Review of Sets and Mathematical Notation

A **set** is a well defined collection of objects. These objects are called **elements** or **members** of the set, and they can be anything, including numbers, letters, people, cities, and even other sets. By convention, sets are usually denoted by capital letters and can be described or defined by listing its elements and surrounding the list by curly braces. For example, we can describe the set A to be the set whose members are the first five prime numbers, or we can explicitly write: $A = \{2, 3, 5, 7, 11\}$. If x is an element of A, then we write $x \in A$. Similarly, if y is not an element of A, then we write $y \notin A$. Two sets A and B are said to be equal, written as $A = B$, if they have the same elements. The order and repetition of elements do not matter, so {red, white, blue} = {blue, white, red} = {red, white, white, blue}. Sometimes, more complicated sets can be defined by using a different notation. For example, the set of all rational numbers, denoted by $\mathbb{Q}$, can be written as: $\{\frac{a}{b} \mid a, b \text{ are integers}, b \neq 0\}$. In English, this is read as "the set of all fractions such that the numerator is an integer and the denominator is a non-zero integer."

Cardinality

We can also talk about the size of a set, or its **cardinality**. If $A = \{1,2,3,4\}$, then the cardinality of A, denoted by $|A|$, is 4. It is possible for the cardinality of a set to be 0. This set is called the **empty set**, denoted by the symbol $\emptyset$. A set can also have an infinite number of elements, such as the set of all integers, prime numbers, or odd numbers.

Subsets and Proper Subsets

If every element of a set A is also in set B, then we say that A is a **subset** of B, written $A \subseteq B$. Equivalently we can write $B \supseteq A$, or B is a superset of A. A **proper subset** is a set A that is strictly contained in B, written as $A \subset B$, meaning that A excludes at least one element of B. For example, consider the set $B = \{1,2,3,4,5\}$. Then $\{1,2,3\}$ is both a subset and a proper subset of B, while $\{1,2,3,4,5\}$ is a subset but not a proper subset of B. Here are a few basic properties regarding subsets:

- The empty set, denote by $\{\}$ or $\emptyset$, is a proper subset of any nonempty set A: $\{\} \subset A$.
- The empty set is a subset of every set B: $\{\} \subseteq B$.
- Every set A is a subset of itself: $A \subseteq A$.

Intersections and Unions

The **intersection** of a set A with a set B, written as $A \cap B$, is a set containing all elements which are in both A and B. Two sets are said to be **disjoint** if $A \cap B = \emptyset$. The **union** of a set A with a set B, written as $A \cup B$, is a set of all elements which are in either A or B or both. For example, if A is the set of all positive even numbers, and B is the set of all positive odd numbers, then $A \cap B = \emptyset$, and $A \cup B = \mathbb{Z}^+$, or the set of all positive integers. Here are a few properties of intersections and unions:

- $A \cup B = B \cup A$

- $A \cup \emptyset = A$
- $A \cap B = B \cap A$
- $A \cap \emptyset = \emptyset$

Complements

If A and B are two sets, then the **relative complement** of A in B, written as $B - A$ or $B \backslash A$, is the set of elements in B, but not in A: $B \backslash A = \{x \in B \mid x \notin A\}$. For example, if $B = \{1,2,3\}$ and $A = \{3,4,5\}$, then $B \backslash A = \{1,2\}$. For another example, if $\mathbb{R}$ is the set of real numbers and $\mathbb{Q}$ is the set of rational numbers, then $\mathbb{R} \backslash \mathbb{Q}$ is the set of irrational numbers. Here are some important properties of complements:

- $A \backslash A = \emptyset$
- $A \backslash \emptyset = A$
- $\emptyset \backslash A = \emptyset$

Significant Sets

In mathematics, some sets are referred to so commonly that they are denoted by special symbols. Some of these numerical sets include:

- $\mathbb{N}$ denotes the set of all natural numbers: $\{0,1,2,3,...\}$.
- $\mathbb{Z}$ denotes the set of all integer numbers: $\{\ldots,-2,-1,0,1,2,\ldots\}$.
- $\mathbb{Q}$ denotes the set of all rational numbers: $\{\frac{a}{b} \mid a,b \in \mathbb{Z}, b \neq 0\}$.
- $\mathbb{R}$ denotes the set of all real numbers.
- $\mathbb{C}$ denotes the set of all complex numbers.

In addition, the **Cartesian product** (also called the **cross product**) of two sets A and B, written as $A \times B$, is the set of all pairs whose first component is an element of A and whose second component is an element of B. In set notation, $A \times B = \{(a,b) \mid a \in A, b \in B\}$. For example, if $A = \{1,2,3\}$ and $B = \{u,v\}$, then $A \times B = \{(1,u),(1,v),(2,u),(2,v),(3,u),(3,v)\}$. Given a set S, another significant set is the **power set** of S, denoted by $\mathscr{P}(S)$, is the set of all subsets of S: $\{T \mid T \subseteq S\}$. For example, if $S = \{1,2,3\}$, then the power set of S is: $\mathscr{P}(S) = \{\{\},\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$. It is interesting to note that, if $|S| = k$, then $|\mathscr{P}(S)| = 2^k$.

Mathematical notation:

Sums and Products:

There is a compact notation for writing sums or products of large numbers of items. For example, to write $1+2+\cdots+n$, without having to say dot dot dot, we write it as $\sum_{i=1}^{n} i$. More generally we can write the sum $f(m)+f(m+1)+\cdots+f(n)$ as $\sum_{i=m}^{n} f(i)$. Thus, $\sum_{i=5}^{n} i^2 = 5^2+6^2+\cdots+n^2$.
To write the product $f(m)f(m+1)\cdots f(n)$, we use the notation $\prod_{i=m}^{n} f(i)$. For example, $\prod_{i=1}^{n} i = 1 \cdot 2 \cdots n$.

Universal and existential quantifiers:

Consider the statement: For all natural numbers n, n^2+n+41 is prime. Here, n is quantified to any element of the set $\mathbb{N}$ of natural numbers. In notation, we write $(\forall n \in \mathbb{N})(n^2+n+41 \text{ is prime})$. Here we have used the *universal quantifier* $\forall$ ("for all"). Is the statement true? If you try to substitute small values of n, you will notice that n^2+n+41 is indeed prime for those values. But if you think harder, you can find larger values of n for which it is not prime. Can you find one? So the statement $(\forall n \in \mathbb{N})(n^2+n+41 \text{ is prime})$ is false.

The *existential quantifer* $\exists$ ("there exists") is used in the following statement: $\exists x \in \mathbb{Z}\ x < 2$ *and* $x^2 = 4$. The statement says there is an integer x which is less than 2, but its square is equal to 4. This is a true statement.

We can also write statements using both kinds of quantifiers:

1. $\forall x \in \mathbb{Z} \exists y \in \mathbb{Z}\ \ y > x$
2. $\exists y \in \mathbb{Z} \forall x \in \mathbb{Z}\ y > x$

The first statement says that, given an integer, we can find a larger one. The second statement says something very different: that there is a largest integer! The first statement is true, the second is not.

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Note 1

1 A Brief Introduction

Welcome to Discrete Math and Probability Theory! You might be wondering what you've gotten yourself into — we're delighted to tell you that the answer is something quite remarkable. For as you will find in this course, computer science is a unique field which straddles the fine line between a wealth of research areas: Natural sciences such as physics and chemistry, applied fields such as engineering, and abstract areas such as mathematics. It is precisely this pervasive diversity that not only allows computer science to have something to offer for everyone, but also makes it a driving force of life as we know it today.

So let us begin by dispelling a common myth: What exactly *is* computer science? Is it programming in Java or C++? Networking via the Internet? Or perhaps videogames and troubleshooting personal computers? The answer is that *all* of these are indeed aspects of computer science; they form part of the "engineering" or applied sides of the field, which works to create useful, reliable, working systems. Yet, this is just part of the answer. For at the heart of computer science lies a fundamental core of highly theoretical areas such as artificial intelligence, algorithms, cryptography, and even quantum computing. These areas investigate questions such as: How can we make a robot smart? Find the shortest path between cities on the map? Or send secretly coded messages to one another? It is this theoretical foundation which makes computer science fundamentally possible.

Sound interesting so far? Good. In this course, our goal is to equip you with the basic tools and language you need to move forward in your study of computer science. Which language are we talking about? Why, mathematics, of course! It turns out that the underlying principles of computer science can be described very elegantly using mathematics, particularly in theoretical branches.

With that said, it is quite understandable if aspects of this material seem at first abstract and difficult to master. Yet, do not be discouraged. Any beginning guitarist, for example, dreams of shredding on stage like, say, Jimi Hendrix (or whoever their favorite musician happens to be). To attain this level of mastery, however, first requires the development of basic terminology and skills, such as the ability to read music and to play scales. In this sense, computer science is no different; there are basic mathematical tools which first need to be internalized. Thus, if at any time it should strike you as difficult to appreciate the material in these notes, fret not, and remind yourself that exciting ideas in computer science lie just ahead!

High-level overview of the course. This course can be thought of as consisting of two halves. The first begins with the basic language of mathematics: Logic and proofs. At first, these two topics may seem completely disconnected from anything related to computers; however, it is precisely their study which led to the invention of computers to begin with! Moreover, logic is directly related to digital circuit design, and proofs by induction play a central role in analyzing programs. Indeed, induction is deeply connected to recursion. In this section, developing your ability to write coherent, rigorous proofs will be an important focus. Next, we move on to the study of a special type of arithmetic known as modular arithmetic. The beautiful properties of such numbers leads directly to schemes for computer security, and for storing and transmitting data so that it is immune to noise.

The second half of the course will introduce you to the basic principles of probability theory. This subject plays a crucial role in every aspect of computer science today, from machine learning and artificial intelligence, to dealing with massive data sets, to computer security.

Finally, let us close this introduction by highlighting one of the primary skills this course will allow you to develop: Problem solving. Indeed, one of the hallmarks of discrete mathematics is its wealth of interesting and mind-stretching puzzles and problems. As you work through the homework questions, you will learn how to attack questions, even when at first you may have no idea what the answer might look like. This kind of problem solving ability will greatly help you in the rest of your computer science career, in addition to being a critical skill for many job interviews and an invaluable asset throughout life.

2 Propositional Logic

To begin, we introduce the language of *propositional logic*. What use is this language? It gives us a clean formalism in which to write and prove mathematical statements. In particular, it treats statements in a black or white fashion — either a statement is true, or it is false; there are no "shades of gray" in between. The formal name for this "black or white" behavior is the *law of the excluded middle*, which states that either a statement or its negation must be true.

The basic construct we begin with is the **proposition**. A proposition is simply a statement which is either true or false, such as:

- $\sqrt{3}$ is irrational.
- $1+1=5$.
- Julius Caesar had 2 eggs for breakfast on his 10^{th} birthday.

Given two propositions P (for example, P could stand for "3 is odd") and Q, we can next *combine* them in a number ways to obtain more interesting propositions. Specifically, we introduce the connectives AND, OR, and NOT for accomplishing this below:

1. **Conjunction (AND)**: $P \wedge Q$ (i.e. "P and Q"). True only when both P and Q are true.
2. **Disjunction (OR)**: $P \vee Q$ (i.e. "P or Q"). True when at least one of P and Q is true.
3. **Negation (NOT)**: $\neg P$ (i.e. "not P"). True when P is false.

We call a statement which is *always true*, regardless of the truth values of its variables, a **tautology**. For example, consider $P \vee \neg P$. In contrast, a statement which is *always false* is a **contradiction**; an example is the statement $P \wedge \neg P$.

Sanity check! Let P denote the proposition "$10^2 = 100$" and Q denote "$4+5=49$". What are the values of $P \wedge Q$, $P \vee Q$, and $\neg Q$? (Answer: False, True, True).

By using connectives to combine propositions, we can obtain complicated statements such as

$$\neg((P_1 \vee P_2) \wedge \neg(P_1 \vee P_3)) \vee ((P_2 \vee P_3) \wedge \neg(P_4 \vee P_5)).$$

However, such statements quickly become difficult to digest. Thankfully, there is a simple equivalent representation of propositions called a **truth table**. A truth table is exactly what you might expect: For a given

proposition, we simply list all possible input values for its variables, along with the outputs given those inputs. (The order does not matter.) To illustrate, here are the truth tables for conjunction (AND) and negation ($\neg$), where T and F stand for true and false, respectively:

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

P	$\neg P$
T	F
F	T

Sanity check! What is the truth table for disjunction (OR)?

So far, we introduced the connectives AND, OR, and NOT. Now that we're warmed up, let us introduce a fourth connective, which is a bit trickier but possibly the most important:

4. **Implication (IMPLIES, $\Longrightarrow$)**: $P \Longrightarrow Q$ (i.e. "P implies Q"). This is the same as "If P, then Q."

Intuitively, the way you should think about implication is that it is only false when P is true and Q is false. Formally, here is the truth table:

P	Q	$P \Longrightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

A useful fact to keep in mind is that $P \Longrightarrow Q$ is equivalent to the statement $\neg P \vee Q$.

Sanity check! Write down the truth table for $\neg P \vee Q$. Does it match that for $P \Longrightarrow Q$, as claimed?

2.1 The contrapositive and converse

Thus far, we have introduced the concept of a proposition, along with connectives AND, OR, NOT, and IMPLIES. We now explore two fundamental statements which are closely related to $P \Longrightarrow Q$. They are the contrapositive and the converse:

1. **Contrapositive**: $\neg Q \Longrightarrow \neg P$.
2. **Converse**: $Q \Longrightarrow P$.

One of these is logically equivalent to $P \Longrightarrow Q$, and the other is not. Can you tell which is which? How would you formally *check* your guess? One way is to write down the truth tables and compare them:

P	Q	$\neg P$	$\neg Q$	$P \Longrightarrow Q$	$Q \Longrightarrow P$	$\neg Q \Longrightarrow \neg P$
T	T	F	F	T	T	T
T	F	F	T	F	T	F
F	T	T	F	T	F	T
F	F	T	T	T	T	T

Note that the columns for $P \implies Q$ and $\neg Q \implies \neg P$ match, whereas $Q \implies P$ differs; thus, we say $P \implies Q$ is *logically equivalent* to $\neg Q \implies \neg P$ (its contrapositive). In general, to denote that statements R and S are equivalent, we use notation $R \iff S$, or in English, "R if and only S" (informally abbreviated as R iff S). (Alternatively, one can use the notation $R \equiv S$.)

Having shown that $P \implies Q$ is *equivalent* to its contrapositive, you might ask: Why are we discussing the contrapositive at all? Well, sometimes the contrapositive is easier to prove than the implication itself.

Finally, where does the *converse* $Q \implies P$ fit into all this? Although it looks superficially similar to the contrapositive, the converse is *not* in general equivalent to $P \implies Q$. Rather, the converse plays a very important role in proving the equivalence of two statements via the following identity:

Sanity check! Show that $P \iff Q$ is logically equivalent to $(P \implies Q)$ AND $(Q \implies P)$. (Hint: Write down the truth tables for $P \iff Q$ and $(P \implies Q)$ AND $(Q \implies P)$, check that they match.)

In particular, this identity says that if you wish to prove equivalence of two statements, proceed by showing the implication in each direction. It will prove repeatedly useful in this course, so keep it in mind!

2.2 Quantifiers

Part of the magic of the formal proof systems we'll discuss in this course is the fact that one can prove statements about *infinite* sets of objects with just a *finite* size proof. But wait a second, have we even learned how to formally write down a statement about infinitely many objects yet? Let's see: How would you use propositions to express the statement "For all integers x, x is either even or odd"?

Here's one approach: Let $P(x)$ denote "x is even" and $Q(x)$ denote "x is odd", and consider the propositional form $P(x) \vee Q(x)$. Does this capture the statement we wished to make? *No.* In particular, we've lost something in the translation — the phrase "for all x". Thus, what we need next is language for dealing with such phrases; such tools are called *quantifiers*.

Here are our two quantifiers of interest:

1. **Universal quantifier**: Denoted $\forall$, read "for all".
2. **Existential quantifier**: Denoted $\exists$, read "there exists".

Note that the universal quantifier $\forall$ is precisely what we needed above — in particular, we can now formally write "for all integers x, x is either even or odd" as $(\forall x \in \mathbb{Z})\ (P(x) \vee Q(x))$. The *existential* quantifier works similarly. For example, suppose we wish to say: "There is a prime integer". Then, we could write $(\exists x \in \mathbb{Z})\ (x \text{ is prime})$. Note that both quantifiers implicitly create a notion of a "universe" U over which the statement is made. In both of the examples above, this universe U is just the set of integers, $\mathbb{Z}$.

Sanity check! Use quantifiers to express the following two statements: "For all integers x, $2x+1$ is odd", and "There exists an integer between 2 and 4".

Note now that there is no reason to limit ourselves to just one quantifier per statement. To demonstrate, let's try something a bit more advanced:

1. $(\forall x \in \mathbb{Z})\ (\exists y \in \mathbb{Z})\ (x < y)$
2. $(\exists y \in \mathbb{Z})\ (\forall x \in \mathbb{Z})\ (x < y)$

Can you explain in English what the two statements are saying? The first says that, given an integer, you can always find a larger integer. The second says something very different: That there is a largest integer! (The first statement is true, the second is not.) Note that this example highlights a key point: In general, the order quantifiers are listed in *does* matter!

2.3 Much ado about negation

Let us close this note with a few useful identities involving negations: Just like tricks in, say, basketball such as dribbling between your legs or behind your back, these mathematical tricks will help you put propositional puzzles involving negations to shame.

The four tricks we present are the following identities involving propositions, the first two of which are known as *De Morgan's Laws*. Let $P(x)$ denote a propositional formula with variable x. Then:

- $\neg(P \vee Q) \iff (\neg P) \wedge (\neg Q)$.
- $\neg(P \wedge Q) \iff (\neg P) \vee (\neg Q)$.
- $\neg(\forall x\, P(x)) \iff \exists x\, \neg P(x)$.
- $\neg(\exists x\, P(x)) \iff \forall x\, \neg P(x)$.

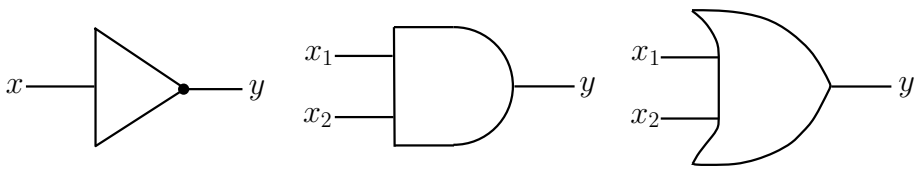
To help develop intuition for these rules, let's illustrate the third rule with an example. Let $P(x)$ denote the statement "x is odd", where we assume x takes values from the universe $U = \mathbb{Z}$. Then, since it is not true that all integers are odd, we have that $\neg(\forall x\, P(x))$ is true. But if all integers are not odd, then there must exist an integer x which is not odd — this is precisely the statement $\exists x\, \neg P(x)$, as desired!

Sanity check! Let $P(x,y)$ be some propositional formula over some universe. Using the rules above, show that $\neg(\forall x\, \exists y\, P(x,y))$ is equivalent to $\exists x\, \forall y\, \neg P(x,y)$. In particular, there is a simple rule this exercise should teach you — each time the negation moves past a quantifier, we "flip" that quantifier (i.e. $\forall$ is flipped to $\exists$ and vice versa).

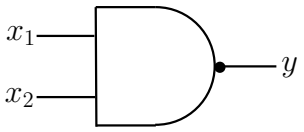
---

3 Exercises

1. Although primitive at first glance, logic is key to fundamental areas of computer science such as digital circuit design. For example, below we depict logic gates used in circuits corresponding respectively to NOT, AND, and OR:

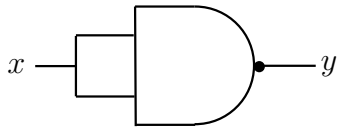


   Here, the wires coming in from the left are the *input* wires, and the wires exiting from the right are the *output* wires. For example, the AND gate takes in input bits x_1 and x_2, and outputs $y = x_1 \wedge x_2$. A remarkable fact regarding circuits is that the NAND gate,



which computes $y = \neg(x_1 \wedge x_2)$ (note the black dot on the output wire, which indicates negation), is *universal*. In other words, each of the other three gates NOT, AND, and OR, can be *simulated* using just the NAND gate! How could this be? Let's work it out for ourselves to find out!

(a) Convince yourself (for example, via truth table), that the following circuit simulates the NOT gate, i.e. that $y = \neg x$:



(b) How would you simulate the AND gate? (Hint: Note that NAND is just AND composed with NOT, and the exercise above just demonstrated how to simulate NOT.)

(c) How would you simulate the OR gate?

2. The rules of logic can also help us reason about *games*. For example, consider a two-player game with the following properties: There are two players, Toby and Fritz, and the game consists of two turns. First, Toby takes a turn, and then Fritz, and then one of them is declared the winner (i.e. there are no ties).

(a) Show that either Toby or Fritz must have a winning strategy in the following sense: The person with the winning strategy *always* wins the game. (Hint: Consider the case where Toby has a winning strategy. Then, we can model this via the proposition "there exists a move by Toby such that for all moves by Fritz, Toby wins." Use the law of the excluded middle to argue that if this statement is false, then its negation must be true — what is the negation of this statement?)

(b) Which property of the game allowed us to apply the law of the excluded middle in part (a) above?

Infinity and Countability

Cardinality

How can we determine whether two sets have the same *cardinality* (or "size")? The answer to this question, reassuringly, lies in early grade school memories: by demonstrating a *pairing* between elements of the two sets. More formally, we need to demonstrate a *bijection* f between the two sets. The bijection sets up a one-to-one correspondence, or pairing, between elements of the two sets. We know how this works for finite sets. In this lecture, we will see what it tells us about *infinite* sets.

Are there more natural numbers $\mathbb{N}$ than there are positive integers $\mathbb{Z}^+$? It is tempting to answer yes, since every positive integer is also a natural number, but the natural numbers have one extra element $0 \notin \mathbb{Z}^+$. Upon more careful observation, though, we see that we can generate a mapping between the natural numbers and the positive integers as follows:

$\mathbb{N}$	0	1	2	3	4	5	...
$\downarrow$		$\searrow$	$\searrow$	$\searrow$	$\searrow$	$\searrow$	$\searrow$
$\mathbb{Z}^+$		1	2	3	4	5	6 ...

Why is this mapping a bijection? Clearly, the function $f : \mathbb{N} \to \mathbb{Z}^+$ is onto because every positive integer is hit. And it is also one-to-one because no two natural numbers have the same image. (The image of n is $f(n) = n+1$, so if $f(n) = f(m)$ then we must have $n = m$.) Since we have shown a bijection between $\mathbb{N}$ and $\mathbb{Z}^+$, this tells us that there are as many natural numbers as there are positive integers! Informally, we have proved that "$\infty + 1 = \infty$."

What about the set of *even* natural numbers $2\mathbb{N} = \{0, 2, 4, 6, ...\}$? In the previous example the difference was just one element. But in this example, there seem to be twice as many natural numbers as there are even natural numbers. Surely, the cardinality of $\mathbb{N}$ must be larger than that of $2\mathbb{N}$ since $\mathbb{N}$ contains all of the odd natural numbers as well. Though it might seem to be a more difficult task, let us attempt to find a bijection between the two sets using the following mapping:

$\mathbb{N}$	0	1	2	3	4	5	...
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$	
$2\mathbb{N}$	0	2	4	6	8	10	...

The mapping in this example is also a bijection. f is clearly one-to-one, since distinct natural numbers get mapped to distinct even natural numbers (because $f(n) = 2n$). f is also onto, since every n in the range is hit: its pre-image is $\frac{n}{2}$. Since we have found a bijection between these two sets, this tells us that in fact $\mathbb{N}$ and $2\mathbb{N}$ have the same cardinality!

What about the set of all integers, $\mathbb{Z}$? At first glance, it may seem obvious that the set of integers is larger

than the set of natural numbers, since it includes negative numbers. However, as it turns out, it is possible to find a bijection between the two sets, meaning that the two sets have the same size! Consider the following mapping:

$0 \to 0, \quad 1 \to -1, \quad 2 \to 1, \quad 3 \to -2, \quad 4 \to 2, \quad \ldots, \quad 124 \to 62, \quad \ldots$

In other words, our function is defined as follows:

$$f(x) = \begin{cases} \frac{x}{2} & \text{if } x \text{ is even} \\ \frac{-(x+1)}{2} & \text{if } x \text{ is odd} \end{cases}$$

We will prove that this function $f : \mathbb{N} \to \mathbb{Z}$ is a bijection, by first showing that it is one-to-one and then showing that it is onto.

Proof (one-to-one): Suppose $f(x) = f(y)$. Then they both must have the same sign. Therefore either $f(x) = \frac{x}{2}$ and $f(y) = \frac{y}{2}$, or $f(x) = \frac{-(x+1)}{2}$ and $f(y) = \frac{-(y+1)}{2}$. In the first case, $f(x) = f(y) \Rightarrow \frac{x}{2} = \frac{y}{2} \Rightarrow x = y$. Hence $x = y$. In the second case, $f(x) = f(y) \Rightarrow \frac{-(x+1)}{2} = \frac{-(y+1)}{2} \Rightarrow x = y$. So in both cases $f(x) = f(y) \Rightarrow x = y$, so f is injective.

Proof (onto): If $y \in \mathbb{Z}$ is non-negative, then $f(2y) = y$. Therefore, y has a pre-image. If y is negative, then $f(-(2y+1)) = y$. Therefore, y has a pre-image. Thus every $y \in \mathbb{Z}$ has a preimage, so f is onto.

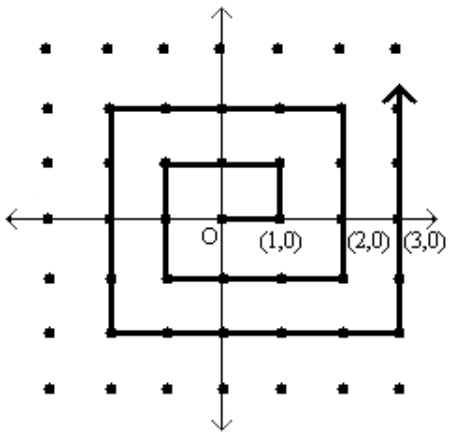
Since f is a bijection, this tells us that $\mathbb{N}$ and $\mathbb{Z}$ have the same size.

Now for an important definition. We say that a set S is **countable** if there is a bijection between S and $\mathbb{N}$ or some subset of $\mathbb{N}$. Thus any finite set S is countable (since there is a bijection between S and the subset $\{0, 1, 2, \ldots, m-1\}$, where $m = |S|$ is the size of S). And we have already seen three examples of countable infinite sets: $\mathbb{Z}^+$ and $2\mathbb{N}$ are obviously countable since they are themselves subsets of $\mathbb{N}$; and $\mathbb{Z}$ is countable because we have just seen a bijection between it and $\mathbb{N}$.

What about the set of all rational numbers? Recall that $\mathbb{Q} = \{\frac{x}{y} \mid x, y \in \mathbb{Z},\ y \neq 0\}$. Surely there are more rational numbers than natural numbers? After all, there are infinitely many rational numbers between any two natural numbers. Surprisingly, the two sets have the same cardinality! To see this, let us introduce a slightly different way of comparing the cardinality of two sets.

If there is a one-to-one function $f : A \to B$, then the cardinality of A is less than or equal to that of B. Now to show that the cardinality of A and B are the same we can show that $|A| \leq |B|$ and $|B| \leq |A|$. This corresponds to showing that there is a one-to-one function $f : A \to B$ and a one-to-one function $g : B \to A$. The existence of these two one-to-one functions implies that there is a bijection $h : A \to B$, thus showing that A and B have the same cardinality. The proof of this fact, which is called the Cantor-Bernstein theorem, is actually quite hard, and we will skip it here.

Back to comparing the natural numbers and the integers. First it is obvious that $|\mathbb{N}| \leq |\mathbb{Q}|$ because $\mathbb{N} \subseteq \mathbb{Q}$. So our goal now is to prove that also $|\mathbb{Q}| \leq |\mathbb{N}|$. To do this, we must exhibit an injection $f : \mathbb{Q} \to \mathbb{N}$. The following picture of a spiral conveys the idea of this injection:



Each rational number $\frac{a}{b}$ (written in its lowest terms, so that $\gcd(a,b) = 1$) is represented by the point (a,b) in the infinite two-dimensional grid shown (which corresponds to $\mathbb{Z} \times \mathbb{Z}$, the set of all pairs of integers). Note that not all points on the grid are valid representations of rationals: e.g., all points on the x-axis have $b = 0$ so none are valid (except for $(0,0)$, which we take to represent the rational number 0); and points such as $(2,8)$ and $(-1,-4)$ are not valid either as the rational number $\frac{1}{4}$ is represented by $(1,4)$. But $\mathbb{Z} \times \mathbb{Z}$ certainly contains all rationals under this representation, so if we come up with an injection from $\mathbb{Z} \times \mathbb{Z}$ to $\mathbb{N}$ then this will also be an injection from $\mathbb{Q}$ to $\mathbb{N}$ (why?).

The idea is to map each pair (a,b) to its position along the spiral, starting at the origin. (Thus, e.g.,$(0,0) \to 0$, $(1,0) \to 1$, $(1,1) \to 2$, $(0,1) \to 3$, and so on.) This mapping certainly maps every rational number to a natural number, because every rational appears somewhere (exactly once) in the grid, and the spiral hits every point in the grid. Why is this mapping an injection? Well, we just have to check that no two rational numbers map to the same natural number. But that is true because no two pairs lie at the same position on the spiral. (Note that the mapping is *not* onto because some positions along the spiral do not correspond to valid representations of rationals; but that is fine.)

This tells us that $|\mathbb{Q}| \leq |\mathbb{N}|$. Since also $|\mathbb{N}| \leq |\mathbb{Q}|$, as we observed earlier, by the Cantor-Bernstein Theorem $\mathbb{N}$ and $\mathbb{Q}$ have the same cardinality.

Our next example concerns the set of all binary strings (of any finite length), denoted $\{0,1\}^*$. Despite the fact that this set contains strings of unbounded length, it turns out to have the same cardinality as $\mathbb{N}$. To see this, we set up a direct bijection $f : \{0,1\}^* \to \mathbb{N}$ as follows. Note that it suffices to *enumerate* the elements of $\{0,1\}^*$ in such a way that each string appears exactly once in the list. We then get our bijection by setting $f(n)$ to be the nth string in the list. How do we enumerate the strings in $\{0,1\}^*$? Well, it's natural to list them in increasing order of length, and then (say) in *lexicographic* order (or, equivalently, numerically increasing order when viewed as binary numbers) within the strings of each length. This means that the list would look like

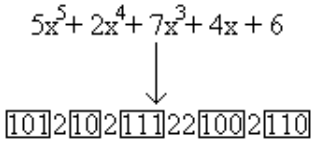
$$\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, 1000, \ldots,$$

where ε denotes the empty string (the only string of length 0). It should be clear that this list contains each binary string once and only once, so we get a bijection with $\mathbb{N}$ as desired.

Our final countable example is the set of all polynomials with natural number coefficients, which we denote $\mathbb{N}(x)$. To see that this set is countable, we will make use of (a variant of) the previous example. Note first that, by essentially the same argument as for $\{0,1\}^*$, we can see that the set of all *ternary* strings $\{0,1,2\}^*$ (that is, strings over the alphabet $\{0,1,2\}$) is countable. To see that $\mathbb{N}(x)$ is countable, it therefore suffices to exhibit an injection $f : \mathbb{N}(x) \to \{0,1,2\}^*$, which in turn will give an injection from $\mathbb{N}(x)$ to $\mathbb{N}$. (It is obvious that there exists an injection from $\mathbb{N}$ to $\mathbb{N}(x)$, since each natural number n is itself trivially a polynomial,

namely the constant polynomial n itself.)

How do we define f? Let's first consider an example, namely the polynomial $p(x) = 5x^5 + 2x^4 + 7x^3 + 4x + 6$. We can list the coefficients of $p(x)$ as follows: $(5,2,7,0,4,6)$. We can then write these coefficients as binary strings: $(101_2, 10_2, 111_2, 0_2, 100_2, 110_2)$. Now, we can construct a ternary string where a "2" is inserted as a separator between each binary coefficient (ignoring coefficients that are 0). Thus we map $p(x)$ to a ternary string as illustrated below:

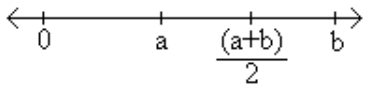


It is easy to check that this is an injection, since the original polynomial can be uniquely recovered from this ternary string by simply reading off the coefficients between each successive pair of 2's. (Notice that this mapping $f : \mathbb{N}(x) \to \{0,1,2\}^*$ is not onto (and hence not a bijection) since many ternary strings will not be the image of any polynomials; this will be the case, for example, for any ternary strings that contain binary subsequences with leading zeros.)

Hence we have an injection from $\mathbb{N}(x)$ to $\mathbb{N}$, so $\mathbb{N}(x)$ is countable.

Cantor's Diagonalization

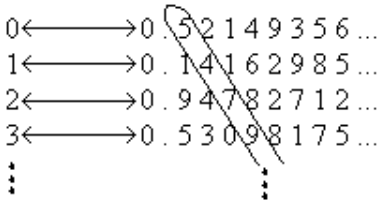
So we have established that $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ all have the same cardinality. What about the real numbers, the set of all points on the real line? Surely they are countable too? After all, the rational numbers, like the real numbers, are dense (i.e., between any two rational numbers there is a rational number):



In fact, between any two *real* numbers there is always a rational number. It is really surprising, then, that there are more real numbers than rationals. That is, there is no bijection between the rationals (or the natural numbers) and the reals. In fact, we will show something even stronger: even the real numbers in the interval $[0,1]$ are uncountable!

Recall that a real number can be written out in an infinite decimal expansion. A real number in the interval $[0,1]$ can be written as $0.d_1d_2d_3...$ Note that this representation is not unique; for example, $1 = 0.999...$[1]; for definiteness we shall assume that every real number is represented as a recurring decimal where possible (i.e., we choose the representation $.999\ldots$ rather than 1).

Cantor's Diagonalization Proof: Suppose towards a contradiction that there is a bijection $f : \mathbb{N} \to \mathbb{R}[0,1]$. Then, we can enumerate the infinite list as follows:



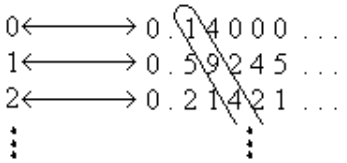
The number circled in the diagonal is some real number $r = 0.5479\ldots$, since it is an infinite decimal expan-

[1] To see this, write $x = .999\ldots$. Then $10x = 9.999\ldots$, so $9x = 9$, and thus $x = 1$.

sion. Now consider the real number s obtained by modifying every digit of r, say by replacing each digit d with $d+2$ mod 10; thus in our example above, $s = 0.7691\ldots$. We claim that s does not occur in our infinite list of real numbers. Suppose for contradiction that it did, and that it was the n^{th} number in the list. Then r and s differ in the n^{th} digit: the n^{th} digit of s is the n^{th} digit of r plus 2 mod 10. So we have a real number s that is not in the range of f. But this contradicts the assertion that f is a bijection. Thus the real numbers are not countable.

Let us remark that the reason that we modified each digit by adding 2 mod 10 as opposed to adding 1 is that the same real number can have two decimal expansions; for example $0.999... = 1.000....$ But if two real numbers differ by more than 1 in any digit they cannot be equal. Thus we are completely safe in our assertion.

With Cantor's diagonalization method, we proved that $\mathbb{R}$ is uncountable. What happens if we apply the same method to $\mathbb{Q}$, in a (futile) attempt to show the rationals are uncountable? Well, suppose for a contradiction that our bijective function $f : \mathbb{N} \to \mathbb{Q}[0,1]$ produces the following mapping:

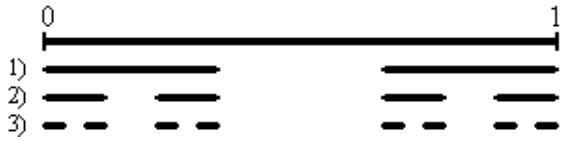


This time, let us consider the number q obtained by modifying every digit of the diagonal, say by replacing each digit d with $d+2$ mod 10. Then in the above example $q = 0.316...$, and we want to try to show that it does not occur in our infinite list of rational numbers. However, we do not know if q is rational (in fact, it is extremely unlikely for the decimal expansion of q to be periodic). This is why the method fails when applied to the rationals. When dealing with the reals, the modified diagonal number was guaranteed to be a real number.

The Cantor Set (Optional)

Please read on only if interested.

The Cantor set is a remarkable set construction involving the real numbers in the interval $[0,1]$. The set is defined by repeatedly removing the middle thirds of line segments infinitely many times, starting with the original interval. For example, the first iteration would involve the removal of the interval $(\frac{1}{3}, \frac{2}{3})$, leaving $[0, \frac{1}{3}] \cup [\frac{2}{3}, 1]$. The first three iterations are illustrated below:



The Cantor set contains all points that have *not* been removed: $C = \{x : x \text{ not thrown out}\}$. How much of the original unit interval is left after this process is repeated infinitely? Well, we start with 1, and after the first iteration we remove $\frac{1}{3}$ of the interval, leaving us with $\frac{2}{3}$. For the second iteration, we keep $\frac{2}{3} \times \frac{2}{3}$ of the original interval. As we repeat the iterations infinitely, we are left with:

$$1 \longrightarrow \frac{2}{3} \longrightarrow \frac{2}{3} \times \frac{2}{3} \longrightarrow \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} \longrightarrow \cdots \longrightarrow \lim_{n \to \infty} (\frac{2}{3})^n = 0$$

According to the calculations, we have removed everything from the original interval! Does this mean that the Cantor set is empty? No, it doesn't. What it means is that the *measure* of the Cantor set is zero; the Cantor set consists of isolated points and does not contain any non-trivial intervals. In fact, not only is the

Cantor set non-empty, it is uncountable![2]

To see why, let us first make a few observations about ternary strings. In ternary notation, all strings consist of digits (called "trits") from the set $\{0,1,2\}$. All real numbers in the interval $[0,1]$ can be written in ternary notation. (E.g., $\frac{1}{3}$ can be written as 0.1, or equivalently as $0.0222\ldots$, and $\frac{2}{3}$ can be written as 0.2 or as $0.1222\ldots$.) Thus, in the first iteration, the middle third removed contains all ternary numbers of the form 0.1xxxxx. The ternary numbers left after the first removal can all be expressed either in the form 0.0xxxxx... or 0.2xxxxx... (We have to be a little careful here with the endpoints of the intervals; but we can handle them by writing $\frac{1}{3}$ as $0.02222\ldots$ and $\frac{2}{3}$ as 0.2.) The second iteration removes ternary numbers of the form 0.01xxxxx and 0.21xxxxx (i.e., any number with 1 in the second position). The third iteration removes 1's in the third position, and so on. Therefore, what remains is all ternary numbers with only 0's and 2's. Thus we have shown that

$$C = \{x \in [0,1] : x \text{ has a ternary representation consisting only of 0's and 2's}\}.$$

Finally, using this characterization, we can set up an *onto* map f from C to $[0,1]$. Since we already know that $[0,1]$ is uncountable, this implies that C is uncountable also. The map f is defined as follows: for $x \in C$, $f(x)$ is defined as the binary decimal obtained by dividing each digit of the ternary representation of x by 2. Thus, for example, if $x = 0.0220$ (in ternary), then $f(x)$ is the binary decimal 0.0110). But the set of all binary decimals $0.xxxxx\ldots$ is in 1-1 correspondence with the real interval $[0,1]$, and the map f is onto because every binary decimal is the image of some ternary string under f (obtained by doubling every binary digit).[3] This completes the proof that C is uncountable.

Power Sets and Higher Orders of Infinity (Optional)

Please read on only if interested.
Let S be any set. Then the *power set* of S, denoted by $\mathscr{P}(S)$, is the set of all subsets of S. More formally, it is defined as: $\mathscr{P}(S) = \{T : T \subseteq S\}$. For example, if $S = \{1,2,3\}$, then $\mathscr{P}(S) = \{\{\}, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$.

What is the cardinality of $\mathscr{P}(S)$? If $|S| = k$ is finite, then $|\mathscr{P}(S)| = 2^k$. To see this, let us think of each subset of S corresponding to a k bit string. In the example above the subset $\{1,3\}$ corresponds to the string 101. A 1 in the i^{th} position indicates that the i^{th} element of S is in the subset and a 0 indicates that it is not. Now the number of binary strings of length k is 2^k, since there are two choices for each bit position. Thus $|\mathscr{P}(S)| = 2^k$. So for finite sets S, the cardinality of the power set of S is exponentially larger than the cardinality of S. What about infinite (countable) sets? We claim that there is no bijection from S to $\mathscr{P}(S)$, so $\mathscr{P}(S)$ is not countable. Thus for example the set of all subsets of natural numbers is not countable, even though the set of natural numbers itself is countable.

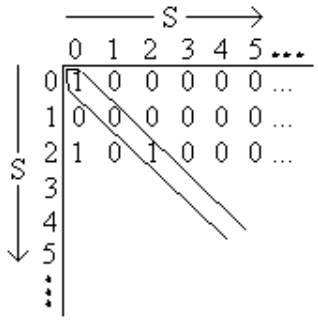
Theorem: Let S be countably infinite. Then $|\mathscr{P}(S)| > |S|$.

Proof: Suppose towards a contradiction that there is a bijection $f : S \to \mathscr{P}(S)$. Recall that we can represent a subset by a binary string, with one bit for each element of S. (So, since S is infinite, the string will be infinitely long. Contrast the case of $\{0,1\}^*$ discussed earlier, which consists of all binary strings of *finite*

[2] It's actually easy to see that C contains at least countably many points, namely the endpoints of the intervals in the construction—i.e., numbers such as $\frac{1}{3}$, $\frac{2}{3}$, $\frac{1}{9}$, $\frac{1}{27}$ etc. It's less obvious that C also contains various other points, such as $\frac{1}{4}$ and $\frac{3}{10}$. (Why?)

[3] Note that f is *not* injective; for example, the ternary strings $0.20222\ldots$ and 0.22 map to binary strings $0.10111\ldots$ and 0.11 respectively, which denote the same real number. Thus f is not a bijection. However, the current proof shows that the cardinality of C is at least that of $[0,1]$, while it is obvious that the cardinality of C is at most that of $[0,1]$ since $C \subset [0,1]$. Hence C has the same cardinality as $[0,1]$ (and as $\mathbb{R}$).

length.) Consider the following diagonalization picture in which the function f maps natural numbers x to binary strings which correspond to subsets of S (e.g., $2 \to 10100... = \{0,2\}$):



In this case, we have assigned the following mapping: $0 \to \{0\}$, $1 \to \{\}$, $2 \to \{0,2\}$, ... (i.e., the nth row describes the n^{th} subset as follows: if there is a 1 in the k^{th} column, then k is in this subset, else it is not.) Using a similar diagonalization argument to the earlier one, flip each bit along the diagonal: $1 \to 0$, $0 \to 1$, and let b denote the resulting binary string. First, we must show that the new element is a subset of S. Clearly it is, since b is an infinite binary string which corresponds to a subset of S. Now suppose b were the n^{th} binary string. This cannot be the case though, since the n^{th} bit of b differs from the n^{th} bit of the diagonal (the bits are flipped). So it's not on our list, but it should be, since we assumed that the list enumerated all possible subsets of S. Thus we have a contradiction, implying that $\mathscr{P}(S)$ is uncountable.

Thus we have seen that the cardinality of $\mathscr{P}(\mathbb{N})$ (the power set of the natural numbers) is strictly larger than the cardinality of $\mathbb{N}$ itself. The cardinality of $\mathbb{N}$ is denoted $\aleph_0$ (pronounced "aleph null"), while that of $\mathscr{P}(\mathbb{N})$ is denoted $2^{\aleph_0}$. It turns out that in fact $\mathscr{P}(\mathbb{N})$ has the same cardinality as $\mathbb{R}$ (the real numbers), and indeed as the real numbers in $[0,1]$. This cardinality is known as $\mathbf{c}$, the "cardinality of the continuum." So we know that $2^{\aleph_0} = \mathbf{c} > \aleph_0$. Even larger infinite cardinalities (or "orders of infinity"), denoted $\aleph_1, \aleph_2, \ldots$, can be defined using the machinery of set theory; these obey (to the uninitiated somewhat bizarre) rules of arithmetic. Several fundamental questions in modern mathematics concern these objects. For example, the famous "continuum hypothesis" asserts that $\mathbf{c} = \aleph_1$ (which is equivalent to saying that there are no sets with cardinality between that of the natural numbers and that of the real numbers).

Self-Reference and Computability

In this lecture we will explore the deep connection between proofs and computation. At the heart of this connection is the notion of self-reference, and it has far-reaching consequences to the limit of computations (the Halting Problem) and the foundation of logic in mathematics (Gödel's incompleteness theorem).

The Liar's Paradox

Recall that propositions are statements that are either true or false. We saw before that some statements are not well defined or too imprecise to be called propositions. But here is a statement that is problematic for more subtle reasons:

"All Cretans are liars,"

so said a Cretan in antiquity, thus giving rise to the so-called liar's paradox which has amused and confounded people over the centuries. Why? Because if the statement above is true, then the Cretan was lying, which implies the statement is false. But actually the above statement isn't really a paradox; it simply yields a contradiction if we assume it is true, but if it is false then there is no problem.

A true formulation of this paradox is the following statement:

"This statement is false."

Is the statement above true? If the statement is true, then what it asserts must be true; namely that it is false. But if it is false, then it must be true. So it really is a paradox, and we see that it arises because of the self-referential nature of the statement. Around a century ago, this paradox found itself at the center of foundational questions about mathematics and computation.

We will now study how this paradox relates to computation. Before doing so, let us consider another manifestation of the paradox, created by the great logician Bertrand Russell. In a village with just one barber, every man keeps himself clean-shaven. Some of the men shave themselves, while others go to the barber. The barber proclaims:

"I shave all and only those men who do not shave themselves."

It seems reasonable then to ask the question: Does the barber shave himself? Thinking more carefully about the question though, we see that we are presented with the same self-referential paradox: a logically impossible scenario. If the barber does not shave himself, then according to what he announced, he shaves himself. If the barber does shave himself, then according to his statement he does not shave himself!

Self-Replicating Programs

Can we use self-reference to design a program that outputs itself? To illustrate the idea, let us consider how we can do this if we could write the program in English. Consider the following instruction:

```
Print out the following sentence:
    "Print out the following sentence:"
```

If we execute the instruction above (interpreting it as a program), then we will get the following output:

> Print out the following sentence:

Clearly this is not the same as the original instruction above, which consists of two lines. We can try to modify the instruction as follows:

```
Print out the following sentence twice:
    "Print out the following sentence twice:"
```

Executing this modified instruction yields the output which now consists of two lines:

> Print out the following sentence twice:
>
> Print out the following sentence twice:

This almost works, except that we are missing the quotes in the second line. We can fix it by modifying the instruction as follows:

```
Print out the following sentence twice, the second time in quotes:
    "Print out the following sentence twice, the second time in quotes:"
```

Then we see that when we execute this instruction, we get exactly the same output as the instruction itself:

> Print out the following sentence twice, the second time in quotes:
>
> "Print out the following sentence twice, the second time in quotes:"

Quines and the Recursion Theorem

In the above section we have seen how to write a self-replicating program in English. But can we do that in a real programming language? In general, a program that prints itself is called a *quine*,[1] and it turns out we can always write quines in any programming language.

As another example, consider the following pseudocode:

```
(Quine "s")
    (s "s")
```

The pseudocode above defines a program `Quine` that takes a string `s` as input, and outputs `(s "s")`, which means we run the string `s` (now interpreted as a program) on itself. Now consider executing the program `Quine` with input `"Quine"`:

```
(Quine "Quine")
```

By definition, this will output

> (Quine "Quine")

which is the same as the instruction that we executed!

This is a simple example, but how do we construct quines in general? The answer is given by the *recursion theorem*, which you can find in the extra credit problem in Homework 8. The recursion theorem states that

[1] Quine is named after the philosopher and logician Willard Van Orman Quine, as popularized in the book *"Gödel, Escher, Bach: An Eternal Golden Braid"* by Douglas Hofstadter.

given any program $P(x,y)$, we can always convert it to another program $Q(x)$ such that $Q(x) = P(x,Q)$, i.e., Q behaves exactly as P would if its second input is the description of the program Q. In this sense we can think of Q as the "self-aware" version of P, since Q essentially has access to its own description.

The Halting Problem

Are there tasks that a computer cannot perform? For example, we would like to ask the following basic question when compiling a program: does it go into an infinite loop? In 1936, Alan Turing showed that there is no program that can perform this test. The proof of this remarkable fact is very elegant and combines two ingredients: self-reference (as in the liar's paradox), and the fact that we cannot separate programs from data. In computers, a program is represented by a string of bits just as integers, characters, and other data are. The only difference is in how the string of bits is interpreted.

We will now examine the Halting Problem. Given the description of a program and its input, we would like to know if the program ever halts when it is executed on the given input. In other words, we would like to write a program `TestHalt` that behaves as follows:

$$\texttt{TestHalt(P,x)} = \begin{cases} \text{"yes"}, & \text{if program } P \text{ halts on input } x \\ \text{"no"}, & \text{if program } P \text{ loops on input } x \end{cases}$$

Why can't such a program exist? First, let us use the fact that a program is just a bit string, so it can be input as data. This means that it is perfectly valid to consider the behavior of `TestHalt(P,P)`, which will output "yes" if P halts on P, and "no" if P loops forever on P. We now prove that such a program cannot exist.

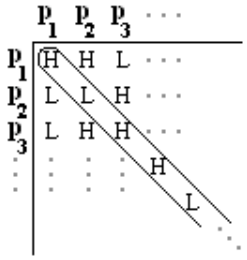
Proof: Define the program

```
Turing(P)
  if TestHalt(P,P) = "yes" then loop forever
  else halt
```

So if the program P when given P as input halts, then `Turing(P)` loops forever; otherwise, `Turing(P)` halts. Assuming we have the program `TestHalt`, we can easily use it as a subroutine in the above program `Turing`.

Now let us look at the behavior of `Turing(Turing)`. There are two cases: either it halts, or it does not. If `Turing(Turing)` halts, then it must be the case that `TestHalt(Turing,Turing)` returned "no." But by definition of `TestHalt`, that would mean that `Turing(Turing)` should not have halted. In the second case, if `Turing(Turing)` does not halt, then it must be the case that `TestHalt(Turing, Turing)` returned "yes," which would mean that `Turing(Turing)` should have halted. In both cases, we arrive at a contradiction which must mean that our initial assumption, namely that the program `TestHalt` exists, was wrong. Thus, `TestHalt` cannot exist, so it is impossible for a program to check if any general program halts. □

What proof technique did we use? This was actually a proof by diagonalization, the same technique that we used in the previous lecture to show that the real numbers are uncountable. Why? Since the set of all computer programs is countable (they are, after all, just finite-length strings over some alphabet, and the set of all finite-length strings is countable), we can enumerate all programs as follows (where P_i represents the i^{th} program):



The $(i, j)^{\text{th}}$ entry in the table above is H if program P_i halts on input P_j, and L if it does not halt. Now if the program `Turing` exists it must occur somewhere on our list of programs, say as P_n. But this cannot be, since if the n^{th} entry in the diagonal is H, meaning that P_n halts on P_n, then by its definition `Turing` loops on P_n; and if the entry is L, then by definition `Turing` halts on P_n. Thus the behavior of `Turing` is different from that of P_n, and hence `Turing` does not appear on our list. Since the list contains all possible programs, we must conclude that the program `Turing` does not exist. And since `Turing` is constructed by a simple modification of `TestHalt`, we can conclude that `TestHalt` does not exist either. Hence the Halting Problem cannot be solved.

In fact, there are many more cases of questions we would like to answer about a program, but cannot. For example, we cannot know if a program ever outputs anything or if it ever executes a specific line. We also cannot check if two programs produce the same output. We cannot even check to see if the program is a virus. These issues are explored in greater detail in the advanced course CS172 (Computability and Complexity).

The Easy Halting Problem

As noted above, the key idea in establishing the uncomputability of the Halting Problem is self-reference: Given a program P, we want to check whether $P(P)$ halts. But in practice, how often do we want to execute a program with its own description as input? Is it possible that if we disallow this kind of self-reference, we can answer the Halting Problem?

That is, given a program P, what if we ask: "Does P halts on input 0?" This looks easier than the Halting Problem (hence the name Easy Halting Problem), since we only need to check whether P halts on a specific input 0, instead of an arbitrary given input. However, it turns out this easier problem is still uncomputable. We prove this claim by showing that if we can solve the Easy Halting Problem, then we can also solve the Halting Problem; since we know the Halting Problem is uncomputable, this implies the Easy Halting Problem must also be uncomputable.

Specifically, suppose we have a program `TestEasyHalt` that answers the Easy Halting Problem:

$$\texttt{TestEasyHalt(P)} = \begin{cases} \text{"yes"}, & \text{if program } P \text{ halts on input } 0 \\ \text{"no"}, & \text{if program } P \text{ loops on input } 0 \end{cases}$$

Then we can use `TestEasyHalt` as a subroutine in the following algorithm that solves the Halting Problem:

```
Halt(P,x)
  define a program P' that returns P(x) on input 0
  return TestEasyHalt(P')
```

The algorithm `Halt` constructs another program P', which depends on both the original program P and the original input x, such that when we call $P'(0)$ we return $P(x)$. An example of such a program P' can simply be:

```
P'(y)
   return P(x)
```

That is, the new program P' ignores the input y and always returns $P(x)$. Then we see that $P'(0)$ halts if and only if $P(x)$ halts. Therefore, if we have such a program `TestEasyHalt`, then `Halt` will correctly answer the Halting Problem. Since we know there cannot be such a program `Halt`, we conclude `TestEasyHalt` does not exist either.

