Course Notes for ECSE 343: Winter 2024

Mohamed-Amine Azzouz

2024-05-02

Contents

Lecture 1: Number Representation

Note: the course is over. Therefore, I am not planning to continue these incomplete notes, since that would require lots of precious time that I judge better spent elsewhere. Nonetheless, the sections on floating-point arithmetic, measures of error, and basic direct methods of solving linear systems (Gaussian Elimination, LU Decomposition, Cholesky) are available. I hope you will appreciate that.

Introduction

These notes are meant to give a more directed form of following the course material for ECSE 343, complementing the course slides. The most important reference used is the textbook [**?**], which is what I am reading over to learn the material. Within a lecture, I present the material as the textbook does.
I will mention all the sections I have read, but will only present a topic if it is mentioned in the slides. The sections that are not mentioned in ECSE 343 will be marked with an asterisk (*). If you want to go over those details I left out, I invite you to read [**?**].

Preliminary to 2.1 Well-posedness and Condition Number of a Problem

- A numerical problem is well-posed if a small change in the parameters given only results in a small change in the result. Otherwise, it is ill-posed.
- This idea is synonymous to numerical stability, which can be quantified with the relative and absolute condition numbers $K(d)$, where d is the data used by the numerical method.

Lecture 6 will expand on those notions.

2.2 Stability of Numerical Methods

Assume the numerical problem we are to solve is well-posed. Oftentimes, we want to simplify the problem posed into a sequence of simpler, but approximate problems.

- A method, which is a sequence of approximate problems, is consistent if that sequence converges to the original numerical problem.
- A method is strongly consistent if each approximate problem gives the same solution as the original problem given the same data.

To measure the convergence of a method (which yields the result x_n to the true result x, we use metrics such as relative and absolute error:

Definition 1.1: Relative and absolute error

- **Absolute error** between x_n and x is defined by

$$\varepsilon = E_{\text{abs}}(x_n) = |x - x_n|$$

- If $x \neq 0$, the **relative error** between x_n and x is defined by

$$\eta = E_{\text{rel}}(x_n) = \frac{|x - x_n|}{|x|}$$

Remark 1.1

The definition above also works whenever x_n and x live in normed vector spaces, where the absolute values are replaced by the appropriate norm.
ASIDE: In the case where x_n and x are vectors or matrices, we can think of an component-wise form of error. In that case, if $x \neq 0$, the **relative error by component** x is defined by

$$E^c_{\text{rel}}(x_n) = \max_{i,j} \frac{|(x - x_n)_{ij}|}{|x_{ij}|}.$$

One last thing about stability and convergence of a numerical is that they are equivalent under certain conditions. It can be shown that:

Proposition 1.1: Lax-Richtmyer Equivalence Theorem

For a consistent numerical method, stability is equivalent to convergence.

Preliminary to 2.3 A priori and a posteriori Analysis

They are means of evaluating the error induced by a numerical method. This will be expanded upon later on Lecture 6.

2.4 Sources of Error in Computational Models

Oftentimes, a numerical problem approximates a mathematical problem, which in terms is a model of a physical problem. We can call it a computational model for said physical problem.

- The **global error** is the difference between the computed solution and the physical solution;

- The global error is the sum of the **mathematical model** and the **computational model**.

2.5 Machine Representation of Numbers

Let $\beta \in \mathbb{N}$ be the base of a number. We need to have $\beta \geq 2$.
First, we shall review representations of integers, mostly focusing on binary:

Definition 1.2: Unsigned integers

Let $x \in \{0\} \cup \mathbb{N}$. We can then represent it as this sum of n terms:

$$x_\beta = \sum_{k=0}^{n-1} x_k \beta^k,$$

where $0 \leq x_k < \beta$ for any $0 \leq k < n$. Note that n entries allow us to represent any integer in $[0, \beta^n - 1]$.

Definition 1.3: Signed integers: sign and magnitude

Let $x \in \mathbb{Z}$. We can then represent it as this sum of n terms and one sign bit s as:

$$x_\beta = (-1)^s \sum_{k=0}^{n-1} x_k \beta^k,$$

where $0 \leq x_k < \beta$ for any $0 \leq k < n$. s is equal to 0 if the number is positive, 1 if it is negative. Note that n entries allow us to represent any integer in $[-\beta^n - 1, \beta^n - 1]$, with 0 having two representations.

However, as you may have seen in ECSE 222 and ECSE 324, the following method of representing signed integers is usually preferred since it makes arithmetic on digital circuits and processors simpler to implement:

Definition 1.4: Signed integers: 2's complement

Let $\beta = 2$, $x \in \mathbb{Z}$. We can then represent x as this sum of n terms and :

$$x = [x_{n-1} x_{n-2} \dots x_1 x_0]$$

where $x_k \in \{0, 1\}$ for any $0 \leq k < n$. Note that n bits allow us to represent any integer in $[-2^n, 2^{n-1} - 1]$. If $x_{n-1} = 0$, x is positive, and $\sum_{k=0}^{n-2} x_k 2^k$. If $x_{n-1} = 1$, it is negative. To find $-x$ from x, flip all of x's bits and add 1.

In general, we can now write down any real number x with finitely many digits as such:

Definition 1.5: Positional system

If x has finitely many digits, one may write it as

$$x_\beta = (-1)^s[x_n x_{n-1} \dots x_1 x_0 . x_{-1} x_{-2} \dots x_{-m}],$$

where $x_m \neq 0$, and $0 \leq x_k < \beta$ for any $-m \leq k \leq n$. s is equal to 0 if the number is positive, 1 if it is negative. In fact, this representation is equivalent to:

$$x_\beta = (-1)^s \sum_{k=-m}^{n} x_k \beta^k.$$

Remark 1.2

For any base β, it is easy to show that for any $\varepsilon > 0$, and $x_\beta \in \mathbb{R}$, there exists $y_\beta \in \mathbb{R}$ with finitely many digits such that $|y_\beta - x_\beta| < \varepsilon$.

This representation, when $\beta = 2$, is called the **fixed-point system**. Definition 1.2 can be rewritten as:

$$x_\beta = (-1)^s[a_{N-2} a_{N-3} \dots a_k . a_{k-1} \dots a_0] = (-1)^s \beta^{-k} \sum_{j=0}^{N-2} a_j \beta^j,$$

giving us N total values: one for the sign bit, k for the fractional part, and $N-k-1$ for the integer part.
Drawback of fixed point: strongly limits the value of the minimum and maximum numbers that can be represented, unless N is very large.
To fix this, the exponential scaling k can be allowed to be varying as such:

Definition 1.6: Floating-point representation

If $x \neq 0$ has finitely many digits, one may write it as

$$x_\beta = (-1)^s \beta^e [0 . a_{t-1} \dots a_1 a_0] = (-1)^s \beta^{e-t} m,$$

where:

- $m = [a_{t-1} \dots a_1 a_0]$ is the **mantissa**, with $0 \leq m \leq \beta^t - 1$;
- t is the number of significant figures a_i, with $0 \leq a_i \leq \beta - 1$;
- e is the exponent, or the number of digits in the integer part.

