

Lecture 1: Getting Started

September 2026

Before we get into the new material for this course, it will be important that you're familiar with some of the basic concepts from probability theory and linear algebra that will come up over and over. Indeed, in a very meaningful sense, a high-level goal of this course is to get you comfortable with using more advanced concepts from these areas in your own work going forward.

1 Probability review

We review some of the definitions of the basic concepts. Since we're not doing a graduate level measure theory class, we're not going to worry about getting the most general formulation of these quantities. This section is also deliberately brief because some background in this material is expected going into this class; if the material here is very unfamiliar to you, you may want to reconsider taking this class.

For us, the basic building block will be that of real-valued random variables, which we typically denote X . These are random events which take values over some subset Ω of $\mathbb{R}^d$, where Ω is usually (but not always) finite. A very common case is $\Omega = \{0, 1\}^n$, i.e. the set of n -length strings. As a running example, consider the setting where X is the outcome of n independent, uniformly random coin flips, where “heads” means 1, and tails means 0.

The behavior of X is described uniquely by its associated probability mass function (or probability density function, if Ω is infinite) $\mu : \Omega \rightarrow \mathbb{R}$. When Ω is finite, this function μ must satisfy $\sum_{x \in \Omega} \mu(x) = 1$ and $\mu(x) \geq 0$ for all $x \in \Omega$, and $\mu(x)$ simply denotes the probability that $X = x$. More generally, for any event $E \subseteq \Omega$, $\mu(E)$ is the probability that $X \in E$:

$$\Pr[X \in E] = \mu(E) = \sum_{x \in E} \mu(x) .$$

When Ω is not finite, it's slightly more complicated, but this definition based on events is taken to be the definition of μ , with sums replaced with integrals:

$$\Pr[X \in E] = \mu(E) = \int_{\mathbb{R}^d} \mu(x) dx , \forall E \subseteq \Omega .$$

The fun begins when we realize that we can manipulate these random variables to form new, interesting distributions. For instance, in the running example, instead of considering X , we can also consider $Y = \frac{1}{n} \sum_{i=1}^n X_i$, the average of the coordinates of X . This is now a new random variable over $\{0, 1/n, \dots, (n-1)/n, 1\}$, with its own associated probability measure μ_Y , depending on the original distribution of X .

Moments of a distribution. Recall that some of the most basic statistics of a distribution are encoded in its moments, such as its mean and variance. Formally, if X is a random variable with associated pmf μ , its expectation is

$$\mathbb{E}[X] = \sum_{x \in \Omega} x \cdot \mu(x) .$$

If X is a univariate random variable, that is $\Omega \subseteq \mathbb{R}$, then its variance, denoted $\text{Var}[X]$, is defined as $\mathbb{E}[(X - \mathbb{E}[X])^2]$, i.e. it is the expectation of the auxiliary random variable $Y = (X - \mathbb{E}[X])^2$. More generally, its k -th central moment is $\mathbb{E}[(X - \mathbb{E}[X])^k]$.

Concentration inequalities. With this, we can now state some fundamental concentration inequalities (also referred to as “tail bounds”), which control the deviation of a random variable from its expectation, based off of its properties, such as its lower order moments. We won’t prove these here, but it is a good exercise if you’re unfamiliar with them to try to prove them yourself.

The most basic one is *Markov’s inequality*:

Theorem 1.1 (Markov’s inequality). *Let X be a non-negative random variable. Then for all $t \geq 0$,*

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

The next one we’ll use is *Chebyshev’s inequality*, which bounds the deviation of any random variable from its mean, in terms of its variance, and follows in one line from Markov’s inequality:

Theorem 1.2 (Chebyshev’s inequality). *Let X be a univariate random variable. Then for all $t \geq 0$:*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\text{Var}[X]}{t^2}.$$

Finally, the most powerful, but also the one that requires the most assumptions, of the “standard” concentration inequalities is the *Chernoff bound*, which is actually a family of inequalities concerned with the concentration of an average of a sum of many independent random variables to their expectation. One standard form of this is the following *Chernoff-Hoeffding bound*:

Theorem 1.3 (Hoeffding’s inequality). *Let $X_1, \dots, X_n$ be n independent, identically distributed random variables. Suppose that $|X_i| \leq 1$ for all i , and let $\mu = \mathbb{E}[X_i]$ denote their shared mean. Then, there is some universal constant $C > 0$ so that for all $t \geq 0$, we have:*

$$\Pr \left[\left| \frac{1}{n} \sum_{i=1}^n X_i - \mu \right| \geq t \right] \leq 2 \exp(-Cnt^2).$$

Again, there are many related forms of this inequality, but roughly speaking, what this inequality says is that you get *exponentially fast* concentration of a sum of bounded random variables to their shared mean. Stated simply, averages converge really fast for sums of random coin flips!

The union bound. Finally, another very useful (but also basic) inequality is the *union bound*, which states that the probability of a union of events is bounded by the sum of their probabilities:

Theorem 1.4 (The union bound.). *Let $E_1, \dots, E_k \subseteq \Omega$ be any set of events. Then*

$$\Pr \left[X \in \bigcup_{i=1}^k E_i \right] \leq \sum_{i=1}^k \Pr[X \in E_i].$$

1.1 Application: The shape of high-dimensional data

A recurring theme throughout this class is that probabilistic reasoning will be useful in many settings, algorithmic or otherwise, when *a priori* no randomness is present in the problem. For instance, consider the following basic question:

What is the largest set of n -length $\{0, 1\}$ strings $x_1, \dots, x_m$ I can take, so that for all $i \neq j$, we have that x_i and x_j differ in at least $n/4$ coordinates?

Among other things, this is a basic question in coding theory: if we think of the x_i as codewords, then this set of codewords has the property that unless you corrupt a constant fraction of the coordinates, you can still recover the original codeword. Perhaps surprisingly, it turns out that you can take this set to be *exponentially large*, and a simple proof is to actually just take a random set of strings:

Theorem 1.5. *There exists a set of strings $x_1, \dots, x_m \in \{0, 1\}^n$ with this property of size $m = \exp(\Omega(n))$.*

Proof. As mentioned above, we will do this by demonstrating that random set of strings will satisfy this property with high probability, and so clearly such a set of strings exists. So, let $X_1, \dots, X_m$ be independent, uniformly random strings.

We first bound the probability that for any $i \neq j$, it holds that their Hamming distance is less than $n/4$. In this case, we say that X_i and X_j are a *bad* pair of strings. To do so, we can define auxiliary random variables Z_ℓ , for $\ell = 1, \dots, n$, where $Z_\ell = 1$ if the ℓ -th coordinates of X_i and X_j match, and 0 otherwise. Since X_i and X_j are independent and uniformly random, Z_ℓ is also uniformly random, so its expectation is $1/2$. Clearly, the bad event is if $\sum_{\ell=1}^n Z_\ell \geq 3n/4$, as this means they disagree in less than $n/3$ coordinates. But for fixed i, j , the Z_ℓ are independent bounded random variables, and so by Hoeffding's inequality, slightly re-written:

$$\Pr \left[\left| \sum_{\ell=1}^n Z_\ell - n/2 \right| \geq tn \right] \leq 2 \exp(-Cnt^2) .$$

Notice that the event that $\sum_{\ell=1}^n Z_\ell \geq 3n/4$ implies that $|\sum_{\ell=1}^n Z_\ell - n/2| \geq n/4$, so in particular, by the above, by setting $t = 1/4$, we get that

$$\Pr[(i, j) \text{ is bad}] = \Pr \left[\sum_{\ell=1}^n Z_\ell \geq 3n/4 \right] \leq 2 \exp(-C'n) ,$$

where $C' = C/16$ is some constant. By a union bound over all $\binom{m}{2}$ pairs (i, j) , we conclude:

$$\Pr[\text{there exists a bad pair } (i, j)] \leq \binom{m}{2} \exp(-C'n) \leq 2m^2 \exp(-C'n) .$$

Thus, as long as $m < \exp(C'n/2)/\sqrt{2}$, the probability that there exists a bad pair is less than 1. But the event complement of this event is the event is exactly the event that all the strings differ in at least $n/4$ coordinates, so we know that there must be some set of strings of size $m = \exp(C'n/2)/\sqrt{2} - 1$ which satisfies the desired property, and so we are done. $\square$

As a side note, this sort of proof is known as the *probabilistic method*, and is often a super powerful way to prove existence of objects that might be otherwise hard to construct.

2 Linear Algebra Review

Another key concept that will come up repeatedly and in a very central fashion throughout this course is linear algebra. We review some of the basics here. Recall a vector $v \in \mathbb{R}^d$ is just a d -tuple of numbers, and denotes a point in d -dimensional space. As is conventional, we will typically write v as a column vector.

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_d \end{bmatrix} .$$

We also define the (squared) ℓ_2 norm of a vector v to be $\|v\|_2^2 = \sum_{i=1}^d v_i^2$, and the inner product to be $\langle u_i, u_j \rangle = \sum_{i=1}^d u_i u_j$.

Bases. When we write a vector v as above, the i -th coordinate really means how far v is in the i -th standard basis direction. At the risk of belaboring the point, more explicitly, we can write the *standard basis* vectors

$e_1, \dots, e_d$, given by

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad e_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \dots \quad e_d = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix},$$

so that $v = \sum_{i=1}^d v_i e_i$.

From this point of view there is nothing truly special about the standard basis vectors: one can do this with any *orthonormal* basis $u_1, \dots, u_d$: recall this just means that $\|u_i\|_2^2 = 1$ for all $i = 1, \dots, d$ and $\langle u_i, u_j \rangle = 0$ for all $i \neq j$. Then, we can always write any vector v as $v = \sum_{i=1}^d c_i u_i$ for some unique coefficients c_i . Crucially, one can verify that $\|v\|_2^2 = \sum_{i=1}^d c_i^2$, that is, the ℓ_2 norm of v is still the sum of the squares of the coefficients in *any* basis. For this reason, the ℓ_2 norm of a vector is a *basis independent* property—this notion of basis independence will be very important throughout this class.

Matrices. The next key concept is that of a matrix. In a basic sense, a $d \times k$ matrix is just a collection of dk numbers, organized into a rectangle.

$$M = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1k} \\ M_{21} & M_{22} & \cdots & M_{2k} \\ \vdots & & \ddots & \vdots \\ M_{d1} & M_{d2} & \cdots & M_{dk} \end{bmatrix}.$$

However, as a warning, this is often the wrong way of thinking about matrices! We will see why throughout this class.

You should be familiar with basic concepts such as matrix-vector product, and matrix-matrix product. Just to recap, if $M \in \mathbb{R}^{d \times k}$ and $v \in \mathbb{R}^k$, then $Mv \in \mathbb{R}^d$, and its i -th coordinate is the inner product of the i -th row of M with v . If $A \in \mathbb{R}^{d \times k}$ and $B \in \mathbb{R}^{k \times m}$, then $AB \in \mathbb{R}^{d \times m}$ where the (i, j) -th position of AB is given by the inner product of the i -th row of A with the j -th column of B . While these are all operationally correct definitions, again they are often not the cleanest way to think about what's happening. For instance, another way to think about the matrix product is as a sum of *outer products* (as you'll see on your homework). Concretely, if this means that if we write

$$A = \begin{bmatrix} | & & | \\ a_1 & \cdots & a_k \\ | & & | \end{bmatrix}, \quad B = \begin{bmatrix} - & b_k^\top & - \\ & \vdots & \\ - & b_k^\top & - \end{bmatrix},$$

then we can write

$$AB = \sum_{i=1}^k a_i b_i^\top$$

so that AB can be written as a sum of k *rank-1* matrices.

2.1 Application: Freivalds' algorithm for Matrix Product Verification

Here's another cool application of randomness, this time with matrices. We consider (a special case of) the *matrix product verification* problem: given two matrices $A, B \in \mathbb{R}^{n \times n}$, our goal is to test if $AB = 0$ or not. Clearly, one way to do this is to just compute AB , but this is kind of slow: the best algorithm for matrix multiplication runs in time n^ω , where $\omega < 2.371177$ is the matrix multiplication constant. In particular, to date, we can't do it in time which is linear in the size of the problem, which is n^2 .

However, if we allow randomness, we can do this in linear time (if we allow some error)! The key fact is the following:

Theorem 2.1. *Let $v \in \mathbb{R}^n$ be a nonzero vector, and let z be uniformly random from $\{-1, 1\}^n$. Then*

$$\Pr[\langle v, X \rangle = 0] \leq \frac{1}{2}.$$

Proof. Consider any $x \in \{-1, 1\}^n$ so that $\langle v, x \rangle = 0$; I claim I can associate a unique $x' \in \{-1, 1\}^n$ so that $\langle v, x' \rangle \neq 0$. That way, at most half of the x 's in the hypercube can have zero inner product with v , which yields the claim. The idea is simple: let i be the first coordinate of v which is nonzero, and let $x'_j = x_j$ for all $j \neq i$, and $x'_i = -x_i$. This way, $\langle v, x' \rangle = \langle v, x \rangle - 2x_i v_i \neq 0$. $\square$

As an immediate corollary, we have:

Fact 2.2. *Let C be a nonzero matrix. Then $\Pr[Cz = \vec{0}] \leq 1/2$, where $\vec{0}$ is the all-zeros vector.*

We can now state our matrix product verification algorithm: simply sample $z \sim \{-1, 1\}^n$, and check if $ABz = 0$. Notice that this can be checked in time $O(n^2)$ by first computing $v = Bz$, then computing Av , both of which can be done in time $O(n^2)$. If it is not zero, we can safely say that $AB \neq 0$. But by the above corollary, we know that the probability that $ABz = 0$ when $AB \neq 0$ is at most $1/2$. So, if we repeat this test k times, by independence, the probability we get $ABz = 0$ every time is at most $1/2^k$. Thus, we obtain the following result:

Theorem 2.3. *Let $\delta > 0$. There is a test which runs in time $O(n^2 \log 1/\delta)$ so that if $AB = 0$, the test always passes, but if $AB \neq 0$, the test passes with probability at most δ .*

In other words, if we're okay with a test that passes 99.9999% of the time, we can do this task in nearly-linear time! This is a variant of a method called Freivalds' algorithm [1]. It is actually an open question if it is possible to do this with a deterministic algorithm.

References

- [1] Rūsiņš Freivalds. Fast probabilistic algorithms. In *Mathematical Foundations of Computer Science 1979* (Proc. 8th Symposium, MFCS '79), Lecture Notes in Computer Science 74. Springer, 1979, pp. 57–69.