The technique that we use here is called a *reduction*. Here we are reducing one problem "Does P halt on x?" into another problem "Does P' halt on 0?", in the sense that if we know how to answer the second problem, then we can use that knowledge to construct an answer for the first problem. This implies that the second problem is actually as difficult as the first, despite the apparently simpler description of the second problem.

Godel's Incompleteness Theorem

In 1900, the great mathematician David Hilbert posed the following two questions about the foundation of logic in mathematics:

1. Is arithmetic consistent?
2. Is arithmetic complete?

To understand the questions above, we recall that mathematics is a formal system based on a list of axioms (for example, Peano's axioms of the natural numbers, axiom of choice, etc.) together with rules of inference. The axioms provide the initial list of true statements in our system, and we can apply the rules of inference to prove other true statements, which we can again use to prove other statements, and so on.

The first question above asks whether it is possible to prove both a proposition P and its negation $\neg P$. If this is the case, then we say that arithmetic is *inconsistent*; otherwise, we say arithmetic is *consistent*. If arithmetic is inconsistent, meaning there are false statements that can be proved, then the entire arithmetic system will collapse because from a false statement we can deduce anything, so every statement in our system will be vacuously true.

The second question above asks whether every true statement in arithmetic can be proved. If this is the case, then we say that arithmetic is *complete*. We note that given a statement, which is either true or false, it can be very difficult to prove which one it is. As a real-world example, consider the following statement, which is known as Fermat's Last Theorem:

$$\forall n \geq 3, \quad \nexists x, y, z \in \mathbb{Z}, \ x^n + y^n = z^n.$$

This theorem was first stated by Pierre de Fermat in 1637,[2] but it has eluded proofs for centuries until it was finally proved by Andrew Wiles in 1994.

In 1928, Hilbert formally posed the questions above as the Entscheidungsproblem. Most people believed that the answer would be "yes," since ideally arithmetic should be both consistent and complete. However, in 1930 Kurt Gödel proved that the answer is in fact "no": Any formal system that is sufficiently rich to formalize computation is either inconsistent (there are false statements that can be proved) or incomplete (there are true statements that cannot be proved). Gödel proved his result by exploiting the deep connection between proofs and computation, but it was not until 1936 that Turing formalized the definition of computation via the notion of Turing machines and computability.

In the rest of this note, we will first understand the essence of Gödel's proof, and then we will provide an easier proof using the Halting Problem.

[2] Along with the famous note: "I have discovered a truly marvelous proof of this, which this margin is too narrow to contain."

Sketch of Gödel's Proof

Suppose we have a formal system F, which consists of a list of axioms and rules of inference, and assume F is sufficiently expressive that we can use it to express computation. Arithmetic is an example of such a system F.

Now suppose we can write a sentence:

$$S(F) = \text{“This sentence is not provable in } F\text{.”}$$

Once we have this sentence, there are two possibilities:

1. Case 1: $S(F)$ is provable. Then the sentence $S(F)$ is true, but by inspecting the content of the sentence itself, we see that this implies $S(F)$ should not be provable. Thus, F is inconsistent in this case.
2. Case 2: $S(F)$ is not provable. By construction, this means the sentence $S(F)$ is true. Thus, F is incomplete in this case, since there is a true statement (namely, $S(F)$) that is not provable.

To complete the proof, it now suffices to construct such a sentence $S(F)$. This is the difficult part of Gödel's proof, which requires a clever encoding (Gödel numbering) of symbols and propositions as natural numbers.

Proof via the Halting Problem

Let us now see how we can prove Gödel's result by reduction to the Halting Problem. Here we proceed by contradiction: Suppose arithmetic is both consistent and complete; we will use this assumption to solve the Easy Halting Problem, which we have seen is as difficult as the Halting Problem.

Recall that in the Easy Halting Problem we want to decide whether a given program P halts on input 0. Let S denote the sentence "P halts on input 0." This is a sentence in arithmetic, and because we assume arithmetic is consistent and complete, we know that S is either true or false, and there is a proof either way.

Recall also that a proof is simply a finite binary string. How many proofs are there? There are countably many, so we can enumerate them and go through them one by one. Now define the program M as follows:

```
M(P)
  for every proof x:
    if x is a proof that P halts on 0, then output "yes"
    if x is a proof that P does not halt on 0, then output "no"
```

The program M takes as input the program P, and proceed to check every possible proof until it finds one that proves either $P(0)$ halts, or $P(0)$ does not halt. By assumption, we know there is always such a proof, so the algorithm M will terminate in finite time, and it will correctly answer the Easy Halting Problem. Since we have established that there cannot be such a program M, our initial assumption must be wrong, so it is not true that arithmetic is both consistent and complete.

Note that here we rely on the fact that given a proof, we can have a program that mechanistically check whether it is a valid proof for our proposition, and the argument above shows how we can use this to answer questions about the limit of computability. This is a manifestation of the intimate ties between proofs and computation in a deep level.

Counting

The next major topic of the course is probability theory. Suppose you toss a fair coin a thousand times. How likely is it that you get exactly 500 heads? And what about 1000 heads? It turns out that the chances of 500 heads are roughly 2.5%, whereas the chances of 1000 heads are so infinitesimally small that we may as well say that it is impossible. But before you can learn to compute or estimate odds or probabilities you must learn to count! That is the subject of this note.

We start by considering a simple scenario. We pick k elements out of an n element set $S = \{1, 2, \cdots, n\}$ one at a time. We wish to count the number of different ways to do this, taking into account the order in which the elements are picked. For example, when $k = 2$, picking 1 and then 2 is considered a different outcome from picking 2 followed by 1. Another way to ask the question is this: we wish to form an ordered sequence of k distinct elements, where each element is picked from the set S. How many different such ordered sequences are there?

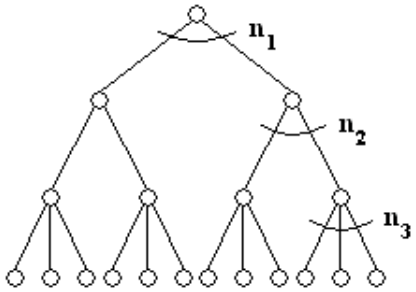
If we were dealing cards, the set would be $S = \{1, \cdots, 52\}$, where each number represents a card in a deck of 52 cards. Picking an element of S in this case refers to dealing one card. Note that once a card is dealt, it is no longer in the deck and so it cannot be dealt again. So the hand of k cards that are dealt consists of k distinct elements from the set S.

For the first card, it is easy to see that we have 52 distinct choices. But now the available choices for the second card depend upon what card we picked first. The crucial observation is that regardless of which card we picked first, there are exactly 51 choices for the second card. So the total number of ways of choosing the first two cards is 52×51. Reasoning in the same way, there are exactly 50 choices for the third card, no matter what our choices for the first two cards. It follows that there are exactly $52 \times 51 \times 50$ sequences of three cards. In general, the number of sequences of k cards is $52 \cdot 51 \cdots (52 - (k-1))$.

This is an example of the first rule of counting:

First Rule of Counting: If an object can be made by a succession of k choices, where there are n_1 ways of making the first choice, and *for every* way of making the first choice there are n_2 ways of making the second choice, and *for every* way of making the first and second choice there are n_3 ways of making the third choice, and so on up to the n_k-th choice, then the total number of distinct objects that can be made in this way is the product $n_1 \cdot n_2 \cdot n_3 \cdots n_k$.

Here is another way of picturing the first rule of counting. Consider the following tree:



It has branching factor n_1 at the root, n_2 at every node at the second level,..., n_k at every node at the k-th level. Each node at level $k+1$ (a leaf node) represents one possible way of making the object by making a succession of k choices. So the number of distinct objects that can be made is equal to the number of leaves in the tree. Moreover, the number of leaves in the tree is the product $n_1 \cdot n_2 \cdot n_3 \cdots n_k$. For example, if $n_1 = 2$, $n_2 = 2$, and $n_3 = 3$, then there are 12 leaves (i.e., outcomes).

Counting Sets

Consider a slightly different question. We would like to pick k distinct elements of $S = \{1, 2, \cdots, n\}$ (i.e. without repetition), but we do not care about the order in which we picked the k elements. For example, picking elements $1, \ldots, k$ is considered the same outcome as picking elements $2, \ldots, k$ and picking 1 as the last (k^{th} element). Now how many ways are there to choose these elements?

When dealing a hand of cards, say a poker hand, it is often more natural to count the number of distinct hands (i.e., the set of 5 cards dealt in the hand), rather than the order in which they were dealt. As we've seen in the section above, if we are considering order, there are $52 \cdot 51 \cdot 50 \cdot 49 \cdot 48 = \frac{52!}{47!}$ outcomes. But how many distinct hands of 5 cards are there? Here is another way of asking the question: each such 5 card hand is just a subset of S of cardinality 5. So we are asking how many 5 element subsets of S are there?

Here is a clever trick for counting the number of distinct subsets of S with exactly 5 elements. Create a bin corresponding to each such 5 element subset. Now take all the sequences of 5 cards and distribute them into these bins in the natural way. Each sequence gets placed in the bin corresponding to the set of 5 elements in the sequence. Thus if the sequence is $(2, 1, 3, 5, 4)$, then it is placed in the bin labeled $\{1, 2, 3, 4, 5\}$. How many sequences are placed in each bin? The answer is exactly 5!, since there are exactly 5! different ways to order 5 cards.

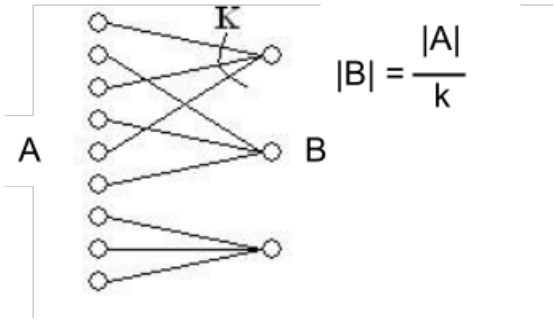
Recall that our goal was to compute the number of 5 element subsets, which now corresponds to the number of bins. We know that there are $\frac{52!}{47!}$ 5-card sequences, and there are 5! sequences placed in each bin. The total number of bins is therefore $\frac{52!}{47!5!}$.

This quantity $\frac{n!}{(n-k)!k!}$ is used so often that there is special notation for it: $\binom{n}{k}$, pronounced *n choose k*. This is the number of ways of picking k distinct elements from S, where the order of placement does not matter. Equivalently, it's the number of ways of choosing k objects out of a total of n objects, where the order of the choices does not matter.

The trick we used above is actually our second rule of counting:

Second Rule of Counting: Assume an object is made by a succession of choices, and the order in which the choices is made does not matter. Let A be the set of ordered objects and let B be the set of unordered objects. If there exists a k to 1 function f from A to B, we can count the number of ordered objects (pretending that the order matters) and divide by k (the number of ordered objects per unordered objects) to obtain $|B|$, the number of unordered objects.

Note that we are assuming the number of ordered objects is the same for every unordered object; the rule cannot be applied otherwise. Here is another way of picturing the second rule of counting:



The function f simply places the ordered outcomes into bins corresponding to the unordered outcomes. In our poker hand example, f will map 5! elements in the domain of the function (the set of ordered 5 card outcomes) to one element in the range (the set of 5 element subsets of S). The number of elements in the range of the function is therefore $\frac{52!}{47!5!}$.

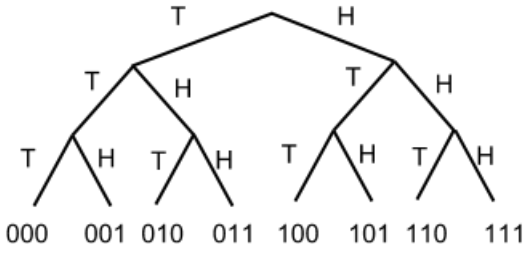
Sampling with Replacement

Sometimes we wish to consider a different scenario where we are still picking k elements out of an n element set $S = \{1, 2, \cdots, n\}$ one at a time (order matters). The difference is that after we pick an element for our sequence, we throw it back into S so we can pick it again. How many such sequences of k elements can we obtain? We can use the first rule of counting. Since we have n choices in each trial, $n_1 = n_2 = \ldots = n_k = n$. Then we have a grand total of n^k sequences.

This type of sampling is called *sampling with replacement*; multiple trials can have the same outcome. Card dealing is a type of *sampling without replacement*, since two trials cannot both have the same outcome (one card cannot be dealt twice).

Coin Flipping

Let us return to coin flipping. How many different outcomes are there if we flip a coin k times? By outcome, we mean the string of results: i.e. 001 would represent 2 tails followed by a heads. We can picture this using the tree below, which depicts the possible outcomes when $k = 3$:



Here $S = \{0, 1\}$. This is a case of sampling with replacement; multiple coin flips could result in tails (we could pick the element 0 from set S in multiple trials). Order also matters - strings 001 and 010 are

considered different outcomes. By the reasoning above (using the first rule of counting) we have a total of $n^k = 2^k$ distinct outcomes, since $n = 2$.

Rolling Dice

Let's say we roll two dice, so $k = 2$ and $S = \{1,2,3,4,5,6\}$. How many possible outcomes are there? In this setting, ordering matters; obtaining 1 with the first die and 2 with the second is different from obtaining 2 with the first and 1 with the second. We are sampling with replacement, since we can obtain the same result on both dice.

The setting is therefore the same as the coin flipping example above (order matters and we are sampling with replacement), so we can use the first rule of counting in the same manner. The number of distinct outcomes is therefore $n^2 = 6^2 = 36$.

Sampling with replacement, but where order does not matter

Say you have unlimited quantities of apples, bananas and oranges. You want to select 5 pieces of fruit to make a fruit salad. How many ways are there to do this? In this example, $S = \{1,2,3\}$, where 1 represents apples, 2 represents bananas, and 3 represents oranges. $k = 5$ since we wish to select 5 pieces of fruit. Ordering does not matter; selecting an apple followed by a banana will lead to the same salad as a banana followed by an apple.

This scenario is much more tricky to analyze. It is natural to apply the second rule of counting because order does not matter. So we first pretend that order matters, and then the number of ordered objects is 3^5 as discussed above. How many ordered options are there for every unordered option? The problem is that this number differs depending on which unordered object we are considering. Let's say the unordered object is an outcome with 5 bananas. There is only one such ordered outcome. But if we are considering 4 bananas and 1 apple, there are 5 such ordered outcomes (represented as 12222,21222,22122,22212,22221).

Now that we see the second rule of counting will not help, can we look at this problem in a different way? Let us first generalize back to our original setting: we have a set $S = \{1,2,\cdots,n\}$ and we would like to know how many ways there are to choose multisets (sets with repetition) of size k. Remarkably, we can model this problem in terms of binary strings.

Assume we have 1 bin for each element from set S, so n bins. For example, if we selected 2 apples and 1 banana, bin 1 would have 2 elements and bin 2 would have 1 element. In order to count the number of multisets, we need to count how many different ways there are to fill these bins with k elements. We don't care about the order of the bins themselves, just how many of the k elements each bin contains. Let's represent each of the k elements by a 0 in the binary string, and separations between bins by a 1. Consider the following picture:

This would be a sample placement where $S = \{1,\cdots,5\}$ and $k = 4$. Counting the number of multi sets is now equivalent to counting the number of placements of the k 0's. We have just reduced what seemed like a very complex problem to a question about a binary string, simply by looking at it from a different perspective!

How many ways can we choose these locations? The length of our binary string is $k+n-1$, and we are

choosing which k locations should contain 0's. The remaining $n-1$ locations will contain 1's. Once we pick a location for one zero, we cannot pick it again; repetition is not allowed. Picking location 1 followed by location 2 is the same as picking location 2 followed by location 1, so ordering does not matter. It follows that all we wish to compute is the number of ways of picking k elements from $k+n-1$ elements, without replacement and where the order of placement does not matter. This is given by $\binom{n+k-1}{k}$, as discussed in the Counting Sets section above. This is therefore the number of ways in which we can choose multisets of size k from set S.

Returning to our example above, the number of ways of picking 5 pieces of fruit is exactly $\binom{3+5-1}{5} = \binom{7}{5}$

Notice that we started with a problem which seemed very different from previous examples, but, by viewing it from a certain perspective, we were able to use previous techniques (those used in counting sets) to find a solution! This is key to many combinatorial arguments as we will explore further in the next section.

Combinatorial Proofs

Combinatorial arguments are interesting because they rely on intuitive counting arguments rather than algebraic manipulation. We can prove complex facts, such as $\binom{n}{k+1} = \binom{n-1}{k} + \binom{n-2}{k} + \cdots + \binom{k}{k}$. You can directly verify this identity by algebraic manipulation. But you can also do this by interpreting what each side means as a combinatorial process. The left hand side is just the number of ways of choosing a $k+1$ element subset from a set of n items. Let us think about a different process that results in the choice of a $k+1$ element subset. We start by picking the lowest numbered element in the subset. It is either the first element of the set or the second or the third or ... If we choose the first element, we must now choose k elements out of the remaining $n-1$ which we can do in $\binom{n-1}{k}$ ways. If instead the lowest numbered element we picked is the second element then we still have to choose k elements from the remaining $n-2$ which can be done in $\binom{n-2}{k}$ ways. Moreover all these subsets are distinct from those where the lowest numbered element was the first one. So we should add the number of ways of choosing each to the grand total. Proceeding in this way, we split up the process into cases according to the first (i.e., lowest-numbered) object we select, to obtain:

$$\textbf{First element selected is either} \begin{cases} \text{element } 1, & \binom{n-1}{k} \\ \text{element } 2, & \binom{n-2}{k} \\ \text{element } 3, & \binom{n-3}{k} \\ \vdots & \\ \text{element}(n-k), & \binom{k}{k} \end{cases}$$

(Note that the lowest-numbered object we select cannot be higher than $n-k$ as we have to select k distinct objects.)

The last combinatorial proof we will do is the following: $\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n$. To see this, imagine that we have a set S with n elements. On the left hand side, the i^{th} term counts the number of ways of choosing a subset of S of size exactly i; so the sum on the left hand side counts the total number of subsets (of any size) of S.

We claim that the right hand side (2^n) does indeed also count the total number of subsets. To see this, just identify a subset with an n-bit vector, where in each position j we put a 1 if the jth element is in the subset, and a 0 otherwise. So the number of subsets is equal to the number of n-bit vectors, which is 2^n (there are 2 options for each bit). Let us look at an example, where $S = \{1,2,3\}$ (so $n = 3$). Enumerate all $2^3 = 8$ possible subsets of S: $\{\{\},\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$. The term $\binom{3}{0}$ counts the number of

ways to choose a subset of S with 0 elements; there is only one such subset, namely the empty set. There are $\binom{3}{1} = 3$ ways of choosing a subset with 1 element, $\binom{3}{2} = 3$ ways of choosing a subset with 2 elements, and $\binom{3}{3} = 1$ way of choosing a subset with 3 elements (namely, the subset consisting of the whole of S). Summing, we get $1+3+3+1 = 8$, as expected.

Introduction to Discrete Probability

In the last note we considered the probabilistic experiment where we flipped a fair coin $10,000$ times and counted the number of Hs. We asked "what is the chance that we get between 4900 and 5100 Hs". One of the lessons was that the remarkable concentration of the fraction of Hs had something to do with the astronomically large number of possible outcomes of $10,000$ coin flips.

In this note we will formalize all these notions for an arbitrary probabilistic experiment. We will start by introducing the space of all possible outcomes of the experiment, called a sample space. Each element of the sample space is assigned a probability - which tells us how likely it is to occur when we actually perform the experiment. The mathematical formalism we introduce might take you some time to get used to. But you should remember that ultimately it is just a precise way to say what we mean when we describe a probabilistic experiment like flipping a coin n times.

Random Experiments

In general, a probabilistic experiment consists of drawing a sample of k elements from a set S of cardinality n. The possible outcomes of such an experiment are exactly the objects that we counted in the last note. Recall from the last note that we considered four possible scenarios for counting, depending upon whether we sampled with or without replacement, and whether the order in which the k elements are chosen does or does not matter. The same will be the case for our probabilistic experiments. The outcome of the random experiment is called a *sample point*. The *sample space*, often denoted by Ω, is the set of all possible outcomes.

An example of such an experiment is tossing a coin 4 times. In this case, $S = \{H, T\}$ and we are drawing 4 elements with replacement. $HTHT$ is an example of a sample point and the sample space has 16 elements:

$\Omega =$

HHHH HTHH
HHHT HTHT
HHTH HTTH
HHTT HTTT
THHH TTHH
THHT TTHT
THTH TTTH
THTT TTTT

How do we determine the chance of each particular outcome, such as HHT, of our experiment? In order to do this, we need to define the probability for each sample point, as we will do below.

Probability Spaces

A probability space is a sample space Ω, together with a probability $\Pr[\omega]$ for each sample point ω, such that

- $0 \leq \Pr[\omega] \leq 1$ for all $\omega \in \Omega$.
- $\sum_{\omega \in \Omega} \Pr[\omega] = 1$, i.e., the sum of the probabilities of all outcomes is 1.

The easiest way to assign probabilities to sample points is uniformly: if $|\Omega| = N$, then $\Pr[x] = \frac{1}{N}\ \forall x \in \Omega$. For example, if we toss a fair coin 4 times, each of the 16 sample points (as pictured above) is assigned probability $\frac{1}{16}$. We will see examples of non-uniform probability distributions soon.

After performing an experiment, we are often interested in knowing whether an event occurred. For example, the might be interested in the event that there were "exactly 2 H's in four tosses of the coin". How do we formally define the concept of an event in terms of the sample space Ω? Here is a beautiful answer. We will identify the event "exactly 2 H's in four tosses of the coin" with the subset consisting of those outcomes in which there are exactly two H's:
$\{HHTT, HTHT, HTTH, THHT, THTH, TTHH\} \subseteq \Omega$. Now we turn this around and say that formally an event A is just a subset of the sample space, $A \subseteq \Omega$.

How should we define the probability of an event A? Naturally, we should just *add up* the probabilities of the sample points in A.

For any event $A \subseteq \Omega$, we define the probability of A to be

$$\Pr[A] = \sum_{\omega \in A} \Pr[\omega].$$

Thus the probability of getting exactly two H's in four coin tosses can be calculated using this definition as follows. A consists of all sequences that have exactly two H's, and so $|A| = \binom{4}{2} = 6$. For this example, there are $2^4 = 16$ possible outcomes for flipping four coins. Thus, each sample point $\omega \in A$ has probability $\frac{1}{16}$; and, as we saw above, there are six sample points in A, giving us $\Pr[A] = 6 \cdot \frac{1}{16} = \frac{3}{8}$.

Examples

We will now look at examples of random experiments and their corresponding sample spaces, along with possible probability spaces and events.

Coin Flipping

Suppose we have a coin of bias p, and our experiment consists of flipping the coin 4 times. The sample space Ω consists of the sixteen possible sequences of H's and T's shown in the figure on the last page.

The probability space depends on p. If $p = \frac{1}{2}$ the probabilities are assigned uniformly; the probability of each sample point is $\frac{1}{16}$. What if the coin comes up heads with probability $\frac{2}{3}$ and tails with probability $\frac{1}{3}$ (i.e. the bias is $p = \frac{2}{3}$)? Then the probabilities are different. For example, $\Pr[HHHH] = \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} = \frac{16}{81}$, while $\Pr[TTHH] = \frac{1}{3} \times \frac{1}{3} \times \frac{2}{3} \times \frac{2}{3} = \frac{4}{81}$. [*Note: We have cheerfully* multiplied *probabilities here; we'll explain why this is OK later. It is* not *always OK!*]

What type of events can we consider in this setting? Let event A be the event that all four coin tosses are the same. Then $A = \{HHHH, TTTT\}$. $HHHH$ has probability $\frac{2}{3}^4$ and $TTTT$ has probability $\frac{1}{3}^4$. Thus, $\Pr[A] = \Pr[HHHH] + \Pr[TTTT] = \frac{2}{3}^4 + \frac{1}{3}^4 = \frac{17}{81}$.

Next, consider event B: the event that there are exactly two heads. The probability of any particular outcome with two heads (such as $HTHT$) is $\frac{2}{3}^2\frac{1}{3}^2$. How many such outcomes are there? There are $\binom{4}{2} = 6$ ways of choosing the positions of the heads, and these choices completely specify the sequence. So $\Pr[B] = 6\frac{2}{3}^2\frac{1}{3}^2 = \frac{24}{81} = \frac{8}{27}$.

More generally, if we flip the coin n times, we get a sample space Ω of cardinality 2^n. The sample points are all possible sequences of n H's and T's. If the coin has bias p, and if we consider any sequence of n coin flips with exactly r H's, then the the probability of this sequence is $p^r(1-p)^{n-r}$.

Now consider the event C that we get exactly r H's when we flip the coin n times. This event consists of exactly $\binom{n}{r}$ sample points. Each has probability $p^r(1-p)^{n-r}$. So the probability of this event, $P[C] = \binom{n}{r}p^r(1-p)^{n-r}$.

Biased coin-tossing sequences show up in many contexts: for example, they might model the behavior of n trials of a faulty system, which fails each time with probability p.

Rolling Dice

The next random experiment we will discuss consists of rolling two dice. In this experiment, $\Omega = \{(i, j) : 1 \leq i, j \leq 6\}$. The probability space is uniform, i.e. all of the sample points have the *same* probability, which must be $\frac{1}{|\Omega|}$. In this case, $|\Omega| = 36$, so each sample point has probability $\frac{1}{36}$. In such circumstances, the probability of any event A is clearly just

$$\Pr[A] = \frac{\text{\# of sample points in } A}{\text{\# of sample points in } \Omega} = \frac{|A|}{|\Omega|}.$$

So for uniform spaces, computing probabilities reduces to *counting* sample points!

Now consider two events: the event A that the sum of the dice is at least 10 and the event B that there is at least one 6. By writing out the number of sample points in each event, we can determine the number of sample points in each event; $|A| = 6$ and $|B| = 11$. By the observation above, it follows that $\Pr[A] = \frac{6}{36} = \frac{1}{6}$ and $\Pr[B] = \frac{11}{36}$.

Card Shuffling

The random experiment consists of shuffling a deck of cards. Ω is equal to the set of the 52! permutations of the deck. The probability space is uniform. Note that we're really talking about an idealized mathematical model of shuffling here; in real life, there will always be a bit of bias in our shuffling. However, the mathematical model is close enough to be useful.

Poker Hands

Here's another experiment: shuffling a deck of cards and dealing a poker hand. In this case, S is the set of 52 cards and our sample space $\Omega = \{\text{all possible poker hands}\}$, which corresponds to choosing $k = 5$ objects without replacement from a set of size $n = 52$ where order does not matter. Hence, as we saw in the previous Note, $|\Omega| = \binom{52}{5} = \frac{52\times51\times50\times49\times48}{5\times4\times3\times2\times1} = 2,598,960$. Since the deck is assumed to be randomly shuffled, the probability of each outcome is equally likely and we are therefore dealing with a uniform probability space.

Let A be the event that the poker hand is a flush. [For those who are not addicted to gambling, a *flush* is a hand in which all cards have the same suit, say Hearts.] Since the probability space is uniform, computing $\Pr[A]$ reduces to simply computing $|A|$, or the number of poker hands which are flushes. There are 13 cards in each suit, so the number of flushes in each suit is $\binom{13}{5}$. The total number of flushes is therefore $4 \cdot \binom{13}{5}$. Then we have

$$\Pr[\text{hand is a flush}] = \frac{4 \cdot \binom{13}{5}}{\binom{52}{5}} = \frac{4 \cdot 13! \cdot 5! \cdot 47!}{5! \cdot 8! \cdot 52!} = \frac{4 \cdot 13 \cdot 12 \cdot 11 \cdot 10 \cdot 9}{52 \cdot 51 \cdot 50 \cdot 49 \cdot 48} \approx 0.002.$$

Balls and Bins

In this experiment, we will throw 20 (labeled) balls into 10 (labeled) bins. Assume that each ball is equally likely to land in any bin, regardless of what happens to the other balls.

If you wish to understand this situation in terms of sampling a sequence of k elements from a set S of cardinality n: here the set S consists of the 10 bins, and we are sampling with replacement $k = 20$ times. The order of sampling matters, since the balls are labeled.

The sample space Ω is equal to $\{(b_1, b_2, \ldots, b_{20}) : 1 \leq b_i \leq 10\}$, where the component b_i denotes the bin in which ball i lands. The cardinality of the sample space, $|\Omega|$, is equal to 10^{20}- each element b_i in the sequence has 10 possible choices, and there are 20 elements in the sequence. More generally, if we throw m balls into n bins, we have a sample space of size n^m. The probability space is uniform; as we said earlier, each ball is equally likely to land in any bin.

Let A be the event that bin 1 is empty. Since the probability space is uniform, we simply need to count how many outcomes have this property. This is exactly the number of ways all 20 balls can fall into the remaining nine bins, which is 9^{20}. Hence, $\Pr[A] = \frac{9^{20}}{10^{20}} = (\frac{9}{10})^{20} \approx 0.12$.

Let B be the event that bin 1 contains at least one ball. This event is the *complement* $\bar{A}$ of A, i.e., it consists of precisely those sample points which are not in A. So $\Pr[B] = 1 - \Pr[A] \approx .88$. More generally, if we throw m balls into n bins, we have:

$$\Pr[\text{bin 1 is empty}] = \left(\frac{n-1}{n}\right)^m = \left(1 - \frac{1}{n}\right)^m.$$

As we shall see, balls and bins is another probability space that shows up very often in Computer Science: for example, we can think of it as modeling a load balancing scheme, in which each job is sent to a random processor.

It is also a more general model for problems we have previously considered. For example, flipping a fair coin 3 times is a special case in which the number of balls (m) is 3 and the number of bins (n) is 2. Rolling two dice is a special case in which $m = 2$ and $n = 6$.

Birthday Paradox

The "birthday paradox" is a remarkable phenomenon that examines the chances that two people in a group have the same birthday. It is a "paradox" not because of a logical contradiction, but because it goes against intuition. For ease of calculation, we take the number of days in a year to be 365. Then $U = \{1, \ldots, 365\}$, and the random experiment consists of drawing a sample of n elements from U, where the elements are the birth dates of n people in a group. Then $|\Omega| = 365^n$. This is because each sample point is a sequence of possible birthdays for n people; so there are n points in the sequence and each point has 365 possible values.

Let A be the event that at least two people have the same birthday. If we want to determine $\Pr[A]$, it might be simpler to instead compute the probability of the complement of A, $\Pr[\bar{A}]$. $\bar{A}$ is the event that no two people have the same birthday. Since $\Pr[A] = 1 - \Pr[\bar{A}]$, we can then easily compute $\Pr[A]$.

We are again working in a uniform probability space, so we just need to determine $|\bar{A}|$. Equivalently, we are computing the number of ways there are for no two people to have the same birthday. There are 365 choices for the first person, 364 for the second, $\ldots$, $365 - n + 1$ choices for the n^{th} person, for a total of $365 \times 364 \times \cdots \times (365 - n + 1)$. Note that this is simply an application of the first rule of counting; we are sampling without replacement and the order matters.

Thus we have $\Pr[\bar{A}] = \frac{|\bar{A}|}{|\Omega|} = \frac{365 \times 364 \times \cdots \times (365-n+1)}{365^n}$. Then $\Pr[A] = 1 - \frac{365 \times 364 \times \cdots \times (365-n+1)}{365^n}$. This allows us to compute $\Pr[A]$ as a function of the number of people, n. Of course, as n increases $\Pr[A]$ increases. In fact, with $n = 23$ people you should be willing to bet that at least two people do have the same birthday, since then $\Pr[A]$ is larger than 50%! For $n = 60$ people, $\Pr[A]$ is over 99%.

The Monty Hall Problem

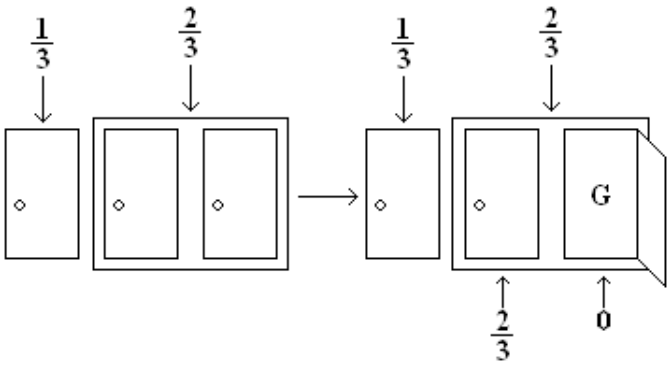
In an (in)famous 1970s game show hosted by one Monty Hall, a contestant was shown three doors; behind one of the doors was a prize, and behind the other two were goats. The contestant picks a door (but doesn't open it). Then Hall's assistant (Carol), opens one of the other two doors, revealing a goat (since Carol knows where the prize is, she can always do this). The contestant is then given the option of sticking with his current door, or switching to the other unopened one. He wins the prize if and only if his chosen door is the correct one. The question, of course, is: Does the contestant have a better chance of winning if he switches doors?

Intuitively, it seems obvious that since there are only two remaining doors after the host opens one, they must have equal probability. So you may be tempted to jump to the conclusion that it should not matter whether or not the contestant stays or switches.

Yet there are other people whose intuition cries out that the contestant is better off switching. So who's correct?

As a matter of fact, the contestant has a better chance of picking the car if he uses the switching strategy. How can you convince yourself that this is true? One way you can do this is by doing a rigorous analysis. You would start by writing out the sample space, and then assign probabilities to each sample point. Finally you would calculate the probability of the event that the contestant wins under the sticking strategy. This is an excellent exercise if you wish to make sure you understand the formalism of probability theory we introduced above.

Let us instead give a more intuitive pictorial argument. Initially when the contestant chooses the door, he has a $\frac{1}{3}$ chance of picking the car. This must mean that the other doors combined have a $\frac{2}{3}$ chance of winning. But after Carol opens a door with a goat behind it, how do the probabilities change? Well, everyone knows that there is a goat behind one of the doors that the contestant did not pick. So no matter whether the contestant is winning or not, Carol is always able to open one of the other doors to reveal a goat. This means that the contestant still has a $\frac{1}{3}$ chance of winning. Also the door that Carol opened has no chance of winning. What about the last door? It must have a $\frac{2}{3}$ chance of containing the car, and so the contestant has a higher chance of winning if he or she switches doors. This argument can be summed up nicely in the following picture:



You will be able to formalize this intuitive argument once we cover conditional probability.

In the meantime, to approach this problem formally, first determine the sample space and the probability space. Just a hint: it is not a uniform probability space! Then formalize the event we have described above (as a subspace of the sample space), and compute the probability of the event. Good luck!

Summary

The examples above illustrate the importance of doing probability calculations systematically, rather than "intuitively." Recall the key steps in all our calculations:

- What is the sample space (i.e., the experiment and its set of possible outcomes)?
- What is the probability of each outcome (sample point)?
- What is the event we are interested in (i.e., which subset of the sample space)?
- Finally, compute the probability of the event by adding up the probabilities of the sample points inside it.

Whenever you meet a probability problem, you should always go back to these basics to avoid potential pitfalls. Even experienced researchers make mistakes when they forget to do this — witness many erroneous "proofs", submitted by mathematicians to newspapers at the time, of the fact that the switching strategy in the Monty Hall problem does not improve the odds.

Introduction

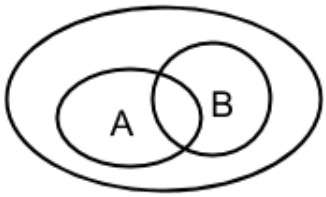
One of the key properties of coin flips is independence: if you flip a fair coin ten times and get ten T's, this does not make it more likely that the next coin flip will be H's. It still has exactly 50% chance of being H's.

By contrast, suppose while dealing cards, the first ten cards are all red (hearts or diamonds). What is the chance that the next card is red? This is easy: we started with exactly 26 red cards and 26 black cards. But after dealing the first ten cards we know that the deck has 16 red cards and 26 black cards. So the chance that the next card is red is $\frac{16}{42}$. So unlike the case of coin flips, now the chance of drawing a red card is no longer independent of the previous card that was dealt. This is the phenomenon we will explore in this note on conditional probability.

Conditional Probability

Let's consider an example with a smaller sample space. Suppose we toss $m = 3$ balls into $n = 3$ bins; this is a uniform sample space with $3^3 = 27$ points. We already know that the probability the first bin is empty is $(1 - \frac{1}{3})^3 = (\frac{2}{3})^3 = \frac{8}{27}$. What is the probability of this event *given that* the second bin is empty? Call these events A, B respectively. In the language of conditional probability we wish to compute the probability $P[A|B]$, which we read to say probability of A given B.

How should we compute $\Pr[A|B]$? Well, since event B is guaranteed to happen, we need to look not at the whole sample space Ω, but at the smaller sample space consisting only of the sample points in B. In terms of the picture below, we are no longer looking at the large oval, but only the oval labeled B:



What should the probabilities of these sample points be? If they all simply inherit their probabilities from Ω, then the sum of these probabilities will be $\sum_{\omega \in B} \Pr[\omega] = \Pr[B]$, which in general is less than 1. So we need to *scale* the probability of each sample point by $\frac{1}{\Pr[B]}$. I.e., for each sample point $\omega \in B$, the new probability becomes

$$\Pr[\omega|B] = \frac{\Pr[\omega]}{\Pr[B]}.$$

Now it is clear how to compute $\Pr[A|B]$: namely, we just sum up these scaled probabilities over all sample points that lie in both A and P:

$$\Pr[A|B] = \sum_{\omega \in A \cap B} \Pr[\omega|B] = \sum_{\omega \in A \cap B} \frac{\Pr[\omega]}{\Pr[B]} = \frac{\Pr[A \cap B]}{\Pr[B]}.$$

Definition 14.1 (Conditional Probability)**.** *For events A, B in the same probability space, such that* $\Pr[B] > 0$*, the conditional probability of A given B is*

$$\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]}.$$

Returning to our example, to compute $\Pr[A|B]$ we need to figure out $\Pr[A \cap B]$. But $A \cap B$ is the event that both the first two bins are empty, i.e., all three balls fall in the third bin. So $\Pr[A \cap B] = \frac{1}{27}$ (why?). Therefore,

$$\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{1/27}{8/27} = \frac{1}{8}.$$

Not surprisingly, $\frac{1}{8}$ is quite a bit less than $\frac{8}{27}$: knowing that bin 2 is empty makes it significantly less likely that bin 1 will be empty.

Example: Card Dealing

Let's apply the ideas discussed above to compute the probability that, when dealing 2 cards and the first card is known to be an ace, the second card is also an ace. Let B be the event that the first card is an ace, and let A be the event that the second card is an ace. Note that $P[A] = P[B] = \frac{1}{13}$.

To compute $\Pr[A|B]$, we need to figure out $\Pr[A \cap B]$. This is the probability that both cards are aces. Note that there are $52 \cdot 51$ sample points in the sample space, since each sample point is a sequence of two cards. A sample point is in $A \cap B$ if both cards are aces. This can happen in $4 \cdot 3 = 12$ ways.

Since each sample point is equally likely, $\Pr[A \cap B] = \frac{12}{52 \cdot 51}$. The probability of event B, drawing an ace in the first trial, is $\frac{4}{52}$. Therefore,

$$\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{3}{51}.$$

Note that this says that if the first card is an ace, it makes it less likely that the second card is also an ace.

Bayesian Inference

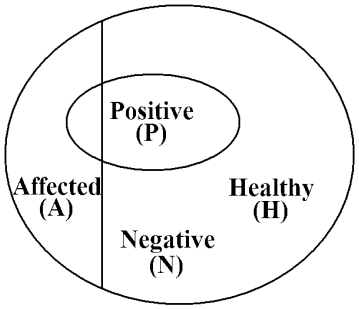
Now that we've introduced the notion of conditional probability, we can see how it is used in real world settings. Conditional probability is at the heart of a subject called *Bayesian inference*, used extensively in fields such as machine learning, communications and signal processing. Bayesian inference is a way to *update knowledge* after making an observation. For example, we may have an estimate of the probability of a given event A. After event B occurs, we can update this estimate to $\Pr[A|B]$. In this interpretation, $\Pr[A]$ can be thought of as a *prior* probability: our assessment of the likelihood of an event of interest A *before* making an observation. It reflects our prior knowledge. $\Pr[A|B]$ can be interpreted as the *posterior* probability of A after the observation. It reflects our new knowledge.

Here is an example of where we can apply such a technique. A pharmaceutical company is marketing a new test for a certain medical disorder. According to clinical trials, the test has the following properties:

1. When applied to an affected person, the test comes up positive in 90% of cases, and negative in 10% (these are called "false negatives").

2. When applied to a healthy person, the test comes up negative in 80% of cases, and positive in 20% (these are called "false positives").

Suppose that the incidence of the disorder in the US population is 5%; this is our prior knowledge. When a random person is tested and the test comes up positive, how can we update this probability? (Note that this is presumably *not* the same as the simple probability that a random person has the disorder, which is just $\frac{1}{20}$.) The implicit probability space here is the entire US population with uniform probabilities.



The sample space here consists of all people in the US — denote their number by N (so $N \approx 250$ million). Let A be the event that a person chosen at random is affected, and B be the event that a person chosen at random tests positive. Now we can rewrite the information above:

- $\Pr[A] = 0.05$, (5% of the U.S. population is affected)
- $\Pr[B|A] = 0.9$ (90% of the affected people test positive)
- $\Pr[B|\overline{A}] = 0.2$ (20% of healthy people test positive)

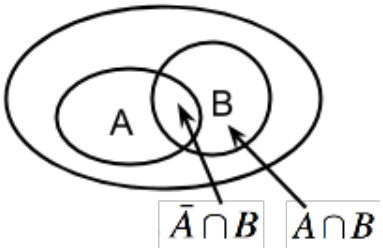
We want to calculate $\Pr[A|B]$. We can proceed as follows:

$$\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{\Pr[B|A]\Pr[A]}{\Pr[B]} \tag{1}$$

We obtained the second equality above by applying the definition of conditional probability:

$$\Pr[B|A] = \frac{\Pr[A \cap B]}{\Pr[A]}$$

Now we need to compute $\Pr[B]$. This is the probability that a random person tests positive. To compute this, we can sum two values: the probability that a healthy person tests positive, $\Pr[\bar{A} \cap B]$ and the probability that an affected person tests positive, $\Pr[A \cap B]$. We can sum because the events $\bar{A} \cap B$ and $A \cap B$ do not intersect:



By again applying the definition of conditional probability we have:

$$\Pr[B] = \Pr[A \cap B] + \Pr[\overline{A} \cap B] = \Pr[B|A]\Pr[A] + \Pr[B|\overline{A}](1 - \Pr[A]) \tag{2}$$

Combining equations (1) and (2), we have expressed $\Pr[A|B]$ in terms of $\Pr[A], \Pr[B|A]$ and $\Pr[B|\overline{A}]$:

$$\Pr[A|B] = \frac{\Pr[B|A]\Pr[A]}{\Pr[B|A]\Pr[A] + \Pr[B|\overline{A}](1-\Pr[A])} \tag{3}$$

By plugging in the values written above, we obtain $\Pr[A|B] = \frac{9}{47} \approx .19$.

Equation (3) is useful for many inference problems. We are given $\Pr[A]$, which is the (unconditional) probability that the event of interest A happens. We are given $\Pr[B|A]$ and $\Pr[B|\overline{A}]$, which quantify how noisy the observation is. (If $\Pr[B|A] = 1$ and $\Pr[B|\overline{A}] = 0$, for example, the observation is completely noiseless.) Now we want to calculate $\Pr[A|B]$, the probability that the event of interest happens given we made the observation. Equation (3) allows us to do just that.

Of course, equations (1), (2) and (3) are derived from the basic axioms of probability and the definition of conditional probability, and are therefore true with or without the above Bayesian inference interpretation. However, this interpretation is very useful when we apply probability theory to study inference problems.

Bayes' Rule and Total Probability Rule

Equations (1) and (2) are very useful in their own right. The first is called **Bayes' Rule** and the second is called the **Total Probability Rule**. Bayes' rule is useful when one wants to calculate $\Pr[A|B]$ but one is given $\Pr[B|A]$ instead, i.e. it allows us to "flip" things around.

The Total Probability rule is an application of the strategy of "dividing into cases" .There are two possibilities: either an event A happens or A does not happen. If A happens the probability that B happens is $\Pr[B|A]$. If A does not happen, the probability that B happens is $\Pr[B|\overline{A}]$. If we know or can easily calculate these two probabilities and also $\Pr[A]$, then the total probability rule yields the probability of event B.

Example: Tennis Match

You are about to play a tennis match against a randomly chosen opponent and you wish to calculate your probability of winning. You know your opponent will be one of two people, X or Y. If person X is chosen, you will win with probability .7. If person Y is chosen, you will win with probability .3. Your opponent is chosen by flipping a coin with bias .6 in favor of X.

Let's first determine which events we are interested in. Let A be the event that you win. Let B_1 be the event that person X is chosen, and let B_2 be the event that person Y is chosen. We wish to calculate $\Pr[A]$. Here is what we know so far:

- $\Pr[A|B_1] = 0.7$, (if person X is chosen, you win with probability .7)
- $\Pr[A|B_2] = 0.3$ (if person Y is chosen, you win with probability .3)
- $\Pr[B_1] = 0.6$ (person X is chosen with probability .6)
- $\Pr[B_2] = 0.4$ (person Y is chosen with probability .4)

By using the Total Probability rule, we have:

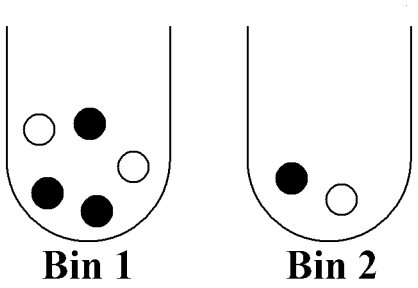
$$\Pr[A] = \Pr[A|B_1]\Pr[B_1] + \Pr[A|B_2]\Pr[B_2].$$

Now we can simply plug in the known values above to obtain $\Pr[A]$:

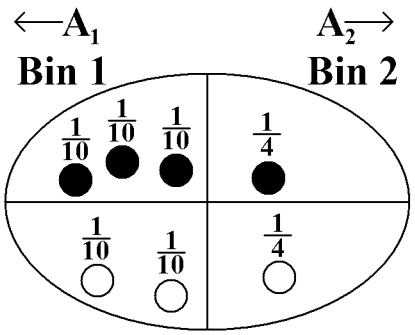
$$\Pr[A] = .7 \times .6 + .3 \times .4 = .54$$

Example: Balls and Bins

Imagine we have two bins containing black and white balls, and further suppose that we wanted to know what is the chance that we picked Bin 1 given that we picked a white ball, i.e., $\Pr[\text{Bin } 1|\bigcirc]$. Assume that we are unbiased when choosing a bin so that each bin is chosen with probability $\frac{1}{2}$.



A wrong approach is to say that the answer is clearly $\frac{2}{3}$, since we know there are a total of three white balls, two of which are in bin 1. However, this picture is misleading because the bins have equal "weight". Instead, what we should do is appropriately scale each sample point as the following picture shows:



This images shows that the sample space Ω is equal to the union of the events contained in bin $1(A_1)$ and bin $2(A_2)$, so $\Omega = A_1 \cup A_2$. We can use the definition of conditional probability to see that

$$\Pr[\text{Bin } 1|\bigcirc] = \frac{\frac{1}{10}+\frac{1}{10}}{\frac{1}{10}+\frac{1}{10}+\frac{1}{4}} = \frac{\frac{2}{10}}{\frac{9}{20}} = \frac{4}{9}$$

Let us try to achieve this probability using Bayes' rule. To apply Bayes' rule, we need to compute $\Pr[\bigcirc|\text{Bin } 1]$, $\Pr[\text{Bin } 1]$ and $\Pr[\bigcirc]$. $\Pr[\bigcirc|\text{Bin } 1]$ is the chance that we pick a white ball given that we picked bin 1, which is $\frac{2}{5}$. $\Pr[\text{ Bin } 1]$ is $\frac{1}{2}$ as given in the description of the problem. Finally, $\Pr[\bigcirc]$ can be computed using the Total Probability rule:

$$\Pr[\bigcirc] = \Pr[\bigcirc|\text{Bin } 1] \times \Pr[\text{Bin } 1] + \Pr[\bigcirc|\text{Bin } 2] \times \Pr[\text{Bin } 2] = \frac{2}{5} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{9}{20}.$$

Observe that we can apply the Total Probability rule here because $\Pr[\text{ Bin } 1]$ is the complement of $\Pr[\text{ Bin } 2]$. Finally, if we plug the above values into Bayes' rule we obtain the probability that we picked bin 1 given that we picked a white ball:

$$\Pr[\text{Bin } 1|\bigcirc] = \frac{\frac{2}{5}\times\frac{1}{2}}{\frac{9}{20}} = \frac{\frac{2}{10}}{\frac{9}{20}} = \frac{4}{9}.$$

All we have done above is combined Bayes' rule and the Total Probability rule; this is also how we obtained Equation (3). We could equivalently have plugged in the appropriate values to Equation (3).

Combinations of events

In most applications of probability in Computer Science, we are interested in things like $\Pr[\bigcup_{i=1}^n A_i]$ and $\Pr[\bigcap_{i=1}^n A_i]$, where the A_i are simple events (i.e., we know, or can easily compute, the $\Pr[A_i]$). The intersection $\bigcap_i A_i$ corresponds to the logical AND of the events A_i, while the union $\bigcup_i A_i$ corresponds to their logical OR. As an example, if A_i denotes the event that a failure of type i happens in a certain system, then $\bigcup_i A_i$ is the event that the system fails.

In general, computing the probabilities of such combinations can be very difficult. In this section, we discuss some situations where it can be done. Let's start with independent events, for which intersections are quite simple to compute.

Independent Events

Definition 14.2 (Independence)**.** *Two events A, B in the same probability space are independent if* $\Pr[A \cap B] = \Pr[A] \times \Pr[B]$.

The intuition behind this definition is the following. Suppose that $\Pr[B] > 0$. Then we have

$$\Pr[A|B] = \frac{\Pr[A \cap B]}{\Pr[B]} = \frac{\Pr[A] \times \Pr[B]}{\Pr[B]} = \Pr[A].$$

Thus independence has the natural meaning that "the probability of A is not affected by whether or not B occurs." (By a symmetrical argument, we also have $\Pr[B|A] = \Pr[B]$ provided $\Pr[A] > 0$.) For events A, B such that $\Pr[B] > 0$, the condition $\Pr[A|B] = \Pr[A]$ is actually *equivalent* to the definition of independence.

Several of our previously mentioned random experiments consist of independent events. For example, if we flip a coin twice, the event of obtaining heads in the first trial is independent to the event of obtaining heads in the second trial. The same applies for two rolls of a die; the outcomes of each trial are independent.

The above definition generalizes to any finite set of events:

Definition 14.3 (Mutual independence)**.** *Events $A_1, \ldots, A_n$ are mutually independent if for every subset $I \subseteq \{1, \ldots, n\}$,*

$$\Pr[\textstyle\bigcap_{i \in I} A_i] = \prod_{i \in I} \Pr[A_i].$$

Note that we need this property to hold for *every* subset I.

For mutually independent events $A_1, \ldots, A_n$, it is not hard to check from the definition of conditional probability that, for any $1 \leq i \leq n$ and any subset $I \subseteq \{1, \ldots, n\} \setminus \{i\}$, we have

$$\Pr[A_i | \textstyle\bigcap_{j \in I} A_j] = \Pr[A_i].$$

Note that the independence of every pair of events (so-called *pairwise independence*) does *not* necessarily imply mutual independence. For example, it is possible to construct three events A, B, C such that each *pair* is independent but the triple A, B, C is *not* mutually independent.

Pairwise Independence Example

Suppose you toss a fair coin twice and let A be the event that the first flip is H's and B be the event that the second flip is H's. Now let C be the event that both flips are the same (i.e. both H's or both T's). Of course A and B are independent. What is more interesting is that so are A and C: given that the first toss came up H's, there is still an even chance that the second flip is the same as the first. Another way of saying this is that $P[A \cap C] = P[A]P[C] = 1/4$ since $A \cap C$ is the event that the first flip is H's and the second is also H's. By the same reasoning B and C are also independent. On the other hand, A, B and C are not mutually independent. For example if we are given that A and B occurred then the probability that C occurs is 1. So even though A, B and C are not mutually independent, every pair of them are independent. In other words, A, B and C are pairwise independent but not mutually independent.

Intersections of events

Computing intersections of independent events is easy; it follows from the definition. We simply multiply the probabilities of each event. How do we compute intersections for events which may not be independent? From the definition of conditional probability, we immediately have the following product rule (sometimes also called the chain rule) for computing the probability of an intersection of events.

Theorem 14.1 (Product Rule)**.** *For any events A, B, we have*

$$\Pr[A \cap B] = \Pr[A] \Pr[B|A].$$

More generally, for any events $A_1, \ldots, A_n$,

$$\Pr[\textstyle\bigcap_{i=1}^{n} A_i] = \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2] \times \cdots \times \Pr[A_n|\textstyle\bigcap_{i=1}^{n-1} A_i].$$

Proof. The first assertion follows directly from the definition of $\Pr[B|A]$ (and is in fact a special case of the second assertion with $n = 2$).