This is the number's **floating-point representation**. Notice that the size of the

Figure 1: Single precision 32 bit representation.

Figure 2: Double precision 64 bit representation.

mantissa is equal to t, unless binary is used. In that case, normalization forces a_{t-1} to 1, making the size of the mantissa to be $t-1$.

Therefore, in terms of computer storage, the N entries (bits when $\beta = 2$) are stored like so:

- 1 bit for the sign s;
- t entries for the mantissa ($t-1$ if binary);
- The remaining $N-1-t$ entries to store the exponent ($N-t$ if binary).

Note that the exponent can only vary between two values $L \leq e \leq U$, where usually $L < 0$ and $U > 0$. This notation is similar to the scientific notation, since we keep track of the significant digits in the mantissa, and exponent gives an idea on the number's scale.
Typically, computers work with two formats for floating-point: single and double precision. In binary ($\beta = 2$), we tend to see:

- The **float** data type, of single precision, where $N = 32$ bits, it can be graphically represented as in Figure 1 above;
- The **double** data type, of double precision, where $N = 64$ bits, it can be graphically represented as in Figure 2 above.

Definition 1.7: Set of floating-point numbers

Define the set of floating-points numbers with t significant figures, base $\beta \geq 2$, $0 \leq a_i \leq \beta - 1$, and exponent range (L, U) with $L \leq e \leq U$ to be

$$\mathbb{F}(\beta, t, L, U) := \{0\} \cup \{x \in \mathbb{R} : x = (-1)^s \beta^{e-t} \sum_{i=0}^{t-1} a_i \beta^i\}.$$

This representation taken at face value gives us the set of **denormalized floating-point numbers**. To make sure each number has a unique representation, we assume that $a_{t-1} \neq 0$, or that $m \geq \beta^{t-1}$. In that case, a_{t-1} is the principal significant digit (always equal to 1 in binary representations), and we end up with the set of **normalized floating-point numbers**, with $\beta^{t-1} \leq m \leq \beta^t - 1$.

Remark 1.3

Just like in sign and magnitude representations, 0 cannot be represented uniquely, since the bit sign s does not affect 0's value. One tends to assume 0 is positive (in other words, assume the unique representation of 0 uses $s = 0$), but using both representations can be useful in representing concepts such that 0^+ and 0^-. This notation can be seen in the context of limits. For example,

$$\lim_{x \to 0^+} f(x), \quad \lim_{x \to 0^-} f(x).$$

Next, because of the sign bit, we notice that $-x \in \mathbb{F}(\beta, t, L, U)$ if and only if $x \in \mathbb{F}(\beta, t, L, U)$. Furthermore, for normalized values, we notice that the values of x are in the range:

$$x_{min} = \beta^{L-1} \leq |x| \leq \beta^U(U - L + 1) + 1 = x_{max}.$$

While the number of possible normalized values is the cardinality:

$$|\mathbb{F}(\beta, t, L, U)| = 2(\beta - 1)\beta^{t-1}(U - L + 1) + 1.$$

To represent numbers smaller than x_{min} in absolute value, one needs to extend $\mathbb{F}$ to the set $\mathbb{F}_D$ of denormalized numbers. If we only do that extension for those small numbers, uniqueness is preserved, while being able to represent numbers whose absolute values are as small as β^{L-t}.
Now, let's look at the distribution of floating-point numbers:

- The absolute distance between two consecutive numbers x and y is $\varepsilon = |x - y| = \beta^{e-t}$.
- However, within any interval $[\beta^e, \beta^{e+1}]$, those numbers are equally spaced, but that spacing increases exponentially as e increases.
- The absolute distance between two consecutive numbers 1 and $y > 1$ is $|1-y| = \beta^{1-t}$, which we call as the **machine epsilon**, denoted ε_m.
- The relative distance between two consecutive numbers x and y is $\eta = \frac{|x-y|}{|x|} = \frac{\beta^{e-t}}{m\beta^{e-t}} = \frac{1}{m}$, where m is the mantissa of x.

We can develop η to give it a range of possible values. First, notice that:

$$\eta = \frac{1}{m} = \frac{1}{a_{t-1} \dots a_0} = \frac{\beta^{-t}}{(0.a_{t-1} \dots a_0)}.$$

Since $\beta^{-1}a_{t-1} \leq (0.a_{t-1} \dots a_0) \leq 1$, it follows that:

$$\beta^{-1}\varepsilon_m = \beta^{-t} = \eta = \frac{|x - y|}{|x|} \leq \beta^{1-t} = \varepsilon_m,$$

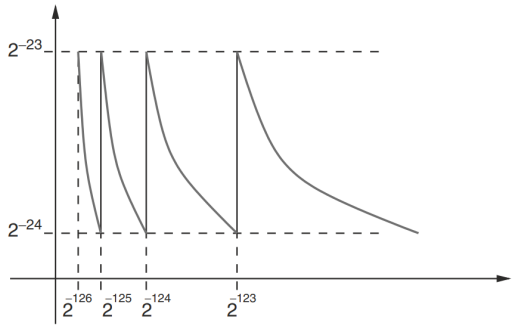

Figure 3: Variation of relative distance for the set of numbers $\mathbb{F}(2, 24, -125, 128)$ IEC559/IEEE754 in single precision.

so the relative distance only depends on the inverse of the mantissa.
Consequentially, as x increases within $[\beta^e, \beta^{e+1}]$, the mantissa increases from β^{t-1} to $\beta^t - 1$ (for normalized values), and the relative distance decreases from β^{1-t} to $(\beta^t - 1)^{-1} \approx \beta^t$. As x reaches β^{e+1}, the relative precision jumps back up to β^{1-t}, as shown on the figure next page. This phenomenon is called **wobbling precision**, and is more pronounced as β increases, so working with binary minimizes this issue.
To get into more practical matters, we should define and discuss the IEC559/IEEE754 standard which is most widely used:

Definition 1.8: IEC559/IEEE754 standards

Define the following sets of floating-point numbers (according to IEC559/IEEE754):

- quarter() $= \mathbb{F}(2, 5, -1, 4)$ ($N = 8$ bits are used);
- half() $= \mathbb{F}(2, 11, -13, 16)$ ($N = 16$ bits are used);
- single() $=$ float() $= \mathbb{F}(2, 24, -125, 128)$ ($N = 32$ bits are used);
- double() $= \mathbb{F}(2, 53, -1021, 1024)$ ($N = 64$ bits are used).

They respectively denote quarter, half, single, and double precision.

For general binary sets $\mathbb{F}(2, t, L, U)$ stored in N bits, we obtain using E to denote the value actually stored in memory, $t - 1$ to be the size of the mantissa, and e to be the exponent as used so far (i.e. the number of entries in the integer part), we get:

- Number of bits to store $E = N - t$;
- bias $= 2^{N-t-1} - 2$;
- range of $E = [0, 2^{N-t} - 1]$;

	$E = 0/e = -2^{N-t-1} + 2$	Other values	$E = 2^{N-t} - 1/e = 2^{N-t-1} + 1$
mantissa $= 0$	± 0	$(-1)^s(0.10)2^e$	$\pm\infty$
mantissa $\neq 0$	denormalized	$(-1)^s(0.1\text{mantissa})2^e$	NaN (not a number)

	$E = 0/e' = -2^{N-t-1} + 1$	Other values	$E = 2^{N-t} - 1/e' = 2^{N-t-1}$
mantissa $= 0$	± 0	$(-1)^s(1.0)2^{e'}$	$\pm\infty$
mantissa $\neq 0$	denormalized	$(-1)^s(1.\text{mantissa})2^{e'}$	NaN (not a number)

- range of $e = [-2^{N-t-1} + 2, 2^{N-t-1} + 1]$.
  Now, we must take two values in the range to take care of the edge cases. Those are, considering that $E = e-$bias:

After removing the edge cases, the range for E is $[1, 2^{N-t} - 2]$, and the range of e is $[-2^{N-t-1} + 3, 2^{N-t-1}]$, which are good values to give for the lower and upper bound parameters ($L = -2^{N-t-1} + 3$, $U = 2^{N-t-1}$).
We can reformulate the same thing in a more similar way to the slides, where we define a new exponent $e' = e - 1$, which is an exponent more closely related to scientific notation. That way, the tables become:

- Number of bits to store $E = N - t$;
- bias' $= 2^{N-t-1} - 1 =$ bias $+1$;
- range of $E = [0, 2^{N-t} - 1]$;
- range of $e' = [-2^{N-t-1} + 1, 2^{N-t-1}]$.

Now, we must take care of the edge cases, as done in the second table above, with $E = e'-$bias'.
After removing the edge cases, the range for E is $[1, 2^{N-t} - 2]$, and the range of e' is $[-2^{N-t-1} + 2, 2^{N-t-1} - 1]$.
We can plug in these values to produce the third table for the IEEE values (woe means "without edge cases"):
Let's now think of other practical issues with floating-point numbers. The first issue is about the idea of best approximating any $x \in \mathbb{R}$ (as long as it is not so big that it goes outside the range of numbers the system can represent). The second issue is about

NOTES	N	t-1	N-t	E range	bias	e range	e woe
quarter()	8	4	3	[0,7]	2	[-2,5]	[-1,4]
half()	16	10	5	[0,31]	14	[-14,17]	[-13,16]
single()	32	23	8	[0,255]	126	[-126,129]	[-125,128]
double()	64	52	11	[0,2047]	1022	[-1022,1025]	[-1021,1024]

SLIDES	N	t-1	N-t	E range	bias'	e' range	e' woe
quarter()	8	4	3	[0,7]	3	[-3,4]	[-2,3]
half()	16	10	5	[0,31]	15	[-15,16]	[-14,15]
single()	32	23	8	[0,255]	127	[-127,128]	[-126,127]
double()	64	52	11	[0,2047]	1023	[-1023,1024]	[-1022,1023]

defining an arithmetic for $\mathbb{F}$, since even if $x, y \in \mathbb{F}$, a binary operation that uses the two won't always yield another element of $x, y \in \mathbb{F}$.
To solve the first problem, we must define, for any appropriate $x \in \mathbb{R}$, an operation which outputs the closest floating-point number. For this define the map $fl : \mathbb{R} \to \mathbb{F}$ by

$$fl(x) = (-1)^s(0.a_{t-1}\dots a_1\widetilde{a_0})\beta^e,\ \widetilde{a_0} = \begin{cases} a_0 \text{ if the first cut off term } a_{-1} < \beta/2 \\ a_0 + 1 \text{ if the first cut off term } a_{-1} \geq \beta/2 \end{cases}$$

This mapping is called the **rounding map**. If instead we always have $\widetilde{a_0} = a_0$ it is instead called the **chopping map**.
Some properties of $fl(x)$:

- $fl(x) = x$ for all $x \in \mathbb{R}$;
- If $x \leq y$, then $fl(x) \leq fl(y)$;
- If $x \in \mathbb{R}$, then $x(1-u) \leq fl(x) \leq x(1+u)$, where $u = \frac{1}{2}\beta^{1-t} = \frac{1}{2}\varepsilon_m$ is called the **roundoff unit**.

In other words, the relative error due to rounding becomes:

$$\eta = E_{\text{rel}}(x) = \frac{|x - fl(x)|}{|x|} \leq u$$

Then, by definition of $fl(x)$, the absolute error is:

$$\varepsilon = E(x) = |x - fl(x)| \leq \beta^{e-t}|(a_{t-1}\dots a_1a_0.a_{-1}\dots) - (a_{t-1}\dots a_1\widetilde{a_0})| \leq \frac{1}{2}\beta^{e-t}.$$

The second issue, which is to create an analogous operation from $\circ$ (which could denote an addition, subtraction, multiplication, or division), is to denote an analogous operation $\bullet : \mathbb{R} \times \mathbb{R} \to \mathbb{F}$, to be defined as:

$$x \bullet y := fl(fl(x) \circ fl(y)).$$

This operation commutes for additions and multiplications, but other properties like associativity are lost. Also, for operations like subtractions, if we want to have the typical roundoff precision of $(x \circ y)(1-u) \leq x \bullet y \leq (x \circ y)(1+u)$, we may need to keep track of other variables, such as rounding digits, which most computers do keep track to make sure the result of a subtraction isn't too far off. An arithmetic that does not have that roundoff precision is called aberrant.

Lecture 3: Linear Equations

Introduction to Linear Equations

Our next concern is this course is the problem of solving systems of linear equations. A system of m equation in n unknowns is a set of algebraic relations of the form

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \text{ for all } 1 \leq i \leq m.$$

x_j are the unknowns we want to solve for, a_{ij} are the coefficients of the system and b_j are the terms on the right hand side. This system can be rewritten in matrix form as:

$$A\mathbf{x} = \mathbf{b}.$$

Let $\mathbb{F}$ be a field, (whose definition will be explained, but most of the time one can just let $\mathbb{F} = \mathbb{R}$). Then, $A = (a_{ij}) \in \mathbb{F}^{m\times n}$ is a m by n coefficient matrix, $\mathbf{x} = (x_i) \in \mathbb{F}^n$ is the unknown vector, and $\mathbf{b} = (b_i) \in \mathbb{F}^m$ is the right hand vector.
In the case where $m = n$, that is, $A \in \mathbb{R}^{n\times n}$, $A\mathbf{x} = \mathbf{b}$ has a unique solution $\mathbf{x} = A^{-1}\mathbf{b}$, if and only if one of those equivalent statements holds:

- A is invertible, or A^{-1} exists;
- $\text{rank}(A) = n$;
- The homogeneous systems $A\mathbf{x} = \mathbf{0}$ only admits the trivial solution $\mathbf{x} = \mathbf{0}$.

If A is invertible, we can use Cramer's rule, but since computing determinants is intensive (which Cramer's rule uses), it is of little practical use. One can also directly compute $\mathbf{x} = A^{-1}\mathbf{b}$. There are advantages and inconvenients in doing so, along with some other facts:

- Once A^{-1} is computed, the system can be easily solved for different right-hand side (RHS) vectors $\mathbf{b}$;
- The numerical complexity of matrix inversion is $O(n^3)$ even for sparse matrices (matrices with mostly 0 entries);
- The matrix inverse A^{-1} is in general dense even when A is sparse.
- Solving using matrix inversion is not as numerically stable as other methods.
- This approach is useful for theoretical considerations.

For now, we will consider direct methods of system solving (methods that yield the solution in finitely many steps) rather than iterative methods (methods that induce a sequence of approximate solutions that converge to the true solution).

3.1 Stability Analysis of Linear Systems*

3.2 Solution of Triangular Systems

Consider the nonsingular linear system:

$$\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Then, we immediately see that the solution for that equation is :

- $x_1 = \frac{b_1}{l_{11}}$,
- $x_2 = \frac{b_2 - l_{21}x_1}{l_{22}}$,
- $x_3 = \frac{b_3 - l_{31}x_1 - l_{32}x_2}{l_{33}}$.

This is called the method of **forward substitution**. In general, the solution of $L\mathbf{x} = \mathbf{b}$, where L is a $n \times n$ nonsingular lower triangular matrix, the same method can be used to yield the following solution:

$$x_i = \frac{1}{l_{ii}}(b_i - \sum_{j=1}^{i-1} l_{ij}x_j) \text{ for any } i = 1, \ldots, n.$$

Similarly, the solution of $U\mathbf{x} = \mathbf{b}$, where U is a $n \times n$ nonsingular upper triangular matrix, the method of **backward substitution** can be used to obtain:

$$x_i = \frac{1}{u_{ii}}(b_i - \sum_{j=i+1}^{n} u_{ij}x_j) \text{ for any } i = n, \ldots, 1.$$

Notes:

- in both cases, if the first index of the sum is greater than the last term, consider the sum to be 0. Also, because of rounding errors, their computer implementation won't yield exact solutions, so one would need to analyze the induced rounding error.
- An application of this is the ability to invert triangular matrices. If $V = T^{-1}$, then the $\mathbf{v_i}$, the i'th column of V, must satisfy $T\mathbf{v_i} = \mathbf{e_i}$, where $\mathbf{e_i}$ is the standard basis vector in $\mathbb{R}^n$ (it is all 0's except 1 at the i'th entry). Then, since a linear transformation in a vector space is entirely determined by the mapping of its basis, we see that $TV = I_n$, so $V = T^{-1}$. Since the basis vectors are "sparse", one could derive a simplified formula for the inverse!

3.3 The Gaussian Elimination Method (GEM)

The goal of Gaussian elimination is to reduce any linear system $A\mathbf{x} = \mathbf{b}$ into a reduced system $U\mathbf{x} = \widehat{\mathbf{b}}$, where U is upper triangular. This method exploits the fact that rescaling and adding up rows and columns in a matrix preserves the solution of the linear system.

Let's say that at the first step, we have the system $A^{(1)}\mathbf{x} = \mathbf{b}^{(1)}$. We proceed row by how, always assuming the diagonal entries are nonzero from the non-singularity of A (if it singular, that may not be true, then just move on to the new row and swap rows to get your matrix to be upper diagonal over the process). Now, to obtain the equivalent system $A^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$, we need to substract an appropriate multiple $m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}}$ for any i to make sure the lower term $a_{21}^{(1)}$ vanishes. In other words,

- $a_{ij}^{(2)} = a_{ij}^{(1)} - m_{i1}a_{1j}^{(1)}$ for any $i, j = 2, \ldots, n$,
- $b_i^{(2)} = b_i^{(1)} - m_{i1}b_1^{(1)}$ for any $i, j = 2, \ldots, n$.

This gives us the equivalent system:

$$\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \ldots & a_{2n}^{(2)} \\ \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & \ldots & a_{nn}^{(2)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{bmatrix}$$

If we keep repeating the same process of substracting appropriate multiples of rows to vanish lower triangular terms, with for $k = 1, \ldots, n-1$, $m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$, and

- $a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik}a_{kj}^{(k)}$ for any $i, j = k+1, \ldots, n$,
- $b_i^{(k+1)} = b_i^{(k)} - m_{ik}b_k^{(k)}$ for any $i, j = k+1, \ldots, n$.

Giving us at the k'th step the equivalent system:

$$\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & \ldots & \ldots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & \ldots & 0 & a_{kk}^{(k)} & \ldots & a_{kn}^{(k)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \ldots & 0 & a_{nk}^{(k)} & \ldots & a_{nn}^{(k)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_k^{(k)} \\ \vdots \\ b_n^{(k)} \end{bmatrix}$$

When $k = n$, we end up with the following upper triangular linear system:

$$\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & \dots & 0 & a_{ii}^{(i)} & \dots & a_{in}^{(i)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \dots & 0 & 0 & \dots & a_{nn}^{(n)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_i^{(i)} \\ \vdots \\ b_n^{(n)} \end{bmatrix}$$

The nonzero terms $a_{ii}^{(i)}$ are called **pivots**. Then we can use backward substitution to solve the system. Note that some things break down for singular cases, and that if this statement of GEM is unclear, there are plenty of examples that can be found online, it's a simple procedure to apply after some practice. The GEM has some advantages and disadvantages:

- It's a systematic algorithm and can be easily implemented as a computer program.
- GEM is accurate can be much faster that matrix inversion for sparse systems.
- However, row operations are performed on the augmented matrix (matrix $\boldsymbol{A}$ and RHS vector $\mathbf{b}$ simultaneously), so we must restart the process from the beginning if we need to solve the system for a different RHS.

3.3 The LU factorization using GEM

One application of Gaussian elimination is the ability to decompose a nonsingular matrix $\boldsymbol{A}$ as a product $\boldsymbol{A} = \boldsymbol{LU}$, where $\boldsymbol{U} = \boldsymbol{A}^{(n)}$ is upper triangular. Furthermore, one can find $\boldsymbol{L}$ by finding the elementary row operation matrices needed to reduce $\boldsymbol{A}$ to $\boldsymbol{U}$. We can then write $\boldsymbol{M}_{n-1} \dots \boldsymbol{M}_1 \boldsymbol{A} = \boldsymbol{A}^{(n)} = \boldsymbol{U}$, implying that:

$$\boldsymbol{A} = \boldsymbol{LU} = (\boldsymbol{M}_{n-1} \dots \boldsymbol{M}_1)^{-1} \boldsymbol{U} \Rightarrow \boldsymbol{L} = (\boldsymbol{M}_{n-1} \dots \boldsymbol{M}_1)^{-1} = \boldsymbol{M}_1^{-1} \dots \boldsymbol{M}_{n-1}^{-1}.$$

We can rewrite the Gaussian elimination formulation as equivalent matrix multiplications:

$$\boldsymbol{M}_k = \begin{bmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & 1 & 0 & & 0 \\ 0 & & -m_{k+1,k} & 1 & & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & & -m_{n,k} & 0 & & 1 \end{bmatrix},$$

with entries $(\boldsymbol{M}_k)_{ip} = \delta_{ip} - m_{ik}\delta_{kp}$, where $\delta_{mn} = \begin{cases} 0, & m \neq n \\ 1, & m = n \end{cases}$. One can show that $\boldsymbol{M}_k^{-1} = 2\boldsymbol{I}_n - \boldsymbol{M}_k$, making it and $\boldsymbol{L}$ lower diagonal with diagonals of 1.
Simpler row reduction matrices are:

- Row rescaling matrix (rescale m'th row)

$$R_m(\lambda) = \begin{bmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & 1 & 0 & & 0 \\ 0 & & 0 & \lambda & & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & & 0 & 0 & & 1 \end{bmatrix}, \; R_m(\lambda)^{-1} = R_m(\lambda^{-1})$$

- Row swapping matrix (swap rows i and j)

$$P_{ij} = \begin{bmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & 1 & \vdots \\ 0 & & 0 & 0 & & 0 \\ 0 & & 0 & 0 & & 0 \\ \vdots & 1 & \vdots & \vdots & \ddots & \vdots \\ 0 & & 0 & 0 & & 1 \end{bmatrix}, \; P_{ij}^{-1} = P_{ij}.$$

- Elimination matrix (add m*row i into row j)

$$E_{ij}(m) = \begin{bmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & 1 & 0 & & 0 \\ 0 & & 0 & 1 & & 0 \\ \vdots & m & \vdots & \vdots & \ddots & \vdots \\ 0 & & 0 & 0 & & 1 \end{bmatrix}, \; E_{ij}(m)^{-1} = E_{ij}(-m).$$

Now, see how the LU decomposition helps us solve the linear system $A\mathbf{x} = \mathbf{b}$: if $A = LU$, $L(U\mathbf{x}) = \mathbf{b}$. We can then solve for $\mathbf{y} = U\mathbf{x}$ with forward substitution, then for $\mathbf{x}$ using back substitution. Obviously, LU decomposition costs the same as GEM computationally.

Definition 3.1: Facts about matrix multiplication

- $$\begin{bmatrix} | & & | \\ \mathbf{A_1} & \dots & \mathbf{A_n} \\ | & & | \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \sum_{i=1}^{n} x_i \mathbf{A_i}$$

- $$\begin{bmatrix} x_1 & \dots & x_n \end{bmatrix} \begin{bmatrix} - & \mathbf{C_1}^T & - \\ & \vdots & \\ - & \mathbf{C_n}^T & - \end{bmatrix} = \sum_{i=1}^{n} x_i \mathbf{C_i}^T$$

The Doolittle Algorithm

There are other more efficient methods to compute the *LU* factorization, called Doolittle's algorithm, producing a compact form of the Gaussian elimination method. It consists in solving this nonlinear system of n^2 equations:

$$a_{ij} = \sum_{r=1}^{\min\{i,j\}} l_{ir} u_{rj}$$

Since it has $n^2 + n$ unknowns (from the number of L and U entries), we have infinitely many solutions. Therefore, we can fix n entries to be 1 and end up with one solution: The Doolittle method sets L's diagonals l_{kk} to be 1:

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ l_{n1} & \dots & 1 \end{bmatrix} = \begin{bmatrix} u_{11} & \dots & u_{1n} \\ \vdots & \ddots & \vdots \\ 0 & \dots & u_{nn} \end{bmatrix}$$

One can then sequentially solve these equations, with solution:

$$u_{kj} = a_{kj} - \sum_{r=1}^{k-1} l_{kr} u_{rj}, \ j = k \dots, n,$$

$$l_{ik} = \frac{1}{u_{kk}} (a_{ik} - \sum_{r=1}^{k-1} l_{ir} u_{rk}), \ i = k+1 \dots, n.$$

In contrast, the Crout method sets U's diagonals u_{kk} to be 1:

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ l_{n1} & \dots & l_{nn} \end{bmatrix} = \begin{bmatrix} 1 & \dots & u_{1n} \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix}$$

In a very similar way, one can then sequentially solve these equations, with solution:

$$l_{ik} = a_{ik} - \sum_{r=1}^{k-1} l_{ir} u_{rk}, \ i = k \dots, n,$$

$$u_{kj} = \frac{1}{l_{kk}} (a_{kj} - \sum_{r=1}^{k-1} l_{kr} u_{rj}), \ j = k+1 \dots, n.$$

3.4 Other Types of Factorization*

- LDM^T Factorization: $A = LDM^T$, where L is lower triangular, M^T is upper triangular, D is diagonal.
- LDL^T/Cholesky Factorization: If A is symmetric, $A = LDL^T$, basically as before with $L = M$. Positive definite matrices has positive entries in D.
- QR Factorization: If A is rectangular, $A = QR$, where Q is orthogonal and R is upper trapezoidal.

Lecture 4: More on Systems of Linear Equations + Gaussian Elimination

Introduction to Linear Systems Using Non-Square Matrices

So far, we mostly talked about systems where A is a square matrix, but now one can realize that Gaussian elimination still works for non square systems, we just need to make sure the elementary matrices that represent row and column operations are of the right dimension. The number of solutions that $A\mathbf{x} = \mathbf{b}$ has depends on A's dimensions, rank, and whether $\mathbf{b} \in \text{col}(A)$.

Definition 4.1: Rank of a matrix

- The rank of a matrix A is the number of independent columns or rows of A.
- Since Gaussian Elimination preserves rank, it also corresponds to the number of nonzero pivots in the fully row reduced echelon form of A.
- $A \in \mathbb{R}^{n\times n}$ is a square matrix, it is invertible iff $\text{rank}(A) = n$.
- If $A \in \mathbb{R}^{m\times n}$, $\text{rank}(A) \leq \min\{m, n\}$.

Definition 4.2: Overdetermined Systems

Let $A \in \mathbb{R}^{m\times n}$, $m > n$, then linear system $A\mathbf{x} = \mathbf{b}$ is overdetermined. If $\mathbf{b} \notin \text{col}(A)$, there is no solution to the system.
If $\text{rank}(A) = \min\{m, n\} = n$ and $\mathbf{b} \in \text{col}(A)$ (which can be found from GEM), then the system has one solution.
If $\text{rank}(A) < \min\{m, n\} = n$, $\mathbf{b} \in \text{col}(A)$ implies infinitely many solutions to the system.
One can minimize the error $\|A\mathbf{x} - \mathbf{b}\|$ with the solution $\mathbf{x} = (A^TA)^{-1}A^T\mathbf{b}$.

Definition 4.3: Underdetermined Systems

Let $A \in \mathbb{R}^{m\times n}$, $m < n$, then linear system $A\mathbf{x} = \mathbf{b}$ is underdetermined. If $\mathbf{b} \notin \text{col}(A)$, there is no solution to the system.
If $\text{rank}(A) = \min\{m, n\} = m$, $\mathbf{b} \in \text{col}(A)$ implies infinitely many solutions to the system.
Given infinitely many solutions, one can find a solution with smallest norm $\|\mathbf{x}\|$ with $\mathbf{x} = A^T(AA^T)^{-1}\mathbf{b}$.

Proposition 4.1: Existence and Uniqueness of Solutions of Linear Systems of Equations

Let r be the rank of matrix $A \in \mathbb{R}^{m\times n}$, and r_b be the rank of the augmented matrix

$[A|b]$ (the matrix formed by appending vector b as an additional column to matrix A).
Existence of Solutions:

$$\text{If } r < r_b, \text{ no solution.}$$
$$\text{If } r = r_b = n, \text{ unique solution.}$$
$$\text{If } r = r_b < n, \text{ infinitely many solutions.}$$

Uniqueness of Solutions:

$$\text{If } r = n, \text{ unique solution.}$$
$$\text{If } r < n, \text{ infinitely many solutions.}$$

Column Space and b:

If b is in the column space of A, and $r = r_b < n$, infinitely many solutions.
If b is not in the column space of A, and $r < r_b$, no solution.

3.5 Pivoting

The GEM process breaks down as soon as a zero pivotal entry is computed. In such an event, one needs to resort to pivoting, which amounts to exchanging rows (or columns) of the system in such a way that nonvanishing pivots are obtained. This is done with adding row-swapping matrices P_k at every elimination step, we then end up with

$$A = LU = (M_{n-1}P_{n-1}\dots M_1P_1)^{-1}U.$$

This is what we call **partial pivoting**.
Let's focus on partial pivoting: the idea for this is to focus on the pivot entries on the diagonal. Before the k'th elimination step, what we do is compare the pivot entry a_{kk} with other entries on the same column, which are of the form a_{kj}. If there exists $j \in \{1,\dots,m\}$ such that $|a_{kj}| > |a_{kk}|$, swap the rows j and k, then $P_k = P_{ik}$. If there are many such i's we simply pick the one with the largest $|a_{kj}|$.
One can also implement **full pivoting**, which also considers the use of column swapping matrices Q_k at every elimination step, giving us:

$$A(Q_{n-1}\dots Q_1) = LU = (M_{n-1}P_{n-1}\dots M_1P_1)^{-1}U.$$

Before the k'th elimination step, what we do is compare the pivot entry a_{kk} with other entries on the whole matrix instead, which are of the form a_{ij}. If there exists $i,j \in \{1,\dots,m\} \times \{1,\dots,n\}$ such that $|a_{ij}| > |a_{kk}|$, swap the rows j and k, and then swap the columns i and k. This amount to left multiplication by $P_k = P_{ik}$ and right multiplication by $Q_k = P_{ik}$. If there are many such ij pairs we simply pick the one with the largest $|a_{ij}|$.
Partial pivoting has a cost: it slows down the process of GEM. This is even more the case with full pivoting.

Pivoting to Improve the Precision of the LU Decomposition

The main purpose of pivoting is to reduce round-off error and increase numerical stability when solving systems of equations.
Without pivoting, one can end up with important rounding errors, as shown in the following example:

Example 4.1

The system:

$$\begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

can be shown to have the unique analytical solution:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 + 10^{-4} \\ 1 - 10^{-4} \end{bmatrix}.$$

Now, let's resolve that system in the case where we are only able to with 3 significant digits ($t = 3$). Then, applying the LU algorithm pivoting gives us:

$$\begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 10^{-4} & 1 \\ 0 & 1 - 10^4 \end{bmatrix} \approx \begin{bmatrix} 10^{-4} & 1 \\ 0 & -10^4 \end{bmatrix} = U$$

This yields $L = \begin{bmatrix} 1 & 0 \\ 10^4 & 1 \end{bmatrix}$, and forward and backward substitution give us $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, which is very far off from the actual solution.
If we include the method of full pivoting beforehand in our algorithm, we get the modified system:

$$\begin{bmatrix} 1 & 1 \\ 1 & 10^{-4} \end{bmatrix} \begin{bmatrix} x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

And we can find U by reducing the system:

$$\begin{bmatrix} 1 & 1 \\ 1 & 10^{-4} \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 \\ 0 & 10^{-4} - 1 \end{bmatrix} \approx \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} = U$$

This yields $L = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, and forward and backward substitution give us $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, giving us a much more accurate solution.

Lecture 5: Basic Linear Algebra

Basic Definitions

To understand some of the concepts in this course, it is useful to review some of the relevant Linear Algebra, which is in the first chapter of the book.
The first thing to understand about linear algebra and matrices is that the values that those matrices contain must be well-behaved. Indeed, they have to satisfy the field axioms:

Definition 5.1: Fields

A set F equipped with two binary operations, addition ($+$) and multiplication ($\cdot$), is called a field if it satisfies the following axioms:

- **Closure under Addition:** For all $a, b \in F$, $a + b \in F$.
- **Associativity of Addition:** For all $a, b, c \in F$, $(a + b) + c = a + (b + c)$.
- **Existence of Additive Identity:** There exists an element $0 \in F$ such that for all $a \in F$, $a + 0 = a$.
- **Existence of Additive Inverse:** For every $a \in F$, there exists an element $(-a) \in F$ such that $a + (-a) = 0$.
- **Closure under Multiplication:** For all $a, b \in F$, $a \cdot b \in F$.
- **Associativity of Multiplication:** For all $a, b, c \in F$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
- **Existence of Multiplicative Identity:** There exists an element $1 \in F$ such that for all $a \in F$, $a \cdot 1 = a$.
- **Existence of Multiplicative Inverse:** For every non-zero element $a \in F$, there exists an element $a^{-1} \in F$ such that $a \cdot a^{-1} = 1$.
- **Distributive Property:** For all $a, b, c \in F$, $a \cdot (b + c) = a \cdot b + a \cdot c$.

We can denote that field as $\mathbb{F} = (F, +, \cdot)$. This is all pretty abstract, so the main thing to remember from this is that fields behave the exact same way as typical real and complex number. Indeed, common examples of fields are $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{C}$ (not $\mathbb{N}$ and $\mathbb{Z}$); other less obvious field are the set $\{0, 1\}$ equipped with modulo-2 addition (XOR) and multiplication (AND), as well as $\{0, \ldots, n-1\}$ equipped with modulo-n addition and multiplication. Therefore, all of those fields are possible building blocks to do linear algebra with, even though it is most useful to stick to $\mathbb{R}$ (and sometimes $\mathbb{C}$) in most settings. Now, let's think of actual linear algebra and state the definition of a vector space:

Definition 5.2: Vector Spaces

Let V be a vector space over a field $\mathbb{F}$ with vector addition $+$ and scalar multiplication $\cdot$. The following axioms must be satisfied:

- **Closure under Vector Addition:** For all $\mathbf{u}, \mathbf{v} \in V$, $\mathbf{u} + \mathbf{v} \in V$.
- **Associativity of Vector Addition:** For all $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$, $(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w})$.
- **Existence of Additive Identity:** There exists a vector $\mathbf{0} \in V$ such that for all $\mathbf{u} \in V$, $\mathbf{u} + \mathbf{0} = \mathbf{u}$.
- **Existence of Additive Inverse:** For every $\mathbf{u} \in V$, there exists a vector $-\mathbf{u} \in V$ such that $\mathbf{u} + (-\mathbf{u}) = \mathbf{0}$.
- **Closure under Scalar Multiplication:** For all $\lambda \in \mathbb{F}$ and $\mathbf{u} \in V$, $\lambda \cdot \mathbf{u} \in V$.
- **Compatibility of Scalar Multiplication with Field Multiplication:** For all $\lambda, \mu \in \mathbb{F}$ and $\mathbf{u} \in V$, $(\lambda \cdot \mu) \cdot \mathbf{u} = \lambda \cdot (\mu \cdot \mathbf{u})$.
- **Identity Element of Scalar Multiplication:** There exists an element $1 \in \mathbb{F}$ such that for all $\mathbf{u} \in V$, $1 \cdot \mathbf{u} = \mathbf{u}$.
- **Distributive Property of Scalar Multiplication with Respect to Vector Addition:** For all $\lambda \in \mathbb{F}$ and $\mathbf{u}, \mathbf{v} \in V$, $\lambda \cdot (\mathbf{u} + \mathbf{v}) = \lambda \cdot \mathbf{u} + \lambda \cdot \mathbf{v}$.
- **Distributive Property of Scalar Multiplication with Respect to Field Addition:** For all $\lambda, \mu \in \mathbb{F}$ and $\mathbf{u} \in V$, $(\lambda + \mu) \cdot \mathbf{u} = \lambda \cdot \mathbf{u} + \mu \cdot \mathbf{u}$.

Based on these axioms, one can deduce some other useful properties of vector spaces:

Proposition 5.1

$\mathbf{0} \in V$ is unique.

Proof. Assume there exist $\mathbf{a} \in V$ such that $\mathbf{a} + \mathbf{v} = \mathbf{v}$ for any $\mathbf{v} \in V$. In particular, $\mathbf{a} + \mathbf{0} = \mathbf{0}$. But axiomatically,

$$\mathbf{a} = \mathbf{a} + \mathbf{0} = \mathbf{0},$$

concluding the proof, since we have shown that any vector with the same properties as $\mathbf{0}$ is necessarily $\mathbf{0}$. ■

Proposition 5.2

For any $\mathbf{v} \in V$, $0\mathbf{v} = \mathbf{0}$.

Proof. By distributivity,

$$\mathbf{v} + 0\mathbf{v} = 1\mathbf{v} + 0\mathbf{v} = (1 + 0)\mathbf{v} = \mathbf{v},$$

but by the previous proposition, $\mathbf{0}$ is the unique additive identity, so $0\mathbf{v} = \mathbf{0}$. ■

When we think of vectors in $\mathbb{R}^n$ (which is an example of a vector space), we see that rescaling or adding up those vectors just consists in doing the same in each of the components.

Proposition 5.3

For any $\mathbf{v} \in V$, $0\mathbf{v} = \mathbf{0}$.

Proof. By distributivity,

$$\mathbf{v} + 0\mathbf{v} = 1\mathbf{v} + 0\mathbf{v} = (1+0)\mathbf{v} = \mathbf{v},$$

but by the previous proposition, $\mathbf{0}$ is the unique additive identity, so $0\mathbf{v} = \mathbf{0}$. ■

Norms and Normed Vector Spaces

Sometimes, it is useful to consider the size of a vector in a vector space:

Definition 5.3: Norms and Normed Vector Spaces

Let V be a vector space over a field $\mathbb{F}$. A normed vector space is defined by a norm function $\|\cdot\| : V \to [0, \infty)$, satisfying the following axioms:

- **Non-negativity:** For all $\mathbf{u} \in V$, $\|\mathbf{u}\| \geq 0$ and $\|\mathbf{u}\| = 0$ if and only if $\mathbf{u} = \mathbf{0}$.
- **Homogeneity:** For all $\lambda \in \mathbb{F}$ and $\mathbf{u} \in V$, $\|\lambda \cdot \mathbf{u}\| = |\lambda| \cdot \|\mathbf{u}\|$.
- **Triangle Inequality:** For all $\mathbf{u}, \mathbf{v} \in V$, $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$.

A vector space equipped with a norm function is called a normed vector space. Any normed vector space induces a metric $d(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|$, which is a way to quantify the distance/error margin between two vectors.

Let's now look at various examples of norms and normed vectors to get a better sense of them:

Example 5.1: Examples of Normed Vector Spaces

- The norm in the real or complex numbers is straightforwardly given by the absolute value:
$$\|x\| = |x|$$
- The $L^p(X)$ norm generalizes norms for functions on a measurable space X, with $p \geq 1$. It is defined as:
$$\|f\|_p = \left(\int_X |f|^p \, d\mu \right)^{1/p}$$

- In the specific case of sequence spaces, the $l^p(X)$ norm is equivalent to the $L^p(\mathbb{N})$ norm, where X is the set of natural numbers. This is expressed as:

$$\|x\|_p = \left(\sum_{n=1}^{\infty} |x_n|^p \right)^{1/p}$$

- For a finite-dimensional vector space $\mathbb{R}^n$, the norm can be seen as the L^p norm over the set $\{1, \ldots, n\}$:

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

- Specific cases for $p = 1$, $p = 2$, and $p = \infty$ in $\mathbb{R}^n$ have well-defined expressions:

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$
$$\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$
$$\|x\|_\infty = \max_{1 \le i \le n} |x_i|$$

- Norms can also be applied to vector spaces of matrices. Two common matrix norms include:

  - The Frobenius norm, denoted as:

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

  - The operator norm, which is the supremum of the ratio of the norm of Ax to the norm of x, over all non-zero vectors x. This can be expressed as an inequality:

$$\|Ax\| \le \|A\|_{\text{op}} \|x\|$$

  The operator norm is defined as:

$$\|A\|_{\text{op}} = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}$$

Fundamental Concepts in Vector Spaces: Subspaces and Bases

Definition 5.4: Vector Subspaces

Let V be a vector space over a field $\mathbb{F}$. A subset W of V is called a *subspace* of V if it satisfies the following conditions:

- **Non-Empty:** W is non-empty, i.e., there exists at least one vector in W.
- **Closure under Addition:** For all $\mathbf{u}, \mathbf{v} \in W$, $\mathbf{u} + \mathbf{v} \in W$.
- **Closure under Scalar Multiplication:** For all $\lambda \in \mathbb{F}$ and $\mathbf{u} \in W$, $\lambda \cdot \mathbf{u} \in W$.

If W is a subspace of V, we denote it as $W \subseteq V$. Basically, a subspace is just a subset of a vector space that is itself a vector space.

We typically consider linear transformations in a vector space. One important way to characterize a vector space is through the basis that generates it, which is to be defined:

Definition 5.5: Basis of a Vector Space

Let V be a vector space over a field $\mathbb{F}$. A set of vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ in V is called a *basis* for V if it satisfies the following two conditions:

- **Linear Independence:** The vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ are linearly independent, meaning that the only solution to the equation $c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \ldots + c_n\mathbf{v}_n = \mathbf{0}$ is $c_1 = c_2 = \ldots = c_n = 0$.
- **Spanning the Vector Space:** Any vector $\mathbf{v}$ in V can be expressed as a linear combination of the basis vectors, i.e., there exist scalars $c_1, c_2, \ldots, c_n$ such that $\mathbf{v} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \ldots + c_n\mathbf{v}_n$.

If V has a basis with n vectors, then V is said to be n-dimensional, and the basis is often denoted as $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$.

For example, we can characterize the way a subspace behaves under a linear transformation solely by looking at what its basis vectors map into. The most ubiquitous basis in $\mathbb{R}^n$ is the standard basis $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$, where $\mathbf{e}_i$ has 0 entries everywhere except a 1 at the i'th entry.

Non-Cartesian Coordinate Systems

However, in specific cases, it is very useful to think of geometrically simplfying nonlinear transformations in $\mathbb{R}^2$ and $\mathbb{R}^3$

Definition 5.6: Polar Coordinates in $\mathbb{R}^2$

In 2D polar coordinates, a point P is represented by the pair (r, θ), where:

- r is the distance from the origin to the point P,
- θ is the angle between the positive x-axis and the line segment from the origin to P.

The coordinates (r, θ) are related to the Cartesian coordinates (x, y) by:

$$x = r\cos(\theta)$$
$$y = r\sin(\theta)$$

Definition 5.7: Cylindrical Coordinates in $\mathbb{R}^3$

In 3D cylindrical coordinates, a point P is represented by the triple (r, θ, z), where:

- r is the distance from the z-axis to the point P in the xy-plane,
- θ is the angle between the positive x-axis and the projection of the point P onto the xy-plane,
- z is the vertical distance from the xy-plane to the point P.

The coordinates (r, θ, z) are related to the Cartesian coordinates (x, y, z) by:

$$x = r\cos(\theta)$$
$$y = r\sin(\theta)$$
$$z = z$$

Definition 5.8: Spherical Coordinates in $\mathbb{R}^3$

In 3D spherical coordinates, a point P is represented by the triple (r, θ, ϕ), where:

- r is the distance from the origin to the point P,
- θ is the angle between the positive x-axis and the projection of the point P onto the xy-plane,
- ϕ is the angle between the positive z-axis and the line segment from the origin to P.

The coordinates (r, θ, ϕ) are related to the Cartesian coordinates (x, y, z) by:

$$\begin{aligned} x &= r\sin(\phi)\cos(\theta) \\ y &= r\sin(\phi)\sin(\theta) \\ z &= r\cos(\phi) \end{aligned}$$

The ranges for the coordinates are typically $0 \leq r < \infty$, $0 \leq \theta < 2\pi$, and $0 \leq \phi \leq \pi$.

Lecture 6: Error and Conditioning

Motivation Behind Error Analysis

Let's consider a number written in terms of a central value with its absolute error $R = r \pm E_{abs}(R)$. We can see by considering the possible error that can occur that

$$A \pm B = (a \pm b) \pm (E_{abs}(A) + E_{abs}(B))$$

However, if A and B are such that $a - b$ is very small, say $A = 1.00 \pm 0.004$ and $B = 0.99 \pm 0.004$, which represent 0.4% relative errors, we end up with $A - B = 0.01 \pm 0.008$, which has a huge relative error of 80%. This is an example of **catastrophic cancellation**. Therefore, error analysis can be necessary to make sure any computations made are within rounding error.

2.1 Well-posedness and Condition Number of a Problem

- A numerical problem is well-posed if a small change in the parameters given only results in a small change in the result. Otherwise, it is ill-posed.
- This idea is synonymous to numerical stability, which can be quantified with the relative and absolute condition numbers $K(d)$, where d is the data used by the numerical method.

FORWARD VS BACKWARD ERROR CONDITION NUMBER, ABSOLUTE, RELATIVE.

2.3 A priori and a posteriori Analysis

They are means of evaluating the error induced by a numerical method. I may elaborate on it once I actually understand it. FORWARD VS BACKWARD ERROR

Example 1: Condition Number for Linear Equations

XXX

Example 2: Condition Number for Single-Variable Functions

XXX

Example 3: Condition Number for Multi-Variable Functions

XXX