $Ax = b$ and $(A + \delta A)\hat{x} = b$, and let $\delta x = \hat{x} - x$ as before. We want to study $\|\delta x\|/\|x\|$, but first we look at the related value $\|\delta x\|/\|\hat{x}\|$.

Theorem 10.1.3. *Let A be nonsingular, $b \neq 0$, $Ax = b$, and $(A + \delta A)\hat{x} = b$. Then*

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq k(A)\frac{\|\delta A\|}{\|A\|}.$$

Proof. Using $Ax = b = (A + \delta A)\hat{x}$, and $\hat{x} = x + \delta x$, we get

$$\delta x = -A^{-1}\delta A\hat{x},$$

meaning that

$$\|\delta x\| \leq \|A^{-1}\|\|\delta x\|\|\hat{x}\|$$

where, since $\|\hat{x}\| > 0$ since $b \neq 0$,

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq \|A^{-1}\|\frac{\|\delta A\|}{\|A\|}\|A\| = k(A)\frac{\|\delta A\|}{\|A\|}. \qquad \square$$

Date: February 2nd, 2018.

Using this we can understand the quantity we really want:

Theorem 10.1.4. *If A is nonsingular, $Ax = b$, $(A + \delta A)\hat{x} = b$, $b \neq 0$, and if*

$$\frac{\|\delta A\|}{\|A\|} < \frac{1}{k(A)},$$

then

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{k(A)\frac{\|\delta A\|}{\|A\|}}{1 - k(A)\frac{\|\delta A\|}{\|A\|}}.$$

Proof. As in the previous theorem, $\|\delta x\| \leq \|A^{-1}\|\|\delta A\|\|\hat{x}\|$. But $\hat{x} = x + \delta x$, so by the triangle inequality $\|\hat{x}\| \leq \|x\| + \|\delta x\|$, whereby

$$\|\delta x\| \leq \|A^{-1}\|\|\delta A\|(\|x\| + \|\delta x\|) = k(A)\frac{\|\delta A\|}{\|A\|}(\|x\| + \|\delta x\|).$$

Thereby

$$\Big(1 - k(A)\frac{\|\delta A\|}{\|A\|}\Big)\|\delta x\| \leq k(A)\frac{\|\delta A\|}{\|A\|}\|x\|.$$

Since both the parenthesis in the left-hand side and $\|x\|$ are positive by assumption,

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{k(A)\frac{\|\delta A\|}{\|A\|}}{1 - k(A)\frac{\|\delta A\|}{\|A\|}}.$$ □

We make two notes. If A is well-conditioned and $\|\delta A\|/\|A\|$ is small, then $\|\delta A\|/\|A\|$ is much smaller than $1/k(A)$, so the denominator in the above theorem is close to 1. So in that case we nearly have

$$\frac{\|\delta x\|}{\|x\|} \leq k(A)\frac{\|\delta A\|}{\|A\|},$$

which looks familiar!

On the other hand, if A is ill-conditioned, then the theorem allows for $\|\delta x\|/\|x\|$ to be large even if $\|\delta A\|/\|A\|$ is small.

10.2 Perturbing Everything

Suppose now that we perturb A to $A + \delta A$ and b to $b + \delta b$.

Theorem 10.2.1. *Let A be nonsingular, $Ax = b$, $\hat{A}\hat{x} = \hat{b}$, with $\hat{A} = A + \delta A$, $\hat{b} = b + \delta b$, and $\hat{x} = x + \delta x$. Then*

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq k(A)\Big(\frac{\|\delta A\|}{\|A\|} + \frac{\|b\|}{\|\hat{b}\|} + \frac{\|\delta A\|}{\|A\|}\frac{\|b\|}{\|\hat{b}\|}\Big).$$

Note that the mixed term in the end above is typically neglible.

Theorem 10.2.2. *Let A be nonsingular,*

$$\frac{\|\delta A\|}{\|A\|} < \frac{1}{k(A)},$$

$Ax = b$, $(A + \delta A)(x + \delta x) = b + \delta b$, and $b \neq 0$. Then

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{k(A)\Big(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|}\Big)}{1 - k(A)\frac{\|\delta A\|}{\|A\|}}.$$

Lecture 11 Error Analysis After The Fact

11.1 A Posteriori Error Analysis Using the Residue

If we solve $Ax = b$ numerically somehow, getting $\hat{x}$, how do error (round-off or others) affect the accuracy? The simple and naive test is this: using $\hat{x}$, whatever that approximate solution might be (it might not even be close, maybe it's random—that's OK, the analysis that follows still applies). Now compute the residual $\hat{r} = b - A\hat{x}$, i.e. difference in our approximate right-hand side to what we wanted.

Now if $\|\hat{r}\|$ (or, better $\|\hat{r}\|/\|b\|$) is small, then we have some confidence that $\hat{x}$ is a decent solution.

How reassuring is the occurrence of a tiny $\|\hat{r}\|$? Let $\delta b = -\hat{r}$, so that $\hat{x}$ is the solution to the perturbed system $A\hat{x} = b + \delta b$.

Now this we can study—we can use the same results as we have for a perturbed right-hand side. This also tells us that this may or may not result in $\hat{x}$ being close to x, even if $\|\hat{r}\|$, i.e. the perturbation, is small; it depends on the condition number $k(A)$!

The following is our old theorem restated in terms of $\hat{r}$:

Theorem 11.1.1. *Let A be nonsingular, $b \neq 0$, and $\hat{x}$ an approximate solution to $Ax = b$. Let $\hat{r} = b - A\hat{x}$. Then*

$$\frac{\|x - \hat{x}\|}{\|x\|} \leq k(A)\frac{\|\hat{r}\|}{\|b\|},$$

where we call the right-hand side the a posteriori error bound.

11.2 Round-Off Errors and Backward Stability

We have previously discussed what might happen if error is introduced, i.e. values are perturbed, but we haven't much discussed where such errors might come from. One inherent source that can't be avoided is round-off in a computer: we would like to maintain, or at least understand, accuracy in the presence of such things.

For instance, we might get accidental cancellation if two values are so close that the computer can't distinguish.

We will discuss the very basics of floating point arithmetic (see IEEE 754 for the standard!). Numbers are stored in memory as, for example,

$$1.23456 \cdot 10^7$$

where we call 1.23456, which is of a fixed length, the ***significand*** or ***mantissa*** and 7 in 10^7 the ***exponent***. Of course in practice computers wouldn't store these as decimals and powers of ten, since they're binary.

This is called a floating point number because the point "floats";

$$1.23456 \cdot 10^7 = 12.3456 \cdot 10^6.$$

There are some elementary facts to keep in mind about these matters:

Date: February 5th, 2018.

1. If the first number of the mantissa is nonzero, we call the number ***normalised***.
2. This arithmetic allows for *very* large and small numbers (in magnitude) to be represented, like
$$-3.456 \cdot 10^{40} \qquad \text{and} \qquad 4.321 \cdot 10^{-32}.$$
3. There is always a fixed amount of space for the mantissa and exponent; limited in the first case the number of digits, and in the second case the magnitude of the exponent.

The effects of these limitations can be seen by way of example. Suppose our mantissa is 4 decimals. Then $2.3334 \cdot 10^5$ can not be represented exactly in our system. Instead we are forced to store
$$2.333 \cdot 10^5,$$
so an error, namely a storage error, is made.

This can also occur in floating point operations, say
$$(2.333 \cdot 10^1)^2 = 2.44429 \cdot 10^2,$$
so this is stored as $2.444 \cdot 10^2$, and another error is made!

There are fundamentally two types of errors: ***overflow***, meaning that we lose digits in a *big* number, and ***underflow***, meaning that a small number is set to 0.

The latter is *usually* harmless, but not always, as we shall see.

Lecture 12 Computing in the Presence of Errors

12.1 IEEE 754

We start by discussing a situation wherein underflow is harmful.

Example 12.1.1. Suppose
$$x = \begin{bmatrix} 10^{-49} & 10^{50} & \dots & 10^{-50} \end{bmatrix}^T \in \mathbb{R}^{101}.$$
Let us compute $\|x\|_2$ in a computing environment that underflows at 10^{-99}.

We have $x_1^2 = 10^{-98}$, and $x_2^2 = x_3^2 = \ldots = x_{101}^2 = 10^{-100}$, all of which are set to 0 in our environment.

Hence we get $\|x\|_2 \approx \sqrt{x_1^2} = 10^{-49}$, but in reality
$$\|x\|_2 = \sqrt{2 \cdot 10^{-98}} = \sqrt{2} 10^{-98} \approx 1.414 \cdot 10^{-49},$$
so our estimate is off by 40%.

Note that this could be avoided by scaling. Let $\beta = \|x\|_\infty$. If $\beta = 0$, we're done since $x = 0$, else consider the scaled vector $\hat{x} = x/\beta$, whence
$$\|x\|_2 = \beta \|\hat{x}\|_2 = \beta \sqrt{\hat{x}_1^2 + \hat{x}_2^2 + \ldots + \hat{x}_n^2}. \qquad \blacktriangle$$

Date: February 7th, 2018.

We looked at numbers in base 10, being accustomed to this, but of course computers calculate and store numbers in base 2. The IEEE 754 standard for floating point arithmetic specifies two types:[3]

- Single precision floating point numbers, and
- Double precision floating point numbers.

The first of these allocates 32 bits per number, wherein 24 are for the significand and 8 are for the exponent. Since $2^{24} \approx 10^7$, this stores numbers to about 7 decimal places, and exponents can be between 2^{-126} and 2^{127} (the reason we're missing a few from $2^8 = 256$ is that there is also space in the specification for special infinity and "not a number" symbols).

Double precision on the other hand uses 64 bits per number, with 53 for the significand and 11 for the exponent. This yields almost 16 decimal precision and exponents between 2^{-126} and 2^{127}.

12.2 Computing in the Presence of Error

Our goal is to assess the cumulative effect of round-off error as we go through long calculations. To this end we want to compare the exact computation C with the floating point result, denoted $\mathrm{fl}(C)$.

For instance, we might compare $xy + x^2$ and $\mathrm{fl}(xy + x^2)$. Of course we have $\mathrm{fl}(C) = C + e$, where e is an ***absolute error***, however we are generally more interested in seeing this as $\varepsilon = e/C$ and hence $\mathrm{fl}(C) = C(1 + \varepsilon)$, where ε is the ***relative error***

The largest relative error that can occur in a given system is called the ***unit round-off***, usually denoted u. If the significand is accurate to s decimals, then the unit round-off is approximately 10^{-s}, though strictly speaking $u = 1/2 \cdot 10^{1-s}$, since we might round up or down depending.

For IEEE 754, it's $2^{-23} \approx 1.2 \cdot 10^{-7}$ for single precision and $2^{-52} \approx 2.2 \cdot 10^{-16}$ for double precision.

Now the idea in computing the relative errors of computations is that, first of all, we can't store numbers exactly before even computing—even storing numbers we incur an error $\hat{x} = x(1 + \varepsilon_x)$ and $\hat{y} = y(1 + \varepsilon_y)$. Let us study the different basic arithmetic operations:

12.3 Multiplication

We have

$$\mathrm{fl}(\hat{x}\hat{y}) = x(1 + \varepsilon_x)y(1 + \varepsilon_y)(1 + \varepsilon_m) = xy(1 + \varepsilon),$$

where by ε_m we mean the relative error incurred in multiplying, and

$$\varepsilon = \varepsilon_x + \varepsilon_y + \varepsilon_m + \varepsilon_x\varepsilon_y + \varepsilon_x\varepsilon_m + \varepsilon_y\varepsilon_m + \varepsilon_x\varepsilon_y\varepsilon_m,$$

but since all of the relative errors are tiny, the higher order terms above are especially tiny, so

$$\varepsilon \approx \varepsilon_x + \varepsilon_y + \varepsilon_m.$$

So if the relative errors are all much smaller than 1, then $\mathrm{fl}(\hat{x}\hat{y}) \approx xy$, so small errors give good output, making multiplication well-behaved.

[3] Actually a few more, but these are the commonly used ones.

12.4 Division

This time we have, for the relative error ε_d or division,

$$\mathrm{fl}\left(\frac{\hat{x}}{\hat{y}}\right) = \frac{x(1+\varepsilon_x)}{y(1+\varepsilon_y)}(1+\varepsilon_d) = \frac{x}{y}(1+\varepsilon_x)(1+\varepsilon_d)\frac{1}{1+\varepsilon_y}.$$

Writing the last factor as a geometric series, we have

$$\frac{1}{1+\varepsilon_y} = 1 - \varepsilon_y + \varepsilon_y^2 - \varepsilon_y^3 + \dots,$$

where the higher order terms are neglible, so

$$\mathrm{fl}\left(\frac{\hat{x}}{\hat{y}}\right) \approx \frac{x}{y}(1+\varepsilon_x)(1+\varepsilon_d)(1-\varepsilon_y) \approx \frac{x}{y}(1+\varepsilon),$$

where $\varepsilon \approx \varepsilon_x - \varepsilon_y + \varepsilon_d$, so division is well-behaved as well.

12.5 Addition

Let us now compute $\mathrm{fl}(\hat{x}+\hat{y})$ and compare with $x+y$. We get

$$\hat{x}+\hat{y} = x(1+\varepsilon_x)+y(1+\varepsilon_y) = x+y+x\varepsilon_x+y\varepsilon_y = (x+y)\Big(1+\frac{x}{x+y}\varepsilon_x+\frac{y}{x+y}\varepsilon_y\Big),$$

where we call the last parenthesis $1+\varepsilon$. Hence addition is well-behaved if $|\varepsilon_x|$ and $|\varepsilon_y|$ are much smaller than 1, unless, and this is important, $1/(x+y)$ is large, in which case $\hat{x}+\hat{y}$ and $x+y$ will be dramatically different.

Hence if $\hat{x} \approx -\hat{y}$ or $x \approx -y$, then addition runs the danger of going astray.

Lecture 13 Backward Error Analysis

13.1 Backward Stability

Suppose we have a long computation to perform, say $C(z_1, z_2, \dots, z_m)$, where z_i are inputs or operands of the computation. We want to examine when $\mathrm{fl}(C(z_1, z_2, \dots, z_m))$ is the *exact* result of performing $C(\hat{z}_1, \hat{z}_2, \dots, \hat{z}_m)$ where $\hat{z}_1, \hat{z}_2, \dots, \hat{z}_m$ are close to $z_1, z_2, \dots, z_m$ (i.e. a modest multiple of the unit round-off).

If this is true for C, then we say that C is ***backward stable***.

Of course, a complete analysis would have to include both backward stability of C and sensitivity analysis, the latter in order to show that small perturbations in the operands lead to similarly small perturbations in $\mathrm{fl}(C)$.

Usually we accept and/or assume the problem is well-conditioned.

The following is an example of a backward stable computation:

Example 13.1.1. Let us solve $Ax = b$ with $\hat{x}$ being some computed solution, and let us study the residue $\hat{r} = b - A\hat{x}$. Then $A\hat{x} = b + \delta b$, where $\delta b = -\hat{r}$.

So if $\|\delta b\|/\|b\| = \|\hat{r}\|/\|b\|$ is tiny, i.e. the residue is relatively small, then $\hat{x}$ is a good solution to the exact nearby system $A\hat{x} = b - \hat{r}$.

Hence a small residue implies backward stability in solving a linear system. ▲

Date: February 9th, 2018.

13.2 Backward Error Analysis for Gaussian Elimination

In many calculations and algorithms we accumulate a large sum along the way, as is the case in Gaussian elimination. The first thing to think about is the order in which we perform the addition; it is associative, but maybe the grouping still affects the propagation of round-off error.

It turns out that this is not the case—addition is backward stable, no matter the order.

Proposition 13.2.1. *Suppose we compute $\sum_{i=1}^{n} w_i$ using floating point arithmetic with unit round-off u. Then*

$$\mathrm{fl}\left(\sum_{i=1}^{n} w_i\right) = \sum_{i=1}^{n} w_i(1+\gamma_i)$$

is the exact solution to a slightly perturbed computation, so we have backward stability, when

$$|\gamma_i| \leq (n-1)u + O(u^2).$$

Theorem 13.2.2. *If G is lower triangular and $Gy = b$ is to be solved via forward substitution using floating point arithmetic, then the computed solution $\hat{y}$ satisfies*

$$(G + \delta G)\hat{y} = b$$

where $\|\delta F\| \leq 2n \cdot u \cdot \|G\|_\infty + O(u^2)$, whence we have backward stability.

These two results combined lead to backward stability of Gaussian elimination if $PA = LU$ provided $\|\hat{L}\|_\infty$ and $\|\hat{U}\|_\infty$ are small. Then the computed solution is a solution to a system $PA + E = \hat{L}\hat{U}$, with E being a perturbation.

In special cases, for example when A is positive definite or diagonally dominant, then we can prove universal backward stability.

In practice, Gaussian elimination with partial or complete pivoting is backward stable; there are pathological examples where it breaks, but they rarely if ever show up in practice.

In general, then, if $k(A) \approx 10^t$ and $t < s$, where s is the accuracy of entries in A and b in terms of decimals, then the computed solution is accurate to $s - t$ decimals.

Lecture 14 The Least-Squares Problem

14.1 The Discrete Least-Squares Problem

Suppose we are given some set of points in $\mathbb{R}^2$, on the form (t_i, y_i), for $i = 1, 2, \ldots, n$, and that they are observations of a function $y = p(t)$, maybe with noise or round-off errors.

We have reasons to believe or hope that $p(t)$ is a function of a known form, perhaps a line, a polynomial of some certain degree, an exponential function, et cetera.

Our goal is to find this presumed function, in such a way that it fits the data as well as possible.

Date: February 12th, 2018.

The way we measure this, which when successful results in the function of "best fit," is based on minimising the error, where by error we mean the difference between the measured y_i and the corresponding y-value coming from our potential function, i.e. $p(t_i)$. So we call $r_i = y_i - p(t_i)$ the residual errors for $i = 1, 2, \ldots, n$, and we want to find $p(t)$ in such a way that

$$\|r\|_2 = \Big(\sum_{i=1}^{n} |y_i - p(t_i)|^2 \Big)^{1/2}$$

is minimised.

Since we're minimising sums of squares, we refer to this problem as a ***least-squares problem***.

Example 14.1.1. Let (t_i, y_i) be a bunch of points, for $i = 1, 2, \ldots, n$, and let $p(t_i) = y_i$, Suppose

$$p(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_{m-1} t^{m-1}$$

where n is large and m is small, i.e. n is much bigger than m.

Note that $p(t)$ of this form lives in the vector space of polynomials of degrees less than or equal to $m - 1$, and this has a basis, say $\{ \phi_1, \phi_2, \ldots, \phi_m \}$. For example, $\phi_i = t^{i-1}$, in this case.

Thus

$$y_i = p(t_i) = \sum_{j=1}^{m} x_j \phi_j(t_i)$$

for $i = 1, 2, \ldots, n$. Written as a linear system this becomes

$$\begin{bmatrix} \phi_1(t_1) & \phi_2(t_1) & \cdots & \phi_m(t_1) \\ \phi_1(t_2) & \phi_2(t_2) & \cdots & \phi_m(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(t_n) & \phi_2(t_n) & \cdots & \phi_m(t_n) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix},$$

where we call the first matrix A, the second one x, the unknown we wish to solve for, and the last one y is our recorded data.

The least-squares problem, then, is to solve $Ax = y$ in such a way that $\|y - Ax\|_2$, the residual error, is minimised.

Of course if y is in the range (i.e. column space) of A, then there exist solution(s) and the optimal norm above is 0. If not, we will try to get close. ▲

The way we attack this, then, is to find the vector x such that Ax is as close to y as possible, which we know must correspond to the orthogonal projection of y onto the range of A. We will develop the theory of how to do this next.

14.2 Orthogonal Matrices

We will view $\mathbb{R}^n$ as an ***inner product space***, i.e. a vector space with an ***inner product***, namely for every $x, y \in \mathbb{R}^n$ we define

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$$

or, equivalently, $\langle x, y\rangle = x^T y = y^T x$.

Since we're confined to the real numbers, this has the following properties: the inner product is commutative, $\langle x, y\rangle = \langle y, x\rangle$, it is linear in both arguments,

$$\langle \alpha_1 x_1 + \alpha_2 x_2, y\rangle = \alpha_1\langle x_1, y\rangle + \alpha_2\langle x_2, y\rangle$$

and

$$\langle x, \alpha_1 y_1 + \alpha_2 y_2\rangle = \alpha_1\langle x, y_1\rangle + \alpha_2\langle x, y_2\rangle.$$

When taken between a vector and itself, it is nonnegative, $\langle x, x\rangle \geq 0$, and moreover $\langle x, x\rangle = 0$ if and only if $x = 0$, and in fact it induces a familiar norm,

$$\sqrt{\langle x, x\rangle} = \|x\|_2,$$

and hence we can write Cauchy-Schwarz inequality as

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

In $\mathbb{R}^2$ we have the property $\langle x, y\rangle = \|x\|_2\|y\|_2\cos(\theta)$, where θ is the angle between the two vectors x and y. Generally in $\mathbb{R}^n$ the angle between two vectors x and y satisfies

$$\theta = \arccos\Big(\frac{\langle x, y\rangle}{\|x\|_2\|y\|_2}\Big).$$

We call x and y ***orthogonal***, or $x \perp y$, if $\langle x, y\rangle = 0$, meaning that $\theta = \pi/2$.

Definition 14.2.1 (Orthogonal matrices)**.** A matrix $Q \in M_n(\mathbb{R})$ is called ***orthogonal*** if $QQ^T = I$.

Hence Q is invertible and in particular $Q^{-1} = Q^T$, and as a consequence $QQ^T = Q^TQ = I$.

Theorem 14.2.2. *If $Q \in M_n(\mathbb{R})$ is orthogonal, then for every $x, y \in \mathbb{R}^n$ we have*

(i) $\langle Qx, Qy\rangle = \langle x, y\rangle$*, and*

(ii) $\|Qx\|_2 = \|x\|_2$.

Notice that what this means is that the transformation by an orthogonal matrix preserves angles and lengths of vectors.

Lecture 15 Orthogonal Matrices

15.1 Orthogonal Matrices Preserve Lengths and Angles

We prove the theorem stated last time:

Proof. For (i), consider

$$\langle Qx, Qy\rangle = (Qy)^T(Qx) = y^TQ^TQx = y^Tx = \langle x, y\rangle,$$

and for (ii) we have

$$\|Qx\|_2^2 = \langle Qx, Qx\rangle = \langle x, x\rangle = \|x\|_2^2$$

by the first part. Since the norms are nonnegative, we may take square roots and acquire $\|Qx\|_2 = \|x\|_2$. □

Date: February 14th, 2018.

Hence multiplication (i.e. transformation) by an orthogonal matrix preserves lengths of vectors and angles between vectors.

In fact, either of the two properties above are sufficient to have Q being orthogonal:

Proof. Suppose we have $y^T Q^T Q x = y^T x$ for every $x, y \in \mathbb{R}^n$. Then take $x = e_i$ and $y = e_j$, whence

$$(Q^T Q)_{ij} = e_j^T Q^T Q e_i = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

whence $Q^T Q = I$. □

Proof. Suppose $\|Qx\|_2 = \|x\|_2$ for every $x \in \mathbb{R}^n$. Then in particular by linearity of multiplication by matrices we have

$$\|x - y\|_2 = \|Qx - Qy\|_2$$

meaning that

$$\langle x - y, x - y \rangle = \langle Qx - Qy, Qx - Qy \rangle$$

and by the linearity of scalar products this eventually yields $\langle x, y \rangle = \langle Qx, Qy \rangle$ for every x and y, and so we have reduced to the previous case. □

Note that *orthogonal* matrices are exactly those matrices having ***orthonormal*** columns (and rows). To see this, think of $Q^T Q = I$ as

$$\begin{bmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_n^T \end{bmatrix} \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} = I$$

so $\|q_i\|_2 = (q_i^T q_i)^{1/2} = 1$ and $q_j^T q_i = 0$ for $i \neq j$, where q_i is the ith column of Q.

15.2 Rotators

Consider the matrix

$$Q = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

This is orthogonal because $\cos(\theta)^2 + \sin(\theta)^2 = 1$. Reading, as we may, the first column as the image of e_1 and the second column as the image of e_2, we see that this matrix is the matrix that rotates vectors by θ anticlockwise. We call Q a ***rotator*** or ***rotation matrix*** in $\mathbb{R}^2$.

Rotators can be used to create zeros in vectors and matrices. Consider the following: if $x = (x_1, x_2)$ and $x_2 \neq 0$, we look for an angle θ such that

$$Q^T x = \begin{bmatrix} y \\ 0 \end{bmatrix}$$

for some $y \in R$.

We then have

$$Q^T x = \begin{bmatrix} x_1\cos(\theta) + x_2\sin(\theta) \\ -x_1\sin(\theta) + x_2\cos(\theta) \end{bmatrix}$$

wherein we want the second coordinate to be 0, so one choice that works is

$$\cos(\theta) = \frac{x_1}{\sqrt{x_1^2 + x_2^2}} = \frac{x_1}{\|x\|_2} \qquad \text{and} \qquad \sin(\theta) = \frac{x_2}{\sqrt{x_1^2 + x_2^2}} = \frac{x_2}{\|x\|_2}.$$

Hence

$$Q^T = \frac{1}{\|x\|_2}\begin{bmatrix} x_1 & x_2 \\ -x_2 & x_1 \end{bmatrix}$$

and so

$$Q^T x = \frac{1}{\|x\|_2}\begin{bmatrix} x_1^2 + x_2^2 \\ 0 \end{bmatrix}.$$

Applying this idea to a matrix

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

we look for Q^T that transforms (a_{11}, a_{21}) into $(r_{11}, 0)$, which makes $Q^T A = R$ an upper triangular matrix, and so $A = QR$, with Q orthogonal, which is a QR decomposition or factorisation of A.

This, of course, like LU factorisation can be used to solve systems, since $Ax = b$ turns into $QRx = b$ and so $Rx = Q^T b = c$, and we can solve $Rx = c$ with backward substitution.

Example 15.2.1. Let us solve the system

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

We find

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

and so

$$R = Q^T A = \frac{1}{\sqrt{5}}\begin{bmatrix} -5 & 9 \\ 0 & 2 \end{bmatrix}$$

and

$$c = Q^T b = \frac{1}{\sqrt{5}}\begin{bmatrix} 4 \\ -3 \end{bmatrix}.$$

Solving $Rz = c$ we get $z_1 = 19/10$ and $z_2 = -3/2$. ▲

We can generalise this to $n \times n$ matrices by

$$Q = \begin{array}{c} \\ \\ i \\ \\ j \\ \\ \end{array}\begin{array}{c} \begin{array}{ccccccc} & & & i & & j & \end{array} \\ \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & & -s & & \\ & & & \ddots & & & \\ & & s & & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \end{array}$$

where $c = \cos(\theta)$ and $s = \sin(\theta)$, so that

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

is its principal (i, j) submatrix.

Such a Q is called a ***Givens*** or ***Jacobi*** rotation or matrix, since Qx rotates x by θ in the (x_i, x_j)-plane.

Theorem 15.2.2. *Let $A \in M_n(\mathbb{R})$. Then there exist an orthogonal matrix $Q \in M_n(\mathbb{R})$ and an upper triangular matrix $R \in M_n(\mathbb{R})$ such that $A = QR$.*

Lecture 16 More on Orthogonal Matrices

Let us first prove the theorem stated at the end of last lecture.

Proof. The idea is very simple. We find a reflector Q_{21} so that Q_{21}^T puts a zero in the $(2, 1)$ position of A, just like how we did in the $\mathbb{R}^2$ case. We then repeat this with $(3, 1)$, and so forth, until $(n, 1)$.

Then we do the same for all subdiagonal elements, eventually ending up with $n(n-1)/2$ rotators multiplied together to get

$$Qn(n-1)^T Q_{n(n-2)}^T \cdots Q_{31}^T Q_{21}^T A = R$$

with R upper triangular. Crucially, since if Q_1 and Q_2 are orthogonal, then $Q_1 Q_2$ is orthogonal too, since

$$(Q_1 Q_2)^T Q_1 Q_2 = Q_2^T Q_1^T Q_1 Q_2 = I,$$

and so the large product of rotators above, call it Q^T, is orthogonal, and hence $A = QR$. □

16.1 Reflectors

Given $v, u \in \mathbb{R}^n$ with $\|u\|_2 = 1$ and $\langle u, v \rangle = 0$, we have that $\{ u, v \}$ is a basis of span$\{ u, v \}$. Hence if $x \in \text{span}\{ u, v \}$ we must have $x = \alpha u + \beta v$ for some $\alpha, \beta \in \mathbb{R}$.

We want to construct a linear transformation $Q \in M_n(\mathbb{R})$ such that Qx is the reflection of x through the line $L = \text{span}\{ v \}$.

Clearly we *want* the result to be $Qx = -\alpha u + \beta v$, since we want to reverse α, which is the coefficient in the u, i.e. orthogonal direction, to L.

Now let $P = uu^T$, which is then an $n \times n$ matrix of rank 1. Notice that $Pu = uu^T u = u$ since $u^T u = 1$, and $Pv = uu^T v = 0$.

Let $Q = I - 2P = I - 2uu^T$, so

$$\begin{aligned} Qx = Q(\alpha u + \beta v) &= \alpha Qu + \beta Qv = \alpha(u - 2Pu) + \beta(v - 2Pv) \\ &= \alpha(u - 2u) + \beta(v - 0) = -\alpha u + \beta v. \end{aligned}$$

Notice that $Q^T = (I - 2uu^T)^T = I - 2uu^T = Q$, so Q is symmetric.

Date: February 16th, 2018.

Moreover

$$QQ^T = I - 2uu^T - 2uu^T + 4uu^T uu^T = I$$

so Q is a symmetric, orthogonal matrix, and we call it a ***reflector***.

Intuivively this makes sense, incidentally: P above is the matrix that projects on the line L, and if we double the vector that brings us onto L, we'll of course reflect through L. Likewise Q should be orthogonal since reflecting and then reflecting again ought to get us back to where we started.

We summarise this in the following theorems:

Theorem 16.1.1. *Let $u \in \mathbb{R}^n$, $\|u\|_2 = 1$, and $P = uu^T \in M_n(\mathbb{R})$. Then*

(*i*) $Pu = u$,

(*ii*) $Pv = 0$ *if and only if* $\langle u, v\rangle = 0$,

(*iii*) $P^2 = P$, *and*

(*iv*) $P^T = P$.

Theorem 16.1.2. *Let $u \in \mathbb{R}^n$, $\|u\|_2 = 1$, and $Q = I - 2uu^T$. Then*

(*i*) $Qu = u$,

(*ii*) $Qv = v$ *if and only if* $\langle u, v\rangle = 0$,

(*iii*) $Q = Q^T$,

(*iv*) $Q^T = Q^{-1}$, *and*

(*v*) $Q^{-1} = Q$.

More generally, we can construct orthogonal matrices like Q by $Q = I - \gamma uu^T$ for $u \neq 0$, where $\gamma = 2/(u^T u)$, where $u^T u$ isn't necessarily 1. These are called ***Householder transformations***.

Theorem 16.1.3. *Let $x, y \in \mathbb{R}^n$, with $x \neq y$ and $\|x\|_2 = \|y\|_2 = 1$. Then there exists a (unique) reflector Q such that $Qx = y$.*

Proof. We need to find u such that $(I - \gamma uu^T)x = y$, with $\gamma = 2/(u^T u)$.

Write $x = (x - y)/2 + (x + y)/2$, and take $u = (x - y)$. Then

$$Qu = Q(x - y) = (I - \gamma uu^T)u = u - \gamma uu^T u = u - 2u = -u = y - x.$$

Also,

$$\langle x + y, x - y\rangle = \langle x, x\rangle - \langle x, y\rangle + \langle y, x\rangle - \langle y, y\rangle = 1 - 1 = 0,$$

so $x + y$ and $x - y$ are orthogonal, so $Q(x + y) = x + y$ and $Q(x - y) = y - x$, whereby $Qx - Qy = y - x$ and $Qx + Qy = x + y$, and adding these we get $Qx = y$. □

Corollary 16.1.4. *For every $x \in \mathbb{R}^n$, there exists a reflector Q such that $Qx = (*, 0, \ldots, 0)$ where $* \neq 0$ if $x \neq 0$ since Q is invertible. In fact, $* = \pm\|x\|_2$ since Q is orthogonal, meaning it preserves lengths.*

It is possible to accomplish QR decomposition using reflectors too.

Proof. We perform induction on n. If $n = 1$, then $Q = [1]$ and $R = [a_{11}]$, so $A = [a_{11}] = QR$, where R is vacuously upper triangular.

Assume we can QR decompose every $(n-1) \times (n-1)$ matrix, and consider the $n \times n$ case.

We find a reflector Q_1^T so that we get 0 everywhere under the first element in the first row of A, making

$$Q_1^T A = \begin{bmatrix} -\tau_1 & \hat{a}_2 \\ 0 & \hat{A}_2, \end{bmatrix}$$

as a block matrix. Then $\hat{A}_2 = \hat{Q}_2 \hat{R}_2$ by the inductive hypothesis, and so if we let

$$\tilde{Q}_2 = \begin{bmatrix} 1 & 0 \\ 0 & \hat{Q}_2 \end{bmatrix}$$

then we have

$$\hat{Q}_2^T Q_1^T A = \begin{bmatrix} -\hat{\tau}_1 & \hat{a}_2 \\ 0 & \hat{R}_2 \end{bmatrix} = R$$

which is upper triangular, and we have $A = QR$, with $Q = Q_1 \tilde{Q}_2$. □

Lecture 17 Uniqueness of QR Decomposition

17.1 Uniqueness

Theorem 17.1.1. *Let $A \in M_n(\mathbb{R})$ be nonsingular. Then there exists* unique *orthogonal matrix $Q \in M_n(\mathbb{R})$ and upper triangular matrix $R \in M_n(\mathbb{R})$ with positive diagonal entries such that $A = QR$.*

Proof. The existence of $A = \hat{Q}\hat{R}$ with $\hat{Q}$ orthogonal, and $\hat{R}$ upper triangular is known and assumed (we did it with both rotators and reflectors last time).

Now since A is nonsingular, $\hat{R}$ is nonsingular, whereby its diagonal elements $r_{jj} \neq 0$ for all $j = 1, 2, \ldots, n$.

Let

$$D = \begin{bmatrix} d_{11} & & & \\ & d_{22} & & \\ & & \ddots & \\ & & & d_{nn} \end{bmatrix}$$

be a diagonal matrix such that

$$d_{jj} = \begin{cases} 1 & \text{if } r_{jj} > 0 \\ -1 & \text{if } r_{jj} < 0. \end{cases}$$

Then clearly $D = D^T = D^{-1}$, and we let $Q = \hat{Q}D^{-1}$ and $R = D\hat{R}$, whence

$$A = \hat{Q}\hat{R} = QDD^{-1}R = QR$$

where Q is orthogonal, being the product of orthogonal matrices, and R is upper triangular with positive diagonal entries by construction.

Date: February 21st, 2018.

Let us now prove uniqueness. Suppose $A = Q_1R_1 = Q_2R_2$ with Q_i orthogonal and R_i upper triangular with positive diagonal entries as described.

Notice that

$$A^TA = R_1^TQ_1^TQ_1R_1 = R_1^TR_1$$

which is the Cholesky factorisation of the positive detinite matrix A^TA, which is unique. Similarly $A^TA = R_2^TR_2$, so $R_1 = R_2$, and hence $Q_2 = AR_2^{-1} = AR_1^{-1} = Q_1$. □

Note for the record that computations with rotators and reflectors is a stable process (i.e. backward stable), as shown in, for instance, Wilkonson's book.

17.2 Generalisation to Complex Numbers

We'll say a few words about how this all works over the complex numbers instead of the reals.

With $x, y \in \mathbb{C}^n$ we then instead define

$$\langle x, y\rangle = \sum_{i=1}^{n} x_i\overline{y_i} = y^*x,$$

where y^* is the conjugate transpose of y. Then $\langle x, y\rangle = \overline{\langle y, x\rangle}$,

$$\langle \alpha_1 x + \alpha_2 y, z\rangle = \alpha_1\langle x, z\rangle + \alpha_2\langle y, z\rangle$$

and

$$\langle x, \alpha_1 y + \alpha_2 z\rangle = \overline{\alpha_1}\langle x, y\rangle + \overline{\alpha_2}\langle x, z\rangle.$$

We moreover have

$$\langle x, x\rangle = \sum_{j=1}^{n} x_j\overline{x_j} = \sum_{j=1}^{n}|x_j|^2 \geq 0,$$

and in particular $\langle x, x\rangle = 0$ if and only if $x = 0$. As before, $\|x\|_2 = \sqrt{\langle x, x\rangle}$.

The notion of an orthogonal matrix generalises to ***unitary***, namely $U^*U = I$, so $U^{-1} = U^*$. For these we then have $\langle Ux, Uy\rangle = \langle x, y\rangle$, $\|Ux\|_2 = \|x\|_2$, and $\|U\|_2 = \|U^{-1}\|_2 = 1$, so $k_2(U) = 1$.

The complex rotators are similar, i.e. given $(a, b) \neq 0$, with $a, b \in \mathbb{C}$, we have

$$U = \frac{1}{\gamma}\begin{bmatrix} a & \overline{-b} \\ b & \overline{a} \end{bmatrix}$$

where $\gamma = \sqrt{|a|^2 + |b|^2}$, which is unitary, and

$$U^*\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \gamma \\ 0 \end{bmatrix}.$$

The complex reflectors are also similar, i.e. $Q = I - 2uu^*$, with $\|u\|_2 = 1$, and so $Qu = -u$, $Qv = v$ for all $\langle u, v\rangle = 0$, and $Q = Q^*$ and $Q^* = Q^{-1} = Q$.

17.3 Solution to the Least Squares Problem

Consider an overdetermined system $Ax = b$ with $A \in M_n(\mathbb{R})$, $x \in \mathbb{R}^m$, and $b \in \mathbb{R}^n$, for $n > m$.

Our task is to find $x \in R^m$ such that $\|r\|_2 = \|b - Ax\|_2$ is minimised. We will show that this always has a solution, i.e. a minimiser exists, and that this solution may or may not be unique.

It is unique if and only if $\operatorname{rank} A$ is as big as possible, i.e. A is full rank, meaning m.

Theorem 17.3.1. *Let $A \in M_{n,m}(\mathbb{R})$, $n > m$. Then there exist orthogonal matrix $Q \in M_n(\mathbb{R})$ and*

$$R = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} \in M_{n,m}(\mathbb{R})$$

where $\hat{R} \in M_m(\mathbb{R})$ is upper triangular such that $A = QR$.

Proof. We form the matrix

$$\breve{A} = \begin{bmatrix} A & \tilde{A} \end{bmatrix} \in M_n(\mathbb{R})$$

with $\tilde{A}$ an arbitrary matrix, maybe the 0-matrix. We know that there exists $Q \in M_n(\mathbb{R})$ orthogonal and $\breve{R} \in M_n(\mathbb{R})$ upper triangular such that $\breve{A} = Q\breve{R}$, and if we let

$$\breve{R} = \begin{bmatrix} R & T \end{bmatrix}$$

with $R \in M_{n,m}(\mathbb{R})$, it follows that $A = QR$ and

$$R = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}$$

and $\hat{R} \in M_m(\mathbb{R})$ is upper triangular. □

Lecture 18 The Least Squares Problem

18.1 Solving the Least Squares problem

Theorem 18.1.1. *Let $A \in M_n(\mathbb{R})$, $b \in \mathbb{R}^n$ with $n < m$, and $\operatorname{rank} A = m$. Then the least squares problem $Ax = b$ has a unique solution which can be found bt solving $\hat{R}x = \hat{c}$ where*

$$\begin{bmatrix} \hat{c} \\ d \end{bmatrix} = c = Q^T b,$$

$\hat{R} \in M_m(\mathbb{R})$ is upper triangular, and $\hat{c} \in \mathbb{R}^m$, with Q and $\hat{R}$ as in the previous theorem.

Proof. Let

$$A = QR = Q \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}$$

as in the previous theorem. We can then transform $Ax = b$ into $Q^T Ax = Q^T b = c$, calling $R = Q^T A$, so we solve $Rx = c$.

Date: February 23rd, 2018.

Now write

$$c = \begin{bmatrix} \hat{c} \\ d \end{bmatrix}$$

with $\hat{c} \in \mathbb{R}^m$, and let

$$s = c - Rx = \begin{bmatrix} \hat{c} \\ d \end{bmatrix} - \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} x = \begin{bmatrix} \hat{c} - \hat{R}x \\ 0 \end{bmatrix}.$$

We want to minimise $\|s\|_2$. Note that

$$\|s\|_2^2 = \sum_{i=1}^{n} |s_i|^2 = \|\hat{c} - \hat{R}x\|_2^2 + \|d\|_2^2.$$

Since $\|d\|_2$ is independent of x, $\|s\|_2$ is minimised at an x that minimuses $\|\hat{c} - \hat{R}x\|_2$. But $\hat{R}$ is invertible (it has rank m and is $m \times m$), so $\hat{R}x = \hat{c}$ has a unique solution (and $\|\hat{c} - \hat{R}x\| = 0$) which minimises $\|s\|_2$. □

Example 18.1.2. Solve the least squares problem

$$\begin{cases} x = 6 \\ 2x = 10, \end{cases}$$

which in matrix form is therefore

$$A\boldsymbol{x} = \begin{bmatrix} 1 \\ s \end{bmatrix} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} 6 \\ 10 \end{bmatrix} = b.$$

Hence we want to find x such that $\|b - A\boldsymbol{x}\|_2$ is minimised. Let us therefore first find the QR factorisation of A, which is

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & 2 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} \sqrt{5} \\ 0 \end{bmatrix} = QR$$

where $\hat{R} = \begin{bmatrix} \sqrt{5} \end{bmatrix}$, so

$$c = Q^T b = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & 2 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} 6 \\ 10 \end{bmatrix} = \frac{1}{\sqrt{5}} \begin{bmatrix} 26 \\ 2 \end{bmatrix}$$

whence $\hat{c} = 26/\sqrt{5}$ and we need to solve $\hat{R}x = \hat{c}$, or $\sqrt{5}x = 26/\sqrt{5}$, meaning that $x = 26/5 = 5.2$. ▲

18.2 The Gram-Schmidt Process

Recall that a set of vectors $\{ q_1, q_2, \ldots, q_k \} \subset \mathbb{R}^n$ being ***orthogonal*** means that

$$\langle q_i, q_j \rangle = 0$$

if $i \neq j$, and $\langle q_i, q_i \rangle \neq 0$.

Recall further that a set of nonzero vectors being orthogonal means that the vectors are linearly independent, and if in addition $\langle q_i, q_i \rangle = 1$ for every $i = 1, 2, \ldots, k$¡ we call the set $\{ q_1, q_2, \ldots, q_k \}$ an ***orthonormal set***.

Recall also that $Q \in M_n(\mathbb{R})$ is orthogonal, i.e. $Q^TQ = I$ if and only if the columns (and rows) form an orthonormal set of vectors.

The idea of the Gram-Schmidt orthogonalisation process is to take a set of linearly independent vectors and turn them into an orthogonal set. We'll illustrate with two vectors w_1 and w_2; our goal is to find v_1 and v_2 such that they are orthogonal and, importantly, they span the same space as w_1 and w_2.

It goes as follows: let $v_1 = w_1$. If we then imagine projecting w_2 orthogonally onto w_1, so that $v_2 = w_2 - cw_1$ for some c, we have

$$0 = \langle v_1, v_2 \rangle = \langle w_1, w_2 - cw_1 \rangle = \langle w_1, w_2 \rangle - c\langle w_1, w_1 \rangle$$

whereby

$$c = \frac{\langle w_1, w_2 \rangle}{\langle w_1, w_1 \rangle}.$$

So $v_1 = w_1$ and

$$v_2 = w_2 - \frac{\langle v_1, w_2 \rangle}{\langle v_1, v_1 \rangle} v_1.$$

In general, we have

Theorem 18.2.1 (Gram-Schmidt)**.** *Let $S = \{ w_1, w_2, \ldots, w_m \} \subset \mathbb{R}^n$ be linearly independent. Define $S' = \{ v_1, v_2, \ldots, v_m \}$ by $v_1 = w_1$ and*

$$v_k = w_k - \sum_{j=1}^{k-1} \frac{\langle w_k, v_j \rangle}{\langle v_j, v_j \rangle} v_j$$

for $k = 2, 3, \ldots, m$. Then S' is an orthogonal set of vectors and

$$\text{span}\{ w_1, w_2, \ldots, w_l \} = \text{span}\{ v_1, v_2, \ldots, v_l \}$$

for $l = 1, 2, \ldots, m$.

Lecture 19 Gram-Schmidt

19.1 Proof of Gram-Schmidt

Proof. Define $S_\ell = \{ w_1, w_2, \ldots, w_\ell \}$ for $\ell = 1, 2, \ldots, m$. We perform induction on ℓ.

For $\ell = 1$, $S_1 = \{ w_1 \}$, with $v_1 = w_1$, and so $S_1' = \{ v_1 \} = S_1$, which is vacuously orthogonal and the spans match.

Let $S_{\ell-1}' = \{ v_1, v_2, \ldots, v_{\ell-1} \}$ and assume it's been constructed with the desired properties. Consider

$$S_\ell' = S_{\ell-1}' \cup \{ v_\ell \},$$

with v_ℓ as described. Then

$$\langle v_\ell, v_i \rangle = \langle w_\ell, v_i \rangle - \sum_{j=1}^{\ell-1} \frac{\langle w_\ell, v_j \rangle}{\langle v_j, v_j \rangle} \langle v_j, v_i \rangle$$

Date: February 26th, 2018.

where by construction $\langle v_j, v_i \rangle = 0$ if $i \neq j$, and is nonzero if $i = j$. Hence the above becomes

$$\langle v_\ell, v_i \rangle = \langle w_\ell, v_i \rangle - \frac{\langle w_\ell, v_i \rangle}{\langle v_i, v_i \rangle} \langle v_i, v_i \rangle = \langle w_\ell, v_i \rangle - \langle w_\ell, v_i \rangle = 0.$$

All by which to say that S'_ℓ is an orthogonal set, and $\text{span}(S'_\ell) = \text{span}(S_\ell)$ since $v \in S'_\ell$ is, by construction, a linear combination of the vectors in S_ℓ. □

19.2 Connection Between Gram-Schmidt and QR

In Gram-Schmidt, we in essence have

$$v_k = w_k - \sum_{j=1}^{k-1} r_{jk} v_j$$

where

$$r_{jk} = \frac{\langle w_k, v_j \rangle}{\langle v_j, v_j \rangle}.$$

If we incorporate normalisation of v_k¡ i.e. replace v_k with $v_k / \|v_k\|_2$, then we can rewrite the above as

$$w_k = r_{kk} v_k + \sum_{j=1}^{k-1} r_{jk} v_j$$

where $r_{jk} = \langle w_{k-1}, v_j \rangle$, and hence for $k = 1$ we have $w_1 = r_{11} v_1$, for $k = 2$ we get $w_2 = r_{22} v_2 + r_{12} v_1$, for $k = 3$ we have $w_3 = r_{33} v_3 + r_{13} v_1 + r_{23} v_2$, and so forth, meaning that in all we have written

$$A = QR$$

where A has columns w_i and Q has columns v_i, where R is the upper triangular matrix composed of the r_{ij}.

Hence we have QR decomposed A.

19.3 Some Numerical Notes

In Gram-Schmidt we hope that the resulting QR decomposition satisfies $\|I - Q^T Q\|_2 \approx cu$, where the left-hand side in some sense measures how close the orthogonalised vectors v_i are to being an orthogonal set, and c is a small constant, with u being the unit round-off.

This of course depends on the condition number of A, but since A is generally not square we need to adjust the definition of this. This is not too bad; we proved that for invertible matrices we could equivalently view the condition number as the maximum magnification divided by the minimum magnification, which is still a reasonable thing do to with rectangulat matrices:

$$k(A) = \frac{\max_{x \neq 0} \frac{\|Ax\|}{\|x\|}}{\min_{x \neq 0} \frac{\|Ax\|}{\|x\|}}.$$

This $k(A)$ can be bad if the columns of A, i.e. the set of vectors to be orthogonalised, is nearly linearly dependent.

The remedy for this is to use multiple re-orthogonalisations, i.e. apply the step to produce v_k via w_k at least twice, substituting w_k with v_k, and recompute.

This was applied heuristically for many years, and the belief was that one reorthogonalisation was enough.

This was proven in 2005.

19.4 Geometric Approach to the Least Squares Problem

We need a few more tools.

Definition 19.4.1 (Orthogonal complement)**.** Given any set $S \subset \mathbb{R}^n$, define the ***orthogonal complement*** of S as

$$S^\perp = \{\, x \in \mathbb{R}^n \mid \langle x, y\rangle = 0 \ \forall y \in S \,\}.$$

That is to say, all vectors orthogonal to everything in S.

There are certain important properties $S^\perp$ has:

1. $S^\perp$ is a subspace of $\mathbb{R}^n$, regardless of S.
2. $\{\, 0 \,\}^\perp = \mathbb{R}^n$ and $(\mathbb{R}^n)^\perp = \{\, 0 \,\}$.
3. If W is a subspace (so as to force $0 \in W$) of $\mathbb{R}^n$, then $W \cap W^\perp$.

   This is easy to prove: let $x \in W \cap W^\perp$, so that $x \in W^\perp$, meaning that x is perpendicular to everything in W, and in particular to x itself, so $\langle x, x\rangle = 0$, which is equivalent with $x = 0$.
4. For $S \subset \mathbb{R}^n$, $(S^\perp)^\perp = \operatorname{span}(S)$.

   (In infinite dimensional vector spaces, it's instead the closure of the span.)

Theorem 19.4.2 (Orthogonal decomposition theorem)**.** *Let S be a subspace of $\mathbb{R}^n$. Then for every $x \in \mathbb{R}^n$, there exists a unique $s \in S$ and $\hat{s} \in S^\perp$ such that $x = s + \hat{s}$.*

Moreover if $\{\, v_1, v_2, \ldots, v_k \,\}$ is an orthonormal basis for S (i.e. $\dim S = k$), then

$$s = \sum_{i=1}^{k} \langle x, v_i\rangle v_i$$

and $\hat{s} = x - s$. The coefficients $\langle x, v_i\rangle$ are called the **Fourier coefficients** *of x in S.*

The resulting s is called the **orthogonal projection** *of x onto S.*

Lecture 20 Orthogonal Projections

20.1 Least Squares Problem Using Orthogonal Projections

We start by proving the Orthogonal decomposition theorem from last time.

Date: February 28th, 2018.

Proof. Let $\{ v_1, v_2, \ldots, v_k \}$ be an orthonormal basis for S. Thus the prescribed

$$s = \sum_{i=1}^{k} \langle x, v_i \rangle v_i$$

is in S. Let $\hat{s} = x - s$, so that $x = s + \hat{s}$. We only need to show $\hat{s} \in S^\perp$. Indeed

$$\langle \hat{s}, v_j \rangle = \langle x - s, v_j \rangle = \left\langle x - \sum_{i=1}^{k} \langle x, v_i \rangle v_i, v_j \right\rangle = \langle x, v_j \rangle - \langle x, v_j \rangle \langle v_j, v_j \rangle = 0,$$

where the sum mostly vanishes since the v_j's are orthonormal, and likewise the last term vanishes for the same reason.

Hence $\hat{s}$ is orthogonal to a basis of S, and so orthogonal to S.

To show uniqueness of s and $\hat{s}$, consider

$$x = s + \hat{s} = s_1 + \hat{s}_1,$$

with $s, s_1 \in S$ and $\hat{s}, \hat{s}_1 \in S^\perp$. Then $s - s_1 = \hat{s}_1 - \hat{s}$, where the left-hand side is in S and the right-hand set is in $S^\perp$, meaning that they're both in $S \cap S^\perp = \{ 0 \}$, so $s = s_1$ and $\hat{s} = \hat{s}_1$. □

Example 20.1.1. Let

$$S = \operatorname{span}\{ u_1, u_2 \} = \operatorname{span}\{ (1,1,0), (1,0,1) \},$$

and let $x = (0,1,1)$ and $y = (2,1,1)$.

We want to write x and y as sums of vectors in S and $S^\perp$. Hence we first need an orthonormal basis for S, so as to apply the orthogonal decomposition theorem, so we use Gram-Schmidt. Let $v_1 = u_1 = (1,1,0)$. Then

$$v_2 = u_2 - \frac{\langle v_1, u_2 \rangle}{\langle u_1, u_1 \rangle} v_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} - \frac{1^2 + 0 + 0}{1^1 + 1^2 + 0} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} - \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}.$$

Now $\{ v_1, v_2 \}$ forms an orthogonal basis for S, and if we replace them by their normalised selves we have $v_1 = 1/\sqrt{2}(1,1,0)$ and $v_2 = 1/\sqrt{6}(1,-1,2)$. Then we have $x = s + \hat{s}$ with

$$s = \langle x, v_1 \rangle v_1 + \langle x, v_2 \rangle v_2 = \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \frac{1}{\sqrt{6}} \frac{1}{\sqrt{6}} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = \frac{3}{6} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \frac{1}{6} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

Hence also

$$\hat{s} = x - s = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} - \frac{1}{3} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} -2 \\ 2 \\ 2 \end{bmatrix}$$

Note also that $y \in S$, so when we decompose it as $y = t + \hat{t}$, with $t \in S$ and $\hat{t} \in S^\perp$, then we will find $y = t$ and $\hat{t} = 0$. ▲

Theorem 20.1.2 (Pythagoras)**.** *Let $x, y \in \mathbb{R}^n$. Then $x \perp y$ if and only if*

$$\|x + y\|_2^2 = \|x\|_2^2 + \|y\|_2^2.$$

Proof. This is straight-forward computation:

$$\|x+y\|_2^2 = \langle x+y, x+y\rangle = \langle x,x\rangle + 2\langle x,y\rangle + \langle y,y\rangle = \|x\|_2^2 + \|y\|_2^2 + 2\langle x,y\rangle,$$

so the result holds if and only if $\langle x,y\rangle = 0$, if and only if $x \perp y$. □

We now get the following corollary to the Orthogonal decomposition theorem:

Corollary 20.1.3. *Let S be a subspace of $\mathbb{R}^n$. Let $x \in \mathbb{R}^n$, and write $x = s + \hat{s}$ with $s \in S$ and $\hat{s} \in S^\perp$. Then for every $y \in S$ we have*

$$\|x-y\|_2 \geq \|x-s\|_2,$$

with equality if and only if $y = s$.

In other words, $s \in S$ is the vector in S closest to the given x.

Proof. Let $x = s + \hat{s}$, so for every $y \in S$ we have $s - y \in S$ since it's a subspace, and so $s - y \perp \hat{s}$. Thus

$$\|x-y\|_2^2 = \|s+\hat{s}-y\|_2^2 = \|(s-y)+\hat{s}\|_2^2 = \|s-y\|_2^2 + \|\hat{s}\|_2^2$$

by Pythagoras theorem, whence if we drop the nonnegative term $\|s-y\|_2^2$ we get

$$\|x-y\|_2^2 \geq \|\hat{s}\|_2^2 = \|x-s\|_2^2.$$

The term we dropped is 0 if and only of $s = y$, and we're done. □

The idea in solving the least squares problem with this is precisely that we want to find, given $A \in M_{n,m}(\mathbb{R})$ and $b \in \mathbb{R}^n$, some $x \in \mathbb{R}^m$ such that $\|b - Ax\|_2$ is minimised.

So in essence we want to find the distance between b and $S = R(A) = \{\, Ax \,|\, x \in \mathbb{R}^m \,\}$, whence all we need to do is decompose $b = s + \hat{s}$ with $s \in S$ and $\hat{s} \in S^\perp$, and then solve the system $Ax = s$ for x, which has an exact solution, though not necessarily unique.

Lecture 21 Normal Equations

21.1 Solving the Least Squares Problem a Third Way

Let us first work an example of the above process.

Example 21.1.1. Solve

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

in the least squares sense. In other words we want to find x such that $\|Ax - b\|_2$ is minimised. This means that we want Ax to be the projection of b onto $R(A)$, so we write $b = s + \hat{s}$, and as we know from a previous example $s = 1/3(2,1,1)$.

Hence we want to solve

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

yielding the unique (because of full rank of A) solution $x_1 = x_2 = 1/3$. ▲

Date: March 2nd, 2018.

In order to solve the least squares problem with so-called normal equations, we first need two lemmata.

Lemma 21.1.2. *Let $A \in M_{n,m}(\mathbb{R})$, $x \in \mathbb{R}^m$, and $y \in \mathbb{R}^n$. Then $\langle Ax, y\rangle = \langle x, A^T y\rangle$.*

Proof. This is a simple computation:

$$\langle Ax, y\rangle = (y^T)(Ax) = (y^T A)(x) = (A^T y)^T x = \langle x, A^T y\rangle. \qquad \square$$

Lemma 21.1.3. *Let $A \in M_{n,m}(\mathbb{R})$. Then* $\operatorname{rank}(A^T A) = \operatorname{rank}(A)$.

Proof. To prove this theorem it suffices to show that $A^T Ax = 0$ if and only if $Ax = 0$, since that means their kernels are equal, whence the dimension of their kernels are equal, whence by the rank-nullity theorem their ranks are equal.

Therefore suppose $Ax = 0$. Then $A^T Ax = A^T 0 = 0$.

On the other hand, if $A^T Ax = 0$, then

$$0 = \langle x, A^T Ax\rangle = \langle Ax, Ax\rangle = \|Ax\|_2^2$$

which implies (in fact is equivalent to) $Ax = 0$. $\square$

Theorem 21.1.4. *Let $A \in M_{n,m}(\mathbb{R})$ and $b \in \mathbb{R}^n$. Then there exists $x_0 \in \mathbb{R}^m$ such that $A^T Ax_0 = A^T b$.*

Furthermore $\|Ax_0 - b\|_2 \leq \|Ax - b\|_2$ for all $x \in \mathbb{R}^m$. In other words, x_0 is the least squares solution to $Ax = b$.

Proof. Suppose x_0 is a solution to $A^T Ax_0 = A^T b$. This is true if and only if $A^T Ax_0 - A^T b = 0$, if and only if $\langle x, A^T Ax_0 - A^T b\rangle = 0$ for all $x \in \mathbb{R}^m$, which is true if and only if $\langle Ax, Ax_0 - b\rangle = 0$ for all x, which is true if and only if $Ax_0 - b \in R(A)^\perp$.

Now $b = Ax_0 = \hat{s}$ by the decomposition theorem, and we are done. $\square$

We call $A^T Ax = A^T b$ the ***normal equations*** for the least squares problem $Ax = b$.

The advantage of this is that it requires no orthogonalisation explicitly.

Note also that by the second lemma, the normal equations have a unique solution if and only if $\operatorname{rank}(A) = \operatorname{rank}(A^T A) = m$, i.e. full rank.

Example 21.1.5. Find the best fit line in the least-squares sense to the date $(1, 2)$, $(3, 4)$, $(5, 7)$, $(7, 9)$, and $(9, 12)$, listed as pairs (t_i, y_i).

Hence we want a line $y = ct + d$ that minimises the sum of squares of errors. This is the same as asking to solve the least squares problem

$$Ax = \begin{bmatrix} 1 & 1 \\ 3 & 1 \\ 5 & 1 \\ 7 & 1 \\ 9 & 1 \end{bmatrix} \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 7 \\ 9 \\ 12 \end{bmatrix} = b.$$

Then

$$A^T A = \begin{bmatrix} 165 & 25 \\ 25 & 5 \end{bmatrix} \qquad \text{and} \qquad A^T b = \begin{bmatrix} 220 \\ 34 \end{bmatrix}.$$

Hence we solve the normal equations

$$\begin{bmatrix} 165 & 25 \\ 25 & 5 \end{bmatrix} \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 220 \\ 34 \end{bmatrix},$$

which has the unique solution

$$x_0 = \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 1.25 \\ 0.55 \end{bmatrix},$$

so the line of best fit is $y = 1.25t + 0.55$. ▲

Lecture 22 Minimal Solutions

22.1 A Twist On Normal Equations

Suppose we have the linear system $Ax = b$ where $b \in R(A)$, i.e. the system is consistend.

Our task is to find a solution s such that $As = b$ and $\|s\|_2 \leq \|u\|_2$ for all other solutions u.

We call such an s a ***minimal solution*** to the system.

Theorem 22.1.1. *Let $A \in M_{n,m}(\mathbb{R})$ and $b \in \mathbb{R}^n$. Sooise $Ax = b$ is consistent. Then*

(*i*) *There exists a minimal solution s, and $s \in R(A^T)$.*

(*ii*) *s is the only solution that lies in $R(A^T)$, i.e. $AA^Tu = b$ implies $A^Tu = s$.*

Proof. (*i*) Let $W = R(A^T)$, i.e. the row space of A, and suppose $Ax = b$. By the orthogonal decomposition theorem we can write $x = s + \hat{s}$, where $s \in W$ and $\hat{s} \in W^\perp$. But $\hat{s} \in W^\perp$ is equivalent with $\hat{s} \perp A^Tu$ for all $u \in \mathbb{R}^n$, which in turn is equivalent with $\langle \hat{s}, A^Tu \rangle =$ for all u, so $\langle A\hat{s}, u \rangle = 0$ for all u, if and only if $A\hat{s} = 0$, if and only if $\hat{s} \in \ker A$.

Thus $b = Ax = A(s + \hat{s}) = As + A\hat{s} = As$, i.e. $s \in W = R(A^T)$, and s is a solution to the system. Moreover s is a minimal solution since if u is any other solution, then $u = s_1 + \hat{s}_1$, with $s_1 \in W$ and $\hat{s}_1 \in W^\perp$, then the two solutions differ by a vector in $\ker A$, since $A(s - u) = b - b = 0$, so $s - u \in \ker A$.

Hence $s_1 = s$. Thus finally

$$\|u\|_2^2 = \|s + \hat{s}_1\|_2^2 = \|s\|_2^2 + \|\hat{s}_1\|_2^2 \geq \|s\|_2^2$$

by Pythagoras' theorem.

(*ii*) Say $Av = b$, with $v \in R(A^T)$, in addition to $As = b$. Then $v - s \in \ker A = R(A^T)^\perp$, as per above. Moreover $v - s \in R(A^T)$ since $v, s \in R(A^T)$, and this is a subspace, so

$$v - s \in R(A^T) \cap R(A^T)^\perp = \{\, 0 \,\},$$

hence $v = s$. □

To find a minimal solution of $Ax = b$, we hence solve $A(A^Tu) = b$ for u, then $s = A^Tu$ is the minimal solution.

Date: March 5th, 2018.

Lecture 23 Singular Value Decomposition

23.1 Introduction

The ***singular value decomposition*** is summarised in the following theorem:

Theorem 23.1.1 (Singular value decomposition)**.** *Let $A \in M_{n,m}(\mathbb{R})$ be a nonzero matrix with* $\operatorname{rank} A = r$ *(meaning $r \leq \min\{ m, n \}$). Then*

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

where $U \in M_n(\mathbb{R})$ and $V \in M_m(\mathbb{R})$ are both orthogonal (i.e. $U^TU = I_n$ and $V^TV = I_m$), and

$$\Sigma = \begin{bmatrix} \sigma_1 & & & & & & \\ & \sigma_2 & & & & & \\ & & \ddots & & & & \\ & & & \sigma_r & & & \\ & & & & 0 & & \\ & & & & & \ddots & \end{bmatrix} \in M_{n,m}(\mathbb{R})$$

is rectangular diagonal with

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \geq \sigma_r > 0.$$

We make a few notes. $A = U\Sigma V^T$ is called the ***singular value decomposition*** of A. The matrices U and V are not necessarily unique, but they have traits related to A.

The numbers $\sigma_1, \sigma_2, \ldots, \sigma_r$ are called the ***singular values*** of A, and the columns of U are the ***left singular vectors*** and the columns of V are the ***right singular vectors*** of A.

Sometimes we include

$$\sigma_{r+1} = \sigma_{r+2} = \ldots = \sigma_k = 0$$

where $k = \min\{ m, n \}$ under the notion of singular values of A.

Note finally that by convention we say that the 0 matrix has all singular values equal to 0—note that $\operatorname{rank} 0 = 0$.

We also make note of the following two equivalent formulations of the theorem, which come in handy sometimes.

First, $A = U\Sigma V^T$ is the same as saying

$$A = \sum_{j=1}^{r} \sigma_j u_j v_j^T$$

where u_j are the columns of U and v_j are the columns of V, which tells us that A is a linear combiniation of the rank 1 matrices $u_j v_j^T$, with coefficients being the singular values.

Date: March 19th, 2018.

With the same notation as in the theorem, we also have

$$Av_i = \begin{cases} \sigma_i u_i, & \text{if } i = 1, 2, \ldots, r \\ 0, & \text{otherwise,} \end{cases}$$

and

$$A^T u_i = \begin{cases} \sigma_i v_i, & \text{if } i = 1, 2, \ldots, r \\ 0, & \text{otherwise,} \end{cases}$$

since $A = U\Sigma V^T$ is equivalent with $AV = USigma$ and $U^T A = \Sigma V^T$, since U and V are orthogonal.

This is interesting because it says that v_i and u_i are almost like eigenvectors, except they switch between themselves.

To prove this we'll need the following lemma:

Lemma 23.1.2. *Given $A \in M_{n,m}(\mathbb{R})$, $A^T A \in M_m(\mathbb{R})$, and all its eigenvalues are nonnegative numbers (i.e. $A^T A$ is Hermitian and positive definite).*

Proof. Suppose $A^T Ax = \lambda x$, with $x \neq 0$, i.e. x is an eigenvector and λ is an eigenvalue. Then $x^T A^T Ax = \lambda x^T x$, so $\langle Ax, Ax\rangle = \lambda\langle x, x\rangle$, whence

$$\lambda = \frac{\langle Ax, Ax\rangle}{\langle x, x\rangle} = \frac{\|Ax\|_2^2}{\|x\|_2^2} \geq 0$$

noting that $\|x\|_2 > 0$. □

Lecture 24 The Existence of Singular Value Decomposition

24.1 Proof of the Singular Value Decomposition Theorem

Proof. Let the eigenvalues of $A^T A$, which is an $m \times m$ matrix, be

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m \geq 0;$$

we proved earlier that the eignevalues are all nonnegative. Since $A^T A$ is Hermitian, to each eigenvalue λ_j there corresponds an eigenvector v_j such that $\{ v_1, v_2, \ldots, v_m \}$ is an orthonormal set.

Now consider the set $\{ Av_1, Av_2, \ldots, Av_m \}$. This is an orthogonal set, since for $i \neq j$ we have

$$(Av_j)^T (Av_i) = v_j^T A^T Av_i$$

but v_i is an eigenvector of $A^T A$, so this is

$$v_j^T \lambda_i v_i = \lambda_i v_j^T v_i = \lambda_i \cdot 0 = 0.$$

In fact, $\{ Av_1, Av_2, \ldots, Av_r \}$ is an orthogonal basis for $R(A)$. To prove this we need only argue that $Av_j \neq 0$ for $j = 1, 2, \ldots, r$, since their span is in $R(A)$, the rank of A is the same as the dimension of $R(A)$, and they're orthogonal, so if they're all nonzero they form a basis.

Date: March 21st, 2018.

Clearly $Av_j \neq 0$, for otherwise $A^TAv_j = 0$, but $A^TAv_j = \lambda_j v_j$, i.e. $\lambda_j = 0$ for some $j = 1, 2, \ldots, r$. But this is a contradiciton since $\operatorname{rank}(A^TA) = \operatorname{rank}(A) = r$, saying that the r largest eigenvalues of A^TA are positive.

We normalise the vectors Av_i, i.e. replace them with $u_j = Av_j/\|Av_j\|_2$, for $j = 1, 2, \ldots, r$. Hence

$$u_j = \frac{Av_j}{\sqrt{v_j^TA^TAv_j}} = \frac{Av_j}{\sqrt{\lambda_j v_j^Tv_j}} = \frac{Av_j}{\sqrt{\lambda_j}\|v_j\|_2} = \frac{Av_j}{\sqrt{\lambda_j}}.$$

Now take this orthonormal basis for $R(A)$ and extend it to an orthonormal basis for $\mathbb{C}^n$, i.e. add arbitrary norm 1 nonzero vectors that are orthogonal to the existing ones so that

$$\{\, u_1, u_2, \ldots, u_r, u_{r+1}, u_{r+2}, \ldots, u_n \,\}$$

spans $\mathbb{C}^n$.

Let U be the matrix whose columns are u_i, and let V be the matrix whose columns are v_i, and finally let Σ be the diagonal $n \times m$ matrix with diagonal elements $\sigma_j = \sqrt{\lambda_j}$ for $j = 1, 2, \ldots, r$. Hence $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ since λ_j are ordered likewise.

We are done, since

$$AV = \begin{bmatrix} Av_1 & Av_2 & \cdots & Av_m \end{bmatrix} = \begin{bmatrix} \sigma_1 u_1 & \sigma_2 u_2 & \cdots & \sigma_r u_r & 0 & \cdots & 0 \end{bmatrix} = U\Sigma,$$

whence $A = U\Sigma V^T$ since $V^T = V^{-1}$. □

Do note that this is a constructive proof! Granted, finding the eigenvalues is nontrivial, but once we have them we get the singular value decomposition almost for free.

24.2 Applications of the Singular Value Decomposition

This is a remarkable useful decomposition, as we shall see in the next handful results.

In what follows we will let A, U, Σ, and V be as described before.

The singular value decomposition lets us easily compute the spectral norm $\|\cdot\|_2$ of a matrix.

Theorem 24.2.1. *Let $A \in \mathbb{R}^{n\times m}$ with $\operatorname{rank} A = r$ and let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ be its singular values. Then*

$$\|A\|_2 = \max_{x\neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \sigma_1.$$

Proof. Suppose $A \neq 0$—if it is not its largest singular value is trivially 0, and likewise $\|A\|_2 = 0$.

We have $Av_1 = \sigma_1 u_1$, whence

$$\frac{\|Av_1\|_2}{\|v_1\|_2} = |\sigma_1|\frac{\|u_1\|_2}{\|v_1\|_2} = \sigma_1.$$

Hence

$$\|A\|_2 \geq \frac{\|Av_1\|_2}{\|v_1\|_2} = \sigma_1.$$

Next let $x \in \mathbb{R}^m \setminus \{0\}$, and write $x = c_1v_1 + c_2v_2 + \ldots + c_mv_m$. Then

$$\|x\|_2^2 = |c_1|^2\|v_1\|_2^2 + |c_2|^2\|v_2\|_2^2 + \ldots + |c_m|^2\|v_m\|_2^2 = |c_1|^2 + |c_2|^2 + \ldots + |c_m|^2$$

by Pythagora's theorem since the v_i are orthonormal.

Moreover

$$Ax = c_1Av_1 + c_2Av_2 + \ldots + c_mAv_m = c_1\sigma_1u_1 + c_2\sigma_2u_2 + \ldots + c_r\sigma_ru_r + 0 + \ldots + 0,$$

whence

$$\|Ax\|_2^2 = |c_1\sigma_1|^2 + |c_2\sigma_2|^2 + \ldots + |c_r\sigma_r|^2$$

as above, and hence if we replace all σ_i by σ_1, the biggest of them, we can bound this by

$$\|Ax\|_2^2 \leq \sigma_1^2(|c_1|^2 + |c_2|^2 + \ldots + |c_r|^2) \leq \sigma_1^2\|x\|_2^2.$$

This then means that

$$\frac{\|Ax\|_2}{\|x\|_2} \leq \sigma_1,$$

and hence

$$\|A\|_2 = \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} \leq \sigma_1$$

and thereby $\|A\|_2 = \sigma_1$ as claimed. □

Theorem 24.2.2. *Let $A \in \mathbb{R}^{n\times m}$ be nonsingular with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0$. Then*

$$k_2(A) = \frac{\sigma_1}{\sigma_n}.$$

Lecture 25 Applications of Singular Value Decomposition

We begin by proving the theorem stated at the end of the last lecture.

Proof. As usual, let $A = U\Sigma V^T$ be a singular value decomposition. Then

$$A^{-1} = (V^T)^{-1}\Sigma^{-1}U^{-1} = (V^{-1})^{-1}\Sigma^{-1}U^T = V\Sigma^{-1}U^T,$$

which is a singular value decomposition of A^{-1} with singular values

$$\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_n^{-1},$$

which when ordered are

$$\sigma_n^{-1} \geq \sigma_{n-1}^{-1} \geq \ldots \geq \sigma_1^{-1} > 0,$$

so by the previous theorem $\|A^{-1}\|_2 = \sigma_n^{-1}$, whereby

$$k_2(A) = \|A\|_2\|A^{-1}\|_2 = \sigma_1\sigma_n^{-1} = \frac{\sigma_1}{\sigma_n}.$$ □

Date: March 23rd, 2018.

25.1 (Numerical) Rank Determination

A question we ask ourselves is this: how does one find the rank of a matrix $A \in \mathbb{R}^{n\times m}$? We have essentially two options:

We could count the (nonzero) pivots in an echelon form of A, i.e. use row reduction or Gaussian elimination. This might be numerically dangerous due to round-off.

Alternatively we could find a singular value decomposition and check how many nonzero singular values it has. It the smallest nonsingular value is σ_r, then $\operatorname{rank}(A) = r$.

This can unfortunately also be problematic.

Example 25.1.1. Consider the matrix

$$A = \begin{bmatrix} 1/3 & 1/3 & 2/3 \\ 2/3 & 2/3 & 4/3 \\ 1/3 & 2/3 & 3/3 \\ 2/5 & 2/5 & 4/5 \\ 3/5 & 1/5 & 4/5 \end{bmatrix}.$$

Notice how the last column is the sum of the previous two. Moreover notice how the first two columns aren't parallel, whence the rank of this matrix is 2.

However, the singular values of this, as computed numerically by MATLAB, are

$$\sigma_1 = 2.5987 > \sigma_2 = 0.3682 > \sigma_3 = 3.7257 \cdot 10^{-16},$$

the last of which, according to the computer, is not *quite* zero, even though we would like it to be. Hence we should decide that σ_3, as computed, is small enough that we consider it 0. ▲

In practice what we do is this: order the singular values, and identify a gap as follows:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k \gg \varepsilon \geq \sigma_{k+1} \geq \ldots,$$

where by $\gg$ we mean that σ_k is *much* bigger than ε, and we take $\varepsilon = 10u\|A\|$, with u being the unit round-off. In other words we want a, in some relative sense, big gap between σ_k and σ_{k+1}.

If this happens we say that the numerical rank of A is k.

Note, however, that we can always fool the computer!

Theorem 25.1.2. *Let $A \in \mathbb{R}^{n\times m}$ with $\operatorname{rank}(A) = r < \min\{ n, m \}$, i.e. not full rank. Let $\varepsilon > 0$. Then we can construct a matrix A_ε such that*

$$\|A - A_\varepsilon\|_2 = \varepsilon$$

and

$$\operatorname{rank} A_\varepsilon = \min\{ n, m \} > \operatorname{rank} A.$$

In other words there exists a full rank matrix however close to A we like, which is to say that full rank matrices are dense.

Proof. Let $A = U\Sigma V^T$ be a singular value decomposition of A where

$$\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r, 0, \ldots, 0) \in \mathbb{R}^{n\times m}.$$

Next define

$$\Sigma_\varepsilon = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r, \varepsilon, \ldots, \varepsilon) \in \mathbb{R}^{n\times m}$$

and use it to define $A_\varepsilon = U\Sigma_\varepsilon V^T$. Then $\|A - A_\varepsilon\|_2 = \|\Sigma - \Sigma_\varepsilon\|_2 = \varepsilon$ since multiplication by orthogonal matrices U and V^T do not affect the norm.

Then moreover $\operatorname{rank} A_\varepsilon = \min\{ n, m \}$. □

Theorem 25.1.3. *Let $A \in \mathbb{R}^{n\times m}$ with $\operatorname{rank} A = r > 0$ and $A = U\Sigma V^T$ a singular value decomposition. Call*

$$\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r, 0, \ldots, 0).$$

For $k = 1, 2, \ldots, r-1$, define $A_k = U\Sigma_k V^T$ where

$$\Sigma_k = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k, 0, \ldots, 0).$$

Then $\operatorname{rank} A_k = k$ and

$$\sigma_{k+1} = \|A - A_k\|_2 = \min\{ \|A - B\|_2 \mid \operatorname{rank} B \le k \}.$$

In other words σ_{k+1} is the minimum distance between A and a matrix of rank less than or equal to k, and this minimum is achieved by A_k.

Proof. That $\operatorname{rank} A_k = k$ is obvious by construction, and likewise $\|A - A_k\|_2 = \sigma_{k+1}$. It remains to show that σ_{k+1} is the minimum of $\|A - B\|_2$ for all matrices B of rank less than or equal to k.

Note that $\dim \ker B = m - \operatorname{rank} B \ge m - k$, and moreover

$$\dim \operatorname{span}\{ v_1, v_2, \ldots, v_{k+1} \} = k + 1$$

where v_i are columns of V. Hence, since the sum of the dimensions exceed m, we know that

$$\operatorname{span}\{ v_1, v_2, \ldots, v_{k+1} \} \cap \ker B \neq \{ 0 \},$$

meaning that there exists some nonzero $\hat{x}$ in this intersection. Since this is in the span of $v_1, v_2, \ldots, v_{k+1}$, we can write it as

$$\hat{x} = c_1 v_1 + c_2 v_2 + \ldots + c_{k+1} v_{k+1}$$

with $c_j \in \mathbb{R}$, and moreover $B\hat{x} = 0$ since it is also in the kernel of B. Hence

$$(A - B)\hat{x} = A\hat{x} = \sum_{i=1}^{k+1} c_i A v_i = \sum_{i=1}^{k+1} c_i \sigma_i u_i,$$

since $Av_i = \sigma_i u_i$. Hence since u_i are orthonormal, by Pythagora's theorem

$$\|(A - B)\hat{x}\|_2^2 = \sum_{i=1}^{k+1} |c_i \sigma_i|^2 \ge \sigma_{k+1}^2 \sum_{i=1}^{k+1} |c_i|^2 = \sigma_{k+1}^2 \|\hat{x}\|_2^2$$

since σ_{k+1} is the smallest of the σ_i at hand. Hence dividing by $\|x\|_2^2$ and taking square roots we have

$$\|A - B\|_2 = \max_{x \neq 0} \frac{\|(A - B)x\|_2}{\|x\|} \ge \frac{\|(A - B)\hat{x}\|_2}{\|\hat{x}\|_2} \ge \sigma_{k+1}.$$ □

Lecture 26 Generalised Inverses

26.1 More Numerical Rank Determination

Corollary 26.1.1. *Let $A \in \mathbb{R}^{n\times m}$ be full rank, i.e.* $\operatorname{rank} A = r = \min\{ n, m \}$. *Let $\sigma_1, \sigma_2, \ldots, \sigma_r$ be the singular values of A. Let $B \in \mathbb{R}^{n\times m}$ such that*

$$\|A - B\|_2 < \sigma_r.$$

Then $\operatorname{rank} B = r$.

Proof. Simply apply the previous theorem for $k = r - 1$. □

Corollary 26.1.2. *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular with singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0$. Suppose*

$$\|A - A_s\|_2 = \min_{C \text{ singular}} \|A - C\|_2,$$

then

$$\|A - A_s\|_2 = \sigma_n$$

and

$$\frac{\|A - A_s\|_2}{\|A\|_2} = \frac{1}{k_2(A)}.$$

Proof. For $\|A - S_2\|_2 = \sigma_n$, use the previous theorem with $k = n - 1$. Then

$$\frac{\|A - A_s\|_2}{\|A\|_2} = \frac{\sigma_n}{\|A\|_2} = \frac{1/\|A^{-1}\|_2}{\|A\|_2} = \frac{1}{\|A\|_2\|A^{-1}\|_2} = \frac{1}{k_2(A)}.$$ □

26.2 Moore-Penrose Inverse

Definition 26.2.1 (Moore-Penrose inverse)**.** Let $A \in \mathbb{R}^{n\times m}$. Then there exists a (unique) matrix $B \in \mathbb{R}^{m\times n}$ such that

(*i*) $ABA = A$,

(*ii*) $BAB = B$,

(*iii*) $(AB)^T = AB$, and

(*iv*) $(BA)^T = BA$.

We call such a matrix B the ***Moore-Penrose inverse*** or ***pseudoinverse*** of A, denoted by $B = A^\dagger$.

Note that if $m = n$ and A is invertible, then A^{-1} is the Moore-Penrose inverse.

Let us prove existence and uniqueness.

Date: March 26th, 2018.

Proof of existence. Let $A = U\Sigma V^T$ be a singular value decomposition of $A \in \mathbb{R}^{n\times m}$ with

$$\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r, 0, \ldots, 0) \in \mathbb{R}^{n\times m}$$

with $r = \operatorname{rank} A$.

Define

$$\Sigma^\dagger = \operatorname{diag}(\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_r^{-1}, 0, \ldots, 0) \in \mathbb{R}^{m\times n},$$

and define $A^\dagger = V\Sigma^\dagger U^T$. This is almost a singular value decomposition, save for the wrong order of the singular values.

We claim that $A^\dagger$ as constructed is indeed a Moore-Penrose inverse of A, which is easy to see. First,

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

Similarly

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

For the symmetric properties, note that $(\Sigma^\dagger\Sigma)^T = \Sigma^\dagger\Sigma$ and likewise $\Sigma\Sigma^\dagger$, whence

$$(AA^\dagger)^T = (U\Sigma V^T V\Sigma^\dagger U^T)^T = U(\Sigma^\dagger\Sigma)^T U^T = U\Sigma V^T V\Sigma^\dagger U^T = AA^\dagger,$$

and analogously for $A^\dagger A$. □

Proof of uniqueness. For uniqueness, suppose we have two matrices B and C such that the four properties hold with respect to A. Then

$$AB = (ACA)B = (AC)^T(AB)^T = C^T A^T B^T A^T = C^T(ABA)^T = C^T A^T = (AC)^T = AC,$$

and by largely identical calculations $BA = CA$. Hence

$$B = BAB = BAC = CAC = C.$$ □

This generalised inverse is quite useful:

Theorem 26.2.2. *Consider the system $Ax = b$ with $A \in \mathbb{R}^{n\times m}$ and $b \in \mathbb{R}^n$. Then $x = A^\dagger b$ is a (unique) least squares solution to the system with minimal norm.*

To prove this we first need a lemma.

Lemma 26.2.3. *Let $A \in \mathbb{R}^{n\times m}$ and $B = A^\dagger$. Then $R(B) = R(A^T)$.*

Proof. We see that

$$B = BAB = (BA)^T B = A^T B^T B,$$

meaning that for every $x \in \mathbb{R}^n$ we have

$$Bx = A^T(B^T Bx) \in R(A^T),$$

so $R(B) \subset R(A^T)$.

Also

$$A^T = (ABA)^T = A^T B^T A^T = (BA)^T A^T = BAA^T,$$

so for every $x \in \mathbb{R}^n$ we have

$$A^T x = B(AA^T x) \in R(B),$$

so $R(A^T) \subset R(B)$, whence $R(A^T) = R(B)$. □

Proof of theorem. We split the proof into two cases. First, suppose $Ax = b$ is consistent, i.e. $b \in R(A)$, so there exists some x_0 such that $Ax_0 = b$. Then

$$AA^\dagger Ax_0 = Ax_0 = b$$

so $AA^\dagger b = b$. Hence $x_0 = A^\dagger b$ is a solution.

Moreover every solution to $Ax = b$ can be written as $z = A^\dagger b + w$ with $w \in \ker A$, so

$$A^\dagger b \in R(A^\dagger) = R(A^T) = \ker(A)^\perp$$

whereby

$$\|z\|_2^2 = \|A^\dagger b + w\|_2^2 = \|A^\dagger b\|_2^2 + \|w\|_2^2 \geq \|A^\dagger b\|_2^2$$

where in the middle we've used Pythagoras theorem since $A^\dagger b$ and w are perpendicular by the above. Therefore $\|A^\dagger b\|_2$ is minimal among all solution.

For the next case, suppose $Ax = b$ is inconsistent, i.e. $b \notin R(A)$. We know that the least squares solution x_0 satisfies the normal equationa $A^T Ax_0 = A^T b$. Indeed $x_0 = A^\dagger b$ satisfies these equations too because

$$\begin{aligned} A^T A(A^\dagger b) &= (V\Sigma^T U^T)(U\Sigma V^T)(V\Sigma^\dagger U^T)b \\ &= V\Sigma^T \Sigma\Sigma^\dagger U^T b = V\Sigma^T U^T b = A^T b. \end{aligned}$$

Similarly to the above considerations, $A^\dagger B$ is in fact of minimal norm. □

Lecture 27 Eigenvalues and Eigenvectors

27.1 Systems of Differential Equations

The following is an application of eigenvalues and eigenvectors, so as to motivate why we want to be able to do these things effectively.

Suppose we want to find two functions $x_1 = x_1(t)$ and $x_2 = x_2(t)$ of time (i.e. $t \geq 0$) such that they satisfy the system

$$\begin{cases} \dot{x}_1 = \frac{d}{dt}x_1(t) = 2x_1(t) + 3x_2(t) = 2x_1 + 3x_2 \\ \dot{x}_2 = \frac{d}{dt}x_2(t) = x_1(t) + 4x_2(t) = x_1 + 4x_2. \end{cases}$$

We set this up as a linear algebra problem,

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix}$$

so that

$$\dot{x} = Ax$$

is the system we wish to solve.

Solving this by hand, as it were, is not necessarily straightforward, especially not if we had a larger system. With eigen considerations it becomes much easier:

The eigenvalues of A are the roots of $\det(tI - A) = (t-1)(t-5)$, so $\lambda_1 = 1$ and $\lambda_2 = 5$. These yield eigenvectors

$$v_1 = \begin{bmatrix} 3 \\ -1 \end{bmatrix} \quad \text{and} \quad v_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Date: March 28th, 2018.

which are linearly independent, whence A is diagonalisable, i.e. if we form the matrix S with v_1 and v_2 as columns, we have

$$S^{-1}AS = D = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$$

where D is the diagonal matrix with the eigenvalues along its diagonal.

Now let us again consider the differential system $\dot{x} = Ax$ and perform the change of variables $y(t) = S^{-1}x(t)$. Then

$$\dot{y} = S^{-1}\dot{x} = S^{-1}Ax = (S^{-1}AS)(S^{-1}x) = DS^{-1}x = Dy$$

so we have a new system of equations $\dot{y} = Dy$. This system is much easier to solve, however, since D is diagonal, so

$$\begin{cases} \dot{y}_1 = y_1 \\ \dot{y}_2 = 5y_2 \end{cases}$$

so the system is ***decoupled***, and the solution is readily found:

$$\begin{cases} y_1(t) = C_1 e^t \\ y_2(t) = C_2 e^{5t} \end{cases}$$

and substitution back for x we have

$$x = Sy = \begin{bmatrix} 3 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} C_1 e^t \\ C_2 e^{5t} \end{bmatrix} = \begin{bmatrix} 3C_1 e^t + C_2 e^{5t} \\ C_2 e^{5t} - C_1 e^t \end{bmatrix}.$$

Hence the eigenvalues give important information about the general solution, both to find it and to analyse asymptotic behaviour.

If A is not diagonalisable, we can accomplish much the same thing by using Jordan canonical forms.

Now in practice one usually solves a system $\dot{x} = Ax$ locally, i.e. for $t \in [0, h]$ for $h > 0$ small to start with, by solving

$$\frac{x(t + \Delta t) - x(t)}{\Delta t} = Ax(t)$$

usually by something like Newton's method.

Note also that theoretically $\dot{x} = Ax$ has a closed form solution in terms of matrix exponentials.

That is to say, for $Y \in \mathbb{R}^{n \times n}$, then we define

$$e^Y = \sum_{k=0}^{\infty} \frac{Y^k}{k!}$$

(which just generalises the power series for e^x). This always exists since

$$\|e^Y\| \leq \sum_{k=0}^{\infty} \frac{\|Y\|^k}{k!} = e^{\|Y\|} < \infty.$$

It is easy to show that the solution to $\dot{x} = Ax$ is

$$x(t) = e^{tA}x(0)$$

for $t \geq 0$ since

$$\dot{x} = \frac{d}{dt}e^{tA}x(0) = Ae^{tA}x(0) = Ax(t).$$

Note for the record that matrix exponentials are in general hard to compute.

27.2 Basic Facts about Eigenvalues and Eigenvectors

Let $A \in \mathbb{C}^{n\times n}$ and $v \in \mathbb{C}^n \setminus \{0\}$. Then if $Av = \lambda v$ for some $\lambda \in \mathbb{C}$, then v is called an ***eigenvector*** of A corresponding to the ***eigenvalue*** λ, and (λ, v) an ***eigenpair***.

The ***spectrum*** of A is

$$\sigma(A) = \{\, \lambda \in \mathbb{C} \mid \text{there exists } v \neq 0 \text{ such that } Av = \lambda v \,\}$$

the set of all eigenvalues of A. Often we will consider it the multiset of eigenvalues, i.e. we include an eigenvalue more than once if it is a repeat root of the characteristic equation.

For instance, we'll write $\sigma(A) = \{1, 1\}$ if $A = I_2$, the 2×2 identity matrix.

We ask ourselves two fundamental questions: first, why do eigenvalues always exist? Secondly, how do we find them?

Theorem 27.2.1. *Let $A \in \mathbb{C}^{n\times n}$. Then $\lambda \in \sigma(A)$ if and only if $\det(\lambda I - A) = 0$.*

Proof. This is straight forward: λ is an eigenvalue if and only if there is a nonzero eigenvector v, i.e. $Av = \lambda v$. This is true if and only if $(A - \lambda I)v = 0$, if and only if $A - \lambda I$ is not invertible (so its columns are linearly dependent), which is true if and only if $\det(A - \lambda I) = \det(\lambda I - A) = 0$.

But

$$\det(\lambda I - A) = \lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1\lambda + a_0$$

is a monic polynomial in λ with $a_i \in \mathbb{C}$, so by the fundamental theorem of algebra $\det(\lambda I - A) = 0$ has n solutions in $\mathbb{C}$. □

Hence finding eigenvalues is a polynomial equation problem.

Lecture 28 The Eigenvalue Problem

28.1 Finding Eigenvalues

We saw that to find eigenvalues of $A \in \mathbb{C}^{n\times n}$ we must solve the characteristic equation $\det(\lambda I - A) = 0$, an nth degree monic polynomial in λ.

It turns out that every monic polynomial is the elementary polynomial of some matrix, and vice versa.

Let

$$p(\lambda) = \lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1\lambda + a_0,$$

with $a_i \in \mathbb{C}$ for all i. Then if we wish to construct a matrix A such that $\det(\lambda I - A) = p(\lambda)$, there are a few options.

The most obvious option is to construct a diagonal matrix with the diagonal values $\lambda_1, \lambda_2, \ldots, \lambda_n$ being the solutions to the characteristic equation. Certainly this satisfies the condition we're after, but it is impractical, because it requires solving the equation, which is a *hard* problem in general.

Date: April 2nd, 2018.

There is a matrix A that has this same property that is easier to construct, called the ***companion matrix***,

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

This matrix also has the property that $\det(\lambda I - A) = p(\lambda)$, although it is not nearly as obvious.

Thus the eigenvalue problem is essentially the root finding problem for polynomials. This is somewhat bad news, since Abel showed that no closed formula exists for roots of polynomials of degree 5 or higher.

Hence there is in general no way to express the eigenvalues directly, in terms of the entries of the matrix. There is no method, and no formula.

As a consequence, the best we can hope for is to approximate the eigenvalues using an iterative method.

It can be proved that to find the whole set of roots of a polynomial, the best way is to form the companion matrix and find eigenvalues of it by iterative methods (this is the work of Jared Aurentz and David Watkins).

Note three things: The ***multiplicity*** of an eigenvalues λ (by which we mean the ***algebraic multiplicity***) coincides with its multiplicity as a root of the characteristic polynomial.

This is related to the ***geometric multiplicity*** of $\lambda \in \sigma(A)$ which is the dimension of the subspace $\ker(\lambda I - A)$.

Example 28.1.1. Consider the matrix

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

The spectrum is $\sigma(A) = \{\, 1, 1 \,\}$, and so the algebraic multiplicity of $\lambda = 1$ is 2, whereas the geometric multiplicity of same is 1, since $\dim \ker(A - I) = 1$. ▲

Note that geometric multiplicity is always less than or equal to algebraic multiplicity.

Secondly, when $A \in \mathbb{R}^{n \times n}$, $\det(\lambda I - A)$ is a real polynomial. As a consequence $\lambda = \alpha + i\beta \in \sigma(A)$ with $\beta \neq 0$ if and only if $\overline{\lambda} = \alpha - i\beta \in \sigma(A)$.

In other words, $\sigma(A)$ is closed under complex conjugation, that is to say non-real eigenvalues come in complex conjugate pairs.

Thirdly, consider

$$A = \begin{bmatrix} A_{11} & * & \cdots & * \\ 0 & A_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ 0 & \cdots & 0 & A_{kk} \end{bmatrix}$$

where A_{jj} are square blocks of size $\ell_j \times \ell_j$ such that $\ell_1 + \ell_2 + \ldots + \ell_k = n$, i.e. A is a block upper triangular matrix.

Then $\sigma(A) = \sigma(A_{11}) \cup \sigma(A_{22}) \cup \ldots \cup \sigma(A_{kk})$.

In particular the diagonal entries of an upper triangular matrix coincide with the eigenvalues (similarly for lower (block) triangular matrices).

To prove this, note that

$$\det(\lambda I - A) = \det(\lambda I - A_{11}) \cdot \det(\lambda I - A_{22}) \cdot \ldots \cdot \det(\lambda I - A_{kk})$$

since determinants are multiplicative.

28.2 The Power Method

Historically, this is the first method to compute (approximate) eigenvalues of a matrix, and perhaps, in many ways, the only method.

The simplest case is this: Assume $A \in \mathbb{C}^{n\times n}$ and $\sigma(A) = \{\, \lambda_1, \lambda_2, \ldots, \lambda_n \,\}$ with

$$|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$$

having corresponding eigenvectors $v_1, v_2, \ldots, v_n$ such that $\{\, v_1, v_2, \ldots, v_n \,\}$ are linearly independent (in other words we're assuming A is ***diagonalisable*** or ***semi-simple***).

Further, we will assume $|\lambda_1| > |\lambda_2|$, i.e. λ_1 is the dominant eigenvalue. (Note that this further assumes $\lambda_1 \neq 0$.)

The idea is to computen λ_1 and v_1 in the following way.

We pick $q \in \mathbb{C}^n$ randomly (sort of—we'll get to this in a bit), and then form the sequence q, Aq, A^2q, and so forth.

Now let us write

$$q = c_1 v_2 + c_2 v_2 + \ldots + c_n v_n$$

with $c_i \in \mathbb{C}^n$. We can do this since $\{\, v_1, v_2, \ldots, v_n \,\}$ is a basis for $\mathbb{C}^n$.

Now suppose $c_1 \neq 0$—this then is what we mean by picking q sort of randomly; we want to ensure it has a component in the direction if v_1.

Then

$$\begin{aligned} A^j q &= c_1 A^j v_1 + c_2 A^j v_2 + \ldots + c_n A^j v_n \\ &= c_1 \lambda_1^j v_1 + c_2 \lambda_2^j v_2 + \ldots + c_n \lambda_n^j v_n \\ &= \lambda_1^j \Big(c_1 v_1 + c_2 \Big(\frac{\lambda_2}{\lambda_1}\Big)^j v_2 + \ldots + c_n \Big(\frac{\lambda_n}{\lambda_1}\Big)^j v_n \Big). \end{aligned}$$

Notice how $|\lambda_k/\lambda_1| < 1$ for $k = 2, 3, \ldots, n$, so as $j \to \infty$, $(\lambda_k/\lambda_1)^j \to 0$.

Lecture 29 The Power Method

29.1 More on the Power Method

Continuing the discussion from last time—

Hence

$$q_j = \frac{A^j q}{\lambda_1^j} = c_1 v_1 + c_2 \Big(\frac{\lambda_2}{\lambda_1}\Big)^j v_2 + \ldots + c_n \Big(\frac{\lambda_n}{\lambda_1}\Big)^j v_n$$

and $q_j \to c_1 v_1$ as $j \to \infty$.

Date: April 4th, 2018.

In fact,

$$\|q_j - c_1 v_1\| = \Big\| \sum_{k=2}^{n} c_k \Big(\frac{\lambda_k}{\lambda_1}\Big)^j v_k \Big\| \leq \sum_{k=2}^{n} |c_k| \Big|\frac{\lambda_k}{\lambda_1}\Big|^j \|v_k\|$$
$$\leq \Big|\frac{\lambda_2}{\lambda_1}\Big|^j \sum_{k=2}^{n} |c_k| \|v_k\| = C \Big|\frac{\lambda_2}{\lambda_1}\Big|^j$$

since λ_2 is the biggest eigenvalue (in modulus), discounding λ_1.

So the convergence of q_j to $c_1 v_1$ depends on the ratio $|\lambda_2/\lambda_1|$, and hence we refer to this quantity as the ratio of convergence of q_j. The smaller this is, the faster q_j converges to $c_1 v_1$.

But q_j the way it is defined here is inaccessible, since it depends on λ_1, which we don't know—that's exactly what we're trying to find.

So in practice we instead let

$$q_{j+1} = \frac{A^j q}{s_{j+1}}$$

for $j = 1, 2, \ldots$, where

$$s_{j+1} = \max_k \{\, |(Aq_j)_k| \,\}.$$

In other words, $\|q_{j+1}\|_\infty = 1$.

This scaling works because $s_{j+1} \geq |\lambda_i|$. This is true because one can prove for any induced matrix norm $\|\cdot\|$,

$$|\lambda_1| = \rho(A) = \max\{\, |\lambda| \mid \lambda \in \sigma(A) \,\} \leq \|A\| = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}.$$

In fact this is true even for matrix norms that aren't induced, but the proof is more challenging.

So from the sequence of vectors q_j we can extract a multiple of v_1, which is an eigenvector of A corresponding to λ_1.

Knowing the eigenvector approximately, we can also compute the approximate eigenvalue, namely since $Aq_j \approx \lambda_1 q_j$ for large j, then

$$\lambda_1 \approx \frac{(Aq_j)_k}{(q_j)_k}$$

for all $k = 1, 2, \ldots, n$.

We make four observations. First, the flop count for $q_{j+1} = Aq_j$ is $2n^2$, plus another $O(n)$ to normalise, so for the power method the complexity is $O(kn^2)$, where k is the number of iterations.

Second, $q_{j+1} = A^{j+1} q$, but we never actually compute large powers of A—this is costly—instead we iteratively compute $q_{j+1} = Aq_j$.

Third, what if we are unlucky and the original q has $c_1 = 0$, so does not have a coordinate in the direction of v_1? Then in theory the power method "converges" in $\text{span}\{\, v_2, v_3, \ldots, v_n \,\}$. In practice there's no problem, for round-off errors in finding q_j will introduce a component in the direction of v_1, which by the domination of λ_1 will come to dominate the other coefficients.

Fourth, the power method still, in some sense, works even when there are multiple dominant eigenvalues, or in cases where the matrix is not diagonalisable. In this case q_j will approach a subspace, rather than a fixed vector. The good news is that this is detectable, because q_j will be skipping around.

29.2 Inverse Power Method

Since the power method only computes a dominant eigenpair, is there anything else we can use the power method for?

The answer is yes. Suppose $A \in \mathbb{C}^{n\times n}$ is as in the power method, but in addition it is also invertible. Then $\sigma(A^{-1}) = \{\, 1/\lambda_n, 1/\lambda_{n-1}, \ldots, 1/\lambda_1 \,\}$, where

$$\left|\frac{1}{\lambda_n}\right| \geq \left|\frac{1}{\lambda_{n-1}}\right| \geq \ldots \geq \left|\frac{1}{\lambda_2}\right| > \left|\frac{1}{\lambda_1}\right|.$$

Now suppose in addition that the first inequality is strict, i.e. A^{-1} has a dominant eigenvalue. Then we apply the power method to A^{-1}, approximating $1/\lambda_n$ and v_n.

Theoretically

$$p_{j+1} = \frac{(A^{-1})p_j}{s_{j+1}}$$

where

$$s_{j+1} = \max_i\{\, |(A^{-1}p_j)_i| \,\},$$

but we can of course avoid computing the inverse.

Lecture 30 Application of the Power Method

30.1 Inverse Power Method continued

To avoid computing A^{-1} in this above scheme we observe that

$$p_{j+1} = \frac{A^{-1}p_j}{s_{j+1}}$$

is equivalent to

$$Ap_{j+1} = \frac{p_j}{s_{j+1}},$$

which we can solve for p_{j+1} using any of our established methods.

The inverse power method can also help us find other eigenvalues, under certain conditions; suppose we have a "good estimate" ρ for some $\lambda \in \sigma(A)$. Then if ρ is closer to λ than any other eigenvalue. Then we can apply the power method to the matrix $A-\rho I$. It has $\lambda-\rho$ as the smallest eigenvalue in modulus.

This works specifically if there is a certain separation between eigenvalues, because we don't want $\lambda - \rho$ being too small, since then $A - \rho I$ is very close to singular, which causes computational issues when using the inverse power method.

30.2 Application of the Power Method

The most famous and high profile use of the power method is Google's PageRank algorithm. The way it works, at least nominally and how it was first described, is this: we construct one very large matrix $A = [a_{ij}]$, where $a_{ij} \neq 0$ if website i links to website j. This gives us an incidence or adjacency matrix of the web,

Date: April 6th, 2018.

in particular a very, very spare matrix, since most websites don't link to the billions upon billions of other websites.

This matrix is then slightly altered by two fudge factors. The first one is a rank 1 matrix of small norm, which serves to mollify and smudge the 0s in most entries, so as to make websites slightly less isolated in this matrix. The second one is a mystery, and has to do with advertisement, giving preference to paying customers.

This matrix is then scaled to make it row-stochastic, meaning rows are nonnegative and add up to 1.

This new matrix is called the Google matrix G.

Now because this is a row-stochastic matrix, the 1-vector is an eigenvector corresponding to the eigenvalue $1 \in \sigma(G)$.

This matrix can be thought of as a stochastic process, a Markov chain, and hence it has a stationary distribution, which is a long vector where each entry represents the probability that a random user is at site j in the long run, and hence the entries of this vector guage the relevance of the sites.

This is how Google ranks pages; users search, they pick a subset of the sites from their very long list, and order them based on the size of their entries in the stationary distribution.

The question then arises of how to compute this stationary distribution. Fortunately it is known that the *left* eigenvector corresponding to 1 is the stationary distribution of any row-stochastic matrix, whence one solves the system $y^T G = y^T$.

(That y^T is nonnegative is a deeper result from the Perron-Frobenius theorem.)

So we need to find y, and fortunately the eigenvalue 1 it corresponds to is the dominant one, so we set up $G^T y = y$ and sovle this approximately using the power method.

Lecture 31 Similarity Transformations

31.1 Basic Results

Definition 31.1.1 (Similarity)**.** Two matrices $A, B \in \mathbb{C}^{n\times n}$ are said to be ***similar*** if there exists some invertible $S \in \mathbb{C}^{n\times n}$ such that

$$B = S^{-1}AS,$$

called a ***similarity transformation***.

It is easy to see that this is an equivalence relation, since clearly every matrix is similar to itself (take $S = I$), if A is similar to B then B is similar to A (just multiply by S and S^{-1} from the respective side), and if A is similar to B and B is similar to C, then combining the respective matrices S and, say, S' will make A similar to C.

An important property of similar matrices is that they share eigenvalues. Suppose $Bx = \lambda x$ with $x \neq 0$. Then if $B = S^{-1}AS$, we have

$$S^{-1}ASx = \lambda x,$$

Date: April 9th, 2018.

so multiplying by S we get $A(Sx) = \lambda(Sx)$, so $Sx \neq 0$ (since S is invertible) is an eigenvector corresponding to λ for A.

Hence $\lambda \in \sigma(B)$ implies $\lambda \in \sigma(A)$, and vice versa by repeating the same argument with the roles of A and B switched.

Also, $x \in \ker(\lambda I - B)$ if and only if $Sx \in \ker(\lambda I - A)$.

If S is unitary (or orthogonal over the reals), i.e. $S^{-1} = S^*$, then we call A and $B = S^{-1}AS = S^*AS$ ***unitarily similar***. It is easy to see that unitary similarities preserve spectral norms $\|\cdot\|_2$ and the corresponding condition number $k_2(\cdot)$.

They also preserve the property of being Hermitian, i.e. $A = A^*$, since

$$(U^*AU)^* = U^*A^*(U^*)^* = U^*AU.$$

Theorem 31.1.2 (Schur's theorem)**.** *Let $A \in \mathbb{C}^{n\times n}$. Then there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ and upper triangular matrix $T \in \mathbb{C}^{n\times n}$ such that*

$$T = U^*AU.$$

(Hence if we can find U, we'll automatically get the eigenvalues of A, since they live on the diagonal of T.)

Proof. We prove it by induction on n. For $n = 1$ the result is vacuously true.

Assume that it is true for $n = k - 1$, and show that as a consequence it is true for $n = k$.

Let $Av = \lambda v$ with $\|v\|_2 = 1$ (so in particular $v \neq 0$). Construct a unitary matrix U_1 whose first column is v, i.e. $U_1e_1 = v$ (this is possible by completing v into a basis for $\mathbb{C}^n$ and applying Gram-Schmidt and then normalising). Let $A_1 = U_1^*AU_1$ with $U_1 = \begin{bmatrix} v & W \end{bmatrix}$ where $W \in \mathbb{C}^{n\times(n-1)}$, so

$$A_1 = \begin{bmatrix} v^* \\ W^* \end{bmatrix} A \begin{bmatrix} v & W \end{bmatrix} = \begin{bmatrix} v^*Av & v^*AW \\ W^*Av & W^*AW \end{bmatrix} = \begin{bmatrix} \lambda & * \\ 0 & \hat{A} \end{bmatrix}$$

since the columns of W are orthogonal to v by construction, and the matrix $\hat{A} \in \mathbb{C}^{(n-1)\times(n-1)}$.

Now given the size of $\hat{A}$, by the inductive hypothesis there exists some $\hat{U}_2 \in \mathbb{C}^{(n-1)\times(n-1)}$ such that $\hat{U}_2^*\hat{A}\hat{U}_2 = \hat{T}$ where $\hat{T}$ is upper triangular. Thus take

$$U_2 = \begin{bmatrix} 1 & 0 \\ 0 & \hat{U}_2 \end{bmatrix} \in \mathbb{C}^{n\times n}$$

being unitary. Then

$$\hat{U}_2A_1U_2 = \begin{bmatrix} \lambda & * \\ 0 & \hat{U}_2^*\hat{A}\hat{U}_2 \end{bmatrix} = \begin{bmatrix} \lambda & * \\ 0 & \hat{T} \end{bmatrix} = T$$

is upper triangular, so

$$U_2^*U_1^*AU_1U_2 = T,$$

and $U = U_1U_2$ is unitary. □

Note a few things. First, the decomposition in Schur's theorem is not unique except for the eigenvalues of A appearing as the diagonal elemengs in T (in any order we like).

This is a kind of pseudo-constructive proof; it's offers a construction, but that construction requires knowledge of eigenpairs.

Finally, this process of deflation in the proof, where we reduce A to A_1 of smaller size, is a useful thing to keep in mind.

31.2 Special Matrices

Theorem 31.2.1. *If $A \in \mathbb{C}^{n\times n}$ is Hermitian (i.e. $A = A^*$), then there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ such that $U^*AU = D$ where D is diagonal.*

That is to say, Hermitian matrices are unitarily diagonalisable.

Proof. Apply Schur's theorem to $A = A^*$, so there exists a unitary matrix U such that $U^*AU = T$, but $T^* = T$ since $A = A^*$, and hence $T = D$ is diagonal. □

In fact, this characterises normal matrices.

Theorem 31.2.2. *A matrix $A \in \mathbb{C}^{n\times n}$ is normal if and only if there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ and a diagonal matrix $D \in \mathbb{C}^{n\times n}$ such that $U^*AU = D$.*

A few facts to keep in mind: If A is Hermitian, then $\sigma(A) \subset \mathbb{R}$. If A is normal and $\sigma(A) \subset \mathbb{R}$, then A is Hermitian. If U^*AU is diagonal, then $A = UDU^*$, and so

$$A = d_1u_1u_1^* + d_2u_2u_2^* + \ldots + d_nu_nu_n^*.$$

Lecture 32 Normal Matrices

32.1 Some Facts about Normal Matrices

If $A \in \mathbb{C}^{n\times n}$ is normal, i.e. $AA^* = A^*A$, so A commutes with its conjugate transpose, then

- $Au = \lambda u$ with $u \neq 0$ if and only if $U^*A = \lambda u^*$, $u \neq 0$¡ so left and right eigenvectors coincide.
- As a consequence of the fact that left or right eigenvalues corresponding to distinct eigenvalues are orthoonal, one can prove that every normal matrix has n linearly independent orthonormal eigenvectors. That is, every normal matrix is diagonalisable via a unitary similarity.

  So A is normal if and only if A is unitarily diagonalisable.
- Hence to construct all normal matrices, take a diagonal matrix $D = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and an arbitrary unitary matrix U and compute U^*DU.
- Hermitian matrices are precisely those normal matrice for which $\sigma(A) \subset \mathbb{R}$.

32.2 Something about the Power Method

One last thing related to the power method:

Date: April 11th, 2018.

Definition 32.2.1 (Convergent matrix)**.** A matrix $A \in \mathbb{C}^{n\times n}$ is called ***convergent*** if

$$\lim_{k\to\infty} A^k = 0.$$

If

$$\lim_{k\to\infty} A^K = B \in \mathbb{C}^{n\times n},$$

then A is called ***semiconvergent***.

Theorem 32.2.2. *A matrix $A \in \mathbb{C}^{n\times n}$ is convergent if and only if its spectral radius*

$$\rho(A) = \max\{\, |\lambda| \mid \lambda \in \sigma(A) \,\} < 1.$$

Theorem 32.2.3. *A matrix A is semiconvergent if and only if $\rho(A) \leq 1$ and if $|\lambda| = 1$ for some $\lambda \in \sigma(A)$, then $\lambda = 1$ and $\operatorname{index}_A(1) = 1$ (i.e. the largest Jordan block of $\lambda = 1$ is 1×1, or equivalently the algebraic and geometric multiplicities of $\lambda = 1$ are the same).*

One can prove

Theorem 32.2.4. *Let $A \in \mathbb{C}^{n\times n}$ such that for some $\lambda \in \sigma(A)$, $|\lambda| = \rho(A) > |\mu|$ for all $\mu \in \sigma(A) \setminus \{\, \lambda \,\}$.*

Also assume λ is simple, i.e. its algebraic and geometric multiplicities are both 1. Then

$$\lim_{k\to\infty} \left(\frac{A}{\rho(A)}\right)^k = xy^*$$

where $Ax = \lambda x$ and $y^*A = \lambda y^*$, with $y^*x = 1$.

This is sometimes known as the Fundamental theorem of demography if A has nonnegative entries.

32.3 QR Algorithm or Francis Algorithm

The story of how the QR algorithm came to be is perhaps apocryphal, but nonetheless interesting.

The story is of a project of Heinz Rutishauser, who between 1954 and 1958 ran a research project with a group of graduate students. The idea was to factor $A = LR$ (with L lower and R upper triangular, i.e. LU factorise), then multiply them back together again to check how close A is to LR.

They had some new, fancy computers, so they decided to do it over and over and over again, so start with A, compute LR, factor again into L_1R_1, compute L_2R_2, and so forth.

This computer was fed programs on punchcards, and as the story goes the students made a mistake and accidentally switched two cards in such a way that the program factored A into L and R, but instead of computing LR and comparing to A, they accidentally computed the switched product RL, then factored this into its LR factorisation, and again switched the order, and so forth.

Now to the students surprise the result started approaching an upper triangular matrix, and since LR and RL are similar (since $R(LR)R^{-1} = RL$, supposing R^{-1} exists), they could find the eigenvalues of A on the diagonal.

This gave birth to the LR-algorithm, an iterative method to compute the full set of eigenvalues of a matrix.

It was subsequently improved by Francis and Kubanskaya independently in the 1970's, christened the QR algorithm.

Lecture 33 QR Algorithm

33.1 Improving the LR Algorithm

There are two shortcomings ot the LR algorithm. First, convergence is rather slow. Secondly, the LR to RL action destroys zeros (and iterates are dense, as it happens).

Francis considered improving the performance relative to both issues.

To improve convergence, the idea is that given a good estimate of an eigenvalue, we can use shifting to improve performance. More precisely, suppose ρ is an estimate of an eigenvalue of A. Then we alter the algorithm to compute $A_{j-1} - \rho I = Q_j R_j$, and then add the eigenvalue back with $A_j = R_j Q_j + \rho I$.

Note that subtracting a multiple of the identity matrix shift the entire spectrum of A correspondingly, and so adding that multiple back ensures we keep the same spectrum.

To preserve spartity, we can precondition A so that it is sparse (lots of zeros) and then adjust the QR algorithm to preserve sparsity. This keeps the flop count low.

33.2 Reduction to Hessenberg and Triangular Forms

The idea is to take $A \in \mathbb{C}^{n\times n}$ and perform a similarity transformation (preferably unitary) that makes A upper triangular. That is to say, a Schur's theorem type problem. But this is extremely hard, since we would need knowledge of eigenvalues!

So we settle for the next best thing, some sort of almost upper triangular matrix.

Definition 33.2.1 (Upper Hessenberg form)**.** A matrix $A \in \mathbb{C}^{n\times n}$ is said to be in ***upper Hessenberg form*** if

$$A = \begin{bmatrix} * & * & \cdots & * \\ * & * & \cdots & * \\ & \ddots & \ddots & \vdots \\ & & * & * \end{bmatrix},$$

i.e. there are subdiagonal nonzero elements.

We can transform any $A \in \mathbb{C}^{n\times n}$ into an upper Hessenberg form via a unitary similarity, essentially by the same argument as the proof of Schur's theorem.

Let

$$A = \begin{bmatrix} a_{11} & c^T \\ b & \hat{A} \end{bmatrix}$$

Date: April 13th, 2018.

with c and b being $(n-1)\times 1$ vectors, and $\hat{A}$ being $(n-1)\times(n-1)$. Now take

$$Q_1 = \begin{bmatrix} 1 & 0^T \\ 0 & \hat{Q}_1 \end{bmatrix}$$

where $\hat{Q}_1 b = \begin{bmatrix} -\tau_1 & 0 & \cdots & 0 \end{bmatrix}^T \in \mathbb{C}^{n-1}$, with $|\tau_1| = \|b\|_2$, which we know we can do; take $\hat{Q}_1$ to be, say, a Householder rotator. Then

$$Q_1 A = \begin{bmatrix} a_{11} & c^T \\ \hat{Q}_1 b & \hat{Q}_1 \hat{A} \end{bmatrix}$$

and since $Q_1 A Q_1^{-1} = Q_1 A Q_1$, this is equal to

$$\begin{bmatrix} a_{11} & c^T \hat{Q}_1 \\ \hat{Q}_1 b & \hat{Q}_1 \hat{A} \hat{Q}_1 \end{bmatrix}.$$

Similarly there exists $\hat{Q}_2 \in \mathbb{C}^{(n-2)\times(n-2)}$ with

$$Q_2 = \begin{bmatrix} 1 & 0 & \\ 0 & 1 & \\ & & \hat{Q}_2 \end{bmatrix}$$

so

$$Q_2(Q_2 A Q_1)Q_2 = \begin{bmatrix} a_{11} & * & \cdots \\ -\tau_1 & * & \cdots \\ & -\tau_2 & \hat{Q}_2 \hat{A}_2 \hat{Q}_2 \\ & 0 & \end{bmatrix}$$

and continuing this QAQ is upper Hessenberg, with $Q = Q_1 Q_2 \cdots Q_{n-2}$. This is $O(n^3)$ (actually $10/3n^3$), which is fairly expensive, but greatly improves the rest of the QR algorithm to make up for it.

Next we'll see how to contruct the QR algorithm so that when applied to upper Hessenberg matrices, it retains the zero structure.

Note for the record that when we are dealing with upper Hessenberg matrices we typically assume the subdiagonal elements are nonzero, since if any subdiagonal is 0 we can split the problem into studying two smaller problems, e.g.

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{bmatrix}$$

in which the two 2×2 matrices on the diagonal can be studied separately.

These upper Hessenberg matrices with nonzero subdiagonal are usually called reduced or proper upper Hessenberg.

Secondly, note that the geometric multiplicity of an eigenvalue λ of reduced upper Hessenberg is always 1. To see this, note that the matrix $A - \lambda I$ is also upper Hessenberg, and if we consider the submatrix acquired by dropping the first column and last row is diagonal, and since the original subdiagonals are nonzero, this new diagonal matrix has nonzero diagonal entries. Hence its rank is $n-1$, whence by the rank-nullity theorem $\ker(A - \lambda I)$ is of dimension 1.

Lecture 34 Francis's Algorithm

34.1 Describing One Step

On the outset, suppose $A \in \mathbb{C}^{n\times n}$ is proper upper Hessenberg—if improper, we would split into several proper upper Hessenberg matrices).

The algorithm then goes as follows: Call $A = A_k$ and $\hat{A} = A_{k+1}$.

First, pick a shift ρ, with the idea being to have it close to some eigenvalue. Let

$$p = \begin{bmatrix} a_{11} - \rho \\ a_{21} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

be the first column of $A - \rho I$.

Next pick a rotator or reflector Q_0 such that

$$Q_0^* p = \begin{bmatrix} * \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Then the transformation from A to $Q_0^* A$ leaves upper Hessenberg form intact because it recombined the first and second rows of A, and moreover $Q_0^* A$ to $Q_0^* A Q_0$ recombines columns one and two, so it creates a "bulge" in the upper Hessenberg form.

In other words,

$$A = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix} \longrightarrow Q_0^* A = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix} \longrightarrow Q_0^* A Q_0 = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ \bullet & * & * & * \\ 0 & 0 & * & * \end{bmatrix}.$$

We return to upper Hessenberg form by a new rotator in the 2-3-plane, say Q_1^*, so

$$Q_1^* Q_0^* A Q_0 = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix},$$

and then $Q_1^* Q_0^* A Q_0 Q_1$ recombines columns two and three, and moves the bulge to

$$Q_1^* Q_0^* A Q_0 Q_1 = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & \bullet & * & * \end{bmatrix}.$$

Doing this one more time we'll have chased the bulge away, since recombining the third and fourth column wouldn't cause a new bulge, so finally we have

Date: April 16th, 2018.

some third rotator Q_2 so that

$$Q_2^* Q_1^* Q_0^* A Q_0 Q_1 Q_2 = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix}.$$

It can be shown that after "many" iterations, the $(n, n-1)$ entry converges to 0, so A_k as $k \to \infty$ reveals an eigenvalue of A.

We then deflate, and continue from there with the $(n-1) \times (n-1)$ leading principal submatrix.

The rate of convergence depends on how close ρ is to the eigenvalue being approximated. We'll justify this in the future.

34.2 The Symmetric Case

When we were dealing with the singular value decomposition we needed eigenvalues for the symmetric matrix $A^T A$, so it is interesting to see how Francis's algorithm works with symmetric matrices.

The first and fundamental observation to make is that upper Hessenberg and symmetric implies that the matrix is tridiagonal.

In practice here, we choose the shifts via the eigenvalues of the trailing 2×2 matrix.

Now it is a fact that Francis's algorithm is for small or relatively small matrices, say maybe $n = 1000$ or $n = 10^5$. For bigger things, there's generally nothing to do, unless there are special properties of the matrix to exploit; maybe it's extremely sparse.

Finally, note that in the symmetric case A going to $Q^* A Q = D$ is diagonal, so the columns of Q automatically give us the eigenvectors corresponding to the eigenvalues on the diagonal of D.

For the general case, $Q^* A Q = T$ is triangular, so $\lambda \in \sigma(A)$ if and only if $\lambda \in \sigma(T)$¡ and solving $(T - \lambda I)u = 0$ gives eigenvectors $x = Qu$.

Lecture 35 Invariant Subspaces

35.1 Basic Eigenfacts

Let $A \in \mathbb{C}^{n \times n}$ and $\lambda \in \mathbb{C}$. If $\lambda \in \sigma(A)$, then

$$S_\lambda \coloneqq \{\, v \in \mathbb{C}^n \mid Av = \lambda v \,\}$$

is a nontrivial subspace of $\mathbb{C}^n$, called the ***eigenspace*** of A corresponding to λ. In fact, and it's not hard to prove, $\lambda \in \sigma(A)$ if and only if $S_\lambda \neq \{\, 0 \,\}$.

Observe that for all $v \in S_\lambda$, $Av \in S_\lambda$ as well.

Definition 35.1.1 (Invariant subspace)**.** A subspace S is $\mathbb{C}^n$ is called ***A-invariant*** or ***invariant under*** $A \in \mathbb{C}^{n \times n}$ if for all $v \in S$ we have $Av \in S$ (i.e. $AS \subset S$).

Date: April 18th, 2018.

We call $\{ 0 \}$ and $\mathbb{C}^n$ trivial invariant subspaces. As we discussed above, S_λ is A-invariant.

A fundamental question to ask is if all A-invariant subspaces are collections of eigenvectors of A.

The answer to this question is no: Take for instance

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix},$$

for which e_1 is clearly an eigenvector corresponding to $\lambda = 1$, but we also have $Ae_2 = e_1 + e_2$, so $S = \text{span}\{ e_1, e_2 \}$ is A-invariant without containing exclusivelt eigenvectors.

One big reason why we care about invariant subspaces is that they help us understand reductions of matrices to block upper triangular form.

Theorem 35.1.2. *Let S be a subspace of $\mathbb{C}^n$ and let $\{ x_1, x_2, \ldots, x_k \}$ be a basis for S. Define*

$$\hat{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_k \end{bmatrix} \in \mathbb{C}^{n \times k}.$$

Then S is A-invariant if and only if there exists some $\hat{B} \in \mathbb{C}^{k \times k}$ such that $A\hat{X} = \hat{X}\hat{B}$.

Proof. For the reverse direction, suppose there exists $\hat{B} \in \mathbb{C}^{k \times k}$ with $A\hat{X} = \hat{X}\hat{B}$ and call $\hat{B} = [b_{ij}]$. Then

$$Ax_j = \sum_{i=1}^{k} x_i b_{ij} \in \text{span}\{ x_1, x_2, \ldots, x_k \} = S$$

for all $j = 1, 2, \ldots, k$. So $AS \subset S$.

For the forward direction, if $AS \subset S$, then $Ax_j \in S$ so there exists some b_{ij} such that

$$Ax_j = x_1 b_{1j} + x_2 b_{2j} + \ldots + x_k b_{kj}$$

for all $j = 1, 2, \ldots, k$, so letting $\hat{B} = [b_{ij}] \in \mathbb{C}^{k \times k}$ we have $A\hat{X} = \hat{X}\hat{B}$. □

Theorem 35.1.3. *Let S be a subspace of $\mathbb{C}^n$, let $A \in \mathbb{C}^{n \times n}$, and $AS \subset S$, so S is A-invariant. Let $\{ x_1, x_2, \ldots, x_n \}$ be a basis for S. Moreover let*

$$\{ x_1, x_2, \ldots, x_k, s_{k+1}, \ldots, x_n \}$$

be a basis for $\mathbb{C}^n$.

Define

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_k & x_{k+1} & \cdots & x_n \end{bmatrix} \in \mathbb{C}^{n \times n}$$

and let $B = X^{-1}AX$. Then B is block upper triangular of the form

$$B = \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}$$

where $B_{11} \in \mathbb{C}^{k \times k}$ and $AX_1 = X_1 B_1 1$, taking

$$X_1 = \begin{bmatrix} x_1 & x_2 & \cdots & x_k \end{bmatrix} \quad \textit{and} \quad X_2 = \begin{bmatrix} x_{k+1} & x_{k+2} & \cdots & x_n \end{bmatrix}.$$

Proof. Note $B = X^{-1}AX$ if and only if $AX = XB$, so if $B = X^{-1}AX$ then Ax_j is the jth column of XB, i.e.

$$Ax_j = \sum_{x=1}^{n} x_i b_{ij}$$

uniquely, since x_i form a basis for $\mathbb{C}^n$. On the other hand, $x_j \in S$ for $j \leq k$, so $Ax_j \in S$ too, hence

$$Ax_j = c_{1j}x_1 + c_{2j}x_2 + \ldots + c_{kj}x_k$$

for some $c_{1j}, c_{2j}, \ldots, c_{kj}$. This forces $b_{ij} = c_{ij}$ for $i = 1, 2, \ldots, k$, and $b_{ij} = 0$ for $i = k+1, k+2, \ldots, n$, for all $j = 1, 2, \ldots, k$.

So

$$B = \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}$$

where the zero block is for $i = k+1, k+2, \ldots, n$ and $j = 1, 2, \ldots, k$ from above.

That $AX_1 = X_1B_{11}$ follows immediately. □

Corollary 35.1.4. *Let* $B = X^{-1}AX$,

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \in \mathbb{C}^{n\times n}.$$

Then B *is upper triangular if and only if* $\operatorname{span}\{ x_1, x_2, \ldots, x_k \mid k = 1, 2, \ldots, n \}$ *are* A*-invariant subspaces.*

Proof. The forward direction is the above theorem iteratively.

The reverse direction, note that the theorem is true in reverse too. □

Lecture 36 Krylov Subspaces and Francis's Algorithm

36.1 Krylov Subspaces

Let $A \in \mathbb{C}^{n\times n}$ and $x \in \mathbb{C}^n \setminus \{ 0 \}$. Generate a nested sequence of the so-called ***Krylov subspaces*** by

$$\begin{aligned} K_1(A, x) &= \operatorname{span}\{ x \}, \\ K_2(A, x) &= \operatorname{span}\{ x, Ax \}, \\ K_3(A, x) &= \operatorname{span}\{ x, Ax, A^2x \}, \\ &\vdots \\ K_j(A, x) &= \operatorname{span}(x, Ax, A^2x, \ldots, A^{j-1}x). \end{aligned}$$

We make a few basic observations. Clearly these are nested; $K_j(A, x) \subset K_{j+1}(A, x)$ for $j = 1, 2, \ldots$.

Since $K_j(A, x)$ is the span of j vectors, $\dim K_j(A, x) \leq j \leq n$.

Date: April 20th, 2018.

This nested growth must stop at some $j \leq n$, say

$$K_1(A,x) \subsetneq K_2(A,x) \subsetneq \ldots \subsetneq K_j(A,x) = K_{j+1}(A,x) = \ldots.$$

Notice how $K_j(A,x)$, the last one to change, is an A-invariant subspace; $AK_j(A,x) \subset K_j(A,x)$.

We can be even more precise:

Theorem 36.1.1. *Let $A, H, Q \in \mathbb{C}^{n\times n}$ with*

$$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix},$$

$q_j \in \mathbb{C}^n$, nonsingular. If $H = Q^{-1}AQ$ and if H is proper upper Hessenberg, then

$$\text{span}\{\, q_1, q_2, \ldots, q_n \,\} = K_j(A, q_1).$$

Proof. We prove this by induction on j. For $j = 1$ it is true by definition, since $K_1(A, q_1) = \text{span}\{\, q_1 \,\}$.

We will show that for any $j < n$, if

$$\text{span}\{\, q_1, q_2, \ldots, q_j \,\} = K_j(A, q_1),$$

then

$$\text{span}\{\, q_1, q_2, \ldots, q_j, q_{j+1} \,\} = K_{j+1}(A, q_1).$$

Write $H = Q^{-1}AQ$ as $AQ = QH$. If we equate the jth columns,

$$Aq_j = Qh_j = \sum_{i=1}^{j+1} q_i h_{ij} = \sum_{i=1}^{j} q_i h_{ij} + q_{j+1}h_{j+1,j}$$

where we've stopped the middle sum at $i = j+1$ since H is assumed upper Hessenberg, i.e. $h_{ij} = 0$ if $i > j+1$. For the same reason $h_{j+1,j} \neq 0$ in the last step, since H is *proper* upper Hessenberg. Hence

$$q_{j+1} = \frac{1}{h_{j+1,j}}\Big(Aq_j - \sum_{i=1}^{j} q_j h_{ij}\Big),$$

and by the inductive hypothesis $Aq_j \in \text{span}\{\, Aq_1, A^2q_1, \ldots, A^jq_1 \,\}$. Moreover $q_i \in K_i(A, q_1) \subset K_{j+1}(A, q_1)$ for $i = 1, 2, \ldots, j$.

Combining these we then have

$$\text{span}\{\, q_1, q_2, \ldots, q_{j+1} \,\} \subset K_{j+1}(A, q_1),$$

and since the dimension of the left-hand side is $j+1$ and the dimension of the right-hand side is at most $j+1$, their dimensions must be equal, so they are equal as subsapces. □

Corollary 36.1.2. *If A is proper upper Hessenberg, then*

$$K_j(A, e_1) = \text{span}\{\, e_1, e_2, \ldots, e_j \,\}$$

for $j = 1, 2, \ldots, n$.

Proof. Apply the previous theorem with $H = A$, so $A = I^{-1}AI$, i.e. $Q = I$. □

Lecture 37 A Sketch of Why Francis's Algorithm Works

37.1 The Idea

Consider a single iteration of Francis's algorithm with (fixed) shift ρ. We start with a matrix A and turn it into a matrix $\hat{A} = Q^*AQ$ with

$$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}$$

where q_1 is proportional to the first column of $A - \rho I$, i.e. $Q_0 e_1 = e_1$, $Q_1 e_1 = e_1$, and so forth.

Also note that $\hat{A}$ is still proper upper Hessenberg, given that A was. So

$$\begin{aligned}\operatorname{span}\{ a_1, a_2, \ldots, a_k \} &= K_k(A, e_1) = K_k(A, (A - \rho I)e_1) = (A - \rho I)K_k(A, e_1) \\ &= (A - \rho I)\operatorname{span}\{ e_1, Ae_1, \ldots, A^{k-1}e_q \} = (A - \rho I)\operatorname{span}\{ e_1, e_2, \ldots, e_k \}\end{aligned}$$

by the proposition above, since A and $A - \rho I$ commute.

Hence by looking at the penultimate step, we see that this is in essence the power method with e_1, for a fixed shift, to the the subspace $\operatorname{span}\{ e_1, e_2, \ldots, e_k \}$.

Now if we number the eigenvalues of A such that

$$|\lambda_1 - \rho| \geq |\lambda_2 - \rho| \geq \ldots \geq |\lambda_n - \rho|.$$

If $|\lambda_k - \rho| > |\lambda_{k+1} - \rho|$ for some k, the subdiagonal entry $a^{(j)}_{k+1,k}$ of the jth iterate $A^{(j)}$ of this process tends to 0 as $j \to \infty$.

Lecture 38 Iterative Methods for Solving Linear Systems

38.1 General Idea

We close this course off with a brief overview of iterative methods to solve linear systems of equations.

The goal, then, is to solve a system $Ax = b$ for x, where $A \in \mathbb{C}^{n \times n}$ is nonsingular. The fundamental idea is to write $A = M - N$, called a ***splitting*** of A, with M and N to be determined.

Then under the assumption that M is invertible, $Ax = b$ is equivalent to $(M - N)x = b$, which in turn is equivalent to $Mx = Nx + b$, so

$$x = M^{-1}Nx + M^{-1}b.$$

We can thing of the above as an operator on the vector x, and then what we're looking for is a fixed point for the operator.

So define the iterative process

$$x_{k+1} = M^{-1}Nx_k + M^{-1}b$$

Date: April 23rd, 2018.
Date: April 25th, 2018.

for $k = 0, 1, 2, \ldots$ with x_0 some initial vector.

If it converges, it should approach the fixed point x above, i.e. the solution to $Ax = b$.

We now want to understand when this happens.

Theorem 38.1.1. *Let $A \in \mathbb{C}^{n\times n}$ and write $A = M - N$ where A and M are nonsingular. Let $H = M^{-1}N$, $c = M^{-1}b$. Then the iterative process*

$$x_{k+1} = Hx_k + c$$

converges to the slution of $Ax = b$ if and only if $\rho(H) < 1$.

Proof. Consider the two equaitons $x_{k+1} = Hx_k + c$ and $x = Hx + c$, and subtract the second from the first:

$$x_{k+1} - x = H(x_k - x) + 0 = \ldots = H^{k+1}(x_0 - x)$$

so $x_{k+1} \to x$ if and only if $H^{k+1} \to 0$ as $k \to \infty$, but we know $H^k \to 0$ as $k \to \infty$ if and only if $\rho(H) < 1$. □

A splitting $A = M - N$ such that M^{-1} exists and $\rho(M^{-1}N) < 1$ is called ***regular***.

Note that of course if we can, we would also like to choose M and N such that $\rho(M^{-1}N) < 1$ is as small as possible, since the smaller it is, the faster the convergence.

The original approach to this problem was to try to pick M such that it is easily invertible and moreover quite sparse, so as to save on computational cost.

Suppose, for instance, we write $A = D + L + U$, where D is the diagonal of A, L is the strictly lower triangular part of A, and U is the strictly upper triangular part of A.

Using $M = D$ (assuming $a_{ii} \neq 0$), so $N = L+U$, we get the so-called ***Jacobi method***.

One could also take $M = D + L$ and $N = U$, again assuming $a_{ii} \neq 0$, and then we get the ***Gauss-Seidel method***.

38.2 Sensitive Overrelaxation Method

The particular weighted splitting

$$H_\omega = (D - \omega L)^{-1}((1 - \omega)D + \omega U),$$

whence

$$x_{k+1} = H_\omega x_k + \omega(D - \omega L)^{-1}b$$

is called the ***sensitive overrelaxation method***.

If $\omega = 1$, this of course gives us Gauss-Seidel.

Theorem 38.2.1. *The sensitive overrelaxation method converges only if $0 < \omega < 2$.*

Proof. Suppose the method converges, meaning that $\rho(H_\omega) < 1$. It suffices to show

$$|\omega - 1| \leq \rho(H_\omega).$$

Notice $\det D^{-1} = \det(D - \omega L)^{-1}$ since L is strictly lower triangular. Hence

$$\begin{aligned}\det H_\omega &= \det(D - \omega L)^{-1} \det((1-\omega)D + \omega U) = \det D^{-1} \det((1-\omega)D + \omega U) \\ &= \det((1-\omega)I + \omega D^{-1}U) = (1-\omega)^n\end{aligned}$$

since $\omega D^{-1}U$ is strictly upper triangular since U is.

Now since $\det H_\omega$ is also the product of the eigenvalues of H_ω, we have that the product of all of its eigenvalues is $(1-\omega)^n$. Hence

$$\rho(H_\omega)^n \geq (1-\omega)^n,$$

so $|1-\omega| \leq \rho(H_\omega) < 1$, i.e. $0 < \omega < 2$. □

So in order for the sensitive overrelaxation method with the splitting

$$A = (D - \omega L) + ((1-\omega)D + \omega U)$$

to converge, we need $0 < \omega < 2$. So what's the role of ω?

The rate of convergence depends on it! For some special type of matrices, one can then choose ω in a special way so as to speed up the convergence.

Index