To prove the second assertion, we will use induction on n (the number of events). The base case is $n = 1$, and corresponds to the statement that $\Pr[A] = \Pr[A]$, which is trivially true. For the inductive step, let $n > 1$ and assume (the inductive hypothesis) that

$$\Pr[\textstyle\bigcap_{i=1}^{n-1} A_i] = \Pr[A_1] \times \Pr[A_2|A_1] \times \cdots \times \Pr[A_{n-1}|\textstyle\bigcap_{i=1}^{n-2} A_i].$$

Now we can apply the definition of conditional probability to the two events A_n and $\bigcap_{i=1}^{n-1} A_i$ to deduce that

$$\begin{aligned}\Pr[\textstyle\bigcap_{i=1}^{n} A_i] = \Pr[A_n \cap (\textstyle\bigcap_{i=1}^{n-1} A_i)] &= \Pr[A_n|\textstyle\bigcap_{i=1}^{n-1} A_i] \times \Pr[\textstyle\bigcap_{i=1}^{n-1} A_i] \\ &= \Pr[A_n|\textstyle\bigcap_{i=1}^{n-1} A_i] \times \Pr[A_1] \times \Pr[A_2|A_1] \times \cdots \times \Pr[A_{n-1}|\textstyle\bigcap_{i=1}^{n-2} A_i],\end{aligned}$$

where in the last line we have used the inductive hypothesis. This completes the proof by induction. □

The product rule is particularly useful when we can view our sample space as a sequence of choices. The next few examples illustrate this point.

Examples

Coin tosses

Toss a fair coin three times. Let A be the event that all three tosses are heads. Then $A = A_1 \cap A_2 \cap A_3$, where A_i is the event that the ith toss comes up heads. We have

$$\begin{aligned}\Pr[A] &= \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2] \\ &= \Pr[A_1] \times \Pr[A_2] \times \Pr[A_3] \\ &= \tfrac{1}{2} \times \tfrac{1}{2} \times \tfrac{1}{2} = \tfrac{1}{8}.\end{aligned}$$

The second line here follows from the fact that the tosses are mutually independent. Of course, we already know that $\Pr[A] = \frac{1}{8}$ from our definition of the probability space in the previous lecture note. Another way of looking at this calculation is that it justifies our definition of the probability space, and shows that it was consistent with assuming that the coin flips are mutually independent.

If the coin is biased with heads probability p, we get, again using independence,

$$\Pr[A] = \Pr[A_1] \times \Pr[A_2] \times \Pr[A_3] = p^3.$$

And more generally, the probability of any sequence of n tosses containing r heads and $n-r$ tails is $p^r(1-p)^{n-r}$. This is in fact the reason we defined the probability space this way in the previous lecture note: we defined the sample point probabilities so that the coin tosses would behave independently.

Monty Hall

Recall the Monty Hall problem from the last lecture: there are three doors and the probability that the prize is behind any given door is $\frac{1}{3}$. There are goats behind the other two doors. The contestant picks a door randomly, and the host opens one of the other two doors, revealing a goat. How do we calculate intersections in this setting? For example, what is the probability that the contestant chooses door 1, the prize is behind door 2, and the host chooses door 3?

Let A_1 be the event that the contestant chooses door 1, let A_2 be the event that the prize is behind door 2, and let A_3 be the event that the host chooses door 3. We would like to compute $\Pr[A_1 \cap A_2 \cap A_3]$. By the product rule:

$$\Pr[A_1 \cap A_2 \cap A_3] = \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2]$$

The probability of A_1 is $\frac{1}{3}$, since the contestant is choosing the door at random. The probability A_2 given A_1 is still $\frac{1}{3}$ since they are independent. The probability of the host choosing door 3 given events A_1 and A_2 is 1; the host cannot choose door 1, since the contestant has already opened it, and the host cannot choose door 2, since the host must reveal a goat (and not the prize). Therefore,

$$\Pr[A_1 \cap A_2 \cap A_3] = \frac{1}{3} \times \frac{1}{3} \times 1 = \frac{1}{9}.$$

Observe that we did need conditional probability in this setting; had we simply multiplied the probabilities of each event, we would have obtained $\frac{1}{27}$ since the probability of A_3 is also $\frac{1}{3}$ (can you figure out why?). What if we changed the situation, and instead asked for the probability that the contestant chooses door 1, the prize is behind door 1, and the host chooses door 2? We can use the same technique as above, but our final answer will be different. This is left as an exercise.

Poker Hands

Let's use the product rule to compute the probability of a flush in a different way. This is equal to $4 \times \Pr[A]$, where A is the probability of a Hearts flush. Intuitively, this should be clear since there are 4 suits; we'll see why this is formally true in the next section. We can write $A = \bigcap_{i=1}^{5} A_i$, where A_i is the event that the ith card we pick is a Heart. So we have

$$\Pr[A] = \Pr[A_1] \times \Pr[A_2|A_1] \times \cdots \times \Pr[A_5|\bigcap_{i=1}^{4} A_i].$$

Clearly $\Pr[A_1] = \frac{13}{52} = \frac{1}{4}$. What about $\Pr[A_2|A_1]$? Well, since we are conditioning on A_1 (the first card is a Heart), there are only 51 remaining possibilities for the second card, 12 of which are Hearts. So $\Pr[A_2|A_1] = \frac{12}{51}$. Similarly, $\Pr[A_3|A_1 \cap A_2] = \frac{11}{50}$, and so on. So we get

$$4 \times \Pr[A] = 4 \times \frac{13}{52} \times \frac{12}{51} \times \frac{11}{50} \times \frac{10}{49} \times \frac{9}{48},$$

which is exactly the same fraction we computed in the previous lecture note.

So now we have two methods of computing probabilities in many of our sample spaces. It is useful to keep these different methods around, both as a check on your answers and because in some cases one of the methods is easier to use than the other.

Unions of Events

You are in Las Vegas, and you spy a new game with the following rules. You pick a number between 1 and 6. Then three dice are thrown. You win if and only if your number comes up on at least one of the dice.

The casino claims that your odds of winning are 50%, using the following argument. Let A be the event that you win. We can write $A = A_1 \cup A_2 \cup A_3$, where A_i is the event that your number comes up on die i. Clearly $\Pr[A_i] = \frac{1}{6}$ for each i. Therefore,

$$\Pr[A] = \Pr[A_1 \cup A_2 \cup A_3] = \Pr[A_1] + \Pr[A_2] + \Pr[A_3] = 3 \times \frac{1}{6} = \frac{1}{2}.$$

Is this calculation correct? Well, suppose instead that the casino rolled six dice, and again you win iff your number comes up at least once. Then the analogous calculation would say that you win with probability $6 \times \frac{1}{6} = 1$, i.e., certainly! The situation becomes even more ridiculous when the number of dice gets bigger than 6.

The problem is that the events A_i are *not disjoint*: i.e., there are some sample points that lie in more than one of the A_i. (We could get really lucky and our number could come up on two of the dice, or all three.) So if we add up the $\Pr[A_i]$ we are counting some sample points more than once.

Fortunately, there is a formula for this, known as the Principle of Inclusion/Exclusion:

Theorem 14.2 (Inclusion/Exclusion)**.** *For events $A_1, \ldots, A_n$ in some probability space, we have*

$$\Pr[\bigcup_{i=1}^{n} A_i] = \sum_{i=1}^{n} \Pr[A_i] - \sum_{\{i,j\}} \Pr[A_i \cap A_j] + \sum_{\{i,j,k\}} \Pr[A_i \cap A_j \cap A_k] - \cdots \pm \Pr[\bigcap_{i=1}^{n} A_i].$$

[In the above summations, $\{i, j\}$ denotes all unordered pairs with $i \neq j$, $\{i, j, k\}$ denotes all unordered triples of distinct elements, and so on.]

I.e., to compute $\Pr[\bigcup_i A_i]$, we start by summing the event probabilities $\Pr[A_i]$, then we *subtract* the probabilities of all pairwise intersections, then we *add* back in the probabilities of all three-way intersections, and so on.

We won't prove this formula here; but you might like to verify it for the special case $n = 3$ by drawing a Venn diagram and checking that every sample point in $A_1 \cup A_2 \cup A_3$ is counted exactly once by the formula. You might also like to prove the formula for general n by induction (in similar fashion to the proof of the Product Rule above).

Taking the formula on faith, what is the probability we get lucky in the new game in Vegas?

$$\Pr[A_1 \cup A_2 \cup A_3] = \Pr[A_1] + \Pr[A_2] + \Pr[A_3] - \Pr[A_1 \cap A_2] - \Pr[A_1 \cap A_3] - \Pr[A_2 \cap A_3] + \Pr[A_1 \cap A_2 \cap A_3].$$

Now the nice thing here is that the events A_i are mutually independent (the outcome of any die does not depend on that of the others), so $\Pr[A_i \cap A_j] = \Pr[A_i]\Pr[A_j] = (\frac{1}{6})^2 = \frac{1}{36}$, and similarly $\Pr[A_1 \cap A_2 \cap A_3] = (\frac{1}{6})^3 = \frac{1}{216}$. So we get

$$\Pr[A_1 \cup A_2 \cup A_3] = \left(3 \times \tfrac{1}{6}\right) - \left(3 \times \tfrac{1}{36}\right) + \tfrac{1}{216} = \tfrac{91}{216} \approx 0.42.$$

So your odds are quite a bit worse than the casino is claiming!

When n is large (i.e., we are interested in the union of many events), the Inclusion/Exclusion formula is essentially useless because it involves computing the probability of the intersection of every non-empty subset of the events: and there are $2^n - 1$ of these! Sometimes we can just look at the first few terms of it and forget the rest: note that successive terms actually give us an overestimate and then an underestimate of the answer, and these estimates both get better as we go along.

However, in many situations we can get a long way by just looking at the first term:

1. **Disjoint events.** If the events A_i are all *disjoint* (i.e., no pair of them contain a common sample point — such events are also called *mutually exclusive*), then

   $$\Pr[\bigcup_{i=1}^n A_i] = \sum_{i=1}^n \Pr[A_i].$$

   [Note that we have already used this fact several times in our examples, e.g., in claiming that the probability of a flush is four times the probability of a Hearts flush — clearly flushes in different suits are disjoint events.]

2. **Union bound.** Always, it is the case that

   $$\Pr[\bigcup_{i=1}^n A_i] \leq \sum_{i=1}^n \Pr[A_i].$$

   This merely says that adding up the $\Pr[A_i]$ can only *over*estimate the probability of the union. Crude as it may seem, in the next lecture note we'll see how to use the union bound effectively in a Computer Science example.

Two Killer Applications: Hashing and Load Balancing

In this lecture we will see that the simple balls-and-bins process can be used to model a surprising range of phenomena. Recall that in this process we distribute k balls in n bins, where each ball is independently placed in a uniformly random bin. We can ask questions such as:

- How large can we choose k while ensuring that with probability at least $1/2$, no two balls land in the same bin?
- If $k = n$, what is the maximum number of balls that are likely to land in the same bin?

As we shall see, these two simple questions provide important insights into two important computer science applications: hashing and load balancing. Our answers to these two questions are based on the application of a simple technique called the union bound, which states that $P[A \cup B] \leq P[A] + P[B]$.

One of the basic issues in hashing is the tradeoff between the size of the hash table and the number of collisions. Here is a simple question that quantifies the tradeoff:

- Suppose a hash function distributes keys evenly over a table of size n. How many keys can we hash before the probability of a collision exceeds (say) $\frac{1}{2}$?

Recall that a hash table is a data structure that supports the storage of a set of keys drawn from a large universe U (say, the names of all 250m people in the US). The set of keys to be stored changes over time, and so the data structure allows keys to be added and deleted quickly. It also rapidly answers given a key whether it is an element in the currently stored set. The crucial question is how large must the hash table be to allow these operations (addition, deletion and membership) to be implemented quickly.

Here is how the hashing works. The hash function h maps U to a table T of modest size. To ADD a key x to our set, we evaluate $h(x)$ (i.e., apply the hash function to the key) and store x at the location $h(x)$ in the table T. All keys in our set that are mapped to the same table location are stored in a simple linked list. The operations DELETE and MEMBER are implemented in similar fashion, by evaluating $h(x)$ and searching the linked list at $h(x)$. Note that the efficiency of a hash function depends on having only few collisions — i.e., keys that map to the same location. This is because the search time for DELETE and MEMBER operations is proportional to the length of the corresponding linked list.

Of course, we could be unlucky and choose keys such that our hash function maps many of them to the same location in the table. But the whole idea behind hashing is that we select our hash function carefully, so that it scrambles up the input key and seems to map it to a random location in the table, making it unlikely that most of the keys we select are mapped to the same location. To quantitatively understand this phenomenon, we will model our hash function as a random function - one that maps each key to a uniformly random location in the table, independently of where all other keys are mapped. The question we will answer is the following: what is the largest number, m, of keys we can store before the probability of a collision reaches $\frac{1}{2}$?

Note that there is nothing special about $\frac{1}{2}$. One can ask, and answer, the same question with different values of collision probability, and the largest number of keys m will change accordingly.

Balls and Bins

Let's begin by seeing how this problem can be put into the balls and bins framework. The balls will be the m keys to be stored, and the bins will be the n locations in the hash table T. Since the hash function maps each key to a random location in the table T, we can see each key (ball) as choosing a hash table location (bin) uniformly and independently from T. Thus the probability space corresponding to this hashing experiment is exactly the same as the balls and bins space.

We are interested in the event A that there is no collision, or equivalently, that all m balls land in different bins. Clearly $\Pr[A]$ will decrease as m increases (with n fixed). Our goal is to find the largest value of m such that $\Pr[A]$ remains above $\frac{1}{2}$. [Note: Really we are looking at different sample spaces here, one for each value of m. So it would be more correct to write $\Pr_m$ rather than just Pr, to make clear which sample space we are talking about. However, we will omit this detail.]

Using the union bound

Let us see how to use the union bound to achieve this goal. We will fix the value of m and try to compute $\Pr[A]$. There are exactly $k = \binom{m}{2} = \frac{m(m-1)}{2}$ possible pairs among our m keys. Imagine these are numbered from 1 to $\binom{m}{2}$ (it doesn't matter how). Let A_i denote the event that pair i has a collision (i.e., both keys in the pair are hashed to the same location). Then the event $\overline{A}$ that *some* collision occurs can be written $\overline{A} = \bigcup_{i=1}^{k} A_i$. What is $\Pr[A_i]$? We claim it is just $\frac{1}{n}$ for every i; this is just the probability that two particular balls land in the same bin.

So, using the union bound from the last lecture, we have

$$\Pr[\overline{A}] \leq \sum_{i=1}^{k} \Pr[A_i] = k \times \frac{1}{n} = \frac{m(m-1)}{2n} \approx \frac{m^2}{2n}.$$

This means that the probability of having a collision is less than $\frac{1}{2}$ provided $\frac{m^2}{2n} \leq \frac{1}{2}$, i.e., provided $m \leq \sqrt{n}$. Thus, if we wish to suffer no collisions, the size of the hash table must be about the square of the cardinality of the set we are trying to store. We will see in the next section on load balancing that the number of collisions does not increase dramatically as we decrease the size of the hash table and make it comparable to the size of the set we are trying to store.

It turns out we can derive a slightly less restrictive bound for m using other techniques from the past several lectures. Although this alternate bound is a little better, both bounds are the same in terms of dependence on n (both are of the form $m = O(\sqrt{n})$). If you are interested in the alternate derivation, please read the section at the end of this note.

Birthday Paradox

Can we do better than the $m = O(\sqrt{n})$ dependence on n? It turns out that the answer is no, and this is related to a surprising phenomenon called the birthday paradox. Suppose there are 23 students in class. What is the chance that two of them have the same birthday? Naively one might answer that since there are 365 days in the year, the chance should be roughly 23/365 $\tilde{6}$%. The correct answer is over 50%!

Suppose there are k people in the room, and let $A =$ "At least two people have the same birthday," and let $\bar{A} =$ "No two people have the same birthday." It turns out it is easier to calculate $\Pr[\bar{A}]$, and then $\Pr[A] = 1 - \Pr[\bar{A}]$.

Our sample space is of cardinality $|\Omega| = 365^k$. And the number of sample points such that no two people to have the same birthday can be calculated as follows: there are 365 choices for the first person, 364 for the second, ..., $365-k+1$ choices for the k^{th} person, for a total of $365 \times 364 \times \cdots \times (365-k+1)$. Thus $\Pr[\bar{A}] = \frac{|\bar{A}|}{|\Omega|} = \frac{365\times 364\times\cdots\times(365-k+1)}{365^k}$. And $\Pr[A] = 1 - \frac{365\times 364\times\cdots\times(365-k+1)}{365^k}$. Substituting $k = 23$, we can check that $\Pr[A]$ is over 50%. And with $k = 60$ $\Pr[A]$ is larger than 99%!

The hashing problem we considered above is a closely related to the birthday problem. Indeed, the birthday problem is the special case of the chasing problem with 365 bins. The $k = 23$ solution to the birthday problem can be seen as k being roughly $\sqrt{365}$.

Application 2: Load balancing

An important practical issue in distributed computing is how to spread the workload in a distributed system among its processors. Here we investigate an extremely simple scenario that is both fundamental in its own right and also establishes a baseline against which more sophisticated methods should be judged.

Suppose we have m identical jobs and n identical processors. Our task is to assign the jobs to the processors in such a way that no processor is too heavily loaded. Of course, there is a simple optimal solution here: just divide up the jobs as evenly as possible, so that each processor receives either $\lceil \frac{m}{n} \rceil$ or $\lfloor \frac{m}{n} \rfloor$ jobs. However, this solution requires a lot of centralized control, and/or a lot of communication: the workload has to be balanced evenly either by a powerful centralized scheduler that talks to all the processors, or by the exchange of many messages between jobs and processors. This kind of operation is very costly in most distributed systems. The question therefore is: What can we do with little or no overhead in scheduling and communication cost?

Back to Balls and Bins

The first idea that comes to mind here is... balls and bins! In other words, each job simply selects a processor uniformly at random and independently of all others, and goes to that processor. (Make sure you believe that the probability space for this experiment is the same as the one for balls and bins.) This scheme requires no communication. However, presumably it won't in general achieve an optimal balancing of the load. Let A_k be the event that the load of some processor is at least k. As designers or users of this load balancing scheme, here's one question we might care about:

Question: Find the smallest value k such that $\Pr[A_k] \leq \frac{1}{2}$.

If we have such a value k, then we'll know that, with good probability (at least $\frac{1}{2}$), every processor in our system will have a load at most k. This will give us a good idea about the performance of the system. Of course, as with our hashing application, there's nothing special about the value $\frac{1}{2}$; we're just using this for illustration. As you can check later (if you choose to read the optional section below), essentially the same analysis can be used to find k such that $\Pr[A_k] \leq 0.05$ (i.e., 95% confidence), or any other value we like. Indeed, we can even find the k's for several different confidence levels and thus build up a more detailed picture of the behavior of the scheme. To simplify our problem, we'll also assume from now on that $m = n$ (i.e., the number of jobs is the same as the number of processors). With a bit more work, we could generalize our analysis to other values of m.

Applying the Union Bound

From Application 1 we know that we get collisions already when $m \approx 1.177\sqrt{n}$. So when $m = n$ the maximum load will certainly be larger than 1 (with good probability). But how large will it be? If we try to

analyze the maximum load directly, we run into the problem that it depends on the number of jobs at *every* processor (or equivalently, the number of balls in every bin). Since the load in one bin depends on those in the others, this becomes very tricky. Instead, what we'll do is analyze the load in any *one* bin, say bin 1; this will be fairly easy. Let $A_k(1)$ be the event that the load in bin 1 is at least k. What we'll do is find k such that

$$\Pr[A_k(1)] \leq \tfrac{1}{2n}. \tag{1}$$

Since all the bins are identical, we will then know that, for the same k,

$$\Pr[A_k(i)] \leq \tfrac{1}{2n} \qquad \text{for } i = 1,2,\ldots,n,$$

where $A_k(i)$ is the event that the load in bin i is at least k. But now, since the event A_k is exactly the union of the events $A_k(i)$ (do you see why?), we can use the "Union Bound" from the previous lecture:

$$\begin{aligned}\Pr[A_k] &= \Pr[\textstyle\bigcup_{i=1}^{n} A_k(i)] \\ &\leq n \times \frac{1}{2n} \\ &= \frac{1}{2}.\end{aligned}$$

It's worth standing back to notice what we did here: we wanted to conclude that $\Pr[A_k] \leq \frac{1}{2}$. We couldn't analyze A_k directly, but we knew that $A_k = \bigcup_{i=1}^{n} A_k(i)$, for much simpler events $A_k(i)$. Since there are n events $A_k(i)$, and all have the same probability, it is enough for us to show that $\Pr[A_k(i)] \leq \frac{1}{2n}$; the union bound then guarantees that $\Pr[A_k] \leq \frac{1}{2}$. This kind of reasoning is very common in applications of probability in Computer Science.

Now all that's left to do is find k which satisfies equation 1. That is, we wish to bound the probability that bin 1 has at least k balls (and find a value of k so that this probability is smaller than $1/2n$). We start by observing that for the event $A_k(1)$ to occur (that bin 1 has at least k balls), there must be some subset S of exactly k balls such that all balls in S ended up in bin 1. We can say this more formally as follows: for a subset S (where $|S| = k$), let B_S be the event that all balls in S land in bin 1. Then the event $A_k(1)$ is a subset of the event $\cup_S B_S$ (where the union is over all sets S of cardinality k). It follows that:

$$\Pr[A_k(1)] \leq \Pr[\cup_S B_S]$$

We can use the union bound on $\Pr[\cup_S B_S]$:

$$\Pr[\cup_S B_S] \leq \sum_S \Pr[B_S]$$

There are $\binom{n}{k}$ sets we are summing over, and for each set S, $\Pr[B_S]$ is simple: it is just the probability that k balls land in bin 1, or $\frac{1}{n^k}$. Using these observations and the above equations, we can compute an upper bound on $\Pr[A_k(1)]$:

$$\Pr[A_k(1)] \leq \binom{n}{k}\frac{1}{n^k}$$

Now to satisfy our original goal (equation 1), we just need to choose k so that $\binom{n}{k}\frac{1}{n^k} \leq \frac{1}{2n}$. But we have

$$\binom{n}{k}\frac{1}{n^k} = \frac{n(n-1)\cdots(n-k+1)}{k!}\frac{1}{n^k} \leq \frac{1}{k!}$$

Setting $k! = 2n$, and simplifying we get that $\left(\frac{k}{e}\right)^k = 2n$. Taking logs we get that $k(\ln k - 1) = \ln 2n$. This gives a value of k of roughly $\frac{\ln n}{\ln \ln n}$.

If you would like to learn more about how to do this, please refer to the additional section on load balancing below.

Finally, here is one punchline from Application 2. Let's say the total US population is about 250 million. Suppose we mail 250 million items of junk mail, each one with a random US address. Then (see the optional section below for more details) with probability at least $\frac{1}{2}$, no one person anywhere will receive more than about a dozen items!

More About Hashing (Optional)

Please read on only if interested. In this section, we will derive the alternate bound described in the hashing section above.

Main Idea

Let's fix the value of m and try to compute $\Pr[A]$. Since our probability space is uniform (each outcome has probability $\frac{1}{n^m}$), it's enough just to count the number of outcomes in A. In how many ways can we arrange m balls in n bins so that no bin contains more than one ball? Well, this is just the number of ways of choosing m things out of n *without* replacement, which as we saw in Note 10 is

$$n \times (n-1) \times (n-2) \times \cdots \times (n-m+2) \times (n-m+1).$$

This formula is valid as long as $m \leq n$: if $m > n$ then clearly the answer is zero. From now on, we'll assume that $m \leq n$.

Now we can calculate the probability of no collision:

$$\begin{aligned} \Pr[A] &= \frac{n(n-1)(n-2)\dots(n-m+1)}{n^m} \\ &= \frac{n}{n} \times \frac{n-1}{n} \times \frac{n-2}{n} \times \cdots \times \frac{n-m+1}{n} \\ &= \left(1-\frac{1}{n}\right) \times \left(1-\frac{2}{n}\right) \times \cdots \times \left(1-\frac{m-1}{n}\right). \end{aligned} \tag{2}$$

Before going on, let's pause to observe that we could compute $\Pr[A]$ in a different way, as follows. View the probability space as a sequence of choices, one for each ball. For $1 \leq i \leq m$, let A_i be the event that the ith ball lands in a different bin from balls $1, 2, \dots, i-1$. Then

$$\begin{aligned} \Pr[A] = \Pr[\textstyle\bigcap_{i=1}^{n} A_i] &= \Pr[A_1] \times \Pr[A_2|A_1] \times \Pr[A_3|A_1 \cap A_2] \times \cdots \times \Pr[A_m|\textstyle\bigcap_{i=1}^{m-1} A_i] \\ &= 1 \times \frac{n-1}{n} \times \frac{n-2}{n} \times \cdots \times \frac{n-m+1}{n} \\ &= \left(1-\frac{1}{n}\right) \times \left(1-\frac{2}{n}\right) \times \cdots \times \left(1-\frac{m-1}{n}\right). \end{aligned}$$

Fortunately, we get the same answer as before! [You should make sure you see how we obtained the conditional probabilities in the second line above. For example, $\Pr[A_3|A_1 \cap A_2]$ is the probability that the third ball lands in a different bin from the first two balls, *given that* those two balls also landed in different bins. This means that the third ball has $n-2$ possible bin choices out of a total of n.]

Essentially, we are now done with our problem: equation (2) gives an exact formula for the probability of no collision when m keys are hashed. All we need to do now is plug values $m = 1, 2, 3, \dots$ into (2) until we find that $\Pr[A]$ drops below $\frac{1}{2}$. The corresponding value of m (minus 1) is what we want.

We can actually make this bound much more useful by turning it around, as we will do below. We will derive an equation which tells us the value of m at which $\Pr[A]$ drops below $\frac{1}{2}$. It turns out that if m is smaller than approximately $1.177\sqrt{n}$, the probability of a collision will be less than $\frac{1}{2}$.

Further Simplification

The bound we gave above (for the largest number m of keys we can store before the probability of a collision reaches $\frac{1}{2}$) is not really satisfactory: it would be much more useful to have a formula that gives the "critical" value of m directly, rather than having to compute $\Pr[A]$ for $m = 1, 2, 3, \ldots$. Note that we would have to do this computation separately for each different value of n we are interested in: i.e., whenever we change the size of our hash table.

So what remains is to "turn equation (2) around", so that it tells us the value of m at which $\Pr[A]$ drops below $\frac{1}{2}$. To do this, let's take logs: this is a good thing to do because it turns the product into a sum, which is easier to handle. We get

$$\ln(\Pr[A]) = \ln\left(1 - \frac{1}{n}\right) + \ln\left(1 - \frac{2}{n}\right) + \cdots + \ln\left(1 - \frac{m-1}{n}\right), \tag{3}$$

where "ln" denotes natural (base e) logarithm. Now we can make use of a standard approximation for logarithms: namely, if x is small then $\ln(1-x) \approx -x$. This comes from the Taylor series expansion

$$\ln(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \ldots.$$

So by replacing $\ln(1-x)$ by $-x$ we are making an error of at most $(\frac{x^2}{2} + \frac{x^3}{3} + \cdots)$, which is at most $2x^2$ when $x \leq \frac{1}{2}$. In other words, we have

$$-x \geq \ln(1-x) \geq -x - 2x^2.$$

And if x is small then the error term $2x^2$ will be much smaller than the main term $-x$. Rather than carry around the error term $2x^2$ everywhere, in what follows we'll just write $\ln(1-x) \approx -x$, secure in the knowledge that we could make this approximation precise if necessary.

Now let's plug this approximation into equation (3):

$$\begin{aligned}
\ln(\Pr[A]) &\approx -\frac{1}{n} - \frac{2}{n} - \frac{3}{n} - \ldots - \frac{m-1}{n} \\
&= -\frac{1}{n}\sum_{i=1}^{m-1} i \\
&= -\frac{m(m-1)}{2n} \\
&\approx -\frac{m^2}{2n}.
\end{aligned} \tag{4}$$

Note that we've used the approximation for $\ln(1-x)$ with $x = \frac{1}{n}, \frac{2}{n}, \frac{3}{n}, \ldots, \frac{m-1}{n}$. So our approximation should be good provided all these are small, i.e., provided n is fairly big and m is quite a bit smaller than n. Once we're done, we'll see that the approximation is actually pretty good even for modest sizes of n.

Now we can undo the logs in (4) to get our expression for $\Pr[A]$:

$$\Pr[A] \approx e^{-\frac{m^2}{2n}}.$$

The final step is to figure out for what value of m this probability becomes $\frac{1}{2}$. So we want the largest m such that $e^{-\frac{m^2}{2n}} \geq \frac{1}{2}$. This means we must have

$$-\frac{m^2}{2n} \geq \ln(\tfrac{1}{2}) = -\ln 2, \tag{5}$$

or equivalently

$$m \leq \sqrt{(2\ln 2)n} \approx 1.177\sqrt{n}. \tag{6}$$

So the bottom line is that we can hash approximately $\underline{m = \lfloor 1.177\sqrt{n} \rfloor \text{ keys}}$ before the probability of a collision reaches $\frac{1}{2}$.

Recall that our calculation was only approximate; so we should go back and get a feel for how much error we made. We can do this by using equation (2) to compute the exact value $m = m_0$ at which $\Pr[A]$ drops below $\frac{1}{2}$, for a few sample values of n. Then we can compare these values with our estimate $m = 1.177\sqrt{n}$.

n	10	20	50	100	200	365	500	1000	10^4	10^5	10^6
$1.177\sqrt{n}$	3.7	5.3	8.3	11.8	16.6	22.5	26.3	37.3	118	372	1177
exact m_0	4	5	8	12	16	22	26	37	118	372	1177

From the table, we see that our approximation is very good even for small values of n. When n is large, the error in the approximation becomes negligible.

Why $\frac{1}{2}$?

As mentioned above, we could consider values other than $\frac{1}{2}$. What we did above was to (approximately) compute $\Pr[A]$ (the probability of no collision) as a function of m, and then find the largest value of m for which our estimate is smaller than $\frac{1}{2}$. If instead we were interested in keeping the collision probability below (say) 0.05 (= 5%), we could derive that we could hash at most $m = \sqrt{(2\ln(20/19))n} \approx 0.32\sqrt{n}$ keys. Of course, this number is a bit smaller than before because our collision probability is now smaller. But no matter what "confidence" probability we specify, our critical value of m will always be $c\sqrt{n}$ for some constant c (which depends on the confidence).

More About Load Balancing (Optional)

In this section, we'll come up with a slightly tighter bound for k and we will also show how to choose k so that the probability of an overloaded processor is less than $\frac{1}{2}$.

We'll start with an alternate calculation of $\Pr[A_k(1)]$. Let $C_j(1)$ be the event that the number of balls in bin 1 is exactly j. It's not hard to write down an *exact* expression for $\Pr[C_j(1)]$]:

$$\Pr[C_j(1)] = \binom{n}{j} \left(\tfrac{1}{n}\right)^j \left(1 - \tfrac{1}{n}\right)^{n-j}. \tag{7}$$

This can be seen by viewing each ball as a biased coin toss: Heads corresponds to the ball landing in bin 1, Tails to all other outcomes. So the Heads probability is $\frac{1}{n}$; and all coin tosses are (mutually) independent. As we saw in earlier lectures, (7) gives the probability of exactly j Heads in n tosses.

Thus we have

$$\Pr[A_k(1)] = \sum_{j=k}^{n} \Pr[C_j(1)] = \sum_{j=k}^{n} \binom{n}{j} \left(\tfrac{1}{n}\right)^j \left(1-\tfrac{1}{n}\right)^{n-j}. \tag{8}$$

Now in some sense we are done: we could try plugging values $k = 1, 2, \ldots$ into (8) until the probability drops below $\frac{1}{2n}$. However, it is also possible to massage equation (8) into a cleaner form from which we can read off the value of k more directly. We will need to do a few calculations and approximations:

$$\begin{aligned}
\Pr[A_k(1)] &= \sum_{j=k}^{n} \binom{n}{j} \left(\tfrac{1}{n}\right)^j \left(1-\tfrac{1}{n}\right)^{n-j} \\
&\leq \sum_{j=k}^{n} \left(\tfrac{n\mathrm{e}}{j}\right)^j \left(\tfrac{1}{n}\right)^j \\
&= \sum_{j=k}^{n} \left(\tfrac{\mathrm{e}}{j}\right)^j
\end{aligned} \tag{9}$$

In the second line here, we used the standard approximation[1] $(\frac{n}{j})^j \leq \binom{n}{j} \leq (\frac{n\mathrm{e}}{j})^j$. Also, we tossed away the $(1-\frac{1}{n})^{n-j}$ term, which is permissible because doing so can only increase the value of the sum (i.e., since $(1-\frac{1}{n})^{n-j} \leq 1$). It will turn out that we didn't lose too much by applying these bounds.

Now we're already down to a much cleaner form in (9). To finish up, we just need to sum the series. Again, we'll make an approximation to simplify our task, and hope that it doesn't cost us too much. The terms in the series in (9) go down at each step by a factor of at least $\frac{\mathrm{e}}{k}$. So we can bound the series by a *geometric* series, which is easy to sum:

$$\sum_{j=k}^{n} \left(\tfrac{\mathrm{e}}{j}\right)^j \leq \left(\tfrac{\mathrm{e}}{k}\right)^k \left(1 + \tfrac{\mathrm{e}}{k} + \left(\tfrac{\mathrm{e}}{k}\right)^2 + \cdots\right) \leq 2\left(\tfrac{\mathrm{e}}{k}\right)^k, \tag{10}$$

where the second inequality holds provided we assume that $k \geq 2\mathrm{e}$ (which it will be, as we shall see in a moment).

So what we have now shown is the following:

$$\Pr[A_k(1)] \leq 2\left(\tfrac{\mathrm{e}}{k}\right)^k \tag{11}$$

(provided $k \geq 2\mathrm{e}$). Note that, even though we have made a few approximations, inequality (11) is completely valid: all our approximations were "$\leq$", so we always have an *upper* bound on $\Pr[X_1 \geq k]$. [You should go back through all the steps and check this.]

Recall from (1) that our goal is to make the probability in (11) less than $\frac{1}{2n}$. We can ensure this by choosing k so that

$$\left(\tfrac{\mathrm{e}}{k}\right)^k \leq \tfrac{1}{4n}. \tag{12}$$

Now we are in good shape: given any value n for the number of jobs/processors, we just need to find the smallest value $k = k_0$ that satisfies inequality (12). We will then know that, with probability at least $\frac{1}{2}$, the maximum load on any processor is at most k_0. The table below shows the values of k_0 for some sample values of n. It is recommended that you perform the experiment and compare these values of k_0 with what happens in practice.

[1] Computer scientists and mathematicians carry around a little bag of tricks for replacing complicated expressions like $\binom{n}{j}$ with simpler approximations. This is just one of these. It isn't too hard to prove the lower bound, i.e., that $\binom{n}{j} \geq (\frac{n}{j})^j$. The upper bound is a bit trickier, and makes use of another approximation for $n!$ known as *Stirling's approximation*, which implies that $j! \geq (\frac{j}{\mathrm{e}})^j$. We won't discuss the details here.

n	10	20	50	100	500	1000	10^4	10^5	10^6	10^7	10^8	10^{15}
exact k_0	6	6	7	7	8	8	9	10	11	12	13	19
$\ln(4n)$	3.7	4.4	5.3	6.0	7.6	8.3	10.6	13.9	15.2	17.5	19.8	36
$\frac{2\ln n}{\ln\ln n}$	5.6	5.4	5.8	6.0	6.8	7.2	8.2	9.4	10.6	11.6	12.6	20

Can we come up with a formula for k_0 as a function of n (as we did for the hashing problem)? Well, let's take logs in (12):

$$k(\ln k - 1) \geq \ln(4n). \tag{13}$$

From this, we might guess that $k = \ln(4n)$ is a good value for k_0. Plugging in this value of k makes the left-hand side of (13) equal to $\ln(4n)(\ln\ln(4n) - 1)$, which is certainly bigger than $\ln(4n)$ provided $\ln\ln(4n) \geq 2$, i.e., $n \geq \frac{1}{4}e^{e^2} \approx 405$. So for $n \geq 405$ we can claim that the maximum load is (with probability at least $\frac{1}{2}$) no larger than $\ln(4n)$. The table above plots the values of $\ln(4n)$ for comparison with k_0. As expected, the estimate is quite good for small n, but becomes rather pessimistic when n is large.

For large n we can do better as follows. If we plug the value $k = \frac{\ln n}{\ln\ln n}$ into the left-hand side of (13), it becomes

$$\frac{\ln n}{\ln\ln n}(\ln\ln n - \ln\ln\ln n - 1) = \ln n\left(1 - \frac{\ln\ln\ln n + 1}{\ln\ln n}\right). \tag{14}$$

Now when n is large this is *just barely* smaller than the right-hand side, $\ln(4n)$. Why? Because the second term inside the parentheses goes to zero as $n \to \infty$,[2] and because $\ln(4n) = \ln n + \ln 4$, which is very close to $\ln n$ when n is large (since $\ln 4$ is a fixed small constant). So we can conclude that, for large values of n, the quantity $\frac{\ln n}{\ln\ln n}$ should be a pretty good estimate of k_0. Actually for this estimate to become good n has to be (literally) astronomically large. For more civilized values of n, we get a better estimate by taking $k = \frac{2\ln n}{\ln\ln n}$. The extra factor of 2 helps to wipe out the lower order terms (i.e., the second term in the parenthesis in (14) and the $\ln 4$) more quickly. The table above also shows the behavior of this estimate for various values of n.

[2] To see this, note that it is of the form $\frac{\ln z + 1}{z}$ where $z = \ln\ln n$, and of course $\frac{\ln z + 1}{z} \to 0$ as $z \to \infty$.

Random Variables: Distribution and Expectation

Example: Coin Flips

Recall our setup of a probabilistic experiment as a procedure of drawing a sample from a set of possible values, and assigning a probability for each possible outcome of the experiment. For example, if we toss a fair coin n times, then there are 2^n possible outcomes, each of which is equally likely and has probability $\frac{1}{2^n}$.

Now suppose we want to make a measurement in our experiment. For example, we can ask what is the number of heads in n coin tosses; call this number X. Of course, X is not a fixed number, but it depends on the actual sequence of coin flips that we obtain. For example, if $n = 4$ and we observe the outcome $\omega = HTHH$, then the number of heads is $X = 3$; whereas if we observe the outcome $\omega = HTHT$, then the number of heads is $X = 2$. In this example of n coin tosses we only know that X is an integer between 0 and n, but we do not know what its exact value is until we observe which outcome of n coin flips is realized and count how many heads there are. Because every possible outcome is assigned a probability, the value X also carries with it a probability for each possible value it can take. The table below lists all the possible values X can take in the example of $n = 4$ coin tosses, along with their respective probabilities.

outcomes ω	value of X (number of heads)	probability occurring
TTTT	0	1/16
HTTT, THTT, TTHT, TTTH	1	4/16
HHTT, HTHT, HTTH, THHT, THTH, TTHH	2	6/16
HHHT, HHTH, HTHH, THHH	3	4/16
HHHH	4	1/16

Such a value X that depends on the outcome of the probabilistic experiment is called a *random variable* (or *r.v.*). As we see from the example above, X does not have a definitive value, but instead only has a probability *distribution* over the set of possible values X can take, which is why it is called random. So the question "What is the number of heads in n coin tosses?" does not exactly make sense because the answer X is a random variable. But the question "What is the *typical* number of heads in n coin tosses?" makes sense: it is asking what is the average value of X (the number of heads) if we repeat the experiment of tossing n coins multiple times. This average value is called the *expectation* of X, and is one of the most useful summary (also called *statistics*) of an experiment.

Example: Permutations

Before we formalize all these notions, let us consider another example to enforce our conceptual understanding of a random variable. Suppose we collect the homeworks of n students, randomly shuffle them, and return them to the students. How many students receive their own homework?

Here the probability space consists of all $n!$ permutations of the homeworks, each with equal probability $\frac{1}{n!}$. If we label the homeworks as $1, 2, \ldots, n$, then each sample point is a permutation $\pi = (\pi_1, \ldots, \pi_n)$ where π_i

is the homework that is returned to the i-th student. Note that $\pi_1, \ldots, \pi_n \in \{1, 2, \ldots, n\}$ are all distinct, so each element in $\{1, \ldots, n\}$ appears exactly once in the permutation π.

In this setting, the i-th student receives her own homework if and only if $\pi_i = i$. Then the question "How many students receive their own homework?" translates into the question of how many indices i's satisfy $\pi_i = i$. These are known as fixed points of the permutation. As in the coin flipping case above, our question does not have a simple numerical answer (such as 4), because the number depends on the particular permutation we choose (i.e., on the sample point). Let's call the number of fixed points X, which is a random variable.

To illustrate the idea concretely, let's consider the example $n = 3$. The following table gives a complete listing of the sample space (of size $3! = 6$), together with the corresponding value of X for each sample point. Here we see that X takes on values 0, 1 or 3, depending on the sample point. You should check that you agree with this table.

permutation π	value of X (number of fixed points)
123	3
132	1
213	1
231	0
312	0
321	1

Random Variables

Let us formalize the concepts discussed above.

Definition 16.1 (Random Variable)**.** *A random variable X on a sample space Ω is a function $X : \Omega \to \mathbb{R}$ that assigns to each sample point $\omega \in \Omega$ a real number $X(\omega)$.*

Until further notice, we will restrict out attention to random variables that are discrete, i.e., they take values in a range that is finite or countably infinite. This means even though we define X to map Ω to $\mathbb{R}$, the actual set of values $\{X(\omega) : \omega \in \Omega\}$ that X takes is a discrete subset of $\mathbb{R}$.

A random variable can be visualized in general by the picture in Figure 1.[1] Note that the term "random variable" is really something of a misnomer: it is a function so there is nothing random about it and it is definitely not a variable! What is random is which sample point of the experiment is realized and hence the value that the random variable maps the sample point to.

Distribution

When we introduced the basic probability space in Note 13, we defined two things:

1. the sample space Ω consisting of all the possible outcomes (sample points) of the experiment;
2. the probability of each of the sample points.

[1] The figures in this note are inspired by figures in Chapter 2 of "Introduction to Probability" by D. Bertsekas and J. Tsitsiklis.

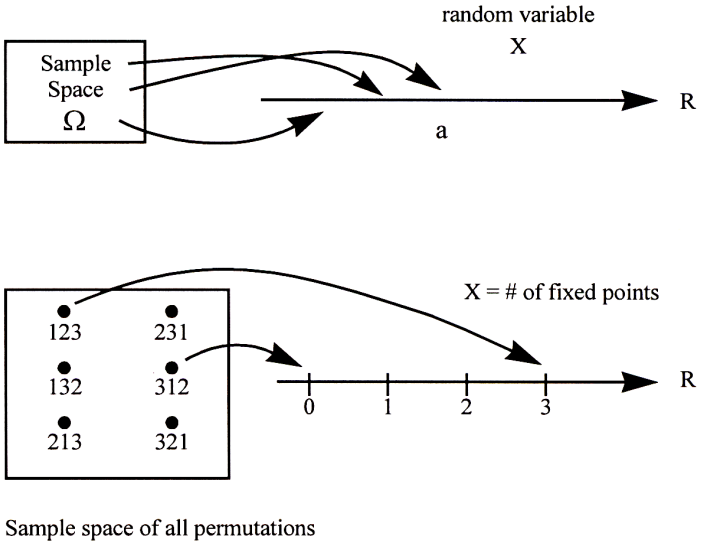


Figure 1: Visualization of how a random variable is defined on the sample space.

Analogously, there are two things important about any random variable:

1. the set of values that it can take;
2. the probabilities with which it takes on the values.

Since a random variable is defined on a probability space, we can calculate these probabilities given the probabilities of the sample points. Let a be any number in the range of a random variable X. Then the set

$$\{\omega \in \Omega : X(\omega) = a\}$$

is an *event* in the sample space (simply because it is a subset of Ω). We usually abbreviate this event to simply "$X = a$". Since $X = a$ is an event, we can talk about its probability, $\Pr[X = a]$. The collection of these probabilities, for all possible values of a, is known as the *distribution* of the random variable X.

Definition 16.2 (Distribution)**.** *The distribution of a discrete random variable X is the collection of values $\{(a, \Pr[X = a]) : a \in \mathscr{A}\}$, where $\mathscr{A}$ is the set of all possible values taken by X.*

Thus, the distribution of the random variable X in our permutation example above is:

$$\Pr[X = 0] = \frac{1}{3}; \qquad \Pr[X = 1] = \frac{1}{2}; \qquad \Pr[X = 3] = \frac{1}{6};$$

and $\Pr[X = a] = 0$ for all other values of a.

The distribution of a random variable can be visualized as a bar diagram, shown in Figure 2. The x-axis represents the values that the random variable can take on. The height of the bar at a value a is the probability $\Pr[X = a]$. Each of these probabilities can be computed by looking at the probability of the corresponding event in the sample space.

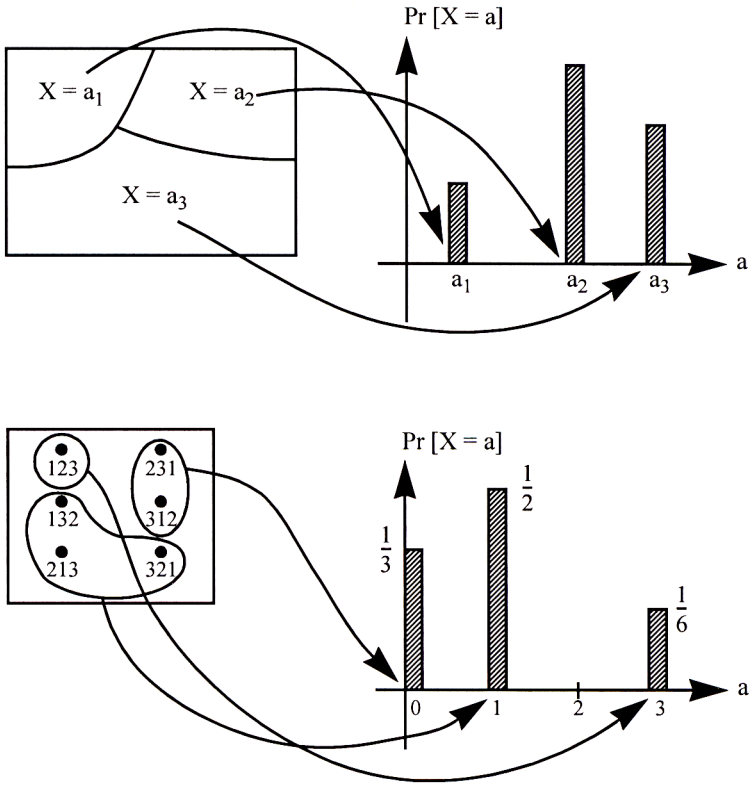


Figure 2: Visualization of how the distribution of a random variable is defined.

Note that the collection of events $X = a$, $a \in \mathscr{A}$, satisfy two important properties:

- Any two events $X = a_1$ and $X = a_2$ with $a_1 \neq a_2$ are disjoint.
- The union of all these events is equal to the entire sample space Ω.

The collection of events thus form a *partition* of the sample space (see Figure 2). Both properties follow directly from the fact that X is a function defined on Ω, i.e., X assigns a unique value to each and every possible sample point in Ω. As a consequence, the sum of the probabilities $\Pr[X = a]$ over all possible values of a is exactly 1. So when we sum up the probabilities of the events $X = a$, we are really summing up the probabilities of all the sample points.

Example: The Binomial Distribution

Let's return to our coin tossing example above, where we defined our random variable X to be the number of heads. More formally, consider the random experiment consisting of n independent tosses of a biased coin which lands on heads with probability p. Each sample point ω is a sequence of tosses. $X(\omega)$ is defined to be the number of heads in ω. For example, when $n = 3$, $X(THH) = 2$.

To compute the distribution of X, we first enumerate the possible values X can take on. They are simply $0, 1, \ldots, n$. Then we compute the probability of each event $X = i$ for $i = 0, 1, \ldots, n$. The probability of the event $X = i$ is the sum of the probabilities of all the sample points with exactly i heads (for example, if $n = 3$ and $i = 2$, there would be three such sample points $\{HHT, HTH, THH\}$). Any such sample point has a

probability $p^i(1-p)^{n-i}$, since the coin flips are independent. There are exactly $\binom{n}{i}$ of these sample points. So

$$\Pr[X=i] = \binom{n}{i} p^i (1-p)^{n-i} \qquad \text{for } i = 0,1,\ldots,n. \tag{1}$$

This distribution, called the *binomial* distribution, is one of the most important distributions in probability. A random variable with this distribution is called a *binomial* random variable, written as

$$X \sim Bin(n,p)$$

An example of a binomial distribution is shown in Figure 3. Notice that due to the properties of X mentioned above, it must be the case that $\sum_{i=0}^{n} \Pr[X=i] = 1$, which implies that $\sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i} = 1$. This provides a probabilistic proof of the Binomial Theorem!

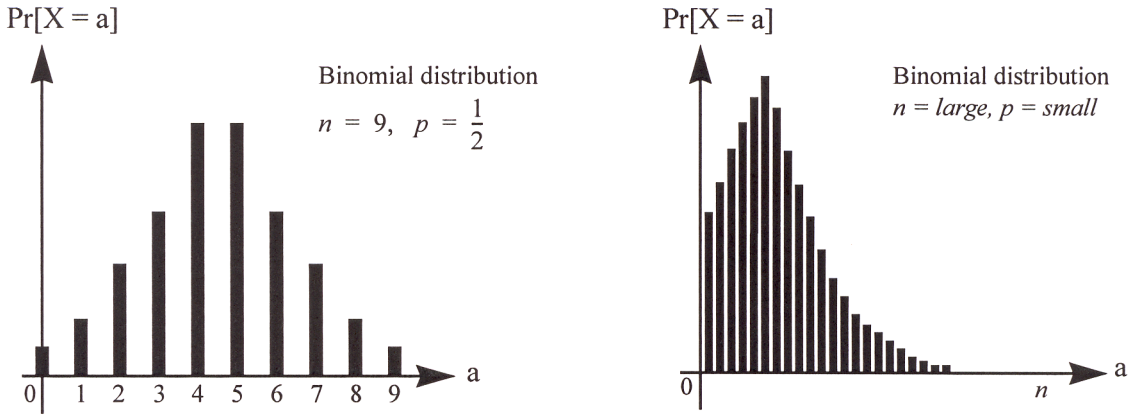


Figure 3: The binomial distributions for two choices of (n,p).

Although we define the binomial distribution in terms of an experiment involving tossing coins, this distribution is useful for modeling many real-world problems. Consider for example the error correction problem studied in Note 9. Recall that we wanted to encode n packets into $n+k$ packets such that the recipient can reconstruct the original n packets from any n packets received. But in practice, the number of packet losses is random, so how do we choose k, the amount of redundancy? If we model each packet getting lost with probability p and the losses are independent, then if we transmit $n+k$ packets, the number of packets received is a random variable X with binomial distribution: $X \sim Bin(n+k, 1-p)$ (we are tossing a coin $n+k$ times, and each coin turns out to be a head (packet received) with probability $1-p$). So the probability of successfully decoding the original data is:

$$\Pr[X \geq n] \;=\; \sum_{i=n}^{n+k} \Pr[X=i] \;=\; \sum_{i=n}^{n+k} \binom{n+k}{i} (1-p)^i p^{n+k-i}.$$

Given fixed n and p, we can choose k such that this probability is no less than, say, 0.99.

Expectation

The distribution of a r.v. contains *all* the probabilistic information about the r.v. In most applications, however, the complete distribution of a r.v. is very hard to calculate. For example, consider the homework permutation example with $n = 20$. In principle, we'd have to enumerate $20! \approx 2.4 \times 10^{18}$ sample points,

compute the value of X at each one, and count the number of points at which X takes on each of its possible values! (though in practice we could streamline this calculation a bit). Moreover, even when we can compute the complete distribution of a r.v., it is often not very informative.

For these reasons, we seek to *compress* the distribution into a more compact, convenient form that is also easier to compute. The most widely used such form is the *expectation* (or *mean* or *average*) of the r.v.

Definition 16.3 (Expectation)**.** *The expectation of a discrete random variable X is defined as*

$$\mathrm{E}(X) = \sum_{a \in \mathscr{A}} a \times \Pr[X = a],$$

where the sum is over all possible values taken by the r.v.

For our simpler permutation example with only 3 students, the expectation is

$$\mathrm{E}(X) = \left(0 \times \frac{1}{3}\right) + \left(1 \times \frac{1}{2}\right) + \left(3 \times \frac{1}{6}\right) = 0 + \frac{1}{2} + \frac{1}{2} = 1.$$

That is, the expected number of fixed points in a permutation of three items is exactly 1.

The expectation can be seen in some sense as a "typical" value of the r.v. (though note that it may not actually be a value that the r.v. ever takes on). The question of how typical the expectation is for a given r.v. is a very important one that we shall return to in a later lecture.

Here is a physical interpretation of the expectation of a random variable: imagine carving out a wooden cutout figure of the probability distribution as in Figure 4. Then the expected value of the distribution is the balance point (directly below the center of gravity) of this object.

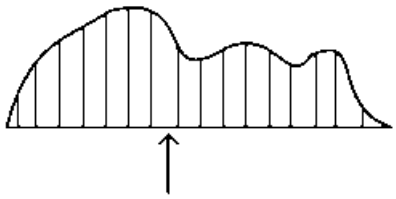


Figure 4: Physical interpretation of expected value as the balance point.

Examples

1. **Single die.** Throw one fair die. Let X be the number that comes up. Then X takes on values $1, 2, \ldots, 6$ each with probability $\frac{1}{6}$, so

$$\mathrm{E}(X) = \frac{1}{6}(1+2+3+4+5+6) = \frac{21}{6} = \frac{7}{2}.$$

   Note that X never actually takes on its expected value $\frac{7}{2}$.

2. **Two dice.** Throw two fair dice. Let X be the sum of their scores. Then the distribution of X is

a	2	3	4	5	6	7	8	9	10	11	12
$\Pr[X = a]$	$\frac{1}{36}$	$\frac{1}{18}$	$\frac{1}{12}$	$\frac{1}{9}$	$\frac{5}{36}$	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{1}{9}$	$\frac{1}{12}$	$\frac{1}{18}$	$\frac{1}{36}$

The expectation is therefore

$$\mathrm{E}(X) = \left(2 \times \frac{1}{36}\right) + \left(3 \times \frac{1}{18}\right) + \left(4 \times \frac{1}{12}\right) + \cdots + \left(12 \times \frac{1}{36}\right) = 7.$$

3. **Roulette.** A roulette wheel is spun (recall that a roulette wheel has 38 slots: the numbers $1, 2, \ldots, 36$, half of which are red and half black, plus 0 and 00, which are green). You bet \$1 on Black. If a black number comes up, you receive your stake plus \$1; otherwise you lose your stake. Let X be your net winnings in one game. Then X can take on the values $+1$ and -1, and $\Pr[X = 1] = \frac{18}{38}$, $\Pr[X = -1] = \frac{20}{38}$. Thus,

$$\mathrm{E}(X) = \left(1 \times \frac{18}{38}\right) + \left(-1 \times \frac{20}{38}\right) = -\frac{1}{19};$$

i.e., you expect to lose about a nickel per game. Notice how the zeros tip the balance in favor of the casino!

Linearity of expectation

So far, we've computed expectations by brute force: i.e., we have written down the whole distribution and then added up the contributions for all possible values of the r.v. The real power of expectations is that in many real-life examples they can be computed much more easily using a simple shortcut. The shortcut is the following:

Theorem 16.1. *For any two random variables X and Y on the same probability space, we have*

$$\mathrm{E}(X+Y) = \mathrm{E}(X) + \mathrm{E}(Y).$$

Also, for any constant c, we have

$$\mathrm{E}(cX) = c\mathrm{E}(X).$$

Proof. To make the proof easier, we will first rewrite the definition of expectation in a more convenient form. Recall from Definition 16.3 that

$$\mathrm{E}(X) = \sum_{a \in \mathscr{A}} a \times \Pr[X = a].$$

Let's look at one term $a \times \Pr[X = a]$ in the above sum. Notice that $\Pr[X = a]$, by definition, is the sum of $\Pr[\omega]$ over those sample points ω for which $X(\omega) = a$. And we know that every sample point $\omega \in \Omega$ is in exactly one of these events $X = a$. This means we can write out the above definition in a more long-winded form as

$$\mathrm{E}(X) = \sum_{\omega \in \Omega} X(\omega) \times \Pr[\omega]. \tag{2}$$

This equivalent definition of expectation will make the present proof much easier (though it is usually less convenient for actual calculations).

Now let's write out $\mathrm{E}(X+Y)$ using equation (2):

$$\begin{aligned}
\mathrm{E}(X+Y) &= \textstyle\sum_{\omega \in \Omega} (X+Y)(\omega) \times \Pr[\omega] \\
&= \textstyle\sum_{\omega \in \Omega} (X(\omega) + Y(\omega)) \times \Pr[\omega] \\
&= \textstyle\sum_{\omega \in \Omega} \Big(X(\omega) \times \Pr[\omega]\Big) + \sum_{\omega \in \Omega} \Big(Y(\omega) \times \Pr[\omega]\Big) \\
&= \mathrm{E}(X) + \mathrm{E}(Y).
\end{aligned}$$

In the last step, we used equation (2) twice.

This completes the proof of the first equality. The proof of the second equality is much simpler and is left as an exercise. □

Theorem 16.1 is very powerful: it says that the expectation of a sum of r.v.'s is the sum of their expectations, with no assumptions about the r.v.'s. We can use Theorem 16.1 to conclude things like $\mathrm{E}(3X-5Y) = 3\mathrm{E}(X)-5\mathrm{E}(Y)$. This property is known as linearity of expectation.

Important caveat: Theorem 16.1 does **not** *say that* $\mathrm{E}(XY) = \mathrm{E}(X)\mathrm{E}(Y)$*, or that* $\mathrm{E}(\frac{1}{X}) = \frac{1}{\mathrm{E}(X)}$*, etc. These claims are not true in general. It is only* sums *and* differences *and* constant multiples *of random variables that behave so nicely.*

Examples

Now let's see some examples of Theorem 16.1 in action.

4. **Two dice again.** Here's a much less painful way of computing $\mathrm{E}(X)$, where X is the sum of the scores of the two dice. Note that $X = X_1 + X_2$, where X_i is the score on die i. We know from example 1 above that $\mathrm{E}(X_1) = \mathrm{E}(X_2) = \frac{7}{2}$. So by Theorem 16.1 we have $\mathrm{E}(X) = \mathrm{E}(X_1) + \mathrm{E}(X_2) = 7$.

5. **More roulette.** Suppose we play the above roulette game not once, but 1000 times. Let X be our expected net winnings. Then $X = X_1 + X_2 + \ldots + X_{1000}$, where X_i is our net winnings in the ith play. We know from earlier that $\mathrm{E}(X_i) = -\frac{1}{19}$ for each i. Therefore, by Theorem 16.1, $\mathrm{E}(X) = \mathrm{E}(X_1) + \mathrm{E}(X_2) + \cdots + \mathrm{E}(X_{1000}) = 1000 \times (-\frac{1}{19}) = -\frac{1000}{19} \approx -53$. So if you play 1000 games, you expect to lose about \$53.

6. **Homeworks.** Let's go back to our homework permutation example with $n = 20$ students. Recall that the r.v. X is the number of students who receive their own homework after shuffling (or equivalently, the number of fixed points). To take advantage of Theorem 16.1, we need to write X as the *sum* of simpler r.v.'s. But since X *counts* the number of times something happens, we can write it as a sum using the following trick:

$$X = X_1 + X_2 + \ldots + X_{20}, \qquad \text{where } X_i = \begin{cases} 1 & \text{if student } i \text{ gets her own hw;} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

[You should think about this equation for a moment. Remember that all the X's are random variables. What does an equation involving random variables mean? What we mean is that, *at every sample point* ω, we have $X(\omega) = X_1(\omega) + X_2(\omega) + \ldots + X_{20}(\omega)$. Why is this true?]

A $\{0,1\}$-valued random variable such as X_i is called an indicator random variable of the corresponding event (in this case, the event that student i gets her own hw). For indicator r.v.'s, the expectation is particularly easy to calculate. Namely,

$$\mathrm{E}(X_i) = (0 \times \Pr[X_i = 0]) + (1 \times \Pr[X_i = 1]) = \Pr[X_i = 1].$$

But in our case, we have

$$\Pr[X_i = 1] = \Pr[\text{student } i \text{ gets her own hw}] = \frac{1}{20}.$$

Now we can apply Theorem 16.1 to (3), to get

$$\mathrm{E}(X) = \mathrm{E}(X_1) + \mathrm{E}(X_2) + \cdots + \mathrm{E}(X_{20}) = 20 \times \frac{1}{20} = 1.$$

So we see that the expected number of students who get their own homeworks in a class of size 20 is 1. But this is exactly the same answer as we got for a class of size 3! And indeed, we can easily see from the above calculation that we would get $\mathrm{E}(X) = 1$ for *any* class size n: this is because we can write $X = X_1 + X_2 + \ldots + X_n$, and $\mathrm{E}(X_i) = \frac{1}{n}$ for each i.

So the expected number of fixed points in a random permutation of n items is always 1, regardless of n. Amazing, but true.

7. **Coin tosses.** Toss a fair coin 100 times. Let the r.v. X be the number of Heads. As in the previous example, to take advantage of Theorem 16.1 we write

$$X = X_1 + X_2 + \ldots + X_{100},$$

where X_i is the indicator r.v. of the event that the i-th toss is Heads. Since the coin is fair, we have

$$\mathrm{E}(X_i) = \Pr[X_i = 1] = \Pr[i\text{-th toss is Heads}] = \tfrac{1}{2}.$$

Using Theorem 16.1, we therefore get

$$\mathrm{E}(X) = \sum_{i=1}^{100} \frac{1}{2} = 100 \times \tfrac{1}{2} = 50.$$

More generally, the expected number of Heads in n tosses of a fair coin is $\frac{n}{2}$. And in n tosses of a biased coin with Heads probability p, it is np (why?). So the expectation of a binomial r.v. $X \sim Bin(n,p)$ is equal to np. Note that it would have been harder to reach the same conclusion by computing this directly from definition of expectation.

8. **Balls and bins.** Throw m balls into n bins. Let the r.v. X be the number of balls that land in the first bin. Then X behaves exactly like the number of Heads in n tosses of a biased coin, with Heads probability $\frac{1}{n}$ (why?). So from Example 7 we get $\mathrm{E}(X) = \frac{m}{n}$.

   In the special case $m = n$, the expected number of balls in any bin is 1. If we wanted to compute this directly from the distribution of X, we'd get into a messy calculation involving binomial coefficients.

   Here's another example on the same sample space. Let the r.v. Y be the number of empty bins. The distribution of Y is horrible to contemplate: to get a feel for this, you might like to write it down for $m = n = 3$ (3 balls, 3 bins). However, computing the expectation $\mathrm{E}(Y)$ is easy using Theorem 16.1. As usual, let's write

$$Y = Y_1 + Y_2 + \cdots + Y_n, \tag{4}$$

   where Y_i is the indicator r.v. of the event "bin i is empty". Again as usual, the expectation of Y_i is easy:

$$\mathrm{E}(Y_i) = \Pr[Y_i = 1] = \Pr[\text{bin } i \text{ is empty}] = \left(1 - \frac{1}{n}\right)^m;$$

   recall that we computed this probability (quite easily) in an earlier lecture. Applying Theorem 16.1 to (4) we therefore have

$$\mathrm{E}(Y) = \sum_{i=1}^{n} \mathrm{E}(Y_i) = n\left(1 - \frac{1}{n}\right)^m,$$

a very simple formula, very easily derived.

Let's see how it behaves in the special case $m = n$ (same number of balls as bins). In this case we get $\mathrm{E}(Y) = n(1-\frac{1}{n})^n$. Now the quantity $(1-\frac{1}{n})^n$ can be approximated (for large enough values of n) by the number $\frac{1}{\mathrm{e}}$.[2] So we see that

$$\mathrm{E}(Y) \to \frac{n}{\mathrm{e}} \approx 0.368n \quad \text{as } n \to \infty.$$

The bottom line is that, if we throw (say) 1000 balls into 1000 bins, the expected number of empty bins is about 368.

[2]More generally, it is a standard fact that for any constant c,

$$(1+\tfrac{c}{n})^n \to \mathrm{e}^c \quad \text{as } n \to \infty.$$

We just used this fact in the special case $c = -1$. The approximation is actually very good even for quite small values of n — try it yourself! E.g., for $n = 20$ we already get $(1-\frac{1}{n})^n = 0.358$, which is very close to $\frac{1}{\mathrm{e}} = 0.367\ldots$. The approximation gets better and better for larger n.

Variance

We have seen in the previous note that if we toss a coin n times with bias p, then the expected number of heads is np. What this means is that if we repeat the experiment multiple times, where in each experiment we toss the coin n times, then on average we get np heads. But in any single experiment, the number of heads observed can be any value between 0 and n. What can we say about how far off we are from the expected value? That is, what is the typical deviation of the number of heads from np?

Random Walk

Let us consider a simpler setting that is equivalent to tossing a fair coin n times, but is more amenable to analysis. Suppose we have a particle that starts at position 0 and performs a random walk. At each time step, the particle moves either one step to the right or one step to the left with equal probability, and the move at each time step is independent of all other moves. We think of these random moves as taking place according to whether a fair coin comes up heads or tails. The expected position of the particle after n moves is back at 0, but how far from 0 should we typically expect the particle to end up at?

Denoting a right-move by $+1$ and a left-move by -1, we can describe the probability space here as the set of all sequences of length n over the alphabet $\{\pm 1\}$, each having equal probability $\frac{1}{2^n}$. Let the r.v. X denote the position of the particle (relative to our starting point 0) after n moves. Thus, we can write

$$X = X_1 + X_2 + \cdots + X_n, \tag{1}$$

where $X_i = +1$ if the i-th move is to the right and $X_i = -1$ otherwise.

Now obviously we have $\mathrm{E}(X) = 0$. The easiest way to see this is to note that $\mathrm{E}(X_i) = (\frac{1}{2} \times 1) + (\frac{1}{2} \times (-1)) = 0$, so by linearity of expectation $\mathrm{E}(X) = \sum_{i=1}^{n} \mathrm{E}(X_i) = 0$. But of course this is not very informative, and is due to the fact that positive and negative deviations from 0 cancel out.

What we are really asking is: What is the expected value of $|X|$, the *distance* of the particle from 0? Rather than consider the r.v. $|X|$, which is a little difficult to work with due to the absolute value operator, we will instead look at the r.v. X^2. Notice that this also has the effect of making all deviations from 0 positive, so it should also give a good measure of the distance from 0. However, because it is the *squared* distance, we will need to take a square root at the end.

We will now show that the expected square distance after n steps is equal to n:

Claim 17.1. *For the random variable X defined in* (1)*, we have* $\mathrm{E}(X^2) = n$.

Proof. We use the expression (1) and expand the square:

$$\begin{aligned} \mathrm{E}(X^2) &= \mathrm{E}((X_1 + X_2 + \cdots + X_n)^2) \\ &= \mathrm{E}(\textstyle\sum_{i=1}^{n} X_i^2 + \sum_{i \neq j} X_i X_j) \\ &= \textstyle\sum_{i=1}^{n} \mathrm{E}(X_i^2) + \sum_{i \neq j} \mathrm{E}(X_i X_j) \end{aligned}$$

In the last line we have used linearity of expectation. To proceed, we need to compute $\mathrm{E}(X_i^2)$ and $\mathrm{E}(X_iX_j)$ (for $i \neq j$). Let's consider first X_i^2. Since X_i can take on only values ± 1, clearly $X_i^2 = 1$ always, so $\mathrm{E}(X_i^2) = 1$. What about $\mathrm{E}(X_iX_j)$? Well, $X_iX_j = +1$ when $X_i = X_j = +1$ or $X_i = X_j = -1$, and otherwise $X_iX_j = -1$. Therefore,

$$\begin{aligned}\Pr[X_iX_j = 1] &= \Pr[(X_i = X_j = +1) \vee (X_i = X_j = -1)] \\ &= \Pr[X_i = X_j = +1] + \Pr[X_i = X_j = -1] \\ &= \Pr[X_i = +1] \times \Pr[X_j = +1] + \Pr[X_i = -1] \times \Pr[X_j = -1] \\ &= \tfrac{1}{4} + \tfrac{1}{4} = \tfrac{1}{2},\end{aligned}$$

where in the above calculation we used the fact that the events $X_i = +1$ and $X_j = +1$ are independent, and similarly the events $X_i = -1$ and $X_j = -1$ are independent. Thus $\Pr[X_iX_j = -1] = \frac{1}{2}$ as well, and hence $\mathrm{E}(X_iX_j) = 0$.

Plugging these values into the above equation gives

$$\mathrm{E}(X^2) = \sum_{i=1}^{n} 1 + \sum_{i \neq j} 0 = n,$$

as claimed. □

So we see that our expected squared distance from 0 is n. One interpretation of this is that we might expect to be a distance of about $\sqrt{n}$ away from 0 after n steps. However, we have to be careful here: we **cannot** simply argue that $\mathrm{E}(|X|) = \sqrt{\mathrm{E}(X^2)} = \sqrt{n}$. (Why not?) We will see later in the lecture how to make precise deductions about $|X|$ from knowledge of $\mathrm{E}(X^2)$.

For the moment, however, let's agree to view $\mathrm{E}(X^2)$ as an intuitive measure of "spread" of the r.v. X. In fact, for a more general r.v. with expectation $\mathrm{E}(X) = \mu$, what we are really interested in is $\mathrm{E}((X-\mu)^2)$, the expected squared distance *from the mean*. In our random walk example, we had $\mu = 0$, so $\mathrm{E}((X-\mu)^2)$ just reduces to $\mathrm{E}(X^2)$.

Definition 17.1 (Variance)**.** *For a r.v. X with expectation $\mathrm{E}(X) = \mu$, the variance of X is defined to be*

$$\mathrm{Var}(X) = \mathrm{E}((X-\mu)^2).$$

The square root $\sigma(X) := \sqrt{\mathrm{Var}(X)}$ is called the standard deviation of X.

The point of the standard deviation is merely to "undo" the squaring in the variance. Thus the standard deviation is "on the same scale as" the r.v. itself. Since the variance and standard deviation differ just by a square, it really doesn't matter which one we choose to work with as we can always compute one from the other immediately. We shall usually use the variance. For the random walk example above, we have that $\mathrm{Var}(X) = n$, and the standard deviation of X, $\sigma(X)$, is $\sqrt{n}$.

The following easy observation gives us a slightly different way to compute the variance that is simpler in many cases.

Theorem 17.1. *For a r.v. X with expectation $\mathrm{E}(X) = \mu$, we have $\mathrm{Var}(X) = \mathrm{E}(X^2) - \mu^2$.*

Proof. From the definition of variance, we have

$$\mathrm{Var}(X) = \mathrm{E}((X-\mu)^2) = \mathrm{E}(X^2 - 2\mu X + \mu^2) = \mathrm{E}(X^2) - 2\mu\mathrm{E}(X) + \mu^2 = \mathrm{E}(X^2) - \mu^2.$$

In the third step above, we used linearity of expectation. Moreover, note that $\mu = \mathrm{E}(X)$ is a constant, so $\mathrm{E}(\mu X) = \mu\mathrm{E}(X) = \mu^2$ and $\mathrm{E}(\mu^2) = \mu^2$. □

Another important property that will come in handy is the following: For any random variable X and constant c, we have

$$\mathrm{Var}(cX) = c^2\mathrm{Var}(X).$$

The proof is simple and left as an exercise.

Examples

Let's see some examples of variance calculations.

1. **Fair die.** Let X be the score on the roll of a single fair die. Recall from the previous note that $\mathrm{E}(X) = \frac{7}{2}$. So we just need to compute $\mathrm{E}(X^2)$, which is a routine calculation:

   $$\mathrm{E}(X^2) = \frac{1}{6}\left(1^2+2^2+3^2+4^2+5^2+6^2\right) = \frac{91}{6}.$$

   Thus from Theorem 17.1,

   $$\mathrm{Var}(X) = \mathrm{E}(X^2) - (\mathrm{E}(X))^2 = \frac{91}{6} - \frac{49}{4} = \frac{35}{12}.$$

2. **Uniform distribution.** More generally, if X is a uniform random variable on the set $\{1,\ldots,n\}$, so X takes on values $1,\ldots,n$ with equal probability $\frac{1}{n}$, then the mean, variance and standard deviation of X are given by:

   $$\mathrm{E}(X) = \frac{n+1}{2}, \qquad \mathrm{Var}(X) = \frac{n^2-1}{12}, \qquad \sigma(X) = \sqrt{\frac{n^2-1}{12}}. \tag{2}$$

   You should verify these as an exercise.

3. **Number of fixed points.** Let X be the number of fixed points in a random permutation of n items (i.e., the number of students in a class of size n who receive their own homework after shuffling). We saw in the previous note that $\mathrm{E}(X) = 1$, regardless of n. To compute $\mathrm{E}(X^2)$, write $X = X_1 + X_2 + \cdots + X_n$, where $X_i = 1$ if i is a fixed point, and $X_i = 0$ otherwise. Then as usual we have

   $$\mathrm{E}(X^2) = \sum_{i=1}^{n} \mathrm{E}(X_i^2) + \sum_{i\neq j} \mathrm{E}(X_iX_j). \tag{3}$$

   Since X_i is an indicator r.v., we have that $\mathrm{E}(X_i^2) = \Pr[X_i = 1] = \frac{1}{n}$. Since both X_i and X_j are indicators, we can compute $\mathrm{E}(X_iX_j)$ as follows:

   $$\mathrm{E}(X_iX_j) = \Pr[X_iX_j = 1] = \Pr[X_i = 1 \wedge X_j = 1] = \Pr[\text{both } i \text{ and } j \text{ are fixed points}] = \frac{1}{n(n-1)}.$$

   Make sure that you understand the last step here. Plugging this into equation (3) we get

   $$\mathrm{E}(X^2) = \sum_{i=1}^{n} \frac{1}{n} + \sum_{i\neq j} \frac{1}{n(n-1)} = (n \times \tfrac{1}{n}) + (n(n-1) \times \tfrac{1}{n(n-1)}) = 1 + 1 = 2.$$

   Thus $\mathrm{Var}(X) = \mathrm{E}(X^2) - (\mathrm{E}(X))^2 = 2 - 1 = 1$. That is, the variance and the mean are both equal to 1. Like the mean, the variance is also independent of n. Intuitively at least, this means that it is unlikely that there will be more than a small number of fixed points even when the number of items, n, is very large.

Independent Random Variables

Independence for random variables is defined in analogous fashion to independence for events:

Definition 17.2 (Independent r.v.'s)**.** *Random variables X and Y on the same probability space are said to be independent if the events $X = a$ and $Y = b$ are independent for all values a, b. Equivalently, the joint distribution of independent r.v.'s decomposes as*

$$\Pr[X = a, Y = b] = \Pr[X = a] \Pr[Y = b] \quad \forall a, b.$$

Mutual independence of more than two r.v.'s is defined similarly. A very important example of independent r.v.'s is indicator r.v.'s for independent events. Thus, for example, if $\{X_i\}$ are indicator r.v.'s for the i-th toss of a coin being Heads, then the X_i are mutually independent r.v.'s.

One of the most important and useful facts about variance is if a random variable X is the sum of *independent* random variables $X = X_1 + \cdots + X_n$, then its variance is the sum of the variances of the individual r.v.'s. In particular, if the individual r.v.'s X_i are identically distributed (i.e., they have the same distribution), then $\mathrm{Var}(X) = \sum_i \mathrm{Var}(X_i) = n \cdot \mathrm{Var}(X_1)$. This means that the standard deviation is $\sigma(X) = \sqrt{n} \cdot \sigma(X_1)$. Note that by contrast, the expected value is $E[X] = n \cdot E[X_1]$. Intuitively this means that whereas the average value of X grows proportionally to n, the spread of the distribution grows proportionally to $\sqrt{n}$, which is much smaller than n. In other words the distribution of X tends to concentrate around its mean.

Let us now formalize these ideas. First, we have the following result which states that the expected value of the product of two independent random variables is equal to the product of their expected values.

Theorem 17.2. *For* independent *random variables X, Y, we have* $\mathrm{E}(XY) = \mathrm{E}(X)\mathrm{E}(Y)$.

Proof. We have

$$\begin{aligned}
\mathrm{E}(XY) &= \sum_a \sum_b ab \times \Pr[X = a, Y = b] \\
&= \sum_a \sum_b ab \times \Pr[X = a] \times \Pr[Y = b] \\
&= \left(\sum_a a \times \Pr[X = a] \right) \times \left(\sum_b b \times \Pr[Y = b] \right) \\
&= \mathrm{E}(X) \times \mathrm{E}(Y),
\end{aligned}$$

as required. In the second line here we made crucial use of independence. □

We now use the above theorem to conclude the nice property that the variance of the sum of independent random variables is equal to the sum of their variances.

Theorem 17.3. *For* independent *random variables X, Y, we have*

$$\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y).$$

Proof. From the alternative formula for variance in Theorem 17.1, we have, using linearity of expectation extensively,

$$\begin{aligned}
\mathrm{Var}(X + Y) &= \mathrm{E}((X + Y)^2) - \mathrm{E}(X + Y)^2 \\
&= \mathrm{E}(X^2) + \mathrm{E}(Y^2) + 2\mathrm{E}(XY) - (\mathrm{E}(X) + \mathrm{E}(Y))^2 \\
&= (\mathrm{E}(X^2) - \mathrm{E}(X)^2) + (\mathrm{E}(Y^2) - \mathrm{E}(Y)^2) + 2(\mathrm{E}(XY) - \mathrm{E}(X)\mathrm{E}(Y)) \\
&= \mathrm{Var}(X) + \mathrm{Var}(Y) + 2(\mathrm{E}(XY) - \mathrm{E}(X)\mathrm{E}(Y)).
\end{aligned}$$

Now *because X, Y are independent*, by Theorem 17.2 the final term in this expression is zero. Hence we get our result. □

Note: The expression $\mathrm{E}(XY) - \mathrm{E}(X)\mathrm{E}(Y)$ appearing in the above proof is called the *covariance* of X and Y, and is a measure of the dependence between X, Y. It is zero when X, Y are independent.

It is very important to remember that **neither** of these two results is true in general, without the assumption that X, Y are independent. As a simple example, note that even for a 0-1 r.v. X with $\Pr[X = 1] = p$, $\mathrm{E}(X^2) = p$ is not equal to $\mathrm{E}(X)^2 = p^2$ (because of course X and X are not independent!). This is in contrast to the case of the expectation, where we saw that the expectation of a sum of r.v.'s is the sum of the expectations of the individual r.v.'s *always*.

Example

Let's return to our motivating example of a sequence of n coin tosses. Let X the the number of Heads in n tosses of a biased coin with Heads probability p (i.e., X has the binomial distribution with parameters n, p). As usual, we write $X = X_1 + X_2 + \cdots + X_n$, where $X_i = 1$ if the i-th toss is Head, and $X_i = 0$ otherwise.

We already know $\mathrm{E}(X) = \sum_{i=1}^{n} \mathrm{E}(X_i) = np$. We can compute $\mathrm{Var}(X_i) = E(X_i^2) - E(X_i)^2 = p - p^2 = p(1-p)$. Since the X_i's are independent, by Theorem 17.3 we get $\mathrm{Var}(X) = \sum_{i=1}^{n} \mathrm{Var}(X_i) = np(1-p)$.

As an example, for a fair coin ($p = \frac{1}{2}$) the expected number of Heads in n tosses is $\frac{n}{2}$, and the standard deviation is $\sqrt{\frac{n}{4}} = \frac{\sqrt{n}}{2}$. Note that since the maximum number of Heads is n, the standard deviation is much less than this maximum number for large n. This is in contrast to the previous example of the uniformly distributed random variable (2), where the standard deviation $\sigma(X) = \sqrt{\frac{n^2-1}{12}} \approx \frac{n}{\sqrt{12}}$ is of the same order as the largest value n. In this sense, the spread of a binomially distributed r.v. is much smaller than that of a uniformly distributed r.v.

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Lecture 18

Chebyshev's Inequality

Problem: Estimating the Bias of a Coin

Suppose we have a biased coin, but we don't know what the bias is. To estimate the bias, we toss the coin n times and count how many Heads we observe. Then our estimate of the bias is given by $\hat{p} = \frac{1}{n}S_n$, where S_n is the number of Heads in the n tosses. Is this a good estimate? Let p denote the true bias of the coin, which is unknown to us. Since $\mathrm{E}(S_n) = np$, we see that the estimator $\hat{p}$ has the correct expected value: $\mathrm{E}(\hat{p}) = \frac{1}{n}\mathrm{E}(S_n) = p$. This means when n is sufficiently large, we can expect $\hat{p}$ to be very close to p; this is a manifestation of the *Law of Large Numbers*, which we shall see at the end of this note.

How large should n be to guarantee that our estimate $\hat{p}$ is within an error ε of the true bias p, i.e., $|\hat{p} - p| \leq \varepsilon$? The answer is that we can never guarantee with absolute certainty that $|\hat{p} - p| \leq \varepsilon$. This is because S_n is a random variable that can take any integer values between 0 and n, and thus $\hat{p} = \frac{1}{n}S_n$ is also a random variable that can take any values between 0 and 1. So regardless of the value of the true bias $p \in (0,1)$, it is possible that in our experiment of n coin tosses we observe n Heads (this happens with probability p^n), in which case $S_n = n$ and $\hat{p} = 1$. Similarly, it is also possible that all n coin tosses come up Tails (this happens with probability $(1-p)^n$), in which case $S_n = 0$ and $\hat{p} = 0$.

So instead of requiring that $|\hat{p} - p| \leq \varepsilon$ with absolute certainty, we relax our requirement and only demand that $|\hat{p} - p| \leq \varepsilon$ with *confidence* δ, namely, $\Pr[|\hat{p} - p| \leq \varepsilon] \geq 1 - \delta$. This means there is a small probability δ that we make an error of more than ε, but with high probability (at least $1 - \delta$) our estimate is very close to p. Now we can state our result: to guarantee that $\Pr[|\hat{p} - p| \leq \varepsilon] \geq 1 - \delta$, it suffices to toss the coin n times where

$$n \geq \frac{1}{4\varepsilon^2\delta}.$$

To prove such a result we use the tool called *Chebyshev's Inequality*, which provides a quantitative bound on how far away a random variable is from its expected value.

Markov's Inequality

Before going to Chebyshev's inequality, we first state the following simpler bound, which applies only to *non-negative* random variables (i.e., r.v.'s which take only values ≥ 0).

Theorem 18.1 (Markov's Inequality). *For a* non-negative *random variable X with expectation $\mathrm{E}(X) = \mu$, and any $\alpha > 0$,*

$$\Pr[X \geq \alpha] \leq \frac{\mathrm{E}(X)}{\alpha}.$$

Proof. From the definition of expectation, we have

$$\begin{aligned} \mathrm{E}(X) &= \textstyle\sum_a a \times \Pr[X = a] \\ &\geq \textstyle\sum_{a \geq \alpha} a \times \Pr[X = a] \\ &\geq \alpha \textstyle\sum_{a \geq \alpha} \Pr[X = a] \\ &= \alpha \Pr[X \geq \alpha]. \end{aligned}$$

Rearranging the inequality gives us the desired result. The crucial step here is the second line, where we have used the fact that X takes on only non-negative values. (Why is this step not valid otherwise?) □

There is an intuitive way of understanding Markov's inequality through an analogy of a seesaw. Imagine that the distribution of a non-negative random variable X is resting on a fulcrum, $\mu = \mathrm{E}(X)$. We are trying to find an upper bound on the percentage of the distribution which lies beyond $k\mu$, i.e., $\Pr[X \geq k\mu]$. In other words, we seek to add as much weight m_2 as possible on the seesaw at $k\mu$ while minimizing the effect it has on the seesaw's balance. This weight will represent the upper bound we are searching for. To minimize the weight's effect, we must imagine that the weight of the distribution which lies beyond $k\mu$ is concentrated at exactly $k\mu$. However, to keep things stable and maximize the weight at $k\mu$, we must add another weight m_1 as far left to the fulcrum as we can so that m_2 is as large as it can be. The farthest we can go to the left is 0, since X is assumed to be non-negative. Moreover, the two weights m_1 and m_2 must add up to 1, since they represent the area under the distribution curve:

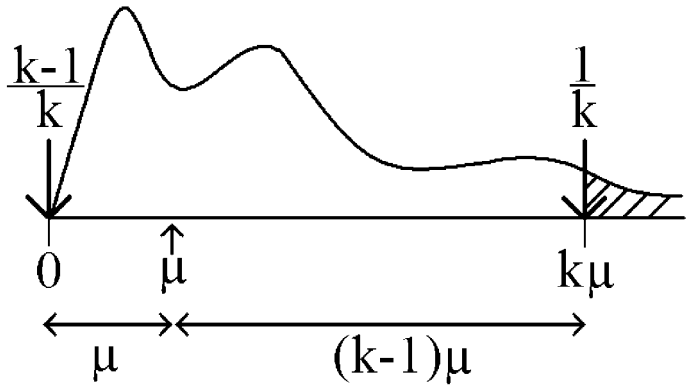


Figure 1: Markov's inequality interpreted as balancing a seesaw.

Since the lever arms are in the ratio $k-1$ to 1, a unit weight at $k\mu$ balances $k-1$ units of weight at 0. So the weights should be $\frac{k-1}{k}$ at 0 and $\frac{1}{k}$ at $k\mu$, which is exactly Markov's bound with $\alpha = k\mu$.

Chebyshev's Inequality

We have seen that, intuitively, the variance (or, more correctly the standard deviation) is a measure of "spread," or deviation from the mean. We can now make this intuition quantitatively precise:

Theorem 18.2 (Chebyshev's Inequality). *For a random variable X with expectation $\mathrm{E}(X) = \mu$, and for any $\alpha > 0$,*

$$\Pr[|X - \mu| \geq \alpha] \leq \frac{\mathrm{Var}(X)}{\alpha^2}.$$

Proof. Define the random variable $Y = (X - \mu)^2$. Note that $\mathrm{E}(Y) = \mathrm{E}((X - \mu)^2) = \mathrm{Var}(X)$. Also, notice that the event that we are interested in, $|X - \mu| \geq \alpha$, is exactly the same as the event $Y = (X - \mu)^2 \geq \alpha^2$.

Therefore, $\Pr[|X-\mu| \geq \alpha] = \Pr[Y \geq \alpha^2]$. Moreover, Y is obviously non-negative, so we can apply Markov's inequality to it to get

$$\Pr[Y \geq \alpha^2] \leq \frac{\mathrm{E}(Y)}{\alpha^2} = \frac{\mathrm{Var}(X)}{\alpha^2}.$$

This completes the proof. □

Let's pause to consider what Chebyshev's inequality says. It tells us that the probability of any given deviation, α, from the mean, either above it or below it (note the absolute value sign), is at most $\frac{\mathrm{Var}(X)}{\alpha^2}$. As expected, this deviation probability will be small if the variance is small. An immediate corollary of Chebyshev's inequality is the following:

Corollary 18.1. *For a random variable X with expectation $\mathrm{E}(X) = \mu$, and standard deviation $\sigma = \sqrt{\mathrm{Var}(X)}$,*

$$\Pr[|X-\mu| \geq \beta\sigma] \leq \frac{1}{\beta^2}.$$

Proof. Plug $\alpha = \beta\sigma$ into Chebyshev's inequality. □

So, for example, we see that the probability of deviating from the mean by more than (say) two standard deviations on either side is at most $\frac{1}{4}$. In this sense, the standard deviation is a good working definition of the "width" or "spread" of a distribution.

In some special cases it is possible to get tighter bounds on the probability of deviations from the mean. However, for general random variables Chebyshev's inequality is essentially the only tool. Its power derives from the fact that it can be applied to *any* random variable.

Example: Estimating the Bias of a Coin

Let us go back to our motivating example of estimating the bias of a coin. Recall that we have a coin of unknown bias p, and our estimate of p is $\hat{p} = \frac{1}{n}S_n$ where S_n is the number of Heads in n coin tosses.

As usual, we will find it helpful to write $S_n = X_1 + \cdots + X_n$, where $X_i = 1$ is the i-th coin toss comes up Heads and $X_i = 0$ otherwise, and the random variables $X_1, \ldots, X_n$ are mutually independent. Then $\mathrm{E}(X_i) = \Pr[X_i = 1] = p$, so by linearity of expectation,

$$\mathrm{E}(\hat{p}) = \mathrm{E}(\tfrac{1}{n}S_n) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}\mathrm{E}(X_i) = p.$$

What about the variance of $\hat{p}$? Note that since the X_i's are independent, the variance of $S_n = \sum_{i=1}^{n} X_i$ is equal to the sum of the variances:

$$\mathrm{Var}(\hat{p}) = \mathrm{Var}(\tfrac{1}{n}S_n) = \tfrac{1}{n^2}\mathrm{Var}(S_n) = \tfrac{1}{n^2}\textstyle\sum_{i=1}^{n}\mathrm{Var}(X_i) = \frac{\sigma^2}{n},$$

where we have written σ^2 for the variance of each of the X_i.

So we see that the variance of $\hat{p}$ decreases linearly with n. This fact ensures that, as we take larger and larger sample sizes n, the probability that we deviate much from the expectation p gets smaller and smaller.

Let's now use Chebyshev's inequality to figure out how large n has to be to ensure a specified accuracy in our estimate of the true bias p. As we discussed in the beginning of this note, a natural way to measure this is for us to specify two parameters, ε and δ, both in the range $(0,1)$. The parameter ε controls the *error* we are prepared to tolerate in our estimate, and δ controls the *confidence* we want to have in our estimate.

Applying Chebyshev's inequality, we have

$$\Pr[|\hat{p} - p| \geq \varepsilon] \leq \frac{\mathrm{Var}(\hat{p})}{\varepsilon^2} = \frac{\sigma^2}{n\varepsilon^2}.$$

To make this less than the desired value δ, we need to set

$$n \geq \frac{\sigma^2}{\varepsilon^2 \delta}. \tag{1}$$

Now recall that $\sigma^2 = \mathrm{Var}(X_i)$ is the variance of a single sample X_i. So, since X_i is an indicator random variable with $\Pr[X_i = 1] = p$, we have $\sigma^2 = p(1-p)$, and inequality (1) becomes

$$n \geq \frac{p(1-p)}{\varepsilon^2 \delta}. \tag{2}$$

Since $p(1-p)$ is takes on its maximum value for $p = 1/2$, we can conclude that it is sufficient to choose n such that:

$$n \geq \frac{1}{4\varepsilon^2 \delta}, \tag{3}$$

as we claimed earlier.

For example, plugging in $\varepsilon = 0.1$ and $\delta = 0.05$, we see that a sample size of $n = 500$ is sufficient to get an estimate $\hat{p}$ that is accurate to within an error of 0.1 with probability at least 95%.

As a concrete example, consider the problem of estimating the proportion p of Democrats in the US population, by taking a small random sample. We can model this as the problem of estimating the bias of a coin above, where each coin toss corresponds to a person that we select randomly from the entire population. Our calculation above shows that to get an estimate $\hat{p}$ that is accurate to within an error of 0.1 with probability at least 95%, it suffices to sample $n = 500$ people. In particular, notice that the size of the sample is independent of the total size of the population! This is how polls can accurately estimate quantities of interest for a population of several hundred million while sampling only a very small number of people.

Estimating a general expectation

What if we wanted to estimate something a little more complex than the bias of a coin? For example, suppose we want to estimate the average wealth of people in the US. We can model this as the problem of estimating the expected value of an unknown probability distribution. Then we can use exactly the same scheme as before, except that now we sample the random variables $X_1, X_2, \ldots, X_n$ independently from our unknown distribution. Clearly $\mathrm{E}(X_i) = \mu$, the expected value that we are trying to estimate. Our estimate of μ will be $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} X_i$, for a suitably chosen sample size n.

Following the same calculation as before, we have $\mathrm{E}(\hat{\mu}) = \mu$ and $\mathrm{Var}(\hat{\mu}) = \frac{\sigma^2}{n}$, where $\sigma^2 = \mathrm{Var}(X_i)$ is the variance of the X_i. (Recall that the only facts we used about the X_i was that they were independent and had the same distribution — actually the same expectation and variance would be enough: why?) This time, however, since we do not have any a priori bound on the mean μ, it makes more sense to let ε be the relative error, i.e., we wish to find an estimate $\hat{\mu}$ that is within an additive error of $\varepsilon\mu$:

$$\Pr[|\hat{\mu} - \mu| \geq \varepsilon\mu] \leq \delta.$$

Using equation (1), but substituting $\varepsilon\mu$ in place of ε, it is enough for the sample size n to satisfy

$$n \geq \frac{\sigma^2}{\mu^2} \times \frac{1}{\varepsilon^2 \delta}. \tag{4}$$

Here ε and δ are the desired relative error and confidence respectively. Now of course we don't know the other two quantities, μ and σ^2, appearing in equation (4). In practice, we would use a lower bound on μ and an upper bound on σ^2 (just as we used a lower bound on p in the coin tossing problem). Plugging these bounds into equation (4) will ensure that our sample size is large enough.

For example, in the average wealth problem we could probably safely take μ to be at least (say) \$20k (probably more). However, the existence of very wealthy people such as Bill Gates means that we would need to take a very high value for the variance σ^2. Indeed, if there is at least one individual with wealth \$50 billion, then assuming a relatively small value of μ means that the variance must be at least about $\frac{(50\times 10^9)^2}{250\times 10^6} = 10^{13}$. There is really no way around this problem with simple uniform sampling: the uneven distribution of wealth means that the variance is inherently very large, and we will need a huge number of samples before we are likely to find anybody who is immensely wealthy. But if we don't include such people in our sample, then our estimate will be way too low.

The Law of Large Numbers

The estimation method we used in the previous sections is based on a principle that we accept as part of everyday life: namely, the Law of Large Numbers (LLN). This asserts that, if we observe some random variable many times, and take the average of the observations, then this average will converge to a *single value*, which is of course the expectation of the random variable. In other words, averaging tends to smooth out any large fluctuations, and the more averaging we do the better the smoothing.

Theorem 18.3 (Law of Large Numbers). *Let $X_1, X_2, \ldots, X_n$ be i.i.d. random variables with common expectation $\mu = \mathrm{E}(X_i)$. Define $A_n = \frac{1}{n}\sum_{i=1}^n X_i$. Then for any $\alpha > 0$, we have*

$$\Pr[|A_n - \mu| \geq \alpha] \to 0 \qquad \text{as } n \to \infty.$$

Proof. Let $\mathrm{Var}(X_i) = \sigma^2$ be the common variance of the r.v.'s; we assume that σ^2 is finite.[1] With this (relatively mild) assumption, the LLN is an immediate consequence of Chebyshev's Inequality. For, as we have seen above, $\mathrm{E}(A_n) = \mu$ and $\mathrm{Var}(A_n) = \frac{\sigma^2}{n}$, so by Chebyshev we have

$$\Pr[|A_n - \mu| \geq \alpha] \leq \frac{\mathrm{Var}(A_n)}{\alpha^2} = \frac{\sigma^2}{n\alpha^2} \to 0 \qquad \text{as } n \to \infty.$$

This completes the proof. □

Notice that the LLN says that the probability of *any* deviation α from the mean, however small, tends to zero as the number of observations n in our average tends to infinity. Thus by taking n large enough, we can make the probability of any given deviation as small as we like.

[1] If σ^2 is not finite, the LLN still holds but the proof is much trickier.

Some Important Distributions

Recall our basic probabilistic experiment of tossing a biased coin n times. This is a very simple model, yet surprisingly powerful. Many important probability distributions that are widely used to model real-world phenomena can be derived from looking at this basic coin tossing model.

The first example, which we have seen in Note 16, is the *binomial distribution* $Bin(n,p)$. This is the distribution of the number of Heads, S_n, in n tosses of a biased coin with probability p to be Heads. Recall that the distribution of S_n is $\Pr[S_n = k] = \binom{n}{k} p^k (1-p)^{n-k}$ for $k \in \{0,1,\ldots,n\}$. The expected value is $\mathrm{E}(S_n) = np$ and the variance is $\mathrm{Var}(S_n) = np(1-p)$. The binomial distribution frequently appears to model the number of successes in a repeated experiment.

Geometric Distribution

Consider tossing a biased coin with Heads probability p repeatedly. Let X denote the number of tosses until the first Head appears. Then X is a random variable that takes values in the set of positive integers $\{1,2,3,\ldots\}$. The event that $X = i$ is equal to the event of observing Tails for the first $i-1$ tosses and getting Heads in the i-th toss, which occurs with probability $(1-p)^{i-1}p$. Such a random variable is called a geometric random variable.

The geometric distribution frequently occurs in applications because we are often interested in how long we have to wait before a certain event happens: how many runs before the system fails, how many shots before one is on target, how many poll samples before we find a Democrat, how many retransmissions of a packet before successfully reaching the destination, etc.

Definition 19.1 (Geometric distribution)**.** *A random variable X for which*

$$\Pr[X = i] = (1-p)^{i-1}p \qquad \text{for } i = 1,2,3,\ldots$$

is said to have the geometric distribution with parameter p. This is abbreviated as $X \sim Geom(p)$.

As a sanity check, we can verify that the total probability of X is equal to 1:

$$\sum_{i=1}^{\infty} \Pr[X = i] = \sum_{i=1}^{\infty} (1-p)^{i-1}p = p\sum_{i=1}^{\infty}(1-p)^{i-1} = p \times \frac{1}{1-(1-p)} = 1,$$

where in the second-to-last step we have used the formula for geometric series.

If we plot the distribution of X (i.e., the values $\Pr[X = i]$ against i) we get a curve that decreases monotonically by a factor of $1-p$ at each step, as shown in Figure 1.

Let us now compute the expectation $\mathrm{E}(X)$. Applying the definition of expected value directly gives us:

$$\mathrm{E}(X) = \sum_{i=1}^{\infty} i \times \Pr[X = i] = p\sum_{i=1}^{\infty} i(1-p)^{i-1}.$$

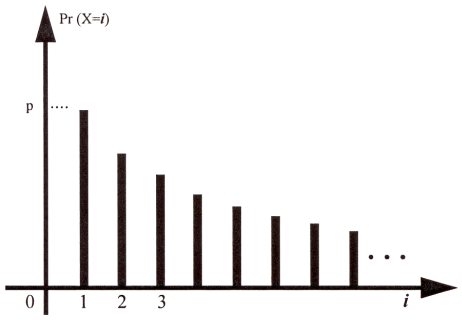


Figure 1: The Geometric distribution.

However, the final summation is difficult to evaluate and requires some calculus trick. Instead, we will use the following alternative formula for expectation.

Theorem 19.1. *Let X be a random variable that takes values in $\{0,1,2,\dots\}$. Then*

$$\mathrm{E}(X) = \sum_{i=1}^{\infty} \Pr[X \geq i].$$

Proof. For notational convenience, let's write $p_i = \Pr[X = i]$, for $i = 0,1,2,\dots$. From the definition of expectation, we have

$$\begin{aligned}
\mathrm{E}(X) &= (0 \times p_0) + (1 \times p_1) + (2 \times p_2) + (3 \times p_3) + (4 \times p_4) + \cdots \\
&= p_1 + (p_2 + p_2) + (p_3 + p_3 + p_3) + (p_4 + p_4 + p_4 + p_4) + \cdots \\
&= (p_1 + p_2 + p_3 + p_4 + \cdots) + (p_2 + p_3 + p_4 + \cdots) + (p_3 + p_4 + \cdots) + (p_4 + \cdots) + \cdots \\
&= \Pr[X \geq 1] + \Pr[X \geq 2] + \Pr[X \geq 3] + \Pr[X \geq 4] + \cdots .
\end{aligned}$$

In the third line, we have regrouped the terms into convenient infinite sums, and each infinite sum is exactly the probability that $X \geq i$ for each i. You should check that you understand how the fourth line follows from the third.

Let us repeat the proof more formally, this time using more compact mathematical notation:

$$\mathrm{E}(X) = \sum_{j=1}^{\infty} j \times \Pr[X = j] = \sum_{j=1}^{\infty} \sum_{i=1}^{j} \Pr[X = j] = \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \Pr[X = j] = \sum_{i=1}^{\infty} \Pr[X \geq i].$$

□

We can now use Theorem 19.1 to compute $\mathrm{E}(X)$ more easily.

Theorem 19.2. *For a geometric random variable $X \sim Geom(p)$, we have $\mathrm{E}(X) = \frac{1}{p}$.*

Proof. The key observation is that for a geometric random variable X,

$$\Pr[X \geq i] = (1-p)^{i-1} \text{ for } i = 1,2,\dots. \tag{1}$$

We can obtain this simply by summing $\Pr[X = j]$ for $j \geq i$. Another way of seeing this is to note that the event "$X \geq i$" means at least i tosses are required. This is equivalent to saying that the first $i-1$ tosses are all

Tails, and the probability of this event is precisely $(1-p)^{i-1}$. Now, plugging equation (1) into Theorem 19.1, we get

$$\mathrm{E}(X) = \sum_{i=1}^{\infty} \Pr[X \geq i] = \sum_{i=1}^{\infty} (1-p)^{i-1} = \frac{1}{1-(1-p)} = \frac{1}{p},$$

where we have used the formula for geometric series. □

So, the expected number of tosses of a biased coin until the first Head appears is $\frac{1}{p}$. Intuitively, if in each coin toss we expect to get p Heads, then we need to toss the coin $\frac{1}{p}$ times to get 1 Head. So for a fair coin, the expected number of tosses is 2, but remember that the actual number of coin tosses that we need can be any positive integers.

Remark: Another way of deriving $\mathrm{E}(X) = \frac{1}{p}$ is to use the interpretation of a geometric random variable X as the number of coin tosses until we get a Head. Consider what happens in the first coin toss: If the first toss comes up Heads, then $X = 1$. Otherwise, we have used one toss, and we repeat the coin tossing process again; the number of coin tosses after the first toss is again a geometric random variable with parameter p. Therefore, we can calculate:

$$\mathrm{E}(X) = \underbrace{p \cdot 1}_{\text{first toss is H}} + \underbrace{(1-p) \cdot (1+\mathrm{E}(X))}_{\text{first toss is T, then toss again}}.$$

Solving for $\mathrm{E}(X)$ yields $\mathrm{E}(X) = \frac{1}{p}$, as claimed.

Let us now compute the variance of X.

Theorem 19.3. *For a geometric random variable $X \sim Geom(p)$, we have* $\mathrm{Var}(X) = \frac{1-p}{p^2}$.

Proof. We will show that $\mathrm{E}(X(X-1)) = \frac{2(1-p)}{p^2}$. Since we already know $\mathrm{E}(X) = \frac{1}{p}$, this will imply the desired result:

$$\begin{aligned} \mathrm{Var}(X) &= \mathrm{E}(X^2) - \mathrm{E}(X)^2 = \mathrm{E}(X(X-1)) + \mathrm{E}(X) - \mathrm{E}(X)^2 \\ &= \frac{2(1-p)}{p^2} + \frac{1}{p} - \frac{1}{p^2} = \frac{2(1-p)+p-1}{p^2} = \frac{1-p}{p^2}. \end{aligned}$$

Now to show $\mathrm{E}(X(X-1)) = \frac{2(1-p)}{p^2}$, we begin with the following identity of geometric series:

$$\sum_{i=0}^{\infty} (1-p)^i = \frac{1}{p}.$$

Differentiating the identity above with respect to p yields (the $i = 0$ term is equal to 0 so we omit it):

$$-\sum_{i=1}^{\infty} i(1-p)^{i-1} = -\frac{1}{p^2}.$$

Differentiating both sides with respect to p again gives us (the $i = 1$ term is equal to 0 so we omit it):

$$\sum_{i=2}^{\infty} i(i-1)(1-p)^{i-2} = \frac{2}{p^3}. \tag{2}$$

Now using the geometric distribution of X and identity (2), we can calculate:

$$\begin{aligned}
\mathrm{E}(X(X-1)) &= \sum_{i=1}^{\infty} i(i-1) \times \Pr[X=i] \\
&= \sum_{i=2}^{\infty} i(i-1)(1-p)^{i-1}p \qquad \text{(the } i=1 \text{ term is equal to 0 so we omit it)} \\
&= p(1-p)\sum_{i=2}^{\infty} i(i-1)(1-p)^{i-2} \\
&= p(1-p) \times \frac{2}{p^3} \qquad \text{(using identity (2))} \\
&= \frac{2(1-p)}{p^2},
\end{aligned}$$

as desired. □

Application: The Coupon Collector's Problem

Suppose we are trying to collect a set of n different baseball cards. We get the cards by buying boxes of cereal: each box contains exactly one card, and it is equally likely to be any of the n cards. How many boxes do we need to buy until we have collected at least one copy of every card?

Let X denote the number of boxes we need to buy in order to collect all n cards. The distribution of X is difficult to compute directly (try it for $n = 3$). But if we are only interested in its expected value $\mathrm{E}(X)$, then we can evaluate it easily using linearity of expectation and what we have just learned about the geometric distribution.

As usual, we start by writing

$$X = X_1 + X_2 + \ldots + X_n \tag{3}$$

for suitable simple random variables X_i. What should the X_i be? Naturally, X_i is the number of boxes we buy while trying to get the i-th new card (starting immediately after we've got the $(i-1)$-st new card). With this definition, make sure you believe equation (3) before proceeding.

What does the distribution of X_i look like? Well, X_1 is trivial: no matter what happens, we always get a new card in the first box (since we have none to start with). So $\Pr[X_1 = 1] = 1$, and thus $\mathrm{E}(X_1) = 1$.

How about X_2? Each time we buy a box, we'll get the same old card with probability $\frac{1}{n}$, and a new card with probability $\frac{n-1}{n}$. So we can think of buying boxes as flipping a biased coin with Heads probability $p = \frac{n-1}{n}$; then X_1 is just the number of tosses until the first Head appears. So X_2 has the geometric distribution with parameter $p = \frac{n-1}{n}$, and

$$\mathrm{E}(X_2) = \frac{n}{n-1}.$$

How about X_3? This is very similar to X_2 except that now we only get a new card with probability $\frac{n-2}{n}$ (since there are now two old ones). So X_3 has the geometric distribution with parameter $p = \frac{n-2}{n}$, and

$$\mathrm{E}(X_3) = \frac{n}{n-2}.$$

Arguing in the same way, we see that, for $i = 1, 2, \ldots, n$, X_i has the geometric distribution with parameter $p = \frac{n-i+1}{n}$, and hence that

$$\mathrm{E}(X_i) = \frac{n}{n-i+1}.$$

Finally, applying linearity of expectation to equation (3), we get

$$\mathrm{E}(X) = \sum_{i=1}^{n} \mathrm{E}(X_i) = \frac{n}{n} + \frac{n}{n-1} + \cdots + \frac{n}{2} + \frac{n}{1} = n \sum_{i=1}^{n} \frac{1}{i}. \tag{4}$$

This is an exact expression for $\mathrm{E}(X)$. We can obtain a tidier form by noting that the sum in it actually has a very good approximation,[1] namely:

$$\sum_{i=1}^{n} \frac{1}{i} \approx \ln n + \gamma,$$

where $\gamma = 0.5772\ldots$ is known as *Euler's constant*.

Thus, the expected number of cereal boxes needed to collect n cards is about $n(\ln n + \gamma)$. This is an excellent approximation to the exact formula (4) even for quite small values of n. So for example, for $n = 100$, we expect to buy about 518 boxes.

Poisson Distribution

Consider the number of clicks of a Geiger counter, which measures radioactive emissions. The average number of such clicks per unit time, λ, is a measure of radioactivity, but the actual number of clicks fluctuates according to a certain distribution called the Poisson distribution. What is remarkable is that the average value, λ, completely determines the probability distribution on the number of clicks X.

Definition 19.2 (Poisson distribution)**.** *A random variable X for which*

$$\Pr[X = i] = \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda} \qquad \text{for } i = 0, 1, 2, \ldots \tag{5}$$

is said to have the Poisson distribution with parameter λ. This is abbreviated as $X \sim Poiss(\lambda)$.

To make sure this is a valid definition, let us check that (5) is in fact a distribution, i.e., that the probabilities sum to 1. We have

$$\sum_{i=0}^{\infty} \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda} = \mathrm{e}^{-\lambda} \sum_{i=0}^{\infty} \frac{\lambda^i}{i!} = \mathrm{e}^{-\lambda} \times \mathrm{e}^{\lambda} = 1.$$

In the second-to-last step, we used the Taylor series expansion $\mathrm{e}^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$.

The Poisson distribution is also a very widely accepted model for so-called "rare events," such as misconnected phone calls, radioactive emissions, crossovers in chromosomes, the number of cases of disease, the number of births per hour, etc. This model is appropriate whenever the occurrences can be assumed to happen randomly with some constant density in a continuous region (of time or space), such that occurrences in disjoint subregions are independent. One can then show that the number of occurrences in a region of unit size should obey the Poisson distribution with parameter λ.

Example: Suppose when we write an article, we make an average of 1 typo per page. We can model this with a Poisson random variable X with $\lambda = 1$. So the probability that a page has 5 typos is

$$\Pr[X = 5] \;=\; \frac{1^5}{5!} \mathrm{e}^{-1} \;=\; \frac{1}{120\,\mathrm{e}} \;\approx\; \frac{1}{326}.$$

[1] This is another of the little tricks you might like to carry around in your toolbox.

Now suppose the article has 200 pages. If we assume the number of typos in each page is independent, then the probability that there is at least one page with exactly 5 typos is

$$\begin{aligned}
\Pr[\exists \text{ a page with exactly 5 typos}] &= 1 - \Pr[\text{every page has} \neq 5 \text{ typos}] \\
&= 1 - \prod_{k=1}^{200} \Pr[\text{page } k \text{ has} \neq 5 \text{ typos}] \\
&= 1 - \prod_{k=1}^{200} (1 - \Pr[\text{page } k \text{ has exactly 5 typos}]) \\
&= 1 - \left(1 - \frac{1}{120\,\mathrm{e}}\right)^{200},
\end{aligned}$$

where in the last step we have used our earlier calculation for $\Pr[X = 5]$. □

Let us now calculate the expectation and variance of a Poisson random variable. As we noted before, the expected value is simply λ. Here we see that the variance is also equal to λ.

Theorem 19.4. *For a Poisson random variable $X \sim Poiss(\lambda)$, we have $\mathrm{E}(X) = \lambda$ and $\mathrm{Var}(X) = \lambda$.*

Proof. We can calculate $\mathrm{E}(X)$ directly from the definition of expectation:

$$\begin{aligned}
\mathrm{E}(X) &= \sum_{i=0}^{\infty} i \times \Pr[X = i] \\
&= \sum_{i=1}^{\infty} i \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda} && \text{(the } i = 0 \text{ term is equal to 0 so we omit it)} \\
&= \lambda \mathrm{e}^{-\lambda} \sum_{i=1}^{\infty} \frac{\lambda^{i-1}}{(i-1)!} \\
&= \lambda \mathrm{e}^{-\lambda} \mathrm{e}^{\lambda} && \text{(since } \mathrm{e}^{\lambda} = \textstyle\sum_{j=0}^{\infty} \frac{\lambda^j}{j!} \text{ with } j = i-1) \\
&= \lambda.
\end{aligned}$$

Similarly, we can calculate $\mathrm{E}(X(X-1))$ as follows:

$$\begin{aligned}
\mathrm{E}(X(X-1)) &= \sum_{i=0}^{\infty} i(i-1) \times \Pr[X = i] \\
&= \sum_{i=2}^{\infty} i(i-1) \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda} && \text{(the } i = 0 \text{ and } i = 1 \text{ terms are equal to 0 so we omit them)} \\
&= \lambda^2 \mathrm{e}^{-\lambda} \sum_{i=2}^{\infty} \frac{\lambda^{i-2}}{(i-2)!} \\
&= \lambda^2 \mathrm{e}^{-\lambda} \mathrm{e}^{\lambda} && \text{(since } \mathrm{e}^{\lambda} = \textstyle\sum_{j=0}^{\infty} \frac{\lambda^j}{j!} \text{ with } j = i-2) \\
&= \lambda^2.
\end{aligned}$$

Therefore,

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 = \mathrm{E}(X(X-1)) + \mathrm{E}(X) - \mathrm{E}(X)^2 = \lambda^2 + \lambda - \lambda^2 = \lambda,$$

as desired. □

A plot of the Poisson distribution reveals a curve that rises monotonically to a single peak and then decreases monotonically. The peak is as close as possible to the expected value, i.e., at $i = \lfloor \lambda \rfloor$. Figure 2 shows an example for $\lambda = 5$.

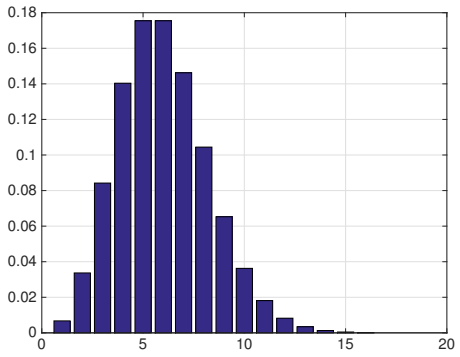


Figure 2: The Poisson distribution with $\lambda = 5$.

Poisson and Coin Flips

To see a concrete example of how Poisson distribution arises, suppose we want to model the number of cell phone users initiating calls in a network during a time period, of duration (say) 1 minute. There are many customers in the network, and all of them can potentially make a call during this time period. However, only a very small fraction of them actually will. Under this scenario, it seems reasonable to make two assumptions:

- The probability of having more than 1 customer initiating a call in any small time interval is negligible.
- The initiation of calls in disjoint time intervals are independent events.

Then if we divide the one-minute time period into n disjoint intervals, than the number of calls X in that time period can be modeled as a binomial random variable with parameter n and probability of success p, i.e., p is the probability of having a call initiated in a time interval of length $1/n$. But what should p be in terms of the relevant parameters of the problem? If calls are initiated at an average rate of λ calls per minute, then $\mathrm{E}(X) = \lambda$ and so $np = \lambda$, i.e., $p = \lambda/n$. So $X \sim \mathit{Bin}(n, \frac{\lambda}{n})$. As we shall see below, as we let n tend to infinity, this distribution tends to the Poisson distribution with parameter λ. We can also see why the Poisson distribution is a model for "rare events." We are thinking of it as a sequence of a large number, n, of coin flips, where we expect only a finite number λ of Heads.

Now we will prove that the Poisson distribution $\mathit{Poiss}(\lambda)$ is the limit of the binomial distribution $\mathit{Bin}(n, \frac{\lambda}{n})$, as n tends to infinity.

Theorem 19.5. *Let $X \sim \mathit{Bin}(n, \frac{\lambda}{n})$ where $\lambda > 0$ is a fixed constant. Then for every $i = 0, 1, 2, \ldots$,*

$$\Pr[X = i] \longrightarrow \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda} \quad \textit{as} \;\; n \to \infty.$$

That is, the probability distribution of X converges to the Poisson distribution with parameter λ.

Proof. Fix $i \in \{0,1,2,\dots\}$, and assume $n \geq i$ (because we will let $n \to \infty$). Then, because X has binomial distribution with parameter n and $p = \frac{\lambda}{n}$,

$$\Pr[X = i] = \binom{n}{i} p^i (1-p)^{n-i} = \frac{n!}{i!(n-i)!} \left(\frac{\lambda}{n}\right)^i \left(1 - \frac{\lambda}{n}\right)^{n-i}.$$

Let us collect the factors into

$$\Pr[X = i] = \frac{\lambda^i}{i!} \left(\frac{n!}{(n-i)!} \cdot \frac{1}{n^i}\right) \cdot \left(1 - \frac{\lambda}{n}\right)^n \cdot \left(1 - \frac{\lambda}{n}\right)^{-i}. \tag{6}$$

The first parenthesis above becomes, as $n \to \infty$,

$$\frac{n!}{(n-i)!} \cdot \frac{1}{n^i} = \frac{n \cdot (n-1) \cdots (n-i+1) \cdot (n-i)!}{(n-i)!} \cdot \frac{1}{n^i} = \frac{n}{n} \cdot \frac{(n-1)}{n} \cdots \frac{(n-i+1)}{n} \to 1.$$

From calculus, the second parenthesis in (6) becomes, as $n \to \infty$,

$$\left(1 - \frac{\lambda}{n}\right)^n \to \mathrm{e}^{-\lambda}.$$

And since i is fixed, the third parenthesis in (6) becomes, as $n \to \infty$,

$$\left(1 - \frac{\lambda}{n}\right)^{-i} \to (1-0)^{-i} = 1.$$

Substituting these results back to (6) gives us

$$\Pr[X = i] \to \frac{\lambda^i}{i!} \cdot 1 \cdot \mathrm{e}^{-\lambda} \cdot 1 = \frac{\lambda^i}{i!} \mathrm{e}^{-\lambda},$$

as desired. □

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Note 2

1 Proofs

In science, evidence is accumulated through experiments to assert the validity of a statement. Mathematics, in contrast, aims for a more absolute level of certainty. A mathematical proof provides a means for *guaranteeing* that a statement is true. Proofs are very powerful and are in some ways like computer programs. Indeed, there is a deep historic link between these two concepts that we will touch upon in this course — the invention of computers is intimately tied to the exploration of the idea of a mathematical proof about a century ago.

So what types of "computer science-related" statements might we want to prove? Here are two examples: (1) Does program P halt on every input? (2) Does program P correctly compute the function $f(x)$, i.e. does it output $f(x)$ on input x, for every x? Note that each of these statements refers to the behavior of a program on *infinitely* many inputs. For such a statement, we can try to provide *evidence* that it is true by testing that it holds for many values of x. Unfortunately, this does not guarantee that the statement holds for the infinitely many values of x that we did not test! To be certain that the statement is true, we must provide a rigorous *proof*.

So what is a proof? A proof is a finite sequence of steps, called *logical deductions*, which establishes the truth of a desired statement. In particular, the power of a proof lies in the fact that using *finite* means, we can guarantee the truth of a statement with *infinitely* many cases.

More specifically, a proof is typically structured as follows. Recall that there are certain statements, called axioms or postulates, that we accept without proof (we have to start somewhere). Starting from these axioms, a proof consists of a sequence of logical deductions: Simple steps that apply the rules of logic. This results in a sequence of statements where each successive statement is necessarily true if the previous statements were true. This property is enforced by the rules of logic: Each statement follows from the previous statements. These rules of logic are a formal distillation of laws that were thought to underlie human thinking. They play a central role in the design of computers, starting with digital logic design or the fundamental principles behind the design of digital circuits. At a more advanced level, these rules of logic play an indispensable role in artificial intelligence, one of whose ultimate goals is to emulate human thought on a computer.

Organization of this note. We begin in Section 2 by setting notation and stating basic mathematical facts used throughout this note. We next introduce four different proof techniques: Direct proof (Section 3), proof by contraposition (Section 4), proof by contradiction (Section 5), and proof by cases (Section 6). We then briefly discuss common pitfalls in and stylistic advice for proofs (Sections 7 and 8, respectively). We close with exercises in Section 9.

2 Notation and basic facts

In this note, we use the following notation and basic mathematical facts. Let $\mathbb{Z}$ denote the set of integers, i.e. $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, and $\mathbb{N}$ the set of natural numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$. Recall that the sum or

product of two integers is an integer, i.e. the set of integers is *closed* under addition and multiplication. The set of natural numbers is also closed under addition and multiplication.

Given integers a and b, we say that a divides b (denoted $a|b$) iff there exists an integer q such that $b = aq$. For example, $2|10$ because there exists an integer $q = 5$ such that $10 = 5 \cdot 2$. We say a natural number p is *prime* if it is divisible only by 1 and itself.

Finally, we use the notation $:=$ to indicate a definition. For example, $q := 6$ defines variable q as having value 6.

3 Direct Proof

With the language of propositional logic from Note 0 under our belts, we can now discuss proof techniques, and the real fun can begin. Are you ready? If so, here is our first technique, known as a *direct proof*. Throughout this section, keep in mind that our goal is give clear and concise proofs. Let's begin with a very simple example.

Theorem 2.1. *For any $a, b, c \in \mathbb{Z}$, if $a|b$ and $a|c$, then $a|(b+c)$.*

Sanity check! Let $P(x,y)$ denote "$x|y$". Convince yourself that the statement above is equivalent to $(\forall a, b, c \in \mathbb{Z})\ (P(a,b) \wedge P(a,c)) \implies P(a,b+c))$.

At a high level, a direct proof proceeds as follows. For each x, the proposition we are trying to prove is of the form $P(x) \implies Q(x)$. A direct proof of this starts by assuming $P(x)$ for a generic value of x and eventually concludes $Q(x)$ through a chain of implications:

Direct Proof
Goal: To prove $P \implies Q$.
Approach: Assume P
$\vdots$
Therefore Q

Proof of Theorem 2.1. Assume that $a|b$ and $a|c$, i.e. there exist integers q_1 and q_2 such that $b = q_1 a$ and $c = q_2 a$. Then, $b + c = q_1 a + q_2 a = (q_1 + q_2)a$. Since the $\mathbb{Z}$ is closed under addition, we conclude that $(q_1 + q_2) \in \mathbb{Z}$, and so $a|(b+c)$, as desired. □

Easy as pie, right? But wait, earlier we said Theorem 2.1 was equivalent to $(\forall a, b, c \in \mathbb{Z})\ (P(a,b) \wedge P(a,c)) \implies P(a,b+c))$; where in the proof above did we encounter the $\forall$ quantifier? The key insight is that the proof did not assume any *specific* values for a, b, and c; indeed, our proof holds for arbitrary $a, b, c \in \mathbb{Z}$! Thus, we have indeed proven the desired claim.

Sanity check! Give a direct proof of the following statement: For any $a, b, c \in \mathbb{Z}$, if $a|b$ and $a|c$, then $a|(b-c)$.

Let's try something a little more challenging.

Theorem 2.2. *Let* $0 < n < 1000$ *be an integer. If the sum of the digits of n is divisible by 9, then n is divisible by 9.*

Observe that this statement is equivalent to

$$(\forall n \in \mathbb{Z}^+)(\text{sum of } n\text{'s digits divisible by } 9 \implies n \text{ divisible by } 9),$$

where $\mathbf{Z}^+$ denotes the set of positive integers, $\{1,2,\ldots\}$. Now the proof proceeds similarly — we start by assuming, for a generic value of n, that the sum of n's digits is divisible by 9. Then we perform a sequence of implications to conclude that n itself is divisible by 9.

Proof of Theorem 2.2. Let n in decimal be written as $n = abc$, i.e. $n = 100a + 10b + c$. Assume that the sum of the digits of n is divisible by 9, i.e.

$$\exists k \in \mathbb{Z} \quad \text{such that} \quad a + b + c = 9k. \tag{1}$$

Adding $99a + 9b$ to both sides of Equation (1), we have

$$100a + 10b + c = n = 9k + 99a + 9b = 9(k + 11a + b).$$

We conclude that n is divisible by 9. □

Is the converse of Theorem 2.2 also true? Recall that the *converse* of $P \implies Q$ is $Q \implies P$. The converse of Theorem 2.2 says that for any integer $0 < n < 1000$, if n is divisible by 9, then the sum of the digits of n is divisible by 9.

Theorem 2.3 (Converse of Theorem 2.2)**.** *Let* $0 < n < 1000$ *be an integer. If n is divisible by 9, then the sum of the digits of n is divisible by 9.*

Proof. Assume that n is divisible by 9. We use the same notation for the digits of n as we used in Theorem 2.2's proof. We proceed as follows.

$$\begin{aligned}
n \text{ is divisible by } 9 &\implies n = 9l \quad \text{for } l \in \mathbb{Z} \\
&\implies 100a + 10b + c = 9l \\
&\implies 99a + 9b + (a + b + c) = 9l \\
&\implies a + b + c = 9l - 99a - 9b \\
&\implies a + b + c = 9(l - 11a - b) \\
&\implies a + b + c = 9k \quad \text{for } k = l - 11a - b \in \mathbf{Z}.
\end{aligned}$$

We conclude that $a + b + c$ is divisible by 9. □

We now come to the moral of this story. We have shown both Theorem 2.2 and its converse, Theorem 2.3. This means that the sum of the digits of n is divisible by 9 *if and only if* n is divisible by 9; in other words, these two statements are logically equivalent. So the key lesson is this: Whenever you wish to prove an equivalence $P \iff Q$, always proceed by showing $P \implies Q$ and $Q \implies P$ separately (as we have done here).

4 Proof by Contraposition

We now move to our second proof technique. Recall from our discussion on propositional logic that any implication $P \implies Q$ is equivalent to its contrapositive $\neg Q \implies \neg P$. Yet, sometimes $\neg Q \implies \neg P$ can be much simpler to prove than $P \implies Q$. Thus, a proof by contraposition proceeds by proving $\neg Q \implies \neg P$ instead of $P \implies Q$.

Proof by Contraposition
Goal: To prove $P \implies Q$.
Approach: Assume $\neg Q$.
$\vdots$
Therefore $\neg P$
Conclusion: $\neg Q \implies \neg P$, which is equivalent to $P \implies Q$.

Consider now the following theorem:

Theorem 2.4. *Let n be a positive integer and let d divide n. If n is odd then d is odd.*

Proving this via the technique of direct proof seems difficult; we would assume n is odd in Step 1, but then what? An approach via contraposition, on the other hand, turns out to be much easier.

Sanity check! What is the contrapositive of Theorem 2.4? (Answer: If d is even, then n is even.)

Proof of Theorem 2.4. We proceed by contraposition. Assume that d is even. Then, by definition, $d = 2k$ for some $k \in \mathbb{Z}$. Because $d|n$, $n = dl$, for some $l \in \mathbb{Z}$. Combining these two statements, we have $n = dl = (2k)l = 2(kl)$. We conclude that n is even. □

Note that this time, the first line of our proof stated our proof technique — this is good practice for any proof, similar to how commenting code is good practice when programming. Stating your proof technique like this an enormous aid to your reader in understanding where your proof will go next. (Let us not forget that a reader who understands your proof, such a teaching assistant or instructor, is much more likely to give you a good grade for it!)

5 Proof by Contradiction

Of all the proof techniques we discuss in this note, it's perhaps hardest to resist the appeal of this one; after all, who wouldn't want to use a technique known as *reductio ad absurdum*, i.e. reduction to an absurdity? The idea in a proof by contradiction is to assume that the claim you wish to prove is *false* (yes, this seems backwards, but bear with us). Then, you show that this leads to a conclusion which is utter nonsense: A contradiction. Hence, you conclude that your claim must in fact have been true.

Sanity check! A proof by contradiction relies crucially on the fact that if a proposition is not false, then it must be true. Which law from a previous lecture embodied this black or white interpretation of a statement?

Proof by Contradiction

Goal: To prove P.

Approach: Assume $\neg P$.

$$\vdots$$
$$R$$
$$\vdots$$
$$\neg R$$

Conclusion: $\neg P \implies \neg R \wedge R$, which is a contradiction. Thus, P.

If you are not convinced by the intuitive explanation thus far as to why proof by contradiction works, here is the formal reasoning: A proof by contradiction shows that $\neg P \implies \neg R \wedge R \equiv$ False. The contrapositive of this statement is hence True $\implies P$.

Let us now take this proof technique on a trial run. Note that in doing so, we are continuing a long-standing legacy — the proof of the theorem below dates back more than 2000 years to the ancient Greek mathematician, Euclid of Alexandria![1]

Theorem 2.5. *There are infinitely many prime numbers.*

To appreciate the power of contradiction, let us pause for a moment to ponder how we might try to prove Theorem 2.5 via a different proof technique, such as, say, a direct proof. It seems very difficult, right? How would you construct infinitely many prime numbers? The remarkable thing about contradiction, however, is that if we assume the statement is false, i.e. there are only finitely many primes, bad things will happen.

To proceed, we now state a simple lemma which is handy in showing Theorem 2.5. Its proof will be deferred to a future lecture in which we learn about induction.

Lemma 2.1. *Every natural number greater than one is either prime or has a prime divisor.*

Proof of Theorem 2.5. We proceed by contradiction. Suppose that Theorem 2.5 is false, i.e. that there are only finitely many primes, say k of them. Then, we can enumerate them: $p_1, p_2, p_3, \ldots, p_k$.

Now, define number $q := p_1 p_2 p_3 \ldots p_k + 1$, which is the product of all primes plus one. We claim that q cannot be prime. Why? Because by definition, it is larger than all the primes p_1 through p_k! By Lemma 2.1, we therefore conclude that q has a prime divisor, p. This will be our statement R.

Next, because $p_1, p_2, p_3, \ldots, p_k$ are all the primes, p must be equal to one of them; thus, p divides $r := p_1 p_2 p_3 \ldots p_k$. Hence, $p|q$ and $p|r$, implying $p|(q-r)$. But $q - r = 1$, implying $p \leq 1$, and hence p is not prime; this is the statement $\neg R$. We thus have $R \wedge \neg R$, which is a contradiction, as desired. □

Now that we're warmed up, let's tackle another classic proof involving contradictions. Recall that a **rational number** is a number that can be expressed as the ratio of two integers. For example, $\frac{2}{3}$, $\frac{3}{5}$, and $\frac{9}{16}$ are rational numbers. Numbers which *cannot* be expressed as fractions, on the other hand, are called **irrational**. Now, how about $\sqrt{2}$? Do you think it's rational or irrational? The answer is as follows.

Theorem 2.6. $\sqrt{2}$ *is irrational.*

[1] It is perhaps worth pausing here to appreciate the true scale of this statement — after all, how many aspects of our human heritage remain relevant after multiple millenia? Music? Fashion? All of these are quickly outdated with time. But mathematics is, in a sense, timeless.

Before giving the proof, let us ask a crucial question: Why should contradiction be a good candidate proof technique to try here? Well, consider this: Theorem 2.5 and Theorem 2.6 share something fundamental in common — in both cases, we wish to show that something *doesn't* exist. For example, for Theorem 2.5, we wished to show that a largest prime doesn't exist, and for Theorem 2.6, we wish to show that integers a and b satisfying $\sqrt{2} = a/b$ don't exist. In general, proving that something *doesn't* exist seems difficult. But this is actually one setting in which proof by contradiction shines.

To prove Theorem 2.6, we use the following simple lemma. In Section 9, we ask you to prove Lemma 2.2.

Lemma 2.2. *If a^2 is even, then a is even.*

Proof of Theorem 2.6. We proceed by contradiction. Assume that $\sqrt{2}$ is rational. By the definition of rational numbers, there are integers a and b with no common factor other than 1, such that $\sqrt{2} = a/b$. Let our assertion R state that a and b share no common factors.

Now, for any numbers x and y, we know that $x = y \implies x^2 = y^2$. Hence $2 = a^2/b^2$. Multiplying both sides by b^2, we have $a^2 = 2b^2$. Since b is an integer, it follows that b^2 is an integer, and thus a^2 is even (by the definition of evenness). Plugging in Lemma 2.2, we hence have that a is even. In other words, there exists integer c such that $a = 2c$.

Combining all our facts thus far, we have that $2b^2 = 4c^2$, or $b^2 = 2c^2$. Since c is an integer, c^2 is an integer, and hence b^2 is even. Thus, again applying Lemma 2.2, we conclude that b is even.

But we have just shown that both a and b are even. In particular, this means they share the common factor 2. This implies $\neg R$. We conclude that $R \vee \neg R$ holds; thus, we have a contradiction, as desired. □

6 Proof by Cases

Here is a proof to tickle your fancy; it relies on another proof technique known as proof by *cases*, which we will touch on informally in this section. Specifically, the idea behind a proof by cases is as follows: Sometimes when we wish to prove a claim, we don't know which of a set of possible cases is true, but we know that *at least one* of the cases is true. What we can do then is to prove the result in *both* cases; then, clearly the general statement must hold.

Theorem 2.7. *There exist irrational numbers x and y such that x^y is rational.*

Proof. We proceed by cases. Note that the statement of the theorem is quantified by an existential quantifier: Thus, to prove our claim, it suffices to demonstrate a single x and y such that x^y is rational. To do so, let $x = \sqrt{2}$ and $y = \sqrt{2}$. Let us divide our proof into two cases, exactly one of which must be true:

(a) $\sqrt{2}^{\sqrt{2}}$ is rational, or

(b) $\sqrt{2}^{\sqrt{2}}$ is irrational.

(Case (a)) Assume first that $\sqrt{2}^{\sqrt{2}}$ is rational. But this immediately yields our claim, since x and y are irrational numbers such that x^y is rational.

(Case (b)) Assume now that $\sqrt{2}^{\sqrt{2}}$ is irrational. Our first guess for x and y was not quite right, but now we have a new irrational number to play with, $\sqrt{2}^{\sqrt{2}}$. So, let's try setting $x = \sqrt{2}^{\sqrt{2}}$ and $y = \sqrt{2}$. Then,

$$x^y = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\sqrt{2}} = \sqrt{2}^2 = 2,$$

where the second equality follows from the axiom $(x^y)^z = x^{yz}$. But now we again started with two irrational numbers x and y and obtained rational x^y.

Since one of case (a) or case (b) must hold, we thus conclude that the statement of Theorem 2.7 is true. □

Before closing, let us point out a peculiarity of the proof above. What were the *actual* numbers x and y satisfying the claim of Theorem 2.7? Were they $x = \sqrt{2}$ and $y = \sqrt{2}$? Or $x = \sqrt{2}^{\sqrt{2}}$ and $y = \sqrt{2}$? Well, since we did a case analysis, it's not clear which of the two choices is actually the correct one. In other words, we have just demonstrated something rather remarkable known as a **nonconstructive** proof: We've proven that some object X exists, but without explicitly revealing what X itself is!

7 Common Errors When Writing Proofs

The ability to write clean and concise proofs is a remarkable thing, and is arguably among the highest forms of intellectual enlightenment one can achieve. It requires your mind to critically reflect on its own inner workings (i.e. your thought processes), and reorganize them into a coherent and logical sequence of thoughts. In other words, your mind is improving itself at a very fundamental level, far transcending the boundaries of computer science or any particular area of study. The benefits of this training will touch every aspect of your life as you know it; indeed, it will shape the way you approach life itself.

As with any such fundamental achievement, developing the ability to write rigorous proofs is likely among the most difficult learning challenges you will face in university, so do not despair if it gives you trouble; you are not alone. There is simply no substitute here for lots and lots of practice. To help get you started on your way, we now raise some red flags regarding common pitfalls in composing proofs. Let us begin with a simple, but common error.

Claim: $-2 = 2$.

Proof? Assume $-2 = 2$. Squaring both sides, we have $(-2)^2 = 2^2$, or $4 = 4$, which is true. We conclude that $-2 = 2$, as desired. ♠

The theorem is obviously false, so what did we do wrong? Our arithmetic is correct, and each step rigorously follows from the previous step. So, the error must lie in the very beginning of the proof, where we made a brazen assumption: That $-2 = 2$. But wait, wasn't this the very statement we were trying to prove? Exactly. In other words, to prove the statement $P \equiv$"$-2 = 2$", we just proved that $P \Longrightarrow$ True, which is not the same as proving P. Lesson #1: When writing proofs, do not assume the claim you aim to prove!

Lesson #2 is about the number zero: In particular, never forget to consider the case where your variables take on the value 0. Otherwise, this can happen:

Claim: $1 = 2$.

Proof? Assume that $x = y$ for integers $x, y \in \mathbb{Z}$. Then,

$$\begin{aligned} x^2 - xy &= x^2 - y^2 && \text{(since } x = y\text{)} \\ x(x-y) &= (x+y)(x-y) && \\ x &= x+y && \text{(divide both sides by } x-y\text{)} \\ x &= 2x. && \end{aligned}$$

Setting $x = y = 1$ yields the claim. ♠

But, clearly $1 \neq 2$, unless your grade school teachers were lying to you. Where did we go wrong? In deriving

the third equality, we divided by $(x-y)$. What is the value of $(x-y)$ in our setting? Zero. Dividing by zero is not well-defined; thus the third equality does not hold.

Lesson #3 says to be careful when mixing negative numbers and inequalities. For example:

Claim: $4 \leq 1$.

Proof? We know that $-2 \leq 1$; squaring both sides of this inequality yields $4 \leq 1$. ♠

Sanity check! To see why this proof fails, ask yourself this: If $a \leq b$, is it necessarily true that $|a| \leq |b|$? Can you give a counterexample?

In addition, do not forget that multiplying an inequality by a negative number flips the direction of the inequality! For example, multiplying both sides of $-2 < 5$ by -1 yields $2 > -5$, as you would expect.

8 Style and substance in proofs

We conclude with some general words of advice. First, get in the habit of thinking carefully before you write down the next sentence of your proof. If you cannot explain clearly why the step is justified, you are making a leap and you need to go back and think some more. In theory, each step in a proof must be justified by appealing to a definition or general axiom. In practice the depth to which one must do this is a matter of taste. For example, we could break down the step, "Since a is an integer, $(2a^2+2a)$ is an integer," into several more steps. [Exercise: what are they?] A justification can be stated without proof only if you are absolutely confident that (1) it is correct and (2) the reader will automatically agree that it is correct.

Notice that in the proof that $\sqrt{2}$ is irrational, we used the result, "For any integer n, if n^2 is even then n is even," twice. This suggests that it may be a useful fact in many proofs. A subsidiary result that is useful in a more complex proof is called a *lemma*. It is often a good idea to break down a long proof into several lemmas. This is similar to the way in which large programming tasks should be divided up into smaller subroutines. Furthermore, make each lemma (like each subroutine) as general as possible so it can be reused elsewhere.

The dividing line between lemmas and theorems is not clear-cut. Usually, when writing a paper, the theorems are those propositions that you want to "export" from the paper to the rest of the world, whereas the lemmas are propositions used locally in the proofs of your theorems. There are, however, some lemmas (for example, the Pumping Lemma and the Lifting Lemma) that are perhaps more famous and important than the theorems they were used to prove.

Finally, you should remember that the point of this lecture was not the specific statements we proved, but the different proof strategies, and their logical structure. Make sure you understand them clearly; you will be using them when you write your own proofs in homework and exams.

9 Exercises

1. Generalize the proof of Theorem 2.2 so that it works for *any* positive integer n. (HINT: Suppose n has k digits, and write a_i for the digits of n, so that $n = \sum_{i=0}^{k-1}(a_i \cdot 10^i)$.)

2. Prove Lemma 2.2. (Hint: First try a direct proof. Then, try contraposition. Which proof approach is better suited to proving this lemma?)

A Brief Introduction to Continuous Probability

Up to now we have focused exclusively on *discrete* probability spaces Ω, where the number of sample points $\omega \in \Omega$ is either finite or countably infinite (such as the integers). As a consequence, we have only been able to talk about *discrete* random variables, which take on only a finite or countably infinite number of values.

But in real life many quantities that we wish to model probabilistically are *real-valued*; examples include the position of a particle in a box, the time at which an certain incident happens, or the direction of travel of a meteorite. In this lecture, we discuss how to extend the concepts we've seen in the discrete setting to this *continuous* setting. As we shall see, everything translates in a natural way once we have set up the right framework. The framework involves some elementary calculus but (at this level of discussion) nothing too scary.

Continuous Uniform Probability Space

Suppose we spin a "wheel of fortune" and record the position of the pointer on the outer circumference of the wheel. Assuming that the circumference is of length ℓ and that the wheel is unbiased, the position is presumably equally likely to take on any value in the real interval $[0,\ell]$. How do we model this experiment using a probability space?

Consider for a moment the analogous discrete setting, where the pointer can stop only at a finite number m of positions distributed evenly around the wheel. (If m is very large, then this is in some sense similar to the continuous setting, which we can think of as the limit $m \to \infty$.) Then we would model this situation using the discrete sample space $\Omega = \{0, \frac{\ell}{m}, \frac{2\ell}{m}, \ldots, \frac{(m-1)\ell}{m}\}$, with uniform probabilities $\Pr[\omega] = \frac{1}{m}$ for each $\omega \in \Omega$.

In the continuous setting, however, we get into trouble if we try the same approach. If we let ω range over all real numbers in $\Omega = [0,\ell]$, what value should we assign to each $\Pr[\omega]$? By uniformity, this probability should be the same for all ω. But if we assign $\Pr[\omega]$ to be any positive value, then because there are infinitely many ω in Ω, the sum of all probabilities $\Pr[\omega]$ will be ∞! Thus, $\Pr[\omega]$ must be zero for all $\omega \in \Omega$. But if all of our sample points have probability zero, then we are unable to assign meaningful probabilities to any events!

To resolve this problem, consider instead any non-empty *interval* $[a,b] \subseteq [0,\ell]$. Can we assign a non-zero probability value to this interval? Since the total probability assigned to $[0,\ell]$ must be 1, and since we want our probability to be uniform, the logical value for the probability of interval $[a,b]$ is

$$\frac{\text{length of } [a,b]}{\text{length of } [0,\ell]} = \frac{b-a}{\ell}.$$

In other words, the probability of an interval is proportional to its length.

Note that intervals are subsets of the sample space Ω and are therefore *events*. So in contrast to discrete probability, where we assigned probabilities to *points* in the sample space, in continuous probability we are assigning probabilities to certain basic events (in this case intervals). What about probabilities of other

events? By specifying the probability of intervals, we have also specified the probability of any event E which can be written as the disjoint union of (a finite or countably infinite number of) intervals, $E = \cup_i E_i$. For then we can write $\Pr[E] = \sum_i \Pr[E_i]$, in analogous fashion to the discrete case. Thus for example the probability that the pointer ends up in the first or third quadrants of the wheel is $\frac{\ell/4}{\ell} + \frac{\ell/4}{\ell} = \frac{1}{2}$. For all practical purposes, such events are all we really need.[1]

Continuous Random Variables

Recall that in the discrete setting we typically work with *random variables* and their distributions, rather than directly with probability spaces and events. The simplest example of a continuous random variable is the position X of the pointer in the wheel of fortune, as discussed above. This random variable has the *uniform* distribution on $[0, \ell]$. How, precisely, should we define the distribution of a continuous random variable? In the discrete case the distribution of a r.v. X is described by specifying, for each possible value a, the probability $\Pr[X = a]$. But for the r.v. X corresponding to the position of the pointer, we have $\Pr[X = a] = 0$ for every a, so we run into the same problem as we encountered above in defining the probability space.

The resolution is the same: instead of specifying $\Pr[X = a]$, we specify $\Pr[a \leq X \leq b]$ for all intervals $[a, b]$.[2] To do this formally, we need to introduce the concept of a *probability density function* (sometimes referred to just as a "density", or a "pdf").

Definition 20.1 (Density)**.** *A* probability density function *for a real-valued random variable X is a function $f : \mathbb{R} \to \mathbb{R}$ satisfying:*

1. *f is non-negative: $f(x) \geq 0$ for all $x \in \mathbb{R}$.*
2. *The total integral of f is equal to 1: $\int_{-\infty}^{\infty} f(x)\,dx = 1$.*

Then the distribution of X is given by:

$$\Pr[a \leq X \leq b] = \int_a^b f(x)dx \qquad \text{for all } a \leq b.$$

Let's examine this definition. Note that the definite integral is just the area under the curve f between the values a and b. Thus f plays a similar role to the "histogram" we sometimes draw to picture the distribution of a discrete random variable. The first condition that f be non-negative ensures that the probability of every event is non-negative. The second condition that the total integral of f equal to 1 ensures that it defines a valid probability distribution, because the r.v. X must take on real values:

$$\Pr[X \in \mathbb{R}] = \Pr[-\infty < X < \infty] = \int_{-\infty}^{\infty} f(x)dx = 1. \tag{1}$$

For example, consider the wheel-of-fortune r.v. X, which has uniform distribution on the interval $[0, \ell]$. This means the density f of X vanishes outside this interval: $f(x) = 0$ for $x < 0$ and for $x > \ell$. Within the interval $[0, \ell]$ we want the distribution of X to be uniform, which means we should take f to be a constant $f(x) = c$ for $0 \leq x \leq \ell$. The value of c is determined by the requirement (1) that the total area under f is 1:

$$1 = \int_{-\infty}^{\infty} f(x)dx = \int_0^{\ell} c\,dx = c\ell,$$

[1] A formal treatment of which events can be assigned a well-defined probability requires a discussion of *measure theory*, which is beyond the scope of this course.

[2] Note that it does not matter whether or not we include the endpoints a, b; since $\Pr[X = a] = \Pr[X = b] = 0$, we have $\Pr[a < X < b] = \Pr[a \leq X \leq b]$.

which gives us $c = \frac{1}{\ell}$. Therefore, the density of the uniform distribution on $[0, \ell]$ is given by

$$f(x) = \begin{cases} 0 & \text{for } x < 0; \\ 1/\ell & \text{for } 0 \le x \le \ell; \\ 0 & \text{for } x > \ell. \end{cases}$$

Remark: Following the "histogram" analogy above, it is tempting to think of $f(x)$ as a "probability." However, $f(x)$ doesn't itself correspond to the probability of anything! In particular, there is no requirement that $f(x)$ be bounded by 1. For example, the density of the uniform distribution on the interval $[0, \ell]$ with $\ell = \frac{1}{2}$ is equal to $f(x) = 1/(\frac{1}{2}) = 2$ for $0 \le x \le \frac{1}{2}$, which is greater than 1. To connect density $f(x)$ with probabilities, we need to look at a very small interval $[x, x+\delta]$ close to x; then we have

$$\Pr[x \le X \le x+\delta] = \int_x^{x+\delta} f(z)dz \approx \delta f(x). \tag{2}$$

Thus, we can interpret $f(x)$ as the "probability per unit length" in the vicinity of x.

Expectation and Variance

As in the discrete case, we define the expectation of a continuous r.v. as follows:

Definition 20.2 (Expectation)**.** *The expectation of a continuous r.v. X with probability density function f is*

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f(x)dx.$$

Note that the integral plays the role of the summation in the discrete formula $\mathrm{E}(X) = \sum_a a \Pr[X = a]$. Similarly, we can define the variance as follows:

Definition 20.3 (Variance)**.** *The variance of a continuous r.v. X with probability density function f is*

$$\mathrm{Var}(X) = \mathrm{E}((X - \mathrm{E}(X))^2) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 = \int_{-\infty}^{\infty} x^2 f(x)dx - \left(\int_{-\infty}^{\infty} x f(x)dx\right)^2.$$

Example: Let X be a uniform r.v. on the interval $[0, \ell]$. Then intuitively, its expected value should be in the middle, $\frac{\ell}{2}$. Indeed, we can use our definition above to compute

$$\mathrm{E}(X) = \int_0^{\ell} x \frac{1}{\ell} dx = \left[\frac{x^2}{2\ell}\right]_0^{\ell} = \frac{\ell}{2},$$

as claimed. We can also calculate its variance using the above definition and plugging in the value $\mathrm{E}(X) = \frac{\ell}{2}$ to get:

$$\mathrm{Var}(X) = \int_0^{\ell} x^2 \frac{1}{\ell} dx - \mathrm{E}(X)^2 = \left[\frac{x^3}{3\ell}\right]_0^{\ell} - \left(\frac{\ell}{2}\right)^2 = \frac{\ell^2}{3} - \frac{\ell^2}{4} = \frac{\ell^2}{12}.$$

The factor of $\frac{1}{12}$ here is not particularly intuitive, but the fact that the variance is proportional to ℓ^2 should come as no surprise. Like its discrete counterpart, this distribution has large variance.

Joint Distribution

Recall that for discrete random variables X and Y, their joint distribution is specified by the probabilities $\Pr[X = a, Y = c]$ for all possible values a, c. Similarly, if X and Y are continuous random variables, then their joint distributions are specified by the probabilities $\Pr[a \le X \le b, c \le Y \le d]$ for all $a \le b$, $c \le d$. Moreover, just as the distribution of X can be characterized by its density function, the joint distribution of X and Y can be characterized by their joint density.

Definition 20.4 (Joint Density)**.** *A* joint density function *for two random variable X and Y is a function $f : \mathbb{R}^2 \to \mathbb{R}$ satisfying:*

1. *f is non-negative: $f(x,y) \ge 0$ for all $x, y \in \mathbb{R}$.*
2. *The total integral of f is equal to 1: $\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\,dx\,dy = 1$.*

Then the joint distribution of X and Y is given by:

$$\Pr[a \le X \le b,\ c \le Y \le d] = \int_c^d \int_a^b f(x,y)\,dx\,dy \qquad \text{for all } a \le b \text{ and } c \le d.$$

In analogy with (2), we can connect the joint density $f(x,y)$ with probabilities by looking at a very small square $[x, x+\delta] \times [y, y+\delta]$ close to (x,y); then we have

$$\Pr[x \le X \le x+\delta,\ y \le Y \le y+\delta] = \int_y^{y+\delta} \int_x^{x+\delta} f(u,v)\,du\,dv \approx \delta^2 f(x,y). \tag{3}$$

Thus we can interpret $f(x,y)$ as the "probability per unit area" in the vicinity of (x,y).

Independence

Recall that two discrete random variables X and Y are said to be independent if the events $X = a$ and $Y = c$ are independent for every possible values a, c. We have a similar definition for continuous random variables:

Definition 20.5 (Independence for Continuous R.V.'s)**.** *Two continuous r.v.'s X, Y are independent if the events $a \le X \le b$ and $c \le Y \le d$ are independent for all $a \le b$ and $c \le d$:*

$$\Pr[a \le X \le b,\ c \le Y \le d] = \Pr[a \le X \le b] \cdot \Pr[c \le Y \le d].$$

What does this definition say about the joint density of independent r.v.'s X and Y? Applying (3) to connect the joint density with probabilities, we get, for small δ:

$$\begin{aligned}
\delta^2 f(x,y) &\approx \Pr[x \le X \le x+\delta,\ y \le Y \le y+\delta] \\
&= \Pr[x \le X \le x+\delta] \cdot \Pr[y \le Y \le y+\delta] \qquad \text{(by independence)} \\
&\approx \delta f_X(x) \times \delta f_Y(y) \\
&= \delta^2 f_X(x) f_Y(y),
\end{aligned}$$

where f_X and f_Y are the (marginal) densities of X and Y respectively. So we get the following result:

Theorem 20.1. *The joint density of independent r.v.'s X and Y is the product of the marginal densities:*

$$f(x,y) = f_X(x)\, f_Y(y) \quad \text{for all } x, y \in \mathbb{R}.$$

Two Important Continuous Distributions

We have already seen one important continuous distribution, namely the uniform distribution. In this section we will see two more: the *exponential* distribution and the *normal* (or *Gaussian*) distribution. These three distributions cover the vast majority of continuous random variables arising in applications.

Exponential Distribution

The exponential distribution is a continuous version of the geometric distribution, which we have already seen in the previous note. Recall that the geometric distribution describes the number of tosses of a coin until the first Head appears; the distribution has a single parameter p, which is the bias (Heads probability) of the coin. Of course, in real life applications we are usually not waiting for a coin to come up Heads but rather waiting for a system to fail, a clock to ring, an experiment to succeed, etc.

In such applications we are frequently not dealing with discrete events or discrete time, but rather with *continuous* time: for example, if we are waiting for an apple to fall off a tree, it can do so at any time at all, not necessarily on the tick of a discrete clock. This situation is naturally modeled by the exponential distribution, defined as follows:

Definition 20.6 (Exponential distribution)**.** *For $\lambda > 0$, a continuous random variable X with density function*

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0; \\ 0 & \text{otherwise.} \end{cases}$$

is called an exponential random variable with parameter λ.

Note that by definition $f(x)$ is non-negative. Moreover, we can check that it satisfies (1):

$$\int_{-\infty}^{\infty} f(x)dx = \int_{0}^{\infty} \lambda e^{-\lambda x} dx = \left[-e^{-\lambda x}\right]_0^{\infty} = 1,$$

so $f(x)$ is indeed a valid probability density function. Figure 1 shows the plot of the density function for the exponential distribution with $\lambda = 2$.

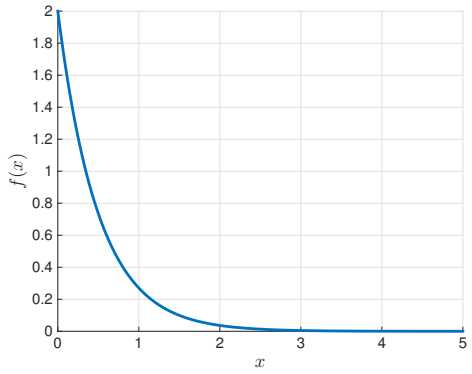


Figure 1: The density function for the exponential distribution with $\lambda = 2$.

Let us now compute its expectation and variance.

Theorem 20.2. *Let X be an exponential random variable with parameter $\lambda > 0$. Then*

$$\mathrm{E}(X) = \frac{1}{\lambda} \qquad \textit{and} \qquad \mathrm{Var}(X) = \frac{1}{\lambda^2}.$$

Proof. We can calculate the expected value using integration by parts:

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f(x)dx = \int_0^{\infty} \lambda x e^{-\lambda x}dx = \left[-xe^{-\lambda x}\right]_0^{\infty} + \int_0^{\infty} e^{-\lambda x}dx = 0 + \left[-\frac{e^{-\lambda x}}{\lambda}\right]_0^{\infty} = \frac{1}{\lambda}.$$

To compute the variance, we first evaluate $\mathrm{E}(X^2)$, again using integration by parts:

$$\mathrm{E}(X^2) = \int_{-\infty}^{\infty} x^2 f(x)dx = \int_0^{\infty} \lambda x^2 e^{-\lambda x}dx = \left[-x^2 e^{-\lambda x}\right]_0^{\infty} + \int_0^{\infty} 2xe^{-\lambda x}dx = 0 + \frac{2}{\lambda}\mathrm{E}(X) = \frac{2}{\lambda^2}.$$

The variance is therefore

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 = \frac{2}{\lambda^2} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2},$$

as claimed. □

Exponential distribution is the continuous time analog of geometric distribution

Like the geometric distribution, the exponential distribution has a single parameter λ, which characterizes the *rate* at which events happen. Note that the exponential distribution satisfies, for any $t \geq 0$,

$$\Pr[X > t] = \int_t^{\infty} \lambda e^{-\lambda x}dx = \left[-e^{-\lambda x}\right]_t^{\infty} = e^{-\lambda t}. \tag{4}$$

In other words, the probability that we have to wait more than time t for our event to happen is $e^{-\lambda t}$, which is an exponential decay with rate λ.

Now consider a discrete-time setting in which we perform one trial every δ seconds (where δ is very small — in fact, we will take $\delta \to 0$ to make time "continuous"), and where our success probability is $p = \lambda\delta$. Making the success probability proportional to δ makes sense, as it corresponds to the natural assumption that there is a fixed *rate* of success *per unit time*, which we denote by $\lambda = p/\delta$. In this discrete setting, the number of trials until we get a success has the geometric distribution with parameter p, so if we let the r.v. Y denote the time (in seconds) until we get a success we have

$$\Pr[Y > k\delta] = (1-p)^k = (1-\lambda\delta)^k \qquad \text{for any } k \geq 0.$$

Hence, for any $t > 0$, we have

$$\Pr[Y > t] = \Pr[Y > (\tfrac{t}{\delta})\delta] = (1-\lambda\delta)^{t/\delta} \approx e^{-\lambda t}, \tag{5}$$

where this final approximation holds in the limit as $\delta \to 0$ with λ and t fixed. (We are ignoring the detail of rounding $\frac{t}{\delta}$ to an integer since we are taking an approximation anyway.)

Comparing the expression (5) with (4), we see that this distribution has the same form as the exponential distribution with parameter λ, where λ (the success rate per unit time) plays an analogous role to p (the probability of success on each trial) — though note that λ is not constrained to be ≤ 1. Thus we may view the exponential distribution as a continuous time analog of the geometric distribution.

Normal Distribution

The last continuous distribution we will look at, and by far the most prevalent in applications, is called the *normal* or *Gaussian* distribution. It has two parameters, μ and σ, which are the mean and standard deviation of the distribution, respectively.

Definition 20.7 (Normal distribution)**.** *For any $\mu \in \mathbb{R}$ and $\sigma > 0$, a continuous random variable X with pdf*

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2}$$

is called a normal random variable with parameters μ and σ. In the special case $\mu = 0$ and $\sigma = 1$, X is said to have the standard normal distribution.

Let's first check that this is a valid definition of a probability density function. Clearly $f(x) \geq 0$ from the definition. For condition (1):

$$\int_{-\infty}^{\infty} f(x)dx = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} e^{-(x-\mu)^2/2\sigma^2} = 1. \tag{6}$$

The fact that this integral evaluates to 1 is a routine exercise in integral calculus, and is left as an exercise (or feel free to look it up in any standard book on probability or on the internet).

A plot of the pdf f reveals a classical "bell-shaped" curve, centered at (and symmetric around) $x = \mu$, and with "width" determined by σ. Figure 2 shows the normal density for several different choices of μ and σ.

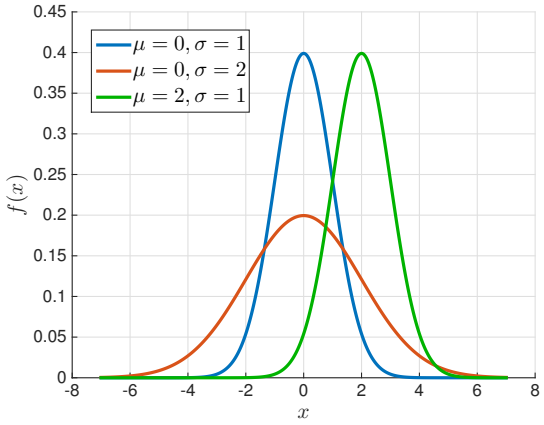


Figure 2: The density function for the normal distribution with several different choices for μ and σ.

The figure above shows that the normal density with different values of μ and σ are very similar to each other. Indeed, the normal distribution has the following nice property with respect to shifting and rescaling.

Lemma 20.1. *If X is a normal random variable with parameters μ and σ, then $Y = \frac{X-\mu}{\sigma}$ is a standard normal random variable. Equivalently, if Y is a standard normal random variable, then $X = \sigma Y + \mu$ has normal distribution with parameters μ and σ.*

Proof. Given that X has normal distribution with parameters μ and σ, we can calculate the distribution of $Y = \frac{X-\mu}{\sigma}$ as:

$$\Pr[a \leq Y \leq b] = \Pr[\sigma a + \mu \leq X \leq \sigma b + \mu] = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{\sigma a+\mu}^{\sigma b+\mu} e^{-(x-\mu)^2/2\sigma^2} dx = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-y^2/2} dy,$$

by a simple change of variable $x = \sigma y + \mu$ in the integral. Hence Y is indeed standard normal. Note that Y is obtained from X just by shifting the origin to μ and scaling by σ. □

Let us now calculate the expectation and variance of a normal random variable.

Theorem 20.3. *For a normal random variable X with parameters μ and σ,*

$$\mathrm{E}(X) = \mu \qquad \textit{and} \qquad \mathrm{Var}(X) = \sigma^2.$$

Proof. Let's first consider the case when X is standard normal, i.e., when $\mu = 0$ and $\sigma = 1$. By definition, its expectation is

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f(x) dx = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} x e^{-x^2/2} dx = \frac{1}{\sqrt{2\pi}} \left(\int_{-\infty}^{0} x e^{-x^2/2} dx + \int_{0}^{\infty} x e^{-x^2/2} dx \right) = 0.$$

The last step follows from the fact that the function $e^{-x^2/2}$ is symmetrical about $x = 0$, so the two integrals are the same except for the sign. For the variance, we have

$$\begin{aligned} \mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}(X)^2 &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} x^2 e^{-x^2/2} dx \\ &= \frac{1}{\sqrt{2\pi}} \left[-x e^{-x^2/2} \right]_{-\infty}^{\infty} + \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-x^2/2} dx \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-x^2/2} dx = 1. \end{aligned}$$

In the first line here we used the fact that $\mathrm{E}(X) = 0$; in the second line we used integration by parts; and in the last line we used (6) in the special case $\mu = 0$, $\sigma = 1$. So the standard normal distribution has expectation $\mathrm{E}(X) = 0 = \mu$ and variance $\mathrm{Var}(X) = 1 = \sigma^2$.

Now consider the general case when X has normal distribution with general parameters μ and σ. By Lemma 20.1, we know that $Y = \frac{X-\mu}{\sigma}$ is a standard normal random variable, so $\mathrm{E}(Y) = 0$ and $\mathrm{Var}(Y) = 1$, as we have just established above. Therefore, we can read off the expectation and variance of X from those of Y. For the expectation, using linearity, we have

$$0 = \mathrm{E}(Y) = \mathrm{E}\left(\frac{X-\mu}{\sigma} \right) = \frac{\mathrm{E}(X) - \mu}{\sigma},$$

and hence $\mathrm{E}(X) = \mu$. For the variance we have

$$1 = \mathrm{Var}(Y) = \mathrm{Var}\left(\frac{X-\mu}{\sigma} \right) = \frac{\mathrm{Var}(X)}{\sigma^2},$$

and hence $\mathrm{Var}(X) = \sigma^2$. □

The bottom line, then, is that the normal distribution has expectation μ and variance σ^2. This explains the notation for the parameters μ and σ.

The fact that the variance is σ^2 (so that the standard deviation is σ) explains our earlier comment that σ determines the "width" of the normal distribution. Namely, by Chebyshev's inequality, a constant fraction of the distribution lies within distance (say) 2σ of the expectation μ.

Note: The above analysis shows that, by means of a simple origin shift and scaling, we can relate any normal distribution to the standard normal. This means that, when doing computations with normal distributions, it's enough to do them for the standard normal. For this reason, books and online sources of mathematical formulas usually contain tables describing the density of the standard normal. From this, one can read off the corresponding information for any normal r.v. X with parameters μ, σ^2, from the formula

$$\Pr[X \le a] = \Pr[Y \le \tfrac{a-\mu}{\sigma}],$$

where Y is standard normal.

The normal distribution is ubiquitous throughout the sciences and the social sciences, because it is the standard model for any aggregate data that results from a large number of independent observations of the same random variable (such as the heights of females in the US population, or the observational error in a physical experiment). Such data, as is well known, tends to cluster around its mean in a "bell-shaped" curve, with the correspondence becoming more accurate as the number of observations increases. A theoretical explanation of this phenomenon is the Central Limit Theorem, which we next discuss.

Sum of Independent Normal Random Variables

An important property of the normal distribution is that the sum of *independent* normal random variables is also normally distributed. We begin with the simple case when X and Y are independent standard normal random variables. In this case the result follows because the joint distribution of X and Y is rotationally symmetric. The general case follows from the translation and scaling property of normal distribution in Lemma 20.1.

Theorem 20.4. *Let X and Y be independent standard normal random variables, and $a, b \in \mathbb{R}$ be constants. Then $Z = aX + bY$ is also a normal random variable with parameters $\mu = 0$ and $\sigma^2 = a^2 + b^2$.*

Proof. [3] Since X and Y are independent, by Theorem 20.1 we know that the joint density of (X,Y) is

$$f(x,y) = f(x) \cdot f(y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}.$$

The key observation is that $f(x,y)$ is rotationally symmetric around the origin, i.e., $f(x,y)$ only depends on the value $x^2 + y^2$, which is the distance of the point (x,y) from the origin $(0,0)$; see Figure 3.

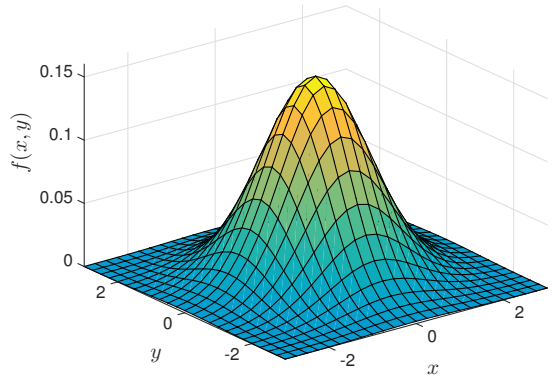


Figure 3: The joint density function $f(x,y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}$ is rotationally symmetric.

Thus, $f(T(x,y)) = f(x,y)$ where T is any rotation of the plane $\mathbb{R}^2$ about the origin. It follows that for any set $A \subseteq \mathbb{R}^2$,

$$\Pr[(X,Y) \in A] = \Pr[(X,Y) \in T(A)] \tag{7}$$

[3] The following proof and figures are adapted from *"Why Is the Sum of Independent Normal Random Variables Normal?"* by B. Eisenberg and R. Sullivan, Mathematics Magazine, Vol. 81, No. 5.

where T is a rotation of $\mathbb{R}^2$. Now given any $t \in \mathbb{R}$, we have

$$\Pr[Z \le t] \;=\; \Pr[aX + bY \le t] \;=\; \Pr[(X,Y) \in A]$$

where A is the half plane $\{(x,y) \mid ax + by \le t\}$. The boundary line $ax + by = t$ lies at a distance $d = \frac{t}{\sqrt{a^2+b^2}}$ from the origin. Therefore, as illustrated in Figure 4, the set A can be rotated into the set

$$T(A) = \Big\{(x,y) \,\Big|\, x \le \frac{t}{\sqrt{a^2+b^2}}\Big\}.$$

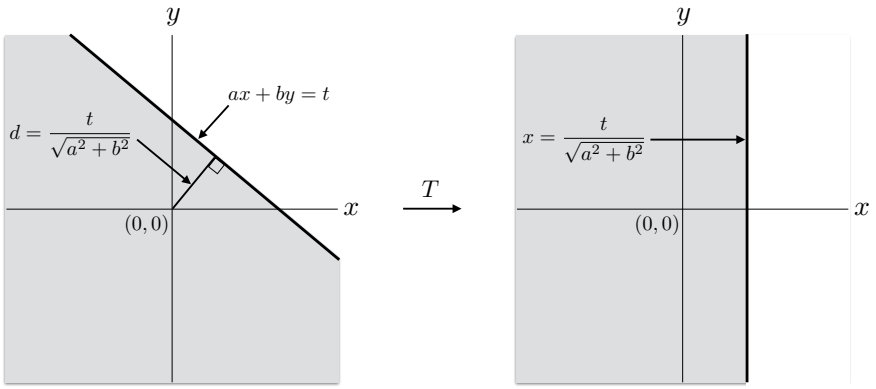


Figure 4: The half plane $ax + by \le t$ is rotated into the half plane $x \le \frac{t}{\sqrt{a^2+b^2}}$.

By (7), this rotation does not change the probability:

$$\Pr[Z \le t] \;=\; \Pr[(X,Y) \in A] \;=\; \Pr[(X,Y) \in T(A)] \;=\; \Pr\Big[X \le \frac{t}{\sqrt{a^2+b^2}}\Big] \;=\; \Pr[\sqrt{a^2+b^2}\,X \le t].$$

Since the equation above holds for all $t \in \mathbb{R}$, we conclude that Z has the same distribution as $\sqrt{a^2+b^2}\,X$. Since X has standard normal distribution, we know by Lemma 20.1 that $\sqrt{a^2+b^2}\,X$ has normal distribution with mean 0 and variance a^2+b^2. Hence we conclude that $Z = aX + bY$ also has normal distribution with mean 0 and variance a^2+b^2. □

The general case now follows easily from Lemma 20.1 and Theorem 20.4.

Corollary 20.1. *Let X and Y be independent normal random variables with parameters (μ_X, σ_X^2) and (μ_Y, σ_Y^2), respectively. Then for any constants $a, b \in \mathbb{R}$, the random variable $Z = aX + bY$ is also normally distributed with mean $\mu = a\mu_X + b\mu_Y$ and variance $\sigma^2 = a^2\sigma_X^2 + b^2\sigma_Y^2$.*

Proof. By Lemma 20.1, $Z_1 = (X - \mu_X)/\sigma_X$ and $Z_2 = (Y - \mu_Y)/\sigma_Y$ are independent standard normal random variables. We can write:

$$Z \;=\; aX + bY \;=\; a(\mu_X + \sigma_X Z_1) + b(\mu_Y + \sigma_Y Z_2) \;=\; (a\mu_X + b\mu_Y) + (a\sigma_X Z_1 + b\sigma_Y Z_2).$$

By Theorem 20.4, $Z' = a\sigma_X Z_1 + b\sigma_Y Z_2$ is normally distributed with mean 0 and variance $\sigma^2 = a^2\sigma_X^2 + b^2\sigma_Y^2$. Since $\mu = a\mu_X + b\mu_Y$ is a constant, by Lemma 20.1 we conclude that $Z = \mu + Z'$ is a normal random variable with mean μ and variance σ^2, as desired. □

The Central Limit Theorem

Recall from Note 18 the Law of Large Numbers for i.i.d. random variables X_i's: it says that the probability of *any* deviation α of the sample average $A_n := \frac{1}{n}\sum_{i=1}^n X_i$ from the mean, however small, tends to zero as the number of observations n in our average tends to infinity. Thus by taking n large enough, we can make the probability of any given deviation as small as we like.

Actually we can say something much stronger than the Law of Large Numbers: namely, the distribution of the sample average A_n, for large enough n, looks like a *normal distribution* with mean μ and variance $\frac{\sigma^2}{n}$. (Of course, we already know that these are the mean and variance of A_n; the point is that the distribution becomes normal!) The fact that the standard deviation decreases with n (specifically, as $\frac{\sigma}{\sqrt{n}}$) means that the distribution approaches a sharp spike at μ.

Recall from the last section that the density of the normal distribution is a symmetrical bell-shaped curve centered around the mean μ. Its height and width are determined by the standard deviation σ as follows: the height at the mean $x = \mu$ is $\frac{1}{\sqrt{2\pi\sigma^2}} \approx \frac{0.4}{\sigma}$; 50% of the mass is contained in the interval of width 0.67σ either side of the mean, and 99.7% in the interval of width 3σ either side of the mean. (Note that, to get the correct scale, deviations are on the order of σ rather than σ^2.)

To state the Central Limit Theorem precisely (so that the limiting distribution is a constant rather than something that depends on n), we shift the mean of A_n to 0 and scale it so that its variance is 1, i.e., we replace A_n by

$$A'_n = \frac{(A_n - \mu)\sqrt{n}}{\sigma} = \frac{\sum_{i=1}^n X_i - n\mu}{\sigma\sqrt{n}}.$$

The Central Limit Theorem then says that the distribution of A'_n converges to the *standard normal* distribution.

Theorem 20.5 (Central Limit Theorem). *Let $X_1, X_2, \ldots, X_n$ be i.i.d. random variables with common expectation $\mu = \mathrm{E}(X_i)$ and variance $\sigma^2 = \mathrm{Var}(X_i)$ (both assumed to be $< \infty$). Define $A'_n = \frac{\sum_{i=1}^n X_i - n\mu}{\sigma\sqrt{n}}$. Then as $n \to \infty$, the distribution of A'_n approaches the standard normal distribution in the sense that, for any real α,*

$$\Pr[A'_n \le \alpha] \to \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\alpha} e^{-x^2/2} dx \quad \text{as } n \to \infty.$$

The Central Limit Theorem is a very striking fact. What it says is the following: If we take an average of n observations of any arbitrary r.v. X, then the distribution of that average will be a bell-shaped curve centered at $\mu = \mathrm{E}(X)$. Thus all trace of the distribution of X disappears as n gets large: all distributions, no matter how complex,[4] look like the normal distribution when they are averaged. The only effect of the original distribution is through the variance σ^2, which determines the width of the curve for a given value of n, and hence the rate at which the curve shrinks to a spike.

Optional: Buffon's Needle

Here is a simple yet interesting application of continuous random variables to the analysis of a classical procedure for estimating the value of π known as *Buffon's needle*, after its 18th century inventor Georges-Louis Leclerc, Comte de Buffon.

Here we are given a needle of length ℓ, and a board ruled with horizontal lines at distance ℓ apart. The experiment consists of throwing the needle randomly onto the board and observing whether or not it crosses

[4] We do need to assume that the mean and variance of X are finite.

one of the lines. We shall see below that (assuming a perfectly random throw) the probability of this event is exactly $2/\pi$. This means that, if we perform the experiment many times and record the *proportion* of throws on which the needle crosses a line, then the Law of Large Numbers tells us that we will get a good estimate of the quantity $2/\pi$, and therefore also of π; and we can use Chebyshev's inequality as in the other estimation problems we considered in that same Note to determine how many throws we need in order to achieve specified accuracy and confidence.

To analyze the experiment, let's consider what random variables are in play. Note that the position where the needle lands is completely specified by two random variables: the vertical distance Y between the midpoint of the needle and the closest horizontal line, and the angle Θ between the needle and the vertical. The r.v. Y ranges between 0 and $\ell/2$, while Θ ranges between $-\pi/2$ and $\pi/2$. Since we assume a perfectly random throw, we may assume that their *joint distribution* has density $f(y,\theta)$ that is uniform over the rectangle $[0,\ell/2]\times[-\pi/2,\pi/2]$. Since this rectangle has area $\frac{\pi\ell}{2}$, the density should be

$$f(y,\theta)=\begin{cases}2/\pi\ell & \text{for } (y,\theta)\in[0,\ell/2]\times[-\pi/2,\pi/2];\\ 0 & \text{otherwise.}\end{cases} \tag{8}$$

Equivalently, Y and Θ are independent random variables, each uniformly distributed in their respective range. As a sanity check, let's verify that the integral of this density over all possible values is indeed 1:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(y,\theta)dyd\theta=\int_{-\pi/2}^{\pi/2}\int_{0}^{\ell/2}\frac{2}{\pi\ell}dyd\theta=\int_{-\pi/2}^{\pi/2}\left[\frac{2y}{\pi\ell}\right]_0^{\ell/2}d\theta=\int_{-\pi/2}^{\pi/2}\frac{1}{\pi}d\theta=\left[\frac{\theta}{\pi}\right]_{-\pi/2}^{\pi/2}=1.$$

This is an analog of equation (1) for our joint distribution; rather than the area under the curve $f(x)$, we are now computing the area under the "surface" $f(y,\theta)$.

Now let E denote the event that the needle crosses a line. How can we express this event in terms of the values of Y and Θ? Well, by elementary geometry the vertical distance of the endpoint of the needle from its midpoint is $\frac{\ell}{2}\cos\Theta$, so the needle will cross the line if and only if $Y\le\frac{\ell}{2}\cos\Theta$. Therefore we have

$$\Pr[E]=\Pr[Y\le\tfrac{\ell}{2}\cos\Theta]=\int_{-\pi/2}^{\pi/2}\int_{0}^{(\ell/2)\cos\theta} f(y,\theta)dyd\theta.$$

Substituting the density $f(y,\theta)$ from equation (8) and performing the integration we get

$$\Pr[E]=\int_{-\pi/2}^{\pi/2}\int_{0}^{(\ell/2)\cos\theta}\frac{2}{\pi\ell}dyd\theta=\int_{-\pi/2}^{\pi/2}\left[\frac{2y}{\pi\ell}\right]_0^{(\ell/2)\cos\theta}d\theta=\frac{1}{\pi}\int_{-\pi/2}^{\pi/2}\cos\theta d\theta=\frac{1}{\pi}\left[\sin\theta\right]_{-\pi/2}^{\pi/2}=\frac{2}{\pi}.$$

This is exactly what we claimed at the beginning of the section!

Buffon's Needle: A Slick Approach

Recall that we toss a unit length needle on (an infinite) board ruled with horizontal lines spaced at unit length apart. We wish to calculate the chance that the needle intersects a horizontal line. That is, let I be the event that the needle intersects a line. We will show that $\Pr[I]=\frac{2}{\pi}$. Let X be a random variable defined to be the number of intersections of such a needle:

$$X=\begin{cases}1 & \text{if the needle intersects a line}\\ 0 & \text{otherwise.}\end{cases}$$

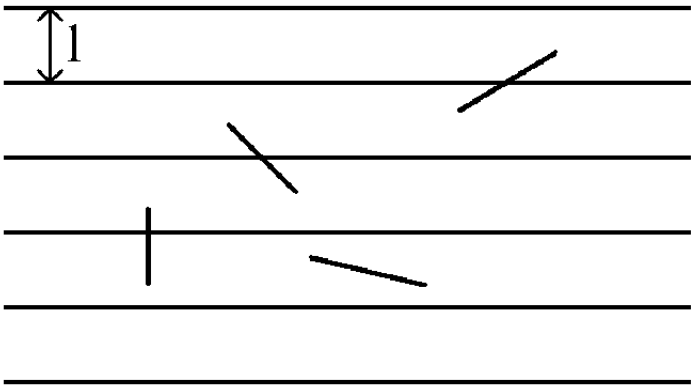


Figure 5: Buffon's Needle.

Since X is an indicator random variable, $\mathrm{E}(X) = \Pr[I]$ (here we are assuming the case in which the needle lies perfectly on two lines cannot happen, since the probability of this particular event is 0). Now, imagine that we were tossing a needle of two unit length, and we let Z be the random variable representing the number of times such a needle intersects horizontal lines on the plane. We can "split" the needle into two parts of unit length and get $Z = X_1 + X_2$, where X_i is 1 if segment i of the needle intersects and 0 otherwise. Thus, $\mathrm{E}(Z) = \mathrm{E}(X_1 + X_2) = \mathrm{E}(X_1) + \mathrm{E}(X_2) = 2\mathrm{E}(X)$, since each segment is of unit length. A similar argument holds if we split a unit length needle into m equal segments, so that $X = X_1 + \cdots + X_m$, where X_i is 1 if segment i (which has length $\frac{1}{m}$) intersects a line and 0 otherwise. We have that $\mathrm{E}(X) = \mathrm{E}(X_1 + \cdots + X_m) = \mathrm{E}(X_1) + \cdots + \mathrm{E}(X_m)$. But each of the $\mathrm{E}(X_i)$'s are equal, so we get $\mathrm{E}(X) = m\mathrm{E}(X_i) \rightarrow \mathrm{E}(X_i) = (\text{length of segment } i) \cdot \mathrm{E}(X)$. Thus, if we drop a needle of length l, and we let Y be the number of intersections, then $\mathrm{E}(Y) = l \cdot \mathrm{E}(X)$. Even if we have a needle which is arbitrarily short, say length ε, we still have $\mathrm{E}(Y) = \varepsilon \cdot \mathrm{E}(X)$.

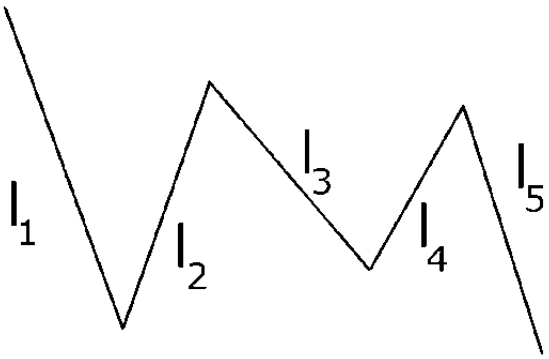


Figure 6: "Noodle" consisting of 5 needles.

Now, consider a "noodle" comprised of two needles of arbitrary lengths l_1 and l_2 (with corresponding random variables X_1 and X_2), where the needles are connected by a rotating joint, we can conclude by linearity of expectation that $\mathrm{E}(X_1 + X_2) = \mathrm{E}(X_1) + \mathrm{E}(X_2)$. In general, we can have a noodle comprised of n needles of arbitrary length: where $\mathrm{E}(X_1 + \cdots + X_n) = \mathrm{E}(X_1) + \cdots + \mathrm{E}(X_n) = l_1\mathrm{E}(X) + \cdots + l_n\mathrm{E}(X)$. Factoring $\mathrm{E}(X)$ out, we get that the expected number of intersections of a noodle is $(\text{length of noodle}) \cdot \mathrm{E}(X)$. In particular, since we are allowed to string together needles at connected rotating joints and since each needle can be

arbitrarily short, we are free to pick any shape, but which one?

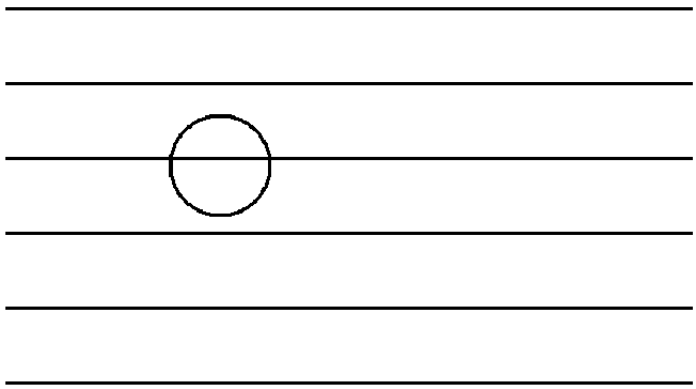


Figure 7: A circle always has two intersections.

Consider a circle with a unit length diameter (and therefore circumference π). Such a shape must always intersect twice: which implies $\mathrm{E}(\text{number of circle intersections}) = \pi \cdot \mathrm{E}(X) = 2$, and thus $\mathrm{E}(X) = \frac{2}{\pi} = \Pr[I]$.

Inference

In this note we revisit the problem of *inference*: Given some data or observations from the world, what can we infer about the underlying process that generates the data? The primary tool that allows us to make such an inference is *Bayes' rule*. We have seen some simple examples of Bayesian inference in Note 14, all of which can be modeled using the balls and bins model. In this note, we will build on this simple model to solve a machine learning application, namely the task of digit classification.

Digit Classification

Handwriting recognition is an important machine learning problem that has many practical applications. Many institutions nowadays regularly use automated systems to recognize handwriting and convert it to digital words or letters, which are easier to process and more efficient than having a human manually look at them one by one. For example, the post office uses handwriting recognition to detect the addresses on mail and sort them to the corresponding destinations, while banks use handwriting recognition to detect the amount of money that are written on checks. These systems have been fine-tuned and achieve accuracy comparable to (or even better than) human performance, which is why they are now so widely used.

To understand how these kinds of systems work, let us consider a simpler problem of digit classification. In this problem, we are given an image of a handwritten digit $(0-9)$ and we have to identify the number written on the image. This is an instance of a *classification* problem: we are given an object (in this case an image) that belongs to one of several classes (the digits), and we have to decide which class to assign the object to. A typical dataset that is used for this problem is the MNIST[1] dataset, where each image is grayscale and has size 28×28 pixels. Figure 1 shows some examples of the digits from MNIST.

Figure 1: Some examples of handwritten digit images from the MNIST dataset.

When we look at the images above, it seems easy for us (humans) to identify the digits. For example, the first image above is clearly a "0" because it looks like a circle, and we know it has to be a "0"; similarly, the second image above is a "1" because it looks like a "1" from our experience, and so on. But in doing so, we are utilizing all our human knowledge and experience with digits. So it is our very good internal understanding of what constitutes a digit that is helping us recognize the digits easily. But if you were a computer, how would you go about approaching this problem?

[1] http://yann.lecun.com/exdb/mnist/

Building a model of the digits

We can imagine that if we have a good understanding of how the digits are generated, then we can solve the digit classification problem easily. For example, suppose for each digit j you have a model $M(j)$ that can generate all possible images corresponding to that digit. Then given an image x, we can identify what digit is written in the image by checking which of the models $M(0), \ldots, M(9)$ can generate x. If we have a perfect model for the digits, then there should be only one model $M(j)$ that can possibly generate the image x; all other models should reject x as being inconsistent with their digit models. In this case, classification is easy: Given an image x, find the unique model $M(j)$ consistent with x, and classify x as having digit j.

Following the argument above, we can try to find a good model for each digit. A reasonable first attempt is to come up with a list of simple rules for writing digits. For example, we can say that "0" looks like a circle, "1" looks like a vertical stroke, "8" looks like two stacked circles, and so on. But we immediately see that these simple rules are not sufficient to capture the wide variations among handwritten digits; see Figure 2 for some more examples from the MNIST dataset.

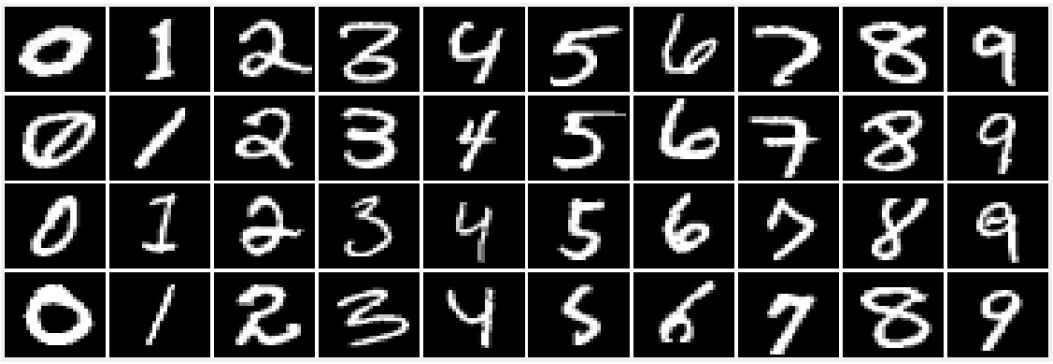


Figure 2: More examples of digit images from the MNIST dataset.

Notice that some of the digits above do not follow standard simple rules that we may have come up with. For example, the second image of "0" has an additional stroke inside, yet we can still recognize it as a "0"; similarly, the last image of "5" is missing the top horizontal stroke, but we know it is still a "5". If we want to be able to recognize these cases, we need to augment our list of rules to take into account all the possible ways to write digits. But even if the same person writes the same digit multiple times, we still see some variations in the result. So this rule-based approach will give us a very large set of rules that becomes unwieldy to use, and we can never be sure that we have captured all the possible variations.

We can try to come up with better models for digits that are more accurate. For example, we can consider modeling the curvature of the digits (how the angles between adjacent points in the digit change), or even start modeling the physical process of writing itself. But trying to capture the underlying variations in a realistic way requires a lot of sophistication, and at some point that task becomes more difficult than the original problem of digit classification that we started with.

So can we get away with a simpler model? The answer is yes: We can indeed use very simple, crude models for the digits. These simple models are themselves not too realistic, but in conjunction with Bayes' rule and the formalism of inference, we can still use them to get a good performance on our digit classification problem.

Probabilistic approach

Recall that the difficulty in the problem is that we don't have a good concise definition that separates one digit from another. Indeed, there are many variations in writing a digit, as we saw in Figure 2. Our approach now is to take a *probabilistic view* of the problem: We treat the variations in the digits as a probability distribution over the images. So for each digit $j \in \{0, 1, \ldots, 9\}$, we have a probability distribution $P_j(x)$ over the images x, which represents our uncertainty or imperfect knowledge about the variations among the images containing digit j.

Under distribution $P_j(x)$, we are a lot more likely to generate images x that look like how we would write digit j; however, there is also a small probability that $P_j(x)$ generates images that are supposed to contain digit j, but by random chance may look like other digits. Figure 3 shows an illustration of the distribution $P_1(x)$. The bulk of the mass (images that are assigned high probability) are good images of digit 1, while on the tails (images with small probability) are some ambiguous images that could be interpreted as digit 1, or possibly other digits.

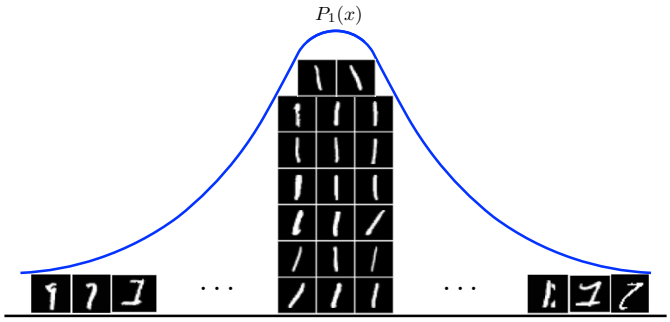


Figure 3: Illustration of the distribution $P_1(x)$ that captures our uncertainty of the variations among digit 1.

The distributions P_j are imperfect models of the digits. Nevertheless, since they are concentrated around good images of each digit, they are still a useful representation of our knowledge. At this point it is still not clear where we get the distributions P_j from, but for the moment let us assume we have them.

With this setup, we can phrase the digit classification problem in terms of balls and bins as follows: We have 10 bins, each one corresponds to a digit. Within each bin we have all possible balls (images) corresponding to that bin (digit). We assume the images within bin j are distributed according to the probability distribution P_j, which means the images in bin j are more likely to have a digit j written in them. Then we are given an image which was taken from one of the bins (the image has a digit written in it), but we don't know which one, and our task is to guess which bin the image was likely to have come from (which digit the image corresponds to).

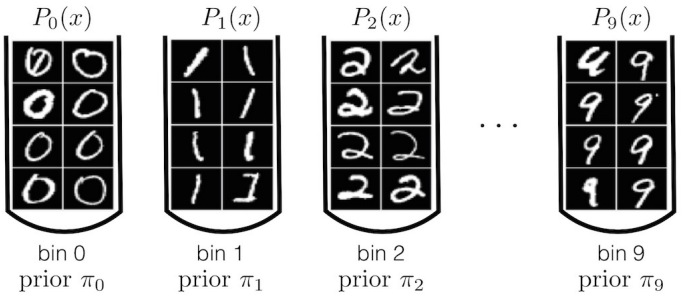


Figure 4: Balls and bins analogy of digit classification. Balls are the images, and bins are the digits.

Classification as inference

Let us assume we have a prior distribution over the bins. If y denotes the bin label, then we write the prior probability distribution as

$$P(y = j) = \pi_j, \qquad j \in \{0, 1, \ldots, 9\}.$$

The prior distribution $(\pi_0, \ldots, \pi_9)$ represents our initial belief on the distribution of the digits before we see any images. For example, we may believe that all 10 digits are equally likely to appear, in which case $\pi_0 = \pi_1 = \cdots = \pi_9 = \frac{1}{10}$. Or maybe for some reason we believe that the digit 1 appears more often than any other digits, say for example $\pi_1 = 0.3$, $\pi_2 = 0.15$, and so on.[2] But for the moment let us assume we have some prior distribution.

As noted before, we assume that within each bin j, the images are distributed according to P_j. Thus, $P_j(x)$ is the conditional probability of generating image x given that the digit is j:

$$P(x \mid y = j) = P_j(x).$$

Now suppose we are given an image x. How do we decide which bin the image came from? Just like in the original balls and bins problem, we apply Bayes' rule to compute the posterior probability of each bin after having observed x:

$$P(y = j \mid x) \;=\; \frac{P(y = j) \cdot P(x \mid y = j)}{P(x)} \;=\; \frac{\pi_j P_j(x)}{\sum_{i=0}^{9} \pi_i P_i(x)}. \tag{1}$$

In the second equality above we have used the total probability rule:

$$\begin{aligned} P(x) &= P(y = 0) \cdot P(x \mid y = 0) + P(y = 1) \cdot P(x \mid y = 1) + \cdots + P(y = 9) \cdot P(x \mid y = 9) \\ &= \pi_0 P_0(x) + \pi_1 P_1(x) + \cdots + \pi_9 P_9(x). \end{aligned}$$

Now that we have the posterior distribution, how should we decide which digit to classify x to? The answer is simple: Choose the digit that maximizes the posterior probability! That is, classify x to the digit:

$$h(x) = \arg\max_j \; P(y = j \mid x) \tag{2}$$

(here "$\arg\max_j$" means we are taking the argument j that maximizes the expression $P(y = j \mid x)$).

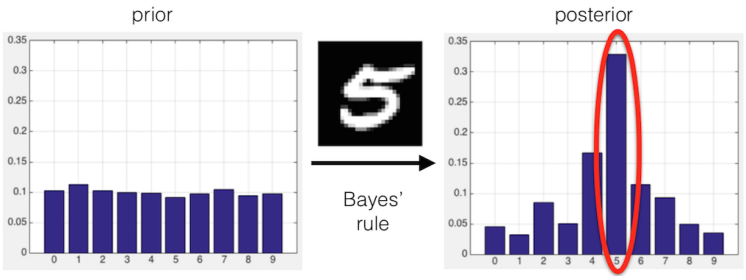


Figure 5: Updating belief by Bayes' rule after seeing image x. We classify x to the bin with highest posterior.

Recall that the posterior distribution (1) represents our updated belief about the digits after seeing the image x. So the rule (2) is doing a simple and reasonable thing: If after seeing x we believe that digit j is more likely than any other digits, then we should use that digit j as our guess to what number is written in x.

[2] There is an interesting justification of this called Benford's law, which states that the number 1 appears as the first significant digit more often than any other numbers. This is closely related to Zipf's law, which we will discuss in the next lecture.

Notice from the Bayes' rule calculation (1) that the denominator $P(x)$ does not depend on j. Therefore, we can omit $P(j)$ from the maximization problem over j in (2), so we can equivalently write:

$$h(x) = \arg\max_j \ \pi_j P_j(x). \tag{3}$$

The estimator in (2) and (3) is called the *Bayes optimal* estimator. It is also known as the *maximum a posteriori (MAP)* estimator, i.e., the one that maximizes the posterior distribution.

Estimating the distributions

We have reduced the problem of classification into inference: To classify an image x, we only need to calculate the prior π_j and the conditional probability $P_j(x)$ for all digits $j \in \{0,1,\ldots,9\}$, and find the one that maximizes their product (3). So if we know the prior distribution π_j and all the conditional distributions P_j, then we are done. But where do they come from?

One way to obtain these distributions is to learn them from the data. Typically, in a machine learning problem we have access to some training data that we can use to train our model, and some test data to test the performance of our model. The MNIST dataset that we are using provides 60000 training examples (where each example consists of a pair (x,y) of an image and the corresponding digit) and 10000 test images.

Estimating the prior. Recall that the prior probability $\pi_j = P(y = j)$ is our initial belief about the distribution of the digits. Suppose we are given n training examples ($n = 60000$ in our case). We divide them into 10 parts based on the digits in the images. Then we estimate the prior probability π_j by:

$$\hat{\pi}_j = \frac{n_j}{n} \tag{4}$$

where n_j is the number of examples having digit j. Intuitively, we are thinking of the training examples as random samples taken from the underlying distribution that generates the digits and the images. If the probability of getting digit j is π_j, then among n samples we expect to get $n_j = n\pi_j$ images with digit j, so our estimate for π_j should be $\hat{\pi}_j = \frac{n_j}{n}$. Notice that $n_0 + n_1 + \cdots + n_9 = n$, so $\hat{\pi}_0 + \hat{\pi}_1 + \cdots + \hat{\pi}_9 = 1$, which means our estimate $(\hat{\pi}_0, \hat{\pi}_1, \ldots, \hat{\pi}_9)$ forms a valid probability distribution.

The prior estimate from the MNIST dataset is as follows (these are displayed on the left plot in Figure 5):

j	0	1	2	3	4	5	6	7	8	9
$\hat{\pi}_j$ (%)	9.87	11.24	9.93	10.22	9.74	9.03	9.86	10.44	9.75	9.92

Estimating the conditional. We see that estimating the prior is easy. But estimating the conditional distribution $P_j(x) = P(x \mid y = j)$ is more difficult. In the MNIST dataset, each image x is grayscale (each pixel value is between 0 and 1) and has size 28×28 pixels. So each image x is a point in the $(28 \times 28 =)$ 784-dimensional space, $x \in [0,1]^{784}$. What kind of distribution should we use to represent $P_j(x)$? In particular, how should we try to model the distribution of the images of digit j?

The important idea is: We don't have to model the distribution of the digits accurately. It turns out that we can use a very simple model that is perhaps unrealistic, but easy to learn. Then with the help of Bayes' rule and the formalism of inference, we can still use this simple model to obtain a good performance in our digit classification task. In the rest of this note, we will see two simple models:

1. **Naive Bayes**: Assume $P_j(x)$ is a product of independent coin flips. This means we assume each pixel value is independent of all other pixels.
2. **Gaussian model**: Assume $P_j(x)$ is a multivariate Gaussian distribution. This allows us to model the correlation between pixel values, but still keeping the model simple and easy to learn.

Naive Bayes

Let us assume for now that we have binary images. That is, we convert our grayscale image $x \in [0,1]^{784}$ into a binary image $x \in \{0,1\}^{784}$ by thresholding each pixel value (if $x_i \geq \frac{1}{2}$, then assign it to $x_i = 1$, otherwise assign it to $x_i = 0$). Then for each digit $j \in \{0,1,\ldots,9\}$, our conditional distribution $P_j(x)$ is a probability distribution over the discrete space $\{0,1\}^{784}$, which has 2^{784} elements. So a general distribution over $\{0,1\}^{784}$ requires specifying 2^{784} numbers that sum up to 1, and that is a very large number of parameters.

In the Naive Bayes model, we make an assumption that greatly simplifies the number of parameters: We assume that within each digit, the individual pixel values are independent $\{0,1\}$-random variables. This means the probability distribution P_j over $x = (x_1, x_2, \ldots, x_{784}) \in \{0,1\}^{784}$ factorizes into a product:

$$P_j(x) = P_{j1}(x_1) \cdot P_{j2}(x_2) \cdots P_{j,784}(x_{784}). \tag{5}$$

Each individual distribution $P_{ji}(x_i)$ is a probability distribution over the value of the i-th pixel $x_i \in \{0,1\}$. We can think of each P_{ji} as a biased coin flip with bias p_{ji}:

$$P_{ji}(x_i = 1) = p_{ji}, \qquad P_{ji}(x_i = 0) = 1 - p_{ji},$$

which we can more concisely write as:

$$P_{ji}(x_i) = p_{ji}^{x_i}(1 - p_{ji})^{1-x_i}, \quad x_i \in \{0,1\}. \tag{6}$$

So now the distribution $P_j(x)$ in (5) only has 784 parameters $(p_{j1}, p_{j2}, \ldots, p_{j,784})$. Moreover, these parameters are easy to learn: Estimating each p_{ji} is the same problem as estimating the bias of a coin, which we know how to do.

Specifically, fix a digit $j \in \{0,1,\ldots,9\}$ and a pixel $i \in \{1,2,\ldots,784\}$. Out of n training examples, let n_j be the number of examples with digit j. Among these n_j examples with digit j, let n_{ji} be the number of examples having $x_i = 1$. Then we estimate p_{ji} by

$$\hat{p}_{ji} = \frac{n_{ji}}{n_j}. \tag{7}$$

Recall the intuition: We are thinking of the n_j examples as random samples drawn from the underlying (conditional) distribution of images with digit j. If the probability of pixel i being on ($x_i = 1$) is equal to p_{ji}, then among these n_j samples we expect to see $n_{ji} = n_j p_{ji}$ images with $x_i = 1$. Therefore, our estimate for p_{ji} should be $\hat{p}_{ji} = \frac{n_{ji}}{n_j}$.

Remark: The estimator (7) is problematic if $n_{ji} = 0$ (we always see $x_i = 0$, in which case $\hat{p}_{ji} = 0$) or $n_{ji} = n_j$ (we always see $x_i = 1$, in which case $\hat{p}_{ji} = 1$). This is because we want to use the estimator $\hat{p}_{ji}$ to make predictions for a future image x. If in our training examples we always see $x_i = 1$ ($n_{ji} = n_j$), then by taking $\hat{p}_{ji} = 1$ we are saying that all future values of x_i will always be equal to 1. However, it can also be the case that the n_j examples that we see by random chance happen to be all $x_i = 1$, even if the true value of p_{ji} is < 1, in which case it is possible for the future values of x_i to be 0. So to hedge our bet, in practice we usually want to use what is called *Laplace smoothing*:

$$\tilde{p}_{ji} = \frac{n_{ji} + 1}{n_j + 2}. \tag{8}$$

This has the effect that even if $n_{ji} = 0$ or $n_{ji} = n_j$, the estimator $\tilde{p}_{ji}$ will be strictly > 0 and < 1. An interpretation of the Laplace smoothing (8) is that it is as if we have two additional examples with digit j: one example with $x_i = 0$, and one example with $x_i = 1$.

Naive Bayes classifier. With the ingredients above, we can summarize the Naive Bayes classifier as follows. The data space is the set of all possible binary images, $\mathscr{X} = \{0,1\}^{784}$. The label space is the set of all possible digits, $\mathscr{Y} = \{0,1,\ldots,9\}$. From the training data, we estimate:

- Prior probability $(\pi_0, \pi_1, \ldots, \pi_9)$ following (4).
- Conditional probability p_{ji} for all digits $0 \le j \le 9$ and pixels $1 \le i \le 784$, following (7) or (8).

Given an image $x = (x_1, \ldots, x_{784}) \in \{0,1\}^{784}$, we can compute its conditional likelihood given digit j as:

$$P_j(x) \;=\; \prod_{i=1}^{784} P_{ji}(x_i) \;=\; \prod_{i=1}^{784} p_{ji}^{x_i}(1-p_{ji})^{1-x_i}.$$

In the first equality above we use the Naive Bayes assumption (5), and in the second equality we use (6). Then we use our inference framework and classify image x to the digit given by the MAP estimator (3):

$$h(x) \;=\; \arg\max_j \; \pi_j P_j(x) \;=\; \arg\max_j \; \pi_j \prod_{i=1}^{784} p_{ji}^{x_i}(1-p_{ji})^{1-x_i}.$$

But notice that in the calculation above we are multiplying many small numbers, so the resulting product will be very small, and may be rounded down to 0 if the value is smaller than the limit of precision of the computer; this is called underflow. To avoid underflow, we can maximize the logarithm instead:[3]

$$h(x) \;=\; \arg\max_j \; \log \pi_j + \sum_{i=1}^{784} \Big(x_i \log p_{ji} + (1-x_i)\log(1-p_{ji}) \Big).$$

This is the entire Naive Bayes classifier, a very simple algorithm. Let's now see how well it works.

Naive Bayes on MNIST. When we apply the Naive Bayes classifier to the MNIST dataset, we obtain 15.4% error rate on 10000 test data (i.e., we misclassified 1540 images). This is already a pretty good performance, because there are 10 classes, so if we were to guess randomly, then we would expect an error rate of 90%. Figure 6 shows some examples of the images that are misclassified.

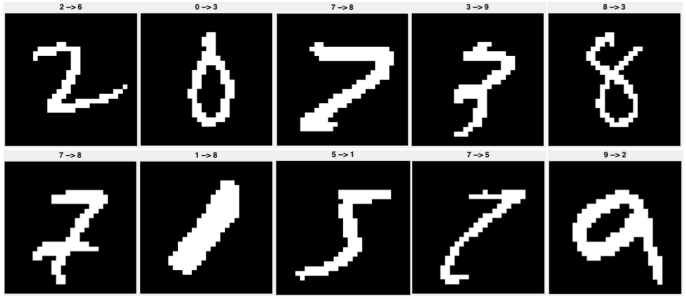


Figure 6: Examples of test images that are misclassified by Naive Bayes.

Let us examine the Naive Bayes model more closely. Figure 7 shows the mean vectors for each digit, namely, the estimated values of p_{ji} (so for each digit j, we render the estimated values $(p_{j1}, p_{j2}, \ldots, p_{j,784})$ as a grayscale 28×28 image). We can also generate samples from the Naive Bayes model we have trained. That is, for each digit j, we can generate a sample $x = (x_1, \ldots, x_{784}) \in \{0,1\}^{784}$ from the conditional distribution P_j by sampling each pixel x_i independently from the distribution P_{ji} (i.e., x_i is a biased coin flip with bias

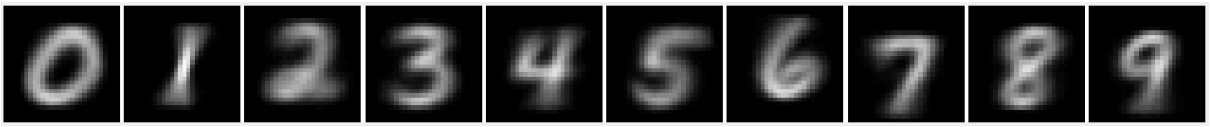


Figure 7: Mean vectors for each digit (the estimated p_{ji}'s).

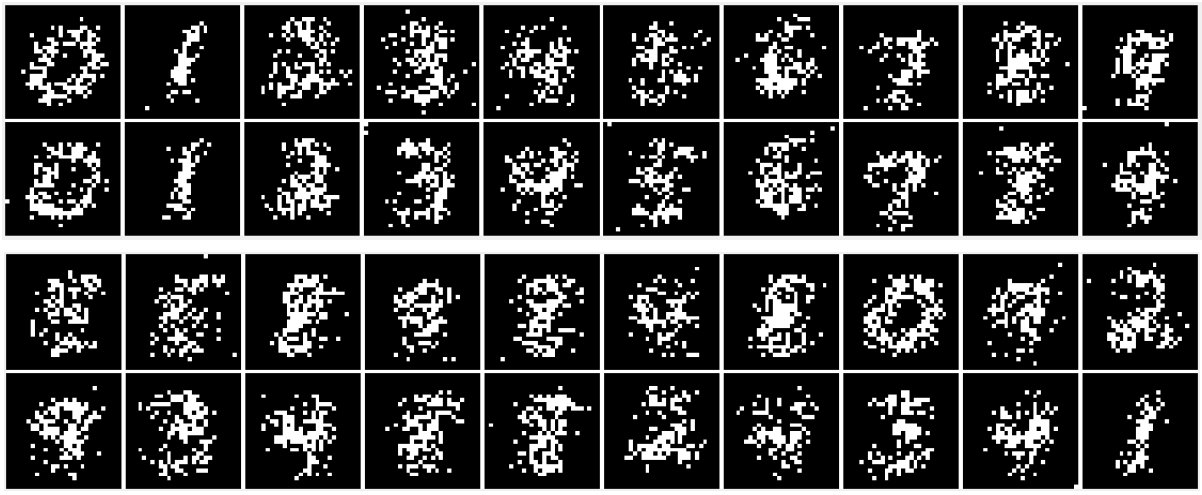


Figure 8: Some samples generated from the trained Naive Bayes model.

p_{ji}). Figure 8 show some samples generated from the model. The first two rows are ordered based on the digits, while the last two rows are arranged in an arbitrary order. Can you still recognize the digits?

From Figure 8 we see that the samples generated from the Naive Bayes model don't look like the training data that we started with (Figure 1 and Figure 2). Some of the samples still look like digits, but others are not recognizable. This shows that the Naive Bayes model does not really capture the true characteristics of the digits. In particular, the independence assumption is clearly dubious, because from the training images we see that the black and white pixels tend to appear in groups, so if a pixel is surrounded by many white pixels then it is also more likely to be white, and vice versa. Yet with this very simple, unrealistic model we are able to achieve a pretty good performance.

We can now try to improve upon this model and try to build a more accurate representation of the digits, thereby obtaining better performance, but at the cost of a more complex model and more expensive learning procedure. This is a general trade-off between model accuracy and complexity that we have to make in many problems.

Gaussian model[4]

The next model we will look at is the Gaussian model. Here we are treating the grayscale image $x \in [0,1]^{784}$ as a vector in the 784-dimensional real space, $x \in \mathbb{R}^{784}$, and we are assuming that within each digit, the images are distributed according to a multivariate Gaussian distribution. This has the advantage that we are also modeling the correlation between different pixel values, while still keeping the model relatively simple and easy to learn.

[3]Note that $z \mapsto \log z$ is a monotonically increasing function, so maximizing an expression $f(j)$ is equivalent to maximizing its logarithm $\log f(j)$. The maximum value will be different, but the argument that achieves the maximum will be the same.

[4]This section requires linear algebra beyond the scope of this class, so this section is optional and only for your own interest.

A brief overview of multivariate Gaussian

What do multivariate Gaussian distributions look like? Recall that the univariate (one-dimensional) Gaussian distribution $N(\mu,\sigma^2)$ is specified by a mean value $\mu \in \mathbb{R}$ and variance $\sigma^2 > 0$. The density of the univariate Gaussian distribution $N(\mu,\sigma^2)$ is given by:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \qquad x \in \mathbb{R}. \tag{9}$$

Figure 9 shows the plot of the density function $p(x)$. Notice the characteristic bell-shaped curve centered around the mean μ and rapidly decreasing as x moves away from μ. This is because the density function $p(x)$ is an exponential function of a negative quadratic function of x, as we can see from (9) above.

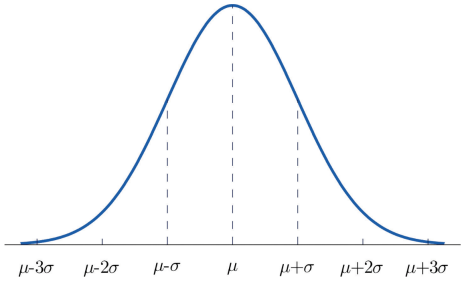


Figure 9: Plot of density function of the univariate Gaussian distribution $N(\mu,\sigma^2)$.

Bivariate Gaussian: same variance. We now consider bivariate (two-dimensional) Gaussian. We start with the simplest case: Let's take two independent univariate Gaussian random variables $X_1 \sim N(\mu_1,\sigma^2)$ and $X_2 \sim N(\mu_2,\sigma^2)$, so they have mean values $\mathrm{E}(X_1) = \mu_1$ and $\mathrm{E}(X_2) = \mu_2$, and the same variance $\mathrm{Var}(X_1) = \mathrm{Var}(X_2) = \sigma^2$. We combine X_1 and X_2 into a column vector X, and μ_1 and μ_2 into a column vector μ:

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \qquad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}.$$

Now we have a two-dimensional random variable $X \in \mathbb{R}^2$, which has a bivariate Gaussian distribution with mean vector $\mathrm{E}(X) = \mu$. The density function of X is the joint density of X_1 and X_2:

$$p(x) \;=\; p(x_1,x_2) \;=\; p_1(x_1)\cdot p_2(x_2)$$

where in the last equality we are using the property that the joint density of independent random variables is the product of their densities. Now plugging in the univariate Gaussian density for $X_1 \sim N(\mu_1,\sigma^2)$ and $X_2 \sim N(\mu_2,\sigma^2)$, we get that the density function of X is:

$$\begin{aligned} p(x) \;&=\; \left\{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_1-\mu_1)^2}{2\sigma^2}\right)\right\} \cdot \left\{\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_2-\mu_2)^2}{2\sigma^2}\right)\right\} \\ &=\; \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right) \end{aligned} \tag{10}$$

where we have used the notation $\|x-\mu\|$ to denote the norm of the vector $x-\mu \in \mathbb{R}^2$:

$$\|x-\mu\| \;=\; \sqrt{(x_1-\mu_1)^2 + (x_2-\mu_2)^2}.$$

From (10), we see that the density $p(x)$ only depends on the distance from the point x to the mean vector μ, so the density function $p(x)$ is rotationally symmetric around the mean μ.[5] Figure 10 shows the contour plot of the density function (10). Each green circle in Figure 10 has the same density value (but the density decreases as the circle gets larger), so this special case is also called the *spherical Gaussian distribution*.

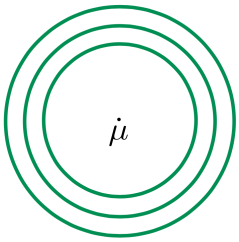


Figure 10: Contour plot of density function $p(x)$ of the spherical two-dimensional Gaussian distribution (10).

Bivariate Gaussian: different variances. Consider two independent univariate Gaussian random variables $X_1 \sim N(\mu_1, \sigma_1^2)$ and $X_2 \sim N(\mu_2, \sigma_2^2)$, so now they have different means $\mathrm{E}(X_1) = \mu_1$, $\mathrm{E}(X_2) = \mu_2$, and different variances $\mathrm{Var}(X_1) = \sigma_1^2$, $\mathrm{Var}(X_2) = \sigma_2^2$. As before, we collect X_1 and X_2 into a vector X; μ_1 and μ_2 into a vector μ; and now we also collect σ_1^2 and σ_2^2 into a matrix Σ:

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \qquad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}.$$

Notice that since X_1 and X_2 are independent, they have zero covariance:

$$\mathrm{Cov}(X_1, X_2) = \mathrm{E}((X_1 - \mu_1)(X_2 - \mu_2)) = 0.$$

So the off-diagonal entries of the matrix Σ, which are equal to 0, are actually the covariance of X_1 and X_2. Then the vector X has a two-dimensional Gaussian distribution with mean vector $\mathrm{E}(X) = \mu$ and diagonal covariance matrix Σ, namely:

$$\mathrm{Cov}(X) \;=\; \begin{pmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) \end{pmatrix} \;=\; \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix} \;=\; \Sigma.$$

The density of X is given by the joint density of (X_1, X_2), which is the product of their individual densities:

$$\begin{aligned} p(x) \;&=\; \left\{ \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left(-\frac{(x_1 - \mu_1)^2}{2\sigma_1^2} \right) \right\} \cdot \left\{ \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp\left(-\frac{(x_2 - \mu_2)^2}{2\sigma_2^2} \right) \right\} \\ &=\; \frac{1}{2\pi\,\sigma_1\,\sigma_2} \exp\left(-\frac{(x_1 - \mu_1)^2}{2\sigma_1^2} - \frac{(x_2 - \mu_2)^2}{2\sigma_2^2} \right) \\ &=\; \frac{1}{2\pi\,|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1} (x - \mu) \right) \end{aligned} \tag{11}$$

In the last equality above we have used the notation $|\Sigma|$ for the determinant of the matrix Σ:

$$|\Sigma| = \det(\Sigma) = \det \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma^2 \end{pmatrix} = \sigma_1^2 \cdot \sigma_2^2.$$

[5] We have seen this property in Note 20 when proving that the sum of two independent normal random variables is also normal.

The expression inside the exponential in (11) is a quadratic function in $x \in \mathbb{R}^2$ involving the inverse covariance matrix Σ^{-1} (here $(x-\mu)^\top$ is a row vector that is the transpose of $(x-\mu)$):

$$(x-\mu)^\top \Sigma^{-1}(x-\mu) \;=\; \begin{pmatrix} x_1-\mu_1 & x_2-\mu_2 \end{pmatrix} \begin{pmatrix} \frac{1}{\sigma_1^2} & 0 \\ 0 & \frac{1}{\sigma_2^2} \end{pmatrix} \begin{pmatrix} x_1-\mu_1 \\ x_2-\mu_2 \end{pmatrix} \;=\; \frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2}.$$

Figure 11 shows the contour plot of the density function (11), where now each contour is not a circle anymore, but an ellipse. We can think of the ellipses as being obtained by stretching the circles in Figure 10 along the axes, where the amount of stretching is proportional to the variances.

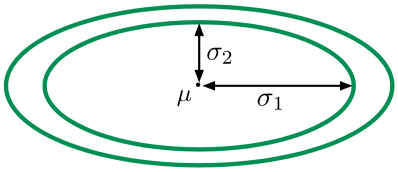


Figure 11: Contour plot of density function $p(x)$ of the diagonal two-dimensional Gaussian distribution (11).

Bivariate Gaussian: general. The general bivariate Gaussian distribution $N(\mu, \Sigma)$ is specified by a mean vector $\mu \in \mathbb{R}^2$ and a 2×2 covariance matrix Σ. The density function is given by the same expression in (11), but now with a general covariance matrix:

$$p(x) \;=\; \frac{1}{2\pi\,|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right). \tag{12}$$

It can be shown that the covariance matrix Σ can be decomposed into perpendicular directions $u_1, u_2 \in \mathbb{R}^2$, such that Σ only acts by scaling along each direction, namely, $\Sigma u_1 = \sigma_1^2 u_1$ and $\Sigma u_2 = \sigma_2^2 u_2$. Thus, having a general covariance Σ is essentially the same as having a diagonal covariance matrix as in (11), except that the axes are now rotated. Figure 12 shows the contour plot of the density function (12). We see that the contours are still ellipses, but oriented along the axes u_1 and u_2.

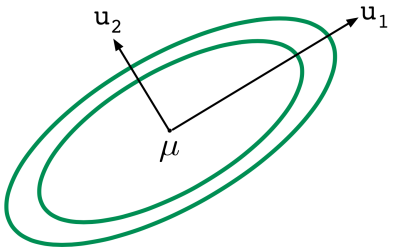


Figure 12: Contour plot of density function $p(x)$ of the general two-dimensional Gaussian distribution (12).

Multivariate Gaussian. The general multivariate Gaussian in $\mathbb{R}^d$ is just the high-dimensional analog of the bivariate case above. The Gaussian distribution $N(\mu, \Sigma)$ in $\mathbb{R}^d$ is specified by a mean vector $\mu \in \mathbb{R}^d$ and a $d \times d$ covariance matrix Σ. The density function is given by:

$$p(x) \;=\; \frac{1}{(2\pi)^{d/2}\,|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right). \tag{13}$$

If $X = (X_1, \ldots, X_d)$ is a random variable with Gaussian $N(\mu, \Sigma)$ distribution, then the expected value of X is equal to μ, and its covariance matrix is equal to Σ:

$$\mathrm{E}(X) = \mu, \qquad \mathrm{Cov}(X) = \mathrm{E}((X-\mu)(X-\mu)^\top) = \Sigma. \tag{14}$$

Explicitly, each component X_i has expected value $\mathrm{E}(X_i) = \mu_i$, and the covariance between components X_i and X_j is given by the (i,j)-th entry of the matrix Σ:

$$\mathrm{Cov}(X_i, X_j) = \mathrm{E}((X_i - \mu_i)(X_j - \mu_j)) = \Sigma_{ij}.$$

In particular, the diagonal entries of Σ are the variances of the components: $\mathrm{Var}(X_i) = \Sigma_{ii}$.

Estimating Gaussian model

Let us now go back to our digit classification problem. Recall that we want to model the conditional distribution of each digit as a multivariate Gaussian distribution. Our images are $x \in \mathbb{R}^{784}$, so for each digit $j \in \{0, 1, \ldots, 9\}$, we assume the conditional distribution $P_j(x)$ is a 784-dimensional Gaussian distribution $N(\mu_j, \Sigma_j)$ parameterized by a mean vector $\mu_j \in \mathbb{R}^{784}$ and a 784×784 covariance matrix Σ_j (note that here the subscript j indicates the digit class, not the component of the mean vector or the covariance matrix). How do we estimate these parameters?

Since μ_j is the expected value (14) of the distribution $N(\mu_j, \Sigma_j)$, we can estimate it by the sample mean. Specifically, recall that out of n training examples, we have n_j images with digit j. Let $x^{(1)}, \ldots, x^{(n_j)} \in \mathbb{R}^{784}$ denote the images with digit j. Then we estimate μ_j via:

$$\hat{\mu}_j \;=\; \frac{1}{n_j} \sum_{i=1}^{n_j} x^{(i)}. \tag{15}$$

Similarly, since Σ_j is the covariance matrix (14), which is the expected value of the matrix $(X - \mu_j)(X - \mu_j)^\top$, we can estimate Σ_j via the sample covariance matrix:

$$\hat{\Sigma}_j \;=\; \frac{1}{n_j} \sum_{i=1}^{n_j} \left(x^{(i)} - \hat{\mu}_j\right)\left(x^{(i)} - \hat{\mu}_j\right)^\top. \tag{16}$$

Note that here we are using the estimated mean vector $\hat{\mu}_j$ in place of the true mean μ_j, which is unknown.

Remark: Notice from (13) that the Gaussian density function involves the inverse covariance matrix. The true covariance matrix Σ_j is always invertible, but our estimator $\hat{\Sigma}_j$ may not be. So in practice, we usually want to do a *regularization*:

$$\tilde{\Sigma}_j \;=\; \hat{\Sigma}_j + \sigma^2 I \tag{17}$$

where $\sigma^2 > 0$ is a constant and I is the identity matrix. Adding a multiple of the identity matrix has the effect of pushing the singular values of $\hat{\Sigma}_j$ in the positive direction, so the resulting matrix $\tilde{\Sigma}_j$ is guaranteed to be invertible. Regularization also has the nice property that it makes the estimator more stable, and typically achieves a better performance (by choosing the value of σ^2 appropriately[6]).

Gaussian model classifier

We now have all the necessary ingredients to define a Gaussian model classifier. The data space is the set of all possible images, $\mathscr{X} = \mathbb{R}^{784}$. The label space is the set of all possible digits, $\mathscr{Y} = \{0, 1, \ldots, 9\}$. From the training data, we estimate:

[6] In our implementation, we use the value $\sigma^2 = 0.1$.

- Prior probability $(\pi_0, \pi_1, \ldots, \pi_9)$ following (4).
- Mean vector μ_j following (15), and covariance matrix Σ_j following (16) or (17), for all $0 \le j \le 9$.

Then given an image $x \in \mathbb{R}^{784}$, we classify it to the digit that is given by the MAP estimator:

$$\begin{aligned} h(x) &= \arg\max_j \ \pi_j P_j(x) \\ &= \arg\max_j \ \log \pi_j + \log P_j(x) \\ &= \arg\max_j \ \log \pi_j - \frac{1}{2}(x-\mu_j)^\top \Sigma_j^{-1}(x-\mu_j) - \frac{1}{2}\log|\Sigma_j| - \frac{784}{2}\log(2\pi) \\ &= \arg\max_j \ \log \pi_j - \frac{1}{2}(x-\mu_j)^\top \Sigma_j^{-1}(x-\mu_j) - \frac{1}{2}\log|\Sigma_j|. \end{aligned}$$

In the second equality above we are switching to maximizing the logarithm of the expression, which is equivalent because logarithm is an increasing function; in the third equality we are substituting the Gaussian density (13) for $P_j(x)$; and in the last equality we omit the last constant term, which does not affect the $\arg\max_j$.

Gaussian model on MNIST. After fitting the model from the MNIST training examples, the Gaussian model classifier achieves 4.58% error rate on 10000 test data (i.e., we misclassified 458 images). This is a big improvement from the Naive Bayes classifier (which has 15.4% error rate). Figure 13 shows some examples of the images that are misclassified, which tend to be more ambiguous.

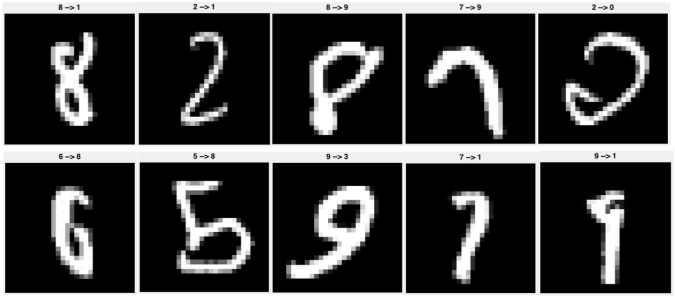


Figure 13: Examples of test images that are misclassified by the Gaussian model.

Let us examine the Gaussian model more closely. Figure 14 shows the mean vectors for each digit, namely, the estimated values of $\mu_j \in \mathbb{R}^{784}$, displayed as 28×28 images. Notice that the μ_j in Figure 14 look identical to the mean vectors p_{ji} from the Naive Bayes model in Figure 7, because in both cases we obtain the mean vectors simply by averaging the training examples (except that in the Naive Bayes model we are averaging the binarized images).

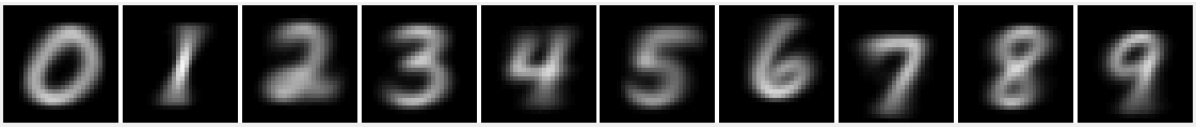


Figure 14: Mean vectors for each digit (the estimated μ_j's).

In the Gaussian model we also have information about the correlations between pixel values, unlike in Naive Bayes where we assume all pixels are independent. Figure 15 shows the estimated covariance matrix Σ_j for

digit $j = 0$ (left) and digit $j = 9$ (right). Both matrices look visually similar, with Σ_0 having stronger values on the corners than Σ_9. But it is difficult to understand what these covariance matrices mean, and whether they make sense for our problem. This is a general issue in complex problems: Once we are working with high-dimensional spaces, it can be hard to have a full understanding of what the model does, because our intuition does not guide us very well in high dimension.

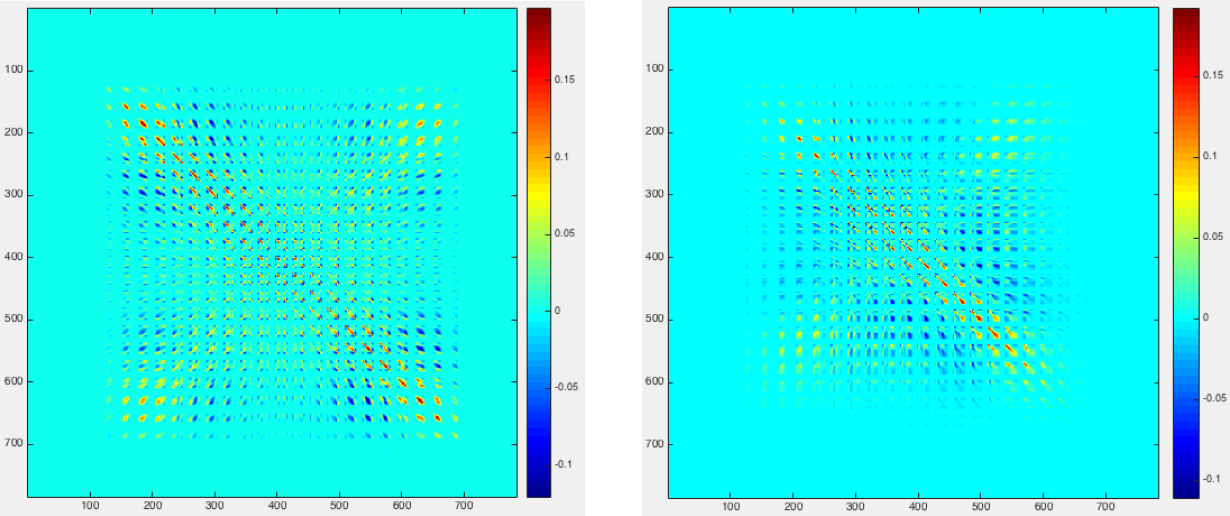


Figure 15: Estimated covariance matrix Σ_j for digit $j = 0$ (left) and $j = 9$ (right)

Finally, we can also generate samples from the Gaussian model we have trained, as shown in Figure 16. The first two rows are ordered based on the digits, while the last two rows are arranged in an arbitrary order. Can you still recognize the digits?

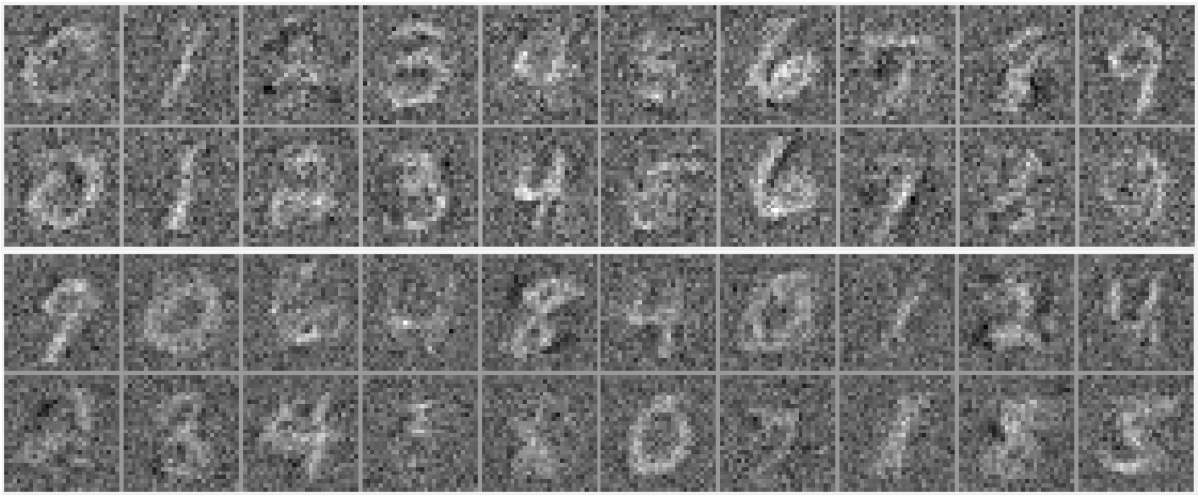


Figure 16: Some samples generated from the trained Gaussian model.

We see above that the samples are very noisy and grainy, but we can still recognize the faint outline of the digits in the images. Note that since we are working with the Gaussian distribution over the entire real space $\mathbb{R}^{784}$, the pixel values in the generated samples can be negative, which does not make sense for real (grayscale) images. Yet we are still able to achieve a good performance in our digit classification task with this model. This again demonstrates that our model does not have to be very accurate, because the formalism of inference with Bayes' rule helps us do the rest of the work.

Zipf's Law and Power Law Distributions

A random graph with n nodes is created by the following process: For each pair of nodes i and j, the edge $\{i, j\}$ is present with probability $p = \frac{d}{n}$, independently of the other edges. So each edge can be modeled as a biased coin flip. From the point of view of a node, the edges adjacent to this node are distributed according to a binomial distribution with mean $(n-1) \cdot \frac{d}{n} \approx d$ and variance $(n-1) \cdot \frac{d}{n} \cdot (1 - \frac{d}{n}) \approx d$ (assuming $n \gg d$).

We know that the binomial distribution (like all distributions we have seen so far) is very concentrated about the mean. For example, if a graph with $n = 20{,}000$ nodes has average degree $d = 4$, then we do not expect any node to have degree 20 or more, since that probability is exponentially small. But in the Internet, the graph of the 20,000 or so autonomous systems which indeed has average degree near 4, it turns out there are about 40 nodes that have degree greater than 20. And in the www (the graph of hyperlinks between $n =$ 10,000,000 documents, if we ignore directions), which is a graph with average degree about 12, more than 100,000 have degree at least 100.

We must conclude that there is some kind of dependence between the edges of these graphs, that they came about by a process that is very different from independent coin tosses we considered above. The degrees of these graphs seem to have a density function that does not decrease exponentially with the difference from the mean.

Zipf's Law

George Kingsley Zipf was a linguist who studied the statistical properties of languages. He noticed that the occurrences of the most frequent words in English (and in other languages) follow a particular pattern. If the most frequent word appears N times in a corpus (say, Shakespeare's plays), then the second most frequent word appears approximately $\frac{N}{2}$ times, and the third most frequent word appears approximately $\frac{N}{3}$ times, and so on. In general, if freq_i denotes the frequency of the i-th most frequent word, then we find that

$$\text{freq}_i \approx \frac{\text{freq}_1}{i}.$$

This phenomenon is quite pervasive, and it is called *Zipf's Law*.

Another place where this law appears is in the population of cities. The largest city of the USA is New York, which has population $\text{pop}_1 \approx 8{,}000{,}000$. The second largest city, Los Angeles, has population $\text{pop}_2 \approx$ 3,800,000, while the third largest city, Chicago, has population $\text{pop}_3 \approx 2{,}700{,}000$, and so on. In general, if pop_i denotes the population of the i-th largest city, then we also find the same relationship:

$$\text{pop}_i \approx \frac{\text{pop}_1}{i}.$$

This means if we plot $\log(\text{pop}_i)$ as a function of $\log(i)$, then we notice that it is almost a straight line with angle 45%; see Figure 1 for an illustration.[1]

[1] Figure is taken from: Xavier Garbaix, *"Zipf's Law for Cities: An Explanation"*, The Quarterly Journal of Economics, 1999.

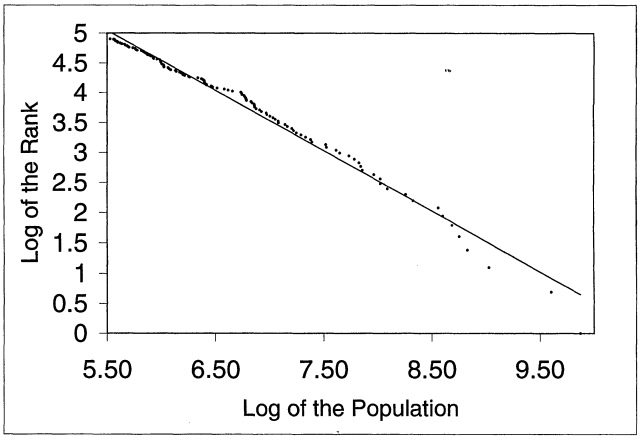


FIGURE I
Log Size versus Log Rank of the 135 largest U. S. Metropolitan Areas in 1991
Source: Statistical Abstract of the United States [1993].

Figure 1: Log-log plot of the population of the largest cities. Straight line indicates power law distribution.

Power Laws

There is a similar phenomenon in the distribution of incomes, first observed by the economist Pareto. He noticed that

$$\Pr(\text{income} \geq x) = \frac{D}{x^{\alpha-1}} \tag{1}$$

for some constant D and power α. This means among a population of N individuals,

$$\frac{\#\text{ of people with income } \geq x}{N} = \frac{D}{x^{\alpha-1}}.$$

In particular, the highest income inc_1 is the value of x such that there is only 1 person with that income:

$$\frac{1}{N} = \frac{D}{\text{inc}_1^{\alpha-1}} \quad \Leftrightarrow \quad \text{inc}_1 = (ND)^{1/(\alpha-1)}.$$

And in general, the i-th highest income inc_i is the value of x such that there are i people with income $\geq x$:

$$\frac{i}{N} = \frac{D}{\text{inc}_1^{\alpha-1}} \quad \Leftrightarrow \quad \text{inc}_i = \left(\frac{ND}{i}\right)^{1/(\alpha-1)}.$$

Substituting the value $\text{inc}_i = (ND)^{1/(\alpha-1)}$ and letting $\beta = \frac{1}{\alpha-1}$ for simplicity, we get the relationship:

$$\text{inc}_i = \frac{\text{inc}_1}{i^\beta}.$$

This kind of distribution is called a *power law*. We see that Zipf's Law is the special case $\beta = 1$ (or $\alpha = 2$), but in general the power β can be arbitrary. In practice, we try to find the best value of β that fits our data.

Heavy tail. The density function of the power law distribution, which is obtained by differentiating (1), is

$$f(x) = \frac{D(\alpha-1)}{x^\alpha}.$$

Notice that it decreases *polynomially*, not exponentially, with x. Contrast this with the other distributions that we have seen previously. For example, the exponential distribution has density

$$f_{\text{Exp}}(x) = \lambda e^{-\lambda x},$$

which decreases exponentially with x, while the normal distribution has density

$$f_{\text{Normal}}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

which decreases exponentially with x^2 as we move away from the mean μ. On the other hand, the power law density only decreases polynomially, which is much slower. Figure 2 shows a comparison of the polynomial and exponential decreases (we shift the exponential functions so they all start at the same point).

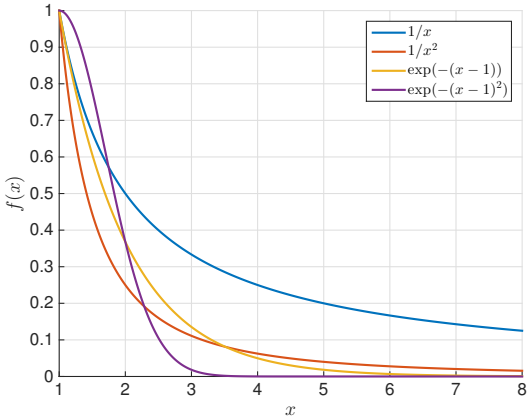


Figure 2: Comparison of the polynomial vs. exponential decrease.

Similarly, the tail probability (1) itself is also decreasing polynomially with x. On the other hand, the other distributions that we have seen so far (binomial, normal, Poisson, geometric, and exponential) all have tail probabilities that are decreasing exponentially with x. For this reason, power law distributions are often called *"heavy tailed"* distributions (the tail probability is larger than normal or other distributions).

Where do power laws come from? We know that the exponentially concentrated distributions are produced by many independent contributions (as in the binomial distribution, or the normal distribution, viz. the law of large numbers). What kind of processes produce power law distributions? In general, power law distributions typically appear in phenomena related to human activities. There are several processes that are known to produce such distributions:

- **Size-independent growth:** If the ratio of a person's income next year to this year's income is a random variable that does not depend on the income, then it can be shown that, after a while, a population whose incomes evolve this way will obey a power law. This is how a network evolves: high-degree nodes are "important," and attract more edges.
- **Trade-offs:** If people try to optimize a single objective, then concentrated distributions often result. But if they must optimize — trade one against the other — two or more objectives simultaneously, then the optimum usually has power law-distributed features. Entities interacting in a complicated environment like the Internet and the www are likely to behave this way.

- **Imitation or copying:** Consider the number of citations that a paper receives. When people write research papers and they need to find a citation for some topic, they typically look at the literature and see what paper is usually cited for that topic, which results in that paper getting more citations. This imitation or copying process results in a power law distribution.
- **Preferential attachment:** The web is arguably constructed by new documents creating hyperlinks to documents discovered by following other hyperlinks. If a document has many links attached to it, then it is more likely to be discovered, and in turn it is more likely to receive more links. Thus, popular documents are likely to become more popular, a phenomenon informally called *"the rich gets richer."* This results in a power law distribution.

Polya Urn

There is a simple model that captures the essence of preferential attachment. Suppose that you have two bins, the white and the black bin, with w and b balls inside, respectively, and you throw a new ball. The balls are magnetic, and they attract the new ball with force proportional to their number; so the new ball will go to the white bin with probability $\frac{w}{w+b}$.

Suppose we start with $w = b = 1$, and throw in n new balls one at a time. How is the resulting distribution going to look like? This process is called a *Polya urn*. Intuitively, the "attraction" rule is going to enhance inequalities, so the bins are not going to be very load-balanced. But how exactly?

For example, what is the probability that, if we throw in 1,000 balls, there will be 900 or more balls that land in the black bin? The answer is simple: 10%. (Notice that, in ordinary non-magnetic balls and bins, the probability of a bin getting 900 or more balls is extremely small, because the number of balls in a bin is a binomial random variable that is sharply concentrated around its mean 500.)

More generally, if we throw in n balls, then we claim that for any $m \in \{0, 1, \ldots, n\}$,

$$\Pr[m \text{ balls go to the white bin}] = \frac{1}{n+1}.$$

In other words, the number of balls that go to any one bin is uniformly distributed! (This is in contrast with ordinary balls and bins, where the number of balls that go to a bin has binomial distribution.)

Here is why. Suppose that we generate a permutation of $\{0, 1, 2, \ldots, n\}$ as follows: we start with a ball labeled 0 somewhere on the line. Then we throw in a ball labeled 1; it will end up left or right of 0 with probability 50%. Suppose it ends up to the right of 0. Then we throw ball 2. It will end up with equal probability to the left of 0, to the right of 1, or between 0 and 1. When we throw ball 3, there are 4 possible positions, and so on. It is easy to convince yourself that, this way, all permutations are generated with equal probability.

Now suppose that we call the balls that ended up to the right of ball 0 "white" and the ones to the left "black". It is easy to see that at each step the new ball is white with probability exactly $\frac{w}{w+b}$; hence this is precisely the Polya urn described above!

One more observation: Where is 0 going to be? Since this is a random permutation, 0 can be at any one of the $n+1$ positions with equal probability. Hence, the number of (new) white balls is going to be $0, 1, \ldots, n$ with equal probability! And the probability that the 0 ball will end up in the 10 percentile (e.g., ≤ 100 white balls and ≥ 900 black balls) is exactly 10%, as claimed.

General case. Suppose now that we start with w balls in the white bin and b balls in the black bins. How does the result change? Notice that this is the same problem as having $k = w + b$ bins, each of which has

1 ball in it. We throw n balls to the k bins following the same magnetic rule as before, and at the end we combine the results in the w bins into one, and the remaining k bins into the other.

So now we have n (new) balls and k colors, not just two. This is the same as having $n+k-1$ balls, where the first $k-1$ ones are just "color boundaries". And the end result is tantamount to selecting $k-1$ from among these balls: The first color is just the balls to the left of the first boundary, the second consists of the balls between the first and second boundary, and so on. Since there are k bins (one for each color) and n balls placed in them, by symmetry the expected number of balls in each bin is $\frac{n}{k}$. So when we combine the w bins into one and the remaining b bins into the other, we see that the expected number of balls in the white bin, where we started with w balls, is equal to $w \cdot \frac{n}{k}$, and the expected number of balls in the black bin, where we started with b balls, is $b \cdot \frac{n}{k}$.

We can also say something more about each of the k bins. We know that the expected number of balls in each bin is $\frac{n}{k}$. We can show that the maximum will be about $\frac{n}{k} \log n$, only a $\log n$ factor above the expectation. But the minimum will be much smaller: $\frac{n}{k^2}$, or about k times smaller than the maximum. In this sense, Polya urns foster imbalances.

There is a model of the web that is a slight extension of Polya urns: Suppose that, once a new ball is thrown, it is added to one of the bins, but also a new bin is started, with one ball in it. This models a graph where nodes arrive, and upon arrival each node u has one edge directed out of it; this edge is directed to a previous node v with probability equal to the degree of the node v at the time. The in-degrees of such graphs are known to be power law-distributed with $\alpha = 3$.

How to Lie with Statistics

"There are three kinds of lies: lies, damned lies, and statistics," as Mark Twain is reputed to have said. In this lecture we will see why statistics are so easy to misuse, and some of the ways we must be careful while evaluating statistical claims.

First a very simple but important point about statistics. Statistics cannot be used to establish causation, they can only show correlations. As an example, the governor of a certain state is concerned about the test scores of high school students in the state. One of his aides brings up an interesting statistic: there is a very strong link between student test scores and the taxes paid by the parents of the student. The parents of high scoring students pay more taxes. The aide's suggestion for increasing student test scores is unusual; sharply increase tax rates. Surely student test scores will follow! The fallacy, of course, is that even though there is a correlation between test scores and parents taxes, there is likely no causal connection. A better explanation is that there is some hidden variable that explains the correlation. In this case the obvious choice is the income of the parents. This determines the taxes paid. And since the quality of high school that a student attends is to a large extent determined by the parent's income, we see a causal link from parent's income to both taxes and test scores.

Let us now turn to a very important paradox in probability called Simpson's paradox, described by Simpson in his 1951 paper. Let us start with an example, which studies the 20 year survival rate of smokers. A paper by Appleton, French, and Vanderpump (1996, *American Statistician*) surveyed 1314 English women in 1972–1974 and again after 20 years. Their results are summarized in the following table:

Smoker	Dead	Alive	Total	% Dead
Yes	139	443	582	24
No	230	502	732	31
Total	369	945	1314	28

From this data one might conclude that non-smoking kills! How does one explain this unusual data? The answer lies in the composition of the two groups. Smoking was unpopular in the middle of the 20th century among women in England, and it increased only in the 1970's; but it was mostly young women who started to smoke. Therefore, the sample of smokers in the study was heavily biased towards young women, whose expected lifespan was of course much larger than 20 years. This becomes clearer when we look more closely at the results of the study broken down by age group:

Age group	18–24		25–34		35–44		45–54		55–54	
Smoker	Y	N	Y	N	Y	N	Y	N	Y	N
Dead	2	1	3	5	11	7	27	12	51	40
Alive	53	61	121	152	95	114	103	66	64	81
Ratio	2.3		0.75		2.4		1.44		1.61	

The interesting thing to notice is that the fatality rates are significantly higher for smokers in almost every age group! The data could be made even more dramatic by increasing the smoking fatalities in the couple of exceptional groups by one or two, thereby achieving the following strange result: in each separate category, the percentage of fatalities among smokers is higher, and yet the overall percentage of fatalities among smokers is lower. This is an example of a phenomenon known as **Simpson's paradox**. (For a nice treatment of this topic, and a different example, you may wish to consult the Wikipedia entry.)

A real word example is provided very close to home by the UC Berkeley gender bias case. In 1973, UC Berkeley was sued for bias against women applying to grad school. Data showed that 44% of men were admitted and only 30% of women. Since admission is decided by departments, the University decided to investigate which departments were "discriminating" against women. It turned out that none of them were! Here is some admissions data for the four largest departments:

Department	#male applicants	#female applicants	%male admit	%female admit
A	825	108	62	82
B	560	25	63	68
C	325	593	37	34
D	417	375	33	35

The explanation is that women applied in larger numbers to departments that had lower admission rates.

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Note 3

1 Mathematical Induction

Introduction. In this note, we introduce the proof technique of *mathematical induction*. Induction is a powerful tool which is used to establish that a statement holds for *all* natural numbers. Of course, there are infinitely many natural numbers — induction provides a way to reason about them by finite means.

Let us demonstrate the intuition behind induction with an example. Suppose we wish to prove the statement: For all natural numbers n, $0+1+2+3+\cdots+n = \frac{n(n+1)}{2}$. More formally, using the universal quantifier from Note 1, we can write this as:

$$\forall n \in \mathbb{N}, \quad \sum_{i=0}^{n} i = \frac{n(n+1)}{2}. \tag{1}$$

How would you prove this? Well, you could begin by checking that it holds for $n = 0, 1, 2$, and so forth. But there are an infinite number of values of n for which it needs to be checked! Moreover, checking just the first few values of n does *not* suffice to conclude the statement holds for all $n \in \mathbb{N}$, as the following example demonstrates:

Sanity check! Consider the statement: $\forall n \in \mathbb{N}$, $n^2 - n + 41$ is a prime number. Check that it holds for the first few natural numbers. (In fact, you could check all the way up to $n = 40$ and not find a counterexample!) Now check the case of $n = 41$.

In mathematical induction, we circumvent this problem by making an interesting observation: Suppose the statement holds for some value $n = k$, i.e. $\sum_{i=0}^{k} i = \frac{k(k+1)}{2}$. (This is called the *induction hypothesis*.) Then:

$$\left(\sum_{i=0}^{k} i\right) + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{(k+1)(k+2)}{2}, \tag{2}$$

i.e. the claim must also hold for $n = k+1$! In other words, if the statement holds for some k, then it must also hold for $k+1$. Let us call the argument above the *inductive step*. The inductive step is a very powerful tool: If we can show the statement holds for k, then the inductive step allows us to conclude that it also holds for $k+1$; but now that $k+1$ holds, the inductive step implies that $k+2$ must hold; in fact, we can repeat this argument indefinitely for all $n \geq k$! So is that it? Have we proven Equation (1)? Almost!

The problem is that in order to apply the inductive step, we first have to establish that Equation (1) holds for some *initial* value of k. Since our aim is to prove the statement for all natural numbers, the obvious choice is $k = 0$. We call this choice of k the *base case*. Then, if the base case holds, the axiom of *mathematical induction* says that the inductive step allows us to conclude that the Equation 1 indeed holds for all $n \in \mathbb{N}$.

Let us now formally re-write this proof.

Theorem 3.1. $\forall n \in \mathbb{N}, \sum_{i=0}^{n} i = \frac{n(n+1)}{2}$.

Proof. We proceed by *induction* on the variable n.

Base case $(n = 0)$: Here, we have $\sum_{i=0}^{0} i = 0 = \frac{0(0+1)}{2}$. Thus, the base case is correct.

Inductive Hypothesis: For arbitrary $n = k \geq 0$, assume that $\sum_{i=0}^{k} i = \frac{k(k+1)}{2}$. In words, the Inductive Hypothesis says "let's assume we have proved the statement for an arbitrary value of $n = k \geq 0$".

Inductive Step: Prove the statement for $n = (k+1)$, i.e. show that $\sum_{i=0}^{k+1} i = \frac{(k+1)(k+2)}{2}$:

$$\sum_{i=0}^{k+1} i = \left(\sum_{i=0}^{k} i\right) + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}, \tag{3}$$

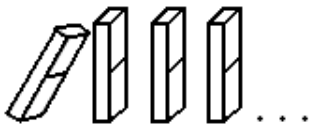
where the second equality follows from the Induction Hypothesis. By the principle of mathematical induction, the claim follows. □

Sanity check! How exactly did the Induction Hypothesis help us show the second equality in Equation (3)? (Hint: We used it to replace $\sum_{i=0}^{k} i$ with something useful.)

Recap. What have we learned so far? Letting $P(n)$ denote the statement $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$, our goal was to prove that $\forall n \in \mathbb{N}$, $P(n)$. The *principle of induction* asserts that to prove this requires three simple steps:

1. **Base Case**: Prove that $P(0)$ is true.
2. **Inductive Hypothesis**: For arbitrary $k \geq 0$, assume that $P(k)$ is true.
3. **Inductive Step**: With the assumption of the Inductive Hypothesis in hand, show that $P(k+1)$ is true.

Let us visualize how these three steps fit together using dominoes. Let statement $P(n)$ be represented by a sequence of dominoes, numbered from $0, 1, 2, ..., n$, such that $P(0)$ corresponds to the 0^{th} domino, $P(1)$ corresponds to the 1^{st} domino, and so on. The dominoes are lined up so that if the k^{th} domino is knocked over, then it in turn knocks over the $k+1^{st}$. Knocking over the k^{th} domino corresponds to proving $P(k)$ is true. And the induction step corresponds to the placement of the dominoes to ensure that if the k^{th} domino falls, in turn it knocks over the $k+1^{st}$ domino. The base case $(n = 0)$ knocks over the $0th$ domino, setting off a chain reaction that knocks down all the dominoes!



Finally, a word about choosing an appropriate base case — in this example, we chose $k = 0$, but in general, the choice of base case will naturally depend on the claim you wish to prove.

Another proof by induction. Let us do another proof by induction. Recall from Note 1 that for integers a and b, we say that *a divides b*, denoted $a|b$, if and only if there exists an integer q satisfying $b = aq$.

Sanity check! How would you write the definition of $a|b$ formally using quantifiers? (Answer: $\forall a, b \in \mathbb{Z}$, $a|b$ iff $\exists\, q \in \mathbb{Z}$ such that $b = aq$.)

Theorem 3.2. *$\forall n \in \mathbb{N}$, $n^3 - n$ is divisible by 3.*

Proof. We proceed by induction over n. Let $P(n)$ denote the statement $\forall n \in \mathbb{N}$, $n^3 - n$ is divisible by 3.

Base case ($n = 0$): $P(0)$ asserts that $3|(0^3 - 0)$ or $3|0$, which is true since any non-zero integer divides 0.

Inductive Hypothesis: Assume for $n = k \geq 0$ that $P(k)$ is true. That is, $3|(k^3 - k)$, or $\exists q \in \mathbb{Z}, k^3 - k = 3q$.

Inductive Step: We show that $P(k+1)$ is true, i.e. that $3|((k+1)^3 - (k+1))$. To show this, we expand the number $((k+1)^3 - (k+1))$ as follows:

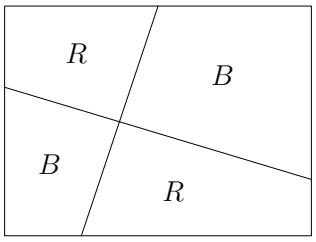
$$\begin{aligned}
(k+1)^3 - (k+1) &= k^3 + 3k^2 + 3k + 1 - (k+1) \\
&= (k^3 - k) + 3k^2 + 3k \\
&= 3q + 3(k^2 + k) \text{ for some } q \in \mathbb{Z} \quad \text{(Induction Hypothesis)} \\
&= 3(q + k^2 + k).
\end{aligned}$$

Sanity check! How exactly did the Induction Hypothesis help us show the third equality above?

We conclude that $3|((k+1)^3 - (k+1))$. Thus, by the principle of induction, $\forall n \in \mathbb{N}$, $3|(n^3 - n)$. □

Two Color Theorem. We now consider a more advanced proof by induction for a simplified version of the famous *four color theorem*. The four color theorem states that any map can be colored with four colors such that any two adjacent countries (which share a border, but not just a point) have different colors. The four color theorem is very difficult to prove, and several bogus proofs were claimed since the problem was first posed in 1852. In fact, it was not until 1976 that a computer-assisted proof of the theorem was finally given Appel and Haken. (For an interesting history of the problem and state-of-the-art proof, which is nonetheless still very challenging, see `www.math.gatech.edu/~thomas/FC/fourcolor.html`).

In this note, we consider a simpler version of the theorem in which our "map" is given by a rectangle which is divided into regions by drawing lines, such that each line divides the rectangle into two regions. Can we color such a map using no more than two colors (say, red and blue) such that no two bordering regions have the same color? To illustrate, here is an example of a two-colored map:



Theorem 3.3. *Let $P(n)$ denote the statement "Any map with n lines is two-colorable". Then, it holds that $\forall n \in \mathbb{N}\ P(n)$.*

Proof. We proceed by induction on n.

Base Case ($n = 0$): Clearly $P(0)$ holds, since if we have $n = 0$ lines, then we can color the entire map using a single color.

Inductive Hypothesis: For any $n = k \geq 0$, assume $P(k)$.

Inductive Step: We prove $P(k+1)$. Specifically, we are given a map with $k+1$ lines and wish to show that it can be two-colored. Let's see what happens if we remove a line. With only k lines on the map, the Induction Hypothesis says we can two-color the map. Let us make the following observation: Given a valid coloring, if we swap red $\leftrightarrow$ blue, we still have a two-coloring. With this in mind, let us place back the line we removed, and leave colors on one side of the line unchanged. On the other side of the line, swap red $\leftrightarrow$ blue. This is illustrated by Figure 1. We claim that this is a valid two-coloring for the map with $k+1$ lines.

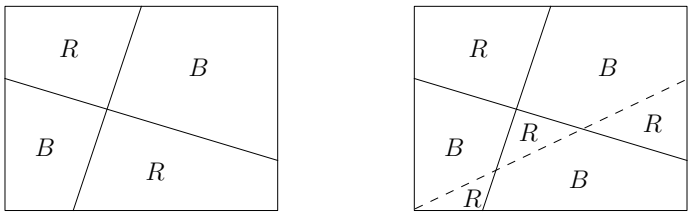


Figure 1: (Left) The map with the $(k+1)$st line removed. (Right) The map with the $(k+1)$st line represented by dashed line.

Why does this work? Consider two regions separated by a shared border. Then, one of two cases must hold: Case 1 is when the shared border is the line that was removed and replaced, i.e. line $k+1$. But by construction, we flipped the colors on one side of this line so that any two regions separated by it have distinct colors. Case 2 is when the shared border is one of the original k lines; here, the Induction Hypothesis guarantees that two regions separated by this border have different colors. Thus, in both cases, the regions separated by the shared border have distinct colors, as required. □

2 Strengthening the Induction Hypothesis

When using induction, it can be very important to choose the correct statement to prove. For example, suppose we wish to prove the statement: $\forall n \geq 1$, the sum of the first n odd numbers is a perfect square. Here is a first proof attempt via induction.

Proof attempt. We proceed by induction on n.

Base Case ($n = 1$): The first odd number is 1, which is a perfect square.

Inductive Hypothesis: Assume that the sum of the first k odd numbers is a perfect square, say m^2.

Inductive Step: The $(k+1)$st odd number is $2k+1$. Thus, by the induction hypothesis, the sum of the first $k+1$ odd numbers is m^2+2k+1. But now we are stuck. Why should m^2+2k+1 be a perfect square? It seems our Induction Hypothesis is too "weak"; it does not give us enough structure to say anything meaningful about the $(k+1)$ case.

So let's take a step back for a moment, and do a preliminary check to ensure our claim isn't obviously false: Let's compute the values of the first few cases. Perhaps in the process, we can also uncover some hidden structure we have not yet identified.

- $n = 1 : 1 = 1^2$ is a perfect square.
- $n = 2 : 1+3 = 4 = 2^2$ is a perfect square.
- $n = 3 : 1+3+5 = 9 = 3^2$ is a perfect square.
- $n = 4 : 1+3+5+7 = 16 = 4^2$ is a perfect square.

It looks like we have good news and even better news: The good news is that we have not yet found a counterexample to our claim. The even better news is that there is a surprising pattern emerging — the sum of the first n odd numbers is not just a perfect square, but is equal precisely to n^2! Motivated by this discovery, let's try something counterintuitive: Let us try to show the following *stronger* claim.

Theorem 3.4. *For all $n \geq 1$, the sum of the first n odd numbers is n^2.*

Proof. We proceed by induction on n.

Base Case ($n = 1$): The first odd number is 1, which is 1^2.

Inductive Hypothesis: Assume that the sum of the first k odd numbers is k^2.

Inductive Step: The $(k+1)$-st odd number is $2k+1$. Applying the Induction Hypothesis, the sum of the first $k+1$ odd numbers is $k^2+(2k+1) = (k+1)^2$. Thus, by the principle of induction the theorem holds. □

So let's get this straight — we couldn't prove our original statement, so instead we hypothesized a stronger one and managed to prove that. Why on Earth did this work? The reason is that our original claim did not capture the true *structure* of the underlying fact we were trying to prove — it was too vague. As a result, our Induction Hypothesis wasn't strong enough to prove our desired result. In contrast, although our second claim is *a priori* stronger, it also has more structure to it; this, in turn, makes our Induction Hypothesis stronger — we can use the fact that not only is the sum of the first k odd numbers a perfect square, but that it in fact equals k^2. This additional structure is exactly what we needed to complete the proof. In summary, we have demonstrated an example in which, although a claim was true, the precise formulation of the Induction Hypothesis made the difference between a failing and successful proof.

Example. Let us now try a second example; this time, we'll let you do some of the work! Suppose we wish to prove the claim: $\forall n \geq 1, \sum_{i=1}^{n} \frac{1}{i^2} \leq 2$.

Sanity check! The "obvious" choice of Induction Hypothesis says the following: Assume that for $n = k$, $\sum_{i=1}^{k} \frac{1}{i^2} \leq 2$. Why does this not suffice to prove the claim for $n = k+1$, i.e. to show that $\sum_{i=1}^{k} \frac{1}{i^2} + \frac{1}{(i+1)^2} \leq 2$? (Hint: Is it possible that $\sum_{i=1}^{k} \frac{1}{i^2}$ actually *equals* 2?)

Now let's again do the unthinkable — let's prove the following *stronger* statement, i.e. let's strengthen our induction hypothesis.

Theorem 3.5. *For all $n \geq 1$, $\sum_{i=1}^{n} \frac{1}{i^2} \leq 2 - \frac{1}{n}$.*

Proof. We proceed by induction on n.

Base Case ($n = 1$): We have $\sum_{i=1}^{1} \frac{1}{i^2} = 1 \leq 2 - \frac{1}{1}$, as required.

Inductive Hypothesis: Assume that the claim holds for $n = k$.

Inductive Step: By the Induction Hypothesis, we have that

$$\sum_{i=1}^{k+1} \frac{1}{i^2} = \sum_{i=1}^{k} \frac{1}{i^2} + \frac{1}{(k+1)^2} \leq 2 - \frac{1}{k} + \frac{1}{(k+1)^2}.$$

Thus, to prove our claim, it suffices to show that

$$-\frac{1}{k} + \frac{1}{(k+1)^2} \leq -\frac{1}{k+1}. \tag{4}$$

Exercise. Show that Equation (4) holds. (Hint: Multiply both sides of the inequality by $(k+1)$.)

By the principle of mathematical induction, the claim holds. □

3 Simple Induction vs. Strong Induction

Thus far, we have been using a notion of induction known as *simple* or *weak* induction. There is another notion of induction, which we now discuss, called *strong* induction. The latter is similar to simple induction, except we have a slightly different inductive hypothesis: Instead of just assuming $P(k)$ is true (as was the case with simple induction), we assume the stronger statement that $P(0)$, $P(1)$, ..., and $P(k)$ are *all* true (i.e. that $\bigwedge_{i=0}^{k} P(i)$ is true).

Is there a difference between the power of strong and weak induction, i.e. can strong induction prove statements which weak induction cannot? *No!* Intuitively, this can be seen by returning to our domino analogy from Section 1. In this picture, weak induction says that if the kth domino falls, so does the $(k+1)$st one, whereas strong induction says that if dominos 1 through k fall, then so does the $(k+1)$st one. But these

scenarios are equivalent: If the first domino falls, then applying weak induction repeatedly we conclude that the first domino in turn topples dominos 2 through k, setting up the case of $k+1$, just as is the case with strong induction. With that said, strong induction does have an appealing advantage — it can make proofs *easier*, since we get to assume a stronger hypothesis. How should we understand this? Consider the analogy of a screwdriver and a power screwdriver — they both accomplish the same task, but the latter is much easier to use!

Let's try a simple example[1] of strong induction. As a bonus, this is our first induction proof requiring *multiple* base cases.

Theorem 3.6. *For every natural number $n \geq 12$, it holds that $n = 4x + 5y$ for some $x, y \in \mathbb{N}$.*

Proof. We proceed by induction on n.

Base Case ($n = 12$): We have $12 = 4 \cdot 3 + 5 \cdot 0$.

Base Case ($n = 13$): We have $13 = 4 \cdot 2 + 5 \cdot 1$.

Base Case ($n = 14$): We have $14 = 4 \cdot 1 + 5 \cdot 2$.

Base Case ($n = 15$): We have $15 = 4 \cdot 0 + 5 \cdot 3$.

Inductive Hypothesis: Assume that the claim holds for all $12 \leq n \leq k$ for $k \geq 15$.

Inductive Step: We prove the claim for $n = k+1 \geq 16$. Specifically, note that $(k+1) - 4 \geq 12$; thus, the Induction Hypothesis implies that $(k+1) - 4 = 4x' + 5y'$ for some $x', y' \in \mathbb{N}$. Setting $x = x' + 1$ and $y = y'$ completes the proof. □

Sanity check! Why would the proof above fail if we used weak induction instead of strong induction?

Let's try a second, more advanced, example. For this, recall that a number $n \geq 2$ is prime if 1 and n are its only divisors.

Theorem 3.7. *Every natural number $n > 1$ can be written as a product of primes.*

Proof. We proceed by induction on n. Let $P(n)$ be the proposition that n can be written as a product of primes. We will prove that $P(n)$ is true for all $n \geq 2$.

Base Case ($n = 2$): We start at $n = 2$. Clearly $P(2)$ holds, since 2 is a prime number.

Inductive Hypothesis: Assume $P(k)$ is true *for all* $2 \leq n \leq k$.

Inductive Step: Prove that $n = k+1$ can be written as a product of primes. We have two cases: either $k+1$ is a prime number, or it is not. For the first case, if $k+1$ is a prime number, then we are done. For the second case, if $k+1$ is not a prime number, then by definition $k+1 = xy$ for some $x, y \in \mathbb{Z}^+$ satisfying $1 < x, y < k+1$. By the Induction Hypothesis, x and y can each be written as a product of primes (since $x, y \leq n$). But this implies that $k+1$ can also be written as a product of primes. □

[1] This problem also goes under the name of the Postage Stamp Problem, which states that every amount of postage over 12 cents can be paid using 4- and 5-cent stamps.

Sanity check! Why does this proof fail if we instead use simple induction? (Hint: Suppose $k = 42$. Since $42 = 6 \times 7$, in the Inductive Step we wish to use the facts that $P(6)$ and $P(7)$ are true. But weak induction does not give us this — which value of P does weak induction allow us to assume is true in this case?)

Finally, for those who wish to have a more formal understanding of why weak and strong induction are equivalent, consider the following. Let $Q(n) = P(0) \wedge P(1) \wedge \cdots \wedge P(n)$. Then, strong induction on P is equivalent to weak induction on Q.

4 Recursion, programming, and induction

There is an intimate connection between induction and recursion, which we explore here via two examples.

Example 1: Fibonacci's rabbits. In the 13th century lived a famous Italian mathematician known as *Fibonacci*[2], who in 1202 considered the following rabbit-based puzzle: Starting with a pair of rabbits, how many rabbits are there after a year, if each month each pair begets a new pair which from the second month on becomes productive? This model of population growth can be modeled by recursively defining a function; the resulting sequence of numbers is nowadays referred to as the *Fibonacci* numbers.

To recursively model our rabbit population, let $F(n)$ denote the number of pairs of rabbits in month n. By the rules above, our initial conditions are as follows: Clearly $F(0) = 0$. In month 1, a single pair of rabbits is introduced, implying $F(1) = 1$. Finally, since it takes a month for new rabbits to become reproductive, we also have $F(2) = 1$.

In month 3, the fun begins. For example, $F(3) = 2$, since the original pair breeds to produce a new pair. But how about $F(n)$ for a general value of n? It seems difficult to give an explicit formula for $F(n)$. However, we can define $F(n)$ *recursively* as follows. In month $n-1$, by definition we have $F(n-1)$ pairs. How many of these pairs were productive? Well, only those that were already alive in the previous month, i.e. $F(n-2)$ of them. Thus, we have $F(n-2)$ new pairs in the nth month in addition to our existing $F(n-1)$ pairs. Hence, we have $F(n) = F(n-1) + F(n-2)$. To summarize:

- $F(0) = 0$.
- $F(1) = 1$.
- For $n \geq 2$, $F(n) = F(n-1) + F(n-2)$.

Pretty neat, except you might wonder why we care about rabbit reproduction in the first place! It turns out this simplified model of population growth illustrates a fundamental principle: Left unchecked, populations grow exponentially over time. In fact, understanding the significance of this unchecked exponential population growth was a key step that led Darwin to formulate his theory of evolution. To quote Darwin: "There is no exception to the rule that every organic being increases at so high a rate, that if not destroyed, the earth would soon be covered by the progeny of a single pair."

Now, we promised you programming in the title of this section, so here it is — a trivial recursive program to evaluate $F(n)$:

[2]Fibonacci was also known as Leonardo of Pisa. If you ever find yourself in Pisa, Italy, you can visit his grave; his remains are buried at Campo Santo.

```
function F(n)
  if n=0 then return 0
  if n=1 then return 1
  else return F(n-1) + F(n-2)
```

Exercise. How long does it take this program to compute $F(n)$, i.e. how many calls to $F(n)$ are needed?

The exercise above should convince you that this is a very inefficient way to compute the nth Fibonacci number. Here is a much faster iterative algorithm to accomplish the same task (this should be a familiar example of turning a tail-recursion into an iterative algorithm):

```
function F2(n)
  if n=0 then return 0
  if n=1 then return 1
  a = 1
  b = 0
  for k = 2 to n do
    temp = a
    a = a + b
    b = temp
  return a
```

Exercise. How long does this iterative algorithm take to compute $F_2(n)$, i.e. how many iterations of its loop are needed? Can you show by induction that this new function $F_2(n) = F(n)$?

Example 2: Binary search. We next use induction to analyze a recursive algorithm which even your grandmother is likely familiar with — binary search! Let us discuss binary search in the context of finding a word in the dictionary: To find word W, we open the dictionary to its middle page. If the letter of that page comes after the first letter of W, we recurse on the first half of the dictionary; else, we recurse on the last half of the dictionary. (For simplicity, we ignore the issue of dividing a dictionary with an *odd* number of pages into two halves.) Once we've narrowed down the dictionary to a single page, we resort to a brute force search to find W on that page. In pseudocode, we have:

1. //precondition: W is a word and D is a subset of the dictionary with at least 1 page.
2. //postcondition: Either the definition of W is returned, or "W not found" is returned.
3. findWord(W, D) {
4. //Base case
5. If (D has precisely one page)
6. Look for W in D by brute force. If found, return its definition; else, return "W not found".
7.
8. //Recursive case
9. Let W' be the first word on the middle page of D.

```
10.     If ( W comes before W' )
11.         return findWord( first half of D )
12.     Else
13.         return findWord( last half of D )
14. }
```

Let us prove using induction that findWord() is *correct*, i.e. it returns the definition of W if W is in the dictionary D. As a bonus, the proof requires us to use *strong* induction instead of simple induction!

Proof of correctness of findWord(). We proceed by strong induction on the number of pages, n, in D.

Base Case ($n = 1$): If D has one page, line 6 searches via brute force for W. Thus, if W is present, it is found and returned; else, we return "W not found", as desired.

Inductive Hypothesis: Assume that findWord() is correct for all $1 \leq n \leq k$. We prove it is correct for $n = k+1$.

Inductive Step: After Step 9, we know W'; thus we can determine whether W must be in the first or second half of D. In the first case, we recurse on the first half of D in line 11, and in the second case we recurse on the second half of D in line 13. By the Induction Hypothesis, the recursive call correctly finds W in the first or second half, or returns "W not found". Since we return the recursive call's answer, we conclude that findWord() correctly finds W in D of size $n = k+1$. By the principle of mathematical induction, the findWord() is correct. □

Sanity check! Why did we require *strong* induction in the proof above?

---

5 False proofs

It is very easy to prove false things if your proof is incorrect! Let's illustrate with a famous example. In the middle of the last century, a colloquial expression in common use was "that is a horse of a different color", referring to something that is quite different from normal or common expectation. The reknowned mathematician George Polya, who was also a great expositor of mathematics for the lay public, gave the following proof to show that there is no horse of a different color!

Theorem 3.8. *All horses are the same color.*

Proof. We proceed by induction on the number of horses, n. Let $P(n)$ denote the statement of the claim.

Base Case ($n = 1$): $P(1)$ is certainly true, since if you have a set containing just one horse, all horses in the set have the same color.

Inductive Hypothesis: Assume $P(n)$ holds for any $n \geq 1$.

Inductive Step: Given a set of $n+1$ horses $\{h_1, h_2, \ldots, h_{n+1}\}$, we can exclude the last horse in the set and apply the inductive hypothesis just to the first n horses $\{h_1, \ldots, h_n\}$, deducing that they all have the same

color. Similarly, we can conclude that the last n horses $\{h_2, \ldots, h_{n+1}\}$ all have the same color. But now the "middle" horses $\{h_2, \ldots, h_n\}$ (i.e., all but the first and the last) belong to both of these sets, so they have the same color as horse h_1 and horse h_{n+1}. It follows, therefore, that all $n+1$ horses have the same color. We conclude, by the principle of induction, that all horses have the same color. □

Clearly, it is not true that all horses are of the same color! So, where did we go wrong? Recall that in order for the principle of mathematical induction to apply, the induction step must show the statement: $\forall n \geq 1$, $P(n) \implies P(n+1)$. We claim that in the proof above, this statement is false — in particular, there exists a choice of n which yields a *counterexample* to this statement.

Exercise. For which value of n is it *not* true that $P(n) \implies P(n+1)$? (Hint: Think about the key property in the proof of having "middle" horses.)

---

6 Practice Problems

1. Prove for any natural number n that $1^2 + 2^2 + 3^2 + \ldots + n^2 = \frac{1}{6}n(n+1)(2n+1)$.

2. Prove that $3^n > 2^n$ for all natural numbers $n \geq 1$.

3. In real analysis, Bernoulli's Inequality is an inequality which approximates the exponentiations of $1+x$. Prove this inequality, which states that $(1+x)^n \geq 1+nx$ if n is a natural number and $1+x > 0$.

4. A common recursively defined function is the *factorial*, defined for a nonnegative number n as $n! = n(n-1)(n-2)...1$, with base case $0! = 1$. Let us reinforce our understanding of the connection between recursion and induction by considering the following theorem involving factorials.

   Theorem: $\forall n \in \mathbb{N}, n > 1 \implies n! < n^n$.

   Prove this theorem using induction. (Hint: In the Inductive Step, write $(n+1)! = (n+1) \cdot n!$, and use the Induction Hypothesis.)

5. A *celebrity* at a party is someone whom everyone knows, yet who knows no one. Suppose that you are at a party with n people. For any pair of people A and B at the party, you can ask A if they know B and receive an honest answer. Give a recursive algorithm to determine whether there is a celebrity at the party, and if so who, by asking at most $3n-4$ questions. (Note: for the purpose of this question you are just visiting the party to ask questions. What you are trying to determine is whether the n people actually attending the party include a celebrity).

   Prove by induction that your algorithm always correctly identifies a celebrity iff there is one, and that the number of questions is at most $3n-4$.

1 The Stable Marriage Problem

In the previous note, we discussed the powerful proof technique of induction. In this note, we apply induction to analyze the solution to an important problem known as the *Stable Marriage Problem*, which we now introduce.

Suppose you run a dating agency, and your task is to match up n men and n women. Each man has an ordered *preference list* of the n women, and each woman has a similar list of the n men. For example, consider the case of $n = 3$, i.e. three men Alex, Bob, and Charles, and three women Anita, Bridget, and Christine, with the following preference lists (lists are ordered from most favorable to least favorable):

Men	Women		
Alex	Anita	Bridget	Christine
Bob	Bridget	Anita	Christine
Charles	Anita	Bridget	Christine

Women	Men		
Anita	Bob	Alex	Charles
Bridget	Alex	Bob	Charles
Christine	Alex	Bob	Charles

What you would like to do as head of the dating agency is to pair up each man with a woman. For example, two possible pairings are {(Alex, Anita), (Bob, Bridget), (Charles, Christine)} and {(Alex, Bridget), (Bob, Christine), (Charles, Anita)}. However, you don't want just any old pairing! In order for your dating firm to be successful, you wish the pairing to make "everyone happy", in that nobody can realistically hope to benefit by switching partners. Can you do this efficiently?

It turns out that there is indeed an algorithm to achieve this; moreover, it's remarkably simple, fast, and widely-used. It's called the *Stable Marriage algorithm* (a.k.a. the Gale-Shapley algorithm), and we present it now.

2 The Stable Marriage Algorithm

Every Morning: Each man proposes to the most preferred woman on his list who has not yet rejected him.

Every Afternoon: Each woman collects all the proposals she received in the morning; to the man she likes best, she responds "maybe, come back tomorrow" (she now has him *on a string*), and to the others, she says "never".

Every Evening: Each rejected man crosses off the woman who rejected him from his list.

The above loop is repeated each successive day until each woman has a man on a string; on this day, each woman marries the man she has on a string.

Let's digest the algorithm by running it on our example above. The following table shows which men propose to which women on the given day (the men in bold are "on a string"):

Days	Women	Proposals
1	Anita	**Alex**, Charles
	Bridget	**Bob**
	Christine	—
2	Anita	**Alex**
	Bridget	**Bob**, Charles
	Christine	—
3	Anita	**Alex**
	Bridget	**Bob**
	Christine	**Charles**

Thus, the algorithm outputs the stable pairing: {(Alex, Anita), (Bob, Bridget), (Charles, Christine)}.

Before analyzing the algorithm's properties further, let's take a moment to ask one of the first questions you should always ask when confronted with a new concept: Why study stable marriage in the first place? What makes it and its solution so interesting? For this, we discuss the very real impact of the Gale-Shapley algorithm on the *Residency Match* program.

3 The Residency Match

Perhaps the most well-known application of the Stable Marriage Algorithm is the residency match program, which pairs medical school graduates and residency slots (internships) at teaching hospitals. Graduates and hospitals submit their ordered preference lists, and the stable pairing produced by a computer matches students with residency programs.

The road to the residency match program was long and twisted. Medical residency programs were first introduced about a century ago. Since interns offered a source of cheap labor for hospitals, soon the number of residency slots exceeded the number of medical graduates, resulting in fierce competition. Hospitals tried to outdo each other by making their residency offers earlier and earlier. By the mid-40s, offers for residency were being made by the beginning of junior year of medical school, and some hospitals were contemplating even earlier offers — to sophomores! The American Medical Association finally stepped in and prohibited medical schools from releasing student transcripts and reference letters until their senior year. This sparked a new problem, with hospitals now making "short fuse" offers to make sure that if their offer was rejected they could still find alternate interns to fill the slot. Once again the competition between hospitals led to an unacceptable situation, with students being given only a few hours to decide whether they would accept an offer.

Finally, in the early 1950's, this unsustainable situation led to the centralized system called the National Residency Matching Program (N.R.M.P.) in which the hospitals ranked the residents and the residents ranked

the hospitals. The N.R.M.P. then produced a pairing between the applicants and the hospitals, though at first this pairing was not stable. It was not until 1952 that the N.R.M.P. switched to the Stable Marriage Algorithm, resulting in a stable pairing.

Finally, if the above still hasn't convinced you of the worth of this algorithm, consider this: In 2012, Lloyd Shapley and Alvin Roth won the Nobel Prize in Economic Sciences by extending the Stable Marriage Algorithm[1]! (The moral of the story? Computer science pays off.)

4 Properties of the Stable Marriage Algorithm

There are two properties we wish to show about the stable marriage algorithm: First, that it doesn't run forever, i.e. it halts, and second, that it outputs a "good" pairing. The former is easy to show; let us do it now.

Lemma 4.1. *The stable marriage algorithm halts.*

Proof. The argument is simple: On each day that the algorithm does not halt, at least one man must eliminate some woman from his list (otherwise the halting condition for the algorithm would be invoked). Since each list has n elements, and there are n lists, this means that the algorithm must terminate in at most n^2 days. □

Next, we'd like to show that the algorithm finds a "good" pairing. Before we do so, let's clarify what we mean by "good".

4.1 Stability

What properties should a good pairing have? Perhaps we'd like to maximize the number of first ranked choices? Alternatively, we could minimize the number of last ranked choices. Or maybe it would be ideal to minimize the sum of the ranks of the choices, which may be thought of as maximizing the average happiness.

In this lecture we will focus on a very basic criterion: *stability*. A pairing is **unstable** if there is a man and a woman who prefer each other to their current partners. We will call such a pair a *rogue couple*. So a pairing of n men and n women is **stable** if it has no rogue couples. Let us now recall our example from earlier:

Men	Women		
Alex	Anita	Bridget	Christine
Bob	Bridget	Anita	Christine
Charles	Anita	Bridget	Christine

Women	Men		
Anita	Bob	Alex	Charles
Bridget	Alex	Bob	Charles
Christine	Alex	Bob	Charles

Sanity check! Consider the following pairing for the example above: {(Alex, Christine), (Bob, Bridget), (Charles, Anita)}. Why is this pairing unstable? (Hint: Alex and Bridget are a rogue couple — why?)

---

[1] See `http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2012/popular-economicsciences2012.pdf` for details.

Here is a stable pairing for our example: {(Bob, Anita), (Alex, Bridget), (Charles, Christine)}. Why is (Alex, Anita) *not* a rogue couple here? It's certainly true that Alex would rather be with Anita than his current partner. Unfortunately for him, Anita would rather be with her current partner than with Alex. Note also that both Charles and Christine are paired with their *least* favorite choice in this matching. Nevertheless, it *is* a stable pairing, since none of their preferred choices would rather be with them.

Before we discuss how to find a stable pairing, let us ask a more basic question: Do stable pairings always exist? Surely the answer is yes, since we could start with any pairing and make it more and more stable as follows: If there is a rogue couple, modify the current pairing so that they are together. Repeat. Surely this procedure must result in a stable pairing? Unfortunately this reasoning is not sound! Why? Although pairing up the rogue couple reduces the number of rogue couples by one, it may also *create new* rogue couples.

Let's further illustrate the fallacy of the reasoning above by applying it to a closely related scenario known as the *Roommates Problem*. In this problem, we have $2n$ people who must be paired up to be roommates (the difference being that unlike in stable marriage, a person can be paired with any of the other $2n-1$ people, i.e. there is no concept of *gender*). Now, suppose our approach of iteratively matching up rogue couples indeed *did* work. Since this approach does not exploit the existence of different genders, we can apply it to the roommates problem. Thus, we conclude there must always exist a stable pairing for roommates. Wouldn't you be surprised then to read the following counterexample, which gives an instance of the roommates problem which does *not* have a stable pairing!

Roommates			
A	B	C	D
B	C	A	D
C	A	B	D
D	–	–	–

We claim that in this instance, there always exists a rogue couple for any pairing. For example, the pairing {(A,B), (C,D)} contains the rogue couple B and C. How about {(B,C), (A,D)}? This pairing is unstable because now A and C are a rogue couple.

Sanity check! Verify that in this example there is *no* stable pairing!

We conclude that any proof that there always exists a stable pairing in the stable marriage problem *must* use the fact that there are two genders in an essential way. In the next section we give such a proof, and a nifty one at that: We prove that there must exist a stable pairing because the stable marriage algorithm always outputs one!

4.2 Analysis

We now prove that the stable marriage algorithm always outputs a stable pairing. Why should this be the case? Consider the following intuitive observation:

Observation 4.1. *Each man begins in the algorithm with his first choice being a possibility; as the algorithm proceeds, however, his best available option can only get* worse *over time. In contrast, women's options can only get* better *with time.*

Thus, as the algorithm progresses, each man's options get worse, while each woman's improves; at some point, the men and the women must "meet" in the middle, and intuitively such a pairing should be stable.

Let us formalize this intuition beginning with a formal statement of Observation 3.1 via the following lemma.

Lemma 4.2 (Improvement Lemma)**.** *If man M proposes to woman W on the kth day, then on every subsequent day W has someone on a string whom she likes at least as much as M.*

Proof. We proceed by induction on the day j, such that $j \geq k$.

Base case $(j = k)$: On day k, two cases are possible: (1) Either W has nobody on a string, in which case she says "maybe" to M, or (2) she has another man M' on a string, in which case, she says "maybe" to M only if she prefers M to M'. In either case, on day k, W has someone on a string who she likes at least as much as M.

Inductive Hypothesis: Suppose the claim is true for $j \geq k$.

Inductive Step: We prove the claim for day $j+1$. By the Induction Hypothesis, on day j, W had a man M' on a string which she likes at least as much as M. On day $j+1$, two cases are possible: (1) Either nobody proposes to W, in which case W keeps M' on a string, or (2) a man M'' proposes to W, in which case, she says maybe to M'' only if she prefers M'' to M'. In either case, W has someone on a string who she likes at least as much as M. □

Detour: The Well-Ordering Princple. Let's take a moment to consider an alternate proof of Lemma 3.2; in the process, we'll uncover a fundamental principle which is equivalent to induction.

Alternate proof of Lemma 3.2. As in our original proof, the claim certainly holds on day $j = k$. Suppose now, for sake of contradiction, that the jth day for $j > k$ is the first counterexample where W has either nobody or some M^* inferior to M on a string. On day $j-1$, she has M' on a string and likes M' at least much as M. According to the algorithm, M' still proposes to W on the jth day since she said "maybe" the previous day. So W has the choice of at least one man on the jth day; moreover, her best choice is at least as good as M', and according to the algorithm she will choose him over M^*. This contradicts our initial assumption. □

What proof technique did we use to prove this lemma? Is it contradiction? Or some other beast entirely? It turns out that this *also* a proof by induction! Why? Well, we began by establishing the base case of $j = k$. Next, instead of proving that for all j, $P(j) \implies P(j+1)$, we showed that the *converse* of this statement is false, i.e. that "there exists j such that $\neg(P(j) \implies P(j+1))$" does not hold; thus, by the principle of the excluded middle, it must indeed hold that for all j, $P(j) \implies P(j+1)$!

Sanity check! Ensure you understand why these two proofs are equivalent.

Let us now be a bit more careful — what exactly allowed us to take this alternate approach? The answer lies in a very special property of the natural numbers known as the *Well-Ordering Principle*. This principle intuitively says that any set of natural numbers contains a "smallest" element. Formally:

Definition 4.1 (Well-ordering principle)**.** *If $S \subseteq \mathbb{N}$ and $S \neq \emptyset$, then S has a smallest element.*

Sanity check! Consider set $S = \{5, 2, 11, 7, 8\} \subset \mathbb{N}$. Verify that the well-ordering principle holds for S.

Where did we use the well-ordering principle in our proof? In the line "Suppose ... that the jth day for $j > k$ is the first counterexample..." — specifically, if the natural numbers did not obey this principle, then the notion of a *first* counterexample would be nonsense! Note also that induction relies on this principle: The statement "for all j, $P(j) \implies P(j+1)$" only makes sense if there is a well-defined order imposed on the natural numbers (i.e. that 3 comes before 4, and 4 comes before 5, etc...). That the natural numbers obey this principle may be a fact you have long taken for granted; but there do exist sets of numbers which do *not* satisfy this property! In this sense, the natural numbers are indeed quite special.

Sanity check! Do the integers satisfy the well-ordering principle? Can you think of a set of numbers you are familiar with which *doesn't* satisfy this principle? (Hint: This set is as "real" as it gets.)

Returning to our analysis of the stable marriage algorithm. Returning to our analysis, we now prove that when the algorithm terminates, all $2n$ people are paired up. Before reading the proof, see if you can convince yourself that this is true. The proof is remarkably short and elegant, and critically uses the Improvement Lemma.

Lemma 4.3. *The stable marriage algorithm terminates with a pairing.*

Proof. We proceed by contradiction. Suppose that there is a man M who is left unpaired when the algorithm terminates. He must have proposed to all n of the women on his list. By the Improvement Lemma, each of these n women has had someone on a string since M proposed to her. Thus, when the algorithm terminates, n women have n men on a string not including M. So there must be at least $n+1$ men. But this is a contradiction, since there are only n men. □

Now, before we prove that the output of the algorithm is a *stable* pairing, let us first do a sample run-through of the stable marriage algorithm. We will use the preference lists given earlier, which are duplicated here for convenience:

Men	Women		
Alex	Anita	Bridget	Christine
Bob	Bridget	Anita	Christine
Charles	Anita	Bridget	Christine

Women	Men		
Anita	Bob	Alex	Charles
Bridget	Alex	Bob	Charles
Christine	Alex	Bob	Charles

The following table shows which men propose to which women on the given day (the circled men are the ones who were told "maybe"):

Days	Women	Proposals
1	A	①, 3
	B	②
	C	—
2	A	①
	B	②, 3
	C	—
3	A	①
	B	②
	C	③

Thus, the stable pairing which the algorithm outputs is: {(1,A), (2,B), (3,C)}.

Theorem 4.1. *The pairing produced by the algorithm is always stable.*

Proof. We give a direct proof that no man M can be involved in a rogue couple. Consider any couple (M,W) in the pairing and suppose that M prefers some woman W^* to W. We will argue that W^* prefers her partner to M, so that (M,W^*) cannot be a rogue couple. Since W^* occurs before W in M's list, he must have proposed to her before he proposed to W. Therefore, according to the algorithm, W^* must have rejected him for somebody she prefers. By the Improvement Lemma, W^* likes her final partner at least as much, and therefore prefers him to M. Thus no man M can be involved in a rogue couple, and the pairing is stable. □

5 Optimality

In Section 4.2, we showed that the stable marriage algorithm always outputs a stable pairing. But is this so impressive? Will your dating service really be successful outputting matches which are just "good enough"? Of course not! To offer the best dating service around, you require some notion of *optimality* in the solutions you obtain.

Consider, for example, the case of 4 men and 4 women with the following preference lists (for simplicity, we use numbers and letters below to label the men and women):

Men	Women			
1	A	B	C	D
2	A	D	C	B
3	A	C	B	D
4	A	B	C	D

Women	Men			
A	1	3	2	4
B	4	3	2	1
C	2	3	1	4
D	3	4	2	1

For these lists, there are exactly two stable pairings: $S = \{(1,A), (2,D), (3,C), (4,B)\}$ and $T = \{(1,A), (2,C), (3,D), (4,B)\}$. The fact that there are *two* stable pairings raises the question: What is the best possible partner for each person? For example, let us consider man 2. The trivial best partner for 2 is his first choice, woman A; unfortunately, A is just not a realistic possibility for him, as pairing him with A would not be stable, since he is so low on her preference list. Indeed, there is no stable pairing in which 2 is paired with A. Examining the two stable pairings, we find that the best possible realistic outcome for man 2 is to be matched to woman D. This demonstrates that the notion of "best partner" can be a bit fuzzy if we are not careful.

So, let us be careful; inspired by the discussion above, here is a rigorous definition of optimality.

Definition 4.2 (Optimal woman for a man)**.** *Among all stable pairings, who is the most preferred woman a man can be paired up with? This woman is the* optimal *woman for this man.*

In other words, the optimal woman is the best that a man can do in a pairing, given that the pairing is stable.

Sanity check! Who are the optimal women for each man A, B, C, and D in our example above? (Hint: The optimal woman for man 2, as discussed above, is D.)

By definition, the best that each man can hope for is to be paired with his optimal woman. But can all men achieve this optimality *simultaneously*? In other words, is there a stable pairing such that each man is paired with his optimal woman? If such a pairing exists, we will call it a *male optimal* pairing. Turning to the example above, S is a male optimal pairing since A is 1's optimal woman, D is 2's optimal woman, C is 3's optimal woman, and B is 4's optimal woman. Similarly, we can define a female optimal pairing, which is the pairing in which each woman is paired with her optimal man.

Sanity check! Check that T is a female optimal pairing.

We can also go in the opposite direction and define the *pessimal* woman for a man to be the lowest ranked woman whom he is ever paired with in some stable pairing. This leads naturally to the notion of a *male pessimal* pairing — can you define it, and also a female pessimal pairing?

Now, we get to the heart of the matter: Who is better off in the Stable Marriage Algorithm — men or women?

Theorem 4.2. *The pairing output by the Stable Marriage algorithm is male optimal.*

Proof. Suppose for sake of contradiction that the pairing is *not* male optimal. Then, there exists a day on which some man was rejected by his optimal woman; let day k be the first such day. On this day, suppose M was rejected by W^* (his optimal mate) in favor of M^*. By definition of optimal woman, there must be exist a stable pairing T in which M and W^* are paired together. Suppose T looks like this: $\{\ldots,(M,W^*),\ldots,(M^*,W'),\ldots\}$. We will argue that (M^*,W^*) is a rogue couple in T, thus contradicting stability.

First, it is clear that W^* prefers M^* to M, since she rejected M in favor of M^* during the execution of the stable marriage algorithm. Moreover, since day k was the first day when some man got rejected by his optimal woman, on day k M^* had not yet been rejected by his optimal woman. Since he proposed to W^* on the k-th day, this implies that M^* likes W^* at least as much as his optimal woman, and therefore at least as much as W'. Therefore, (M^*,W^*) form a rogue couple in T, and so T is not stable. Thus, we have a contradiction, implying the pairing is male optimal. □

What proof techniques did we use to prove this Theorem 3.2? Again, it is a proof by induction, structured as an application of the well-ordering principle.

Sanity check! Where did we use the well-ordering principle in the proof of Theorem 3.2?

Exercise. Can you rewrite the proof of Theorem 3.2 as a regular induction proof? (Hint: The proof is really showing by induction on k the following statement: For every k, no man gets rejected by his optimal woman on the kth day.)

We conclude that men fare very well by following this algorithm. How about the women? The following theorem confirms the sad truth:

Theorem 4.3. *If a pairing is male optimal, then it is also female pessimal.*

Proof. We proceed by contradiction. Let $T = \{\ldots,(M,W),\ldots\}$ be the male optimal pairing (which we know is output by the algorithm by Theorem 3.2). Suppose for the sake of contradiction that there exists a stable pairing $S = \{\ldots,(M^*,W),\ldots,(M,W'),\ldots\}$ such that M^* is lower on W's list than M (i.e., M is not her pessimal man). We will argue that S cannot be stable by showing that (M,W) is a rogue couple in S. By assumption, W prefers M to M^* since M^* is lower on her list. And M prefers W to his partner W' in S because W is his partner in the male optimal pairing T. Contradiction. Therefore, the male optimal pairing is female pessimal. □

In light of these findings, what does the stable marriage algorithm teach us? When it comes to relationships, it pays to make the first move!

Let us close with some final comments about the National Residency Matching Program. Until recently the stable marriage algorithm was run with the hospitals doing the proposing, and so the pairings produced were hospital optimal. In the nineties, the roles were reversed so that the medical students were proposing to the hospitals. More recently, there were other improvements made to the algorithm which the N.R.M.P. used. For example, the pairing takes into account preferences for married students for positions at the same or nearby hospitals.

6 Further reading (optional)

Though it was in use 10 years earlier, the propose-and-reject algorithm was first properly analyzed by Gale and Shapley, in a famous paper dating back to 1962 that still stands as one of the great achievements in the analysis of algorithms. The full reference is:

D. Gale and L.S. Shapley, "College Admissions and the Stability of Marriage," *American Mathematical Monthly* **69** (1962), pp. 9–14.

Stable marriage and its numerous variants remains an active topic of research in computer science. Although it is by now 25 years old, the following very readable book covers many of the interesting developments since Gale and Shapley's algorithm:

D. Gusfield and R.W. Irving, *The Stable Marriage Problem: Structure and Algorithms*, MIT Press, 1989.

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Note 5

1 Graph Theory: An Introduction

One of the fundamental ideas in computer science is the notion of *abstraction*: capturing the essence or the core of some complex situation by a simple model. Some of the largest and most complex entities we might deal with include the internet, the brain, maps, and social networks. In each case, there is an underlying "network" or *graph* that captures the important features that help us understand these entities more deeply. In the case of the internet, this network or graph specifies how web pages link to one another. In the case of the brain, it is the interconnection network between neurons. We can describe these ideas in the beautiful framework of *graph theory*, which is the subject of this lecture.

Remarkably, graph theory has its origins in a simple evening pastime of the residents of Königsberg, Prussia (nowadays Kaliningrad, Russia) a few centuries ago. Through their city ran the Pregel river, depicted on the left in Figure 1 below, separating Königsberg into two banks A and D and islands B and C. Connecting the islands to the mainland were seven bridges. As the residents of the city took their evening walks, many would try to solve the challenge of picking a route that would cross each of the seven bridges precisely once and return to the starting point.

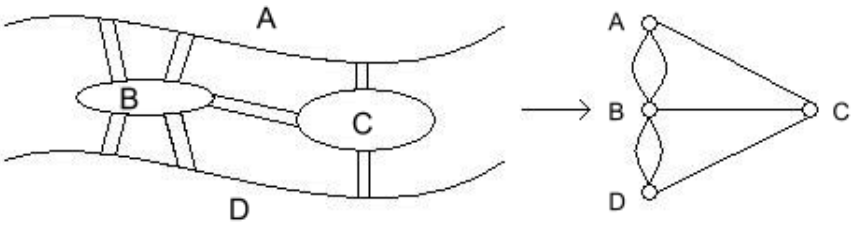


Figure 1: (Left) The city of Königsberg. (Right) The (multi-)graph modeling the bridge connections in Königsberg.

In 1736, the brilliant mathematician Leonhard Euler proved this task to be impossible. How did he do it? The key is to realize that for the purpose of choosing such a route, Figure 1a can be replaced with Figure 1b, where each land mass A, B, C, and D is replaced by a small circle, and each bridge by a line segment. With this abstraction in place, the task of choosing a route can be restated as follows: trace through all the line segments and return to the starting point without lifting the pen, and without traversing any line segment more than once. The proof of impossibility is simple. Under these tracing rules, the pen must enter each small circle as many times as it exits it, and therefore the number of line segments incident to that circle must be even. But in Figure 1b, each circle has an odd number of line segments incident to it, so it is impossible to carry out such a tracing. Actually Euler did more. He gave a precise condition under which the tracing can be carried out. For this reason, Euler is generally hailed as the inventor of graph theory.

1.1 Formal definitions

Formally, a (undirected) graph is defined by a set of vertices V and a set of edges E. The vertices correspond to the little circles in Figure 1 above, and the edges correspond to the line segments between the vertices. In Figure 1, $V = \{A,B,C,D\}$ and $E = \{\{A,B\},\{A,B\},\{A,C\},\{B,C\},\{B,D\},\{B,D\},\{C,D\}\}$. However, note that here E is a multiset (a set where an element can appear multiple times). This is because in the Königsberg example there are multiple bridges between a pair of banks. We will generally not consider such a situation of multiple edges between a single pair of vertices, so in our definition, we require E to be a set, not a multi-set. What this means is that between any pair of vertices there is either 0 or 1 edge. If there are multiple edges between a pair of vertices, then we collapse them into a single edge.

More generally, we can also define a directed graph. If an edge in an undirected graph represents a street, then an edge in a directed graph represents a one-way street. To make this formal, let V be a set denoting the vertices of a graph G. For example, we can have $V = \{1,2,3,4\}$. Then, the set of (directed) edges E is a subset of $V \times V$, i.e. $E \subseteq V \times V$. (Recall here that $U \times V$ denotes the Cartesian product of sets U and V, defined as $U \times V = \{(u,v) : u \in U \text{ and } v \in V\}$.) Continuing with our example, let $E = \{(1,2),(1,3),(1,4)\}$. Then, the corresponding graph is given by G_1 below.

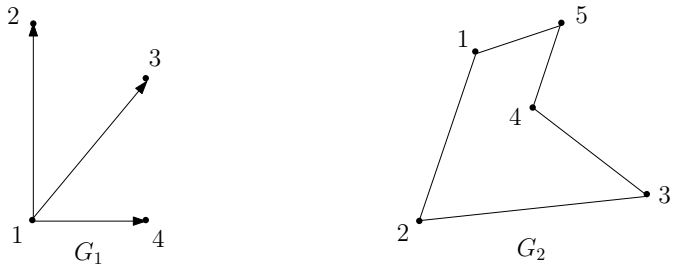


Figure 2: Examples of directed and undirected graphs, respectively.

Note that each edge in G_1 has a *direction* specified by an arrow; thus, for example, $(1,2) \in E$ but $(2,1) \notin E$. Such graphs are useful in modeling one-way relationships, such as one-way streets between two locations, and are called *directed*. On the other hand, if each edge goes in both directions, i.e., $(u,v) \in E$ iff $(v,u) \in E$, then we call the graph *undirected*. For undirected graphs we drop the ordered pair notation for edges, and simply denote the edge between u and v by the set $\{u,v\}$. Undirected graphs model relationships such as two-way streets between locations naturally, and an example is given by G_2 above. For simplicity, we omit the arrowheads when drawing edges in undirected graphs. We conclude that a graph is thus formally specified as an ordered pair $G = (V,E)$, where V is the vertex set and E is the edge set.

Sanity check! What are the vertex and edge sets V and E for graph G_2?

Let us continue our discussion with a working example from *social networks*, an area in which graph theory plays a fundamental role. Suppose you wish to model a social network in which vertices correspond to people, and edges correspond to the following relationship between people: We say Alex *recognizes* Bridget if Alex knows who Bridget is, but Bridget does not know who Alex is. If, on the other hand, Alex knows Bridget and Bridget knows Alex, then we say they *know each other*.

Sanity check! Suppose first that an edge between two people (say) Alex and Bridget means that Alex recognizes Bridget; would you use a directed or undirected graph for this? How about if an edge instead means Alex and Bridget know each other? (Answer: directed and undirected, respectively.)

Moving on with our example, we say that edge $e = \{u, v\}$ (or $e = (u, v)$) is *incident* on vertices u and v, and that u and v are *neighbors* or *adjacent*. If G is undirected, then the *degree* of vertex $u \in V$ is the number of edges incident to u, i.e., $degree(u) = |\{v \in V : \{u, v\} \in E\}|$. A vertex u whose degree is 0 is called an *isolated* vertex, since there is no edge which connects u to the rest of the graph.

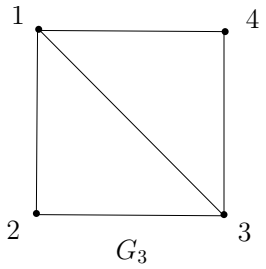
Sanity check! What does the degree of a vertex represent in our *undirected* social network in which an edge $\{u, v\}$ means u and v know each other? How should we interpret an isolated vertex?

A directed graph, on the other hand, has two different notions of degree due to the directions on the edges. Specifically, the *in-degree* of a vertex u is the number of edges from other vertices to u, and the *out-degree* of u is the number of edges from u to other vertices.

Sanity check! What do the in-degree and out-degree of a vertex represent in our *directed* social network in which an edge (u, v) means u recognizes v?

Finally, our definition of a graph thus far allows edges of the form $\{u, u\}$ (or (u, u)), i.e., a *self-loop*. In our social network, however, this gives us no interesting information (it means that person A recognizes him/herself!). Thus, here and in general in these notes, we shall assume that our graphs have no self-loops, unless stated otherwise. We shall also not allow multiple edges between a pair of vertices (unlike the case of the Seven Bridges of Königsberg).

Paths, walks, and cycles. Let $G = (V, E)$ be an undirected graph. A *path* in G is a sequence of edges $\{v_1, v_2\}, \{v_2, v_3\}, ..., \{v_{n-2}, v_{n-1}\}, \{v_{n-1}, v_n\}$. In this case we say that there is a path *between* v_1 *and* v_n. For example, suppose the graph G_3 below models a residential neighborhood in which each vertex corresponds to a house, and two houses u and v are neighbors if there exists a direct road from u to v.



G_3

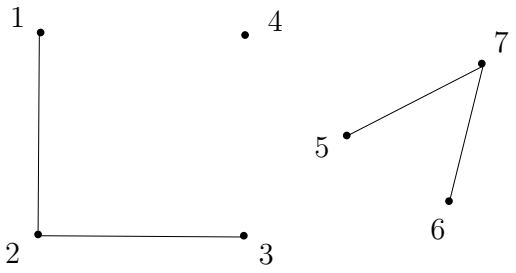
Sanity check! What is the shortest path from house 1 to house 3 in G_3? How about the longest path, assuming no house is visited twice?

Usually, we assume a path is *simple*, meaning $v_1, \ldots, v_n$ are distinct. This makes complete sense in our housing example G_3; if you wanted drive from house 1 to 3 via house 2, why would you visit house 2 more than once? A *cycle* (or *circuit*) is a sequence of edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-2}, v_{n-1}\}, \{v_{n-1}, v_n\}, \{v_n, v_1\}$, where $v_1, \ldots, v_n$ are distinct (i.e., a cycle is a path which starts and ends on the same vertex v_1).

Sanity check! Give a cycle starting at house 1 in G_3.

Suppose now that your aim is not to go from 1 to 3 as quickly as possible, but to take a leisurely stroll from 1 to 3 via the sequence $\{1,2\},\{2,1\},\{1,4\},\{4,3\}$. A sequence of edges with repeated vertices, such as this one, is called a *walk* from 1 to 3. Analogous to the relationship between paths and cycles, a *tour* is a walk which starts and ends at the same vertex. For example, $\{1,2\},\{2,3\},\{3,1\}$ is a tour.

Connectivity. Much of what we discuss in this note revolves around the notion of connectivity. A graph is said to be *connected* if there is a path between any two distinct vertices. For example, our residential network G_3 above is connected, since one can drive from any house to any other house via *some* sequence of direct roads. On the other hand, the network below is *disconnected*.



Sanity check! What would a disconnected vertex represent in our residential network? Why would you not want to design a neighborhood this way?

Note that *any* graph (even a disconnected one) always consists of a collection of *connected components*, i.e., sets $V_1,\ldots,V_k$ of vertices, such that all vertices in a set V_i are connected. For example, the graph above is not connected, but nevertheless consists of three connected components: $V_1 = \{1,2,3\}$, $V_2 = \{4\}$, and $V_3 = \{5,6,7\}$.

2 Revisiting the Seven Bridges of Koenigsberg: Eulerian Tours

With a formal underpinning in graph theory under our belts, we are ready to revisit the Seven Bridges of Königsberg. What exactly is this problem asking? It says: Given a graph G (in this case, G is an abstraction of Königsberg), is there a walk in G that uses each edge exactly once? We call any such walk in a graph an *Eulerian walk*. (In contrast, by definition a walk can normally visit each edge or vertex as many times as desired.) Moreover, if an Eulerian walk is closed, i.e., it ends at its starting point, then it is called an *Eulerian tour*. Thus, the Seven Bridges of Königsberg asks: Given a graph G, does it have an Eulerian tour? We now give a precise characterization of this in terms of simpler properties of the graph G. For this, define an *even degree* graph as a graph in which all vertices have even degree.

Theorem 5.1 (Euler's Theorem (1736))**.** *An undirected graph $G = (V,E)$ has an Eulerian tour iff G is even degree, and connected (except possibly for isolated vertices).*

Proof. To prove this, we must establish two directions: if, and only if.

Only if. We give a direct proof for the forward direction, i.e., if G has an Eulerian tour, then it is connected and has even degree. Assume that G has an Eulerian tour. This means every vertex that has an edge adjacent to it (i.e., every non-isolated vertex) must lie on the tour, and is therefore connected with all other vertices on the tour. This proves that the graph is connected (except for isolated vertices).

Next, we prove that each vertex has even degree by showing that all edges incident to a vertex can be paired up. Notice that every time the tour enters a vertex along an edge it exits along a different edge. We can pair these two edges up (they are never again traversed in the tour). The only exception is the start vertex, where the first edge leaving it cannot be paired in this way. But notice that by definition, the tour necessarily ends at the start vertex. Therefore, we can pair the first edge with the last edge entering the start vertex. So all edges adjacent to any vertex of the tour can be paired up, and therefore each vertex has even degree.

If. We give a recursive algorithm for finding an Eulerian tour, and prove by induction that it correctly outputs an Eulerian tour.

We start with a useful subroutine, FINDTOUR(G,s), which finds a tour (not necessarily Eulerian) in G. FINDTOUR is very simple: it just starts walking from a vertex $s \in V$, at each step choosing any untraversed edge incident to the current vertex, until it gets stuck because there is no more adjacent untraversed edge. We now prove that FINDTOUR must in fact get stuck at the original vertex s.

Claim: FINDTOUR(G,s) must get stuck at s.

Proof of claim: An easy proof by induction on the length of the walk shows that when FINDTOUR enters any vertex $v \neq s$, it will have traversed an odd number of edges incident to v, while when it enters s it will have traversed an even number of edges incident to s. Since every vertex in G has even degree, this means every time it enters $v \neq s$, there is at least one untraversed edge incident to v, and therefore the walk cannot get stuck. So the only vertex it can get stuck at is s. The formal proof is left as an exercise. □

The algorithm FINDTOUR(G,s) returns the tour it has traveled when it gets stuck at s. Note that while FINDTOUR(G,s) always succeeds in finding a tour, it does not always return an Eulerian tour.

We now give a recursive algorithm EULER(G,s) that outputs an Eulerian tour starting and ending at s. EULER(G,s) invokes another subroutine SPLICE$(T,T_1,\ldots,T_k)$ which takes as input a number of edge disjoint tours $T,T_1,\ldots,T_k$ ($k \geq 1$), with the condition that the tour T intersects each of the tours $T_1,\ldots,T_k$ (i.e., T shares a vertex with each of the T_i's). The procedure SPLICE$(T,T_1,\ldots,T_k)$ outputs a single tour T' that traverses all the edges in $T,T_1,\ldots,T_k$, i.e., it splices together all the tours. The combined tour T' is obtained by traversing the edges of T, and whenever it reaches a vertex s_i that intersects another tour T_i, it takes a detour to traverse T_i from s_i back to s_i again, and only then it continues traversing T.

The algorithm EULER(G,s) is given as follows:

Function EULER(G,s)
$T =$ FINDTOUR(G,s)
Let $G_1,\ldots,G_k$ be the connected components when the edges in T are removed from G, and let s_i be the first vertex in T that intersects G_i
Output SPLICE$(T,$EULER$(G_1,s_1),\ldots,$EULER$(G_k,s_k))$
end EULER

We prove by induction on the size of G that EULER(G,s) outputs an Eulerian Tour in G. The same proof works regardless of whether we think of size as number of vertices or number of edges. For concreteness, here we use number of edges m of G.

Base case: $m = 0$, which means G is empty (it has no edges), so there is no tour to find.

Induction hypothesis: EULER(G,s) outputs an Eulerian Tour in G for any even degree, connected graph with at most $m \geq 0$ edges.

Induction step: Suppose G has $m+1$ edges. Recall that $T =$ FINDTOUR(G,s) is a tour, and therefore has even degree at every vertex. When we remove the edges of T from G, we are therefore left with an even degree graph with less than m edges, but it might be disconnected. Let $G_1,\ldots,G_k$ be the connected

components. Each such connected component has even degree and is connected (up to isolated vertices). Moreover, T intersects each of the G_i, and as we traverse T there is a first vertex where it intersects G_i. Call it s_i. By the induction hypothesis $\text{EULER}(G_i, s)$ outputs an Eulerian tour of G_i. Now by the definition of SPLICE, it splices the individual tours together into one large tour whose union is all the edges of G, hence an Eulerian tour. □

Sanity check! Why does Theorem 5.1 imply the answer to the Seven Bridges of Königsberg is no?

---

3 Important classes of graphs

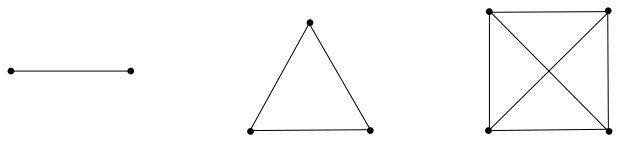
As we have seen, graphs are an abstract and general construct allowing us to represent relationships between objects, such as houses and roads. In practice, certain classes of graphs prove especially useful. For example, imagine that our graph represents the interconnection between routers on the internet. To send a packet from one node to another, we need to find a path in this graph from our source and destination. Therefore, to be able to send a packet from any node to any other node, we only need the graph to be connected. A minimally connected graph is called a *tree*, and it is the most efficient graph (i.e., with minimum number of edges) we can use to connect any set of vertices.

But now suppose the connections between some routers are not too strong, so sometimes we can lose an edge between two vertices in the graph. If our graph is a tree, then removing an edge from it results in a disconnected graph, which means there are some vertices in the graph that we cannot reach. To avoid this bad case, we want some sort of redundancy or robustness in the graph connectivity. Clearly the most connected graph is the complete graph, in which all nodes are connected to all other nodes. However, as we shall see below, the complete graph uses exponentially many edges, which makes it impractical for large-scale problems.

There is also a nice family of graphs called the hypercube graphs, which combines the best of both worlds: they are robustly connected, but do not use too many edges. In this section, we study these three classes of graphs in more detail.

3.1 Complete graphs

We start with the simplest class of graphs, the *complete* graphs. Why are such graphs called complete? Because they contain the *maximum* number of edges possible. In other words, in an undirected complete graph, every pair of (distinct) vertices u and v are connected by an edge $\{u, v\}$. For example, below we have complete graphs on $n = 2, 3, 4$ vertices, respectively.



K_2 K_3 K_4

Here, the notation K_n denotes the *unique* complete graph on n vertices. Formally, we can write $K_n = (V, E)$ for $|V| = n$ and $E = \big\{\{v_i, v_j\} \mid v_i \neq v_j \text{ and } v_i, v_j \in V\big\}$.

Sanity check!

1. Can you draw K_5, the complete graph on $n = 5$ vertices?
2. What is the degree of every vertex in K_n?

Exercise. How many edges are there in K_n? (Answer: $n(n-1)/2$.) Verify that the K_5 you drew above has this many edges.

Next, let us return to the theme of connectivity. A complete graph is special in that each vertex is neighbors with every other vertex. Thus, such a graph is very "strongly connected" in that a large number of edges must be removed before we disconnect the graph into two components. Why might this be a good property to have (say) in a communications network, where vertices correspond to mainframes, and edges correspond to communications channels?

Sanity check! What is the minimum number of edges which must be removed from K_n to obtain an isolated vertex?

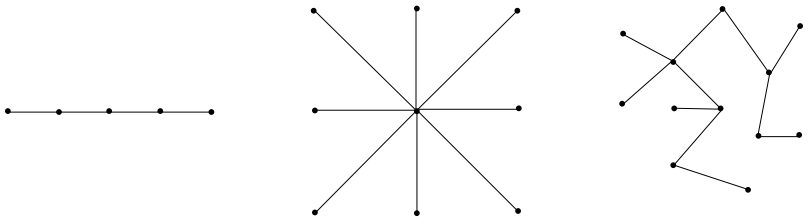
Finally, we can also discuss complete graphs for *directed* graphs, which are defined as you might expect: For any pair of vertices u and v, both $(u,v),(v,u) \in E$.

3.2 Trees

If complete graphs are "maximally connected," then trees are the opposite: Removing just a single edge disconnects the graph! Formally, there are a number of equivalent definitions of when a graph $G = (V,E)$ is a tree, including:

1. G is connected and contains no cycles.
2. G is connected and has $n-1$ edges (where $n = |V|$).
3. G is connected, and the removal of any single edge disconnects G.
4. G has no cycles, and the addition of any single edge creates a cycle.

Here are three examples of trees:

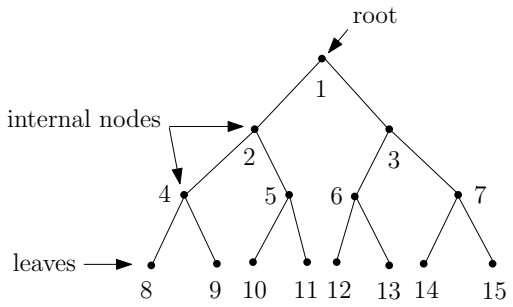


Sanity check!

1. Convince yourself that the three graphs above satisfy all four equivalent definitions of a tree.

2. Give an example of a graph which is *not* a tree.

Why would we want to study such funny-looking graphs? One reason is that many graph-theoretical problems which are computationally intractable on arbitrary graphs, such as the Maximum Cut problem, are easy to solve on trees. Another reason is that they model many types of natural relationships between objects. To demonstrate, we now introduce the concept of a *rooted* tree, an example of which is given below.



In a rooted tree, there is a designated node called the *root*, which we think of as sitting at the top of the tree. The bottom-most nodes are called *leaves*, and the intermediate nodes are called *internal nodes*. The *depth d* of the tree is the length of the longest path from the root to a leaf. Moreover, the tree can be thought of as grouped into layers or *levels*, where the k-th level for $k \in \{0, 1, \ldots, d\}$ is the set of vertices which are connected to the root via a path consisting of precisely k edges.

Sanity check!

1. What is the depth of the tree above? (Answer: 3)
2. Which vertices are on level 0 of the tree above? How about on level 3?

Where do rooted trees come in handy? Consider, for example, the setting of bacterial cell division. In this case, the root might represent a single bacterium, and each subsequent layer corresponds to cell division in which the bacterium divides into two new bacteria. Rooted trees can also be used to allow fast searching of ordered sets of elements, such as in *binary search trees*, which you may have already encountered in your studies.

One of the nice things about trees is that induction works particularly well in proving properties of trees. Let us demonstrate with a case in point: We shall prove that the first two definitions of a tree given above are indeed equivalent.

Theorem 5.2. *The statements "G is connected and contains no cycles" and "G is connected and has $n-1$ edges" are equivalent.*

Proof. We proceed by showing the forward and converse directions.

Forward direction. We prove using strong induction on n that if G is connected and contains no cycles, then G is connected and has $n-1$ edges. Assume $G = (V, E)$ is connected and contains no cycles.

Base case ($n = 1$): In this case, G is a single vertex and has no edges. Thus, the claim holds.

Inductive hypothesis: Assume the claim is true for $1 \leq n \leq k$.

Inductive proof: We show the claim for $n = k+1$. Remove an arbitrary vertex $v \in V$ from G along with its incident edges, and call the resulting graph G'. Clearly, *removing* a vertex cannot create a cycle; thus, G' contains no cycles. However, removing v may result in a *disconnected* graph G', in which case the induction hypothesis cannot be applied to G' as a whole. Thus, we have two cases to examine — either G' is connected, or G' is disconnected. Here, we show the former case, as it is simpler and captures the essential proof ideas. The latter case is left as an exercise below.

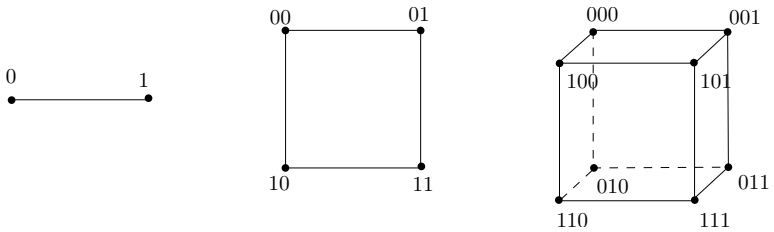
So, assume G' is connected. But now G' is a connected graph with no cycles on k vertices, so we can apply the induction hypothesis to G' to conclude that G' is connected and has $k-1$ edges. Let us now add v back to G' to obtain G. How many edges can be incident on v? Well, since G' is connected, then if v is incident on more than one edge, G will contain a cycle. But by assumption G has no cycles! Thus, v must be incident on one edge, implying G has $(k-1)+1 = k$ edges, as desired.

Converse direction. We prove using contradiction that if G is connected and has $n-1$ edges, then G is connected and contains no cycles. Assume G is connected, has $n-1$ edges, and contains a cycle. Then, by definition of a cycle, removing any edge in the cycle does not disconnect the graph G. In other words, we can remove an edge in the cycle to obtain a new connected graph G' consisting of $n-2$ edges. However, we claim that G' must be disconnected, which will yield our desired contradiction. This is because in order for a graph to be connected, it must have at least $n-1$ edges. This is a fact that you have to prove in the exercise below. This completes the proof of the converse direction. □

3.3 Hypercubes

We have discussed how complete graphs are a class of graphs whose vertices are particularly "well-connected." However, to achieve this strong connectivity, a large number of edges is required, which in many applications of graph theory is infeasible. Consider the example of the *Connection Machine*, which was a massively parallel computer by the company Thinking Machines in the 1980s. The idea of the Connection Machine was to have a million processors working in parallel, all connected via a communications network. If you were to connect each pair of such processors with a direct wire to allow them to communicate (i.e., if you used a complete graph to model your communications network), this would require 10^{12} wires! What the builders of the Connection Machine thus decided was to instead use a 20-dimensional *hypercube* to model their network, which still allowed a strong level of connectivity, while limiting the number of neighbors of each processor in the network to 20. This section is devoted to studying this particularly useful class of graphs, known as hypercubes.

The vertex set of the n-dimensional hypercube $G = (V,E)$ is given by $V = \{0,1\}^n$, where recall $\{0,1\}^n$ denotes the set of all n-bit strings. In other words, each vertex is labeled by a unique n-bit string, such as $00110\cdots0100$. The edge set E is defined as follows: Two vertices x and y are connected by edge $\{x,y\}$ if and only if x and y differ in exactly one bit position. For example, $x = 0000$ and $y = 1000$ are neighbors, but $x = 0000$ and $y = 0011$ are not. More formally, $x = x_1x_2\ldots x_n$ and $y = y_1y_2\ldots y_n$ are neighbors if and only if there is an $i \in \{1,\ldots,n\}$ such that $x_j = y_j$ for all $j \neq i$, and $x_i \neq y_i$. To help you visualize the hypercube, we depict the 1-, 2-, and 3-dimensional hypercubes below.



There is an alternative and useful way to define the n-dimensional hypercube via recursion, which we now discuss. Define the 0-subcube (respectively, 1-subcube) as the $(n-1)$-dimensional hypercube with vertices labeled by $0x$ for $x \in \{0,1\}^{n-1}$ (respectively, $1x$ for $x \in \{0,1\}^{n-1}$). Then, the n-dimensional hypercube is obtained by placing an edge between each pair of vertices $0x$ in the 0-subcube and $1x$ in the 1-subcube.

Sanity check! Where are the 0- and 1-subcubes in the 3-dimensional hypercube depicted above? Can you use these along with the recursive definition above to draw the 4-dimensional hypercube?

Exercise. Prove that the n-dimensional hypercube has 2^n vertices. Hint: Use the fact that each bit has two possible settings, 0 or 1.

We began this section by singing praises for the hypercube in terms of its connectivity properties; we now investigate these claims formally. Let us begin by giving two proofs of a simple property of the hypercube. Each proof relies on one of our two equivalent (namely, direct and recursive) definitions of the hypercube.

Lemma 5.1. *The total number of edges in an n-dimensional hypercube is $n2^{n-1}$.*

Proof 1. The degree of each vertex is n, since n bit positions can be flipped in any $x \in \{0,1\}^n$. Since each edge is counted twice, once from each endpoint, this yields a total of $n2^n/2 = n2^{n-1}$ edges. □

Proof 2. By the second definition of the hypercube, it follows that $E(n) = 2E(n-1) + 2^{n-1}$, and $E(1) = 1$, where $E(n)$ denotes the number of edges in the n-dimensional hypercube. A straightforward induction shows that $E(n) = n2^{n-1}$. □

Exercise. Using induction to show that in Proof 2 above, $E(n) = n2^{n-1}$.

Let us focus on the question of connectivity, and prove that the n-dimensional hypercube is well-connected in the following sense: To disconnect any subset $S \subseteq V$ of vertices from the rest of the graph, a large number of edges must be discarded. In particular, we shall see that the number of discarded edges must scale with $|S|$. In the theorem below, recall that $V - S = \{v \in V : v \notin S\}$ is the set of vertices that are not in S.

Theorem 5.3. *Let $S \subseteq V$ be such that $|S| \leq |V - S|$ (i.e., that $|S| \leq 2^{n-1}$), and let E_S denote the set of edges connecting S to $V - S$, i.e.,*

$$E_S := \{\{u,v\} \in E \mid u \in S \text{ and } v \in V - S\}.$$

Then, it holds that $|E_S| \geq |S|$.

Proof. We proceed by induction on n.

Base case ($n = 1$): The 1-dimensional hypercube graph has two vertices 0 and 1, and one edge $\{0,1\}$. We also have the assumption $|S| \leq 2^{1-1} = 1$, so there are two possibilities. First, if $|S| = 0$, then the claim trivially holds. Otherwise, if $|S| = 1$, then $S = \{0\}$ and $V - S = \{1\}$, or vice versa. In either case we have $E_S = \{0,1\}$, so $|E_S| = 1 = |S|$.

Inductive hypothesis: Assume the claim holds for $1 \leq n \leq k$.

Inductive step: We prove the claim for $n = k+1$. Recall that we have the assumption $|S| \leq 2^k$. Let S_0 (respectively, S_1) be the vertices from the 0-subcube (respectively, 1-subcube) in S. We have two cases to examine: Either S has a fairly equal intersection size with the 0- and 1-subcubes, or it does not.

1. **Case 1:** $|S_0| \leq 2^{k-1}$ **and** $|S_1| \leq 2^{k-1}$

   In this case, we can apply the induction hypothesis separately to the 0- and 1-subcubes. This says that restricted to the 0-subcube itself, there are at least $|S_0|$ edges between $|S_0|$ and its complement (in the 0-subcube), and similarly there are at least $|S_1|$ edges between $|S_1|$ and its complement (in the 1-subcube). Thus, the total number of edges between S and $V - S$ is at least $|S_0| + |S_1| = |S|$, as desired.

2. **Case 2:** $|S_0| > 2^{k-1}$

   In this case, S_0 is unfortunately too large for the induction hypothesis to apply. However, note that since $|S| \leq 2^k$, we have $|S_1| = |S| - |S_0| \leq 2^{k-1}$, so we *can* apply the hypothesis to S_1. As in Case 1, this allows us to conclude that there are at least $|S_1|$ edges in the 1-subcube crossing between S and $V - S$.

   What about the 0-subcube? Here, we cannot apply the induction hypothesis directly, but there is a way to apply it after a little massaging. Consider the set $V_0 - S_0$, where V_0 is the set of vertices in the 0-subcube. Note that $|V_0| = 2^k$ and $|V_0 - S_0| = |V_0| - |S_0| = 2^k - |S_0| < 2^k - 2^{k-1} = 2^{k-1}$. Thus, we *can* apply the inductive hypothesis to the set $V_0 - S_0$. This yields that the number of edges between S_0 and $V_0 - S_0$ is at least $2^k - |S_0|$. Adding our totals for the 0-subcube and the 1-subcube so far, we conclude there are at least $2^k - |S_0| + |S_1|$ crossing edges between S and $V - S$. However, recall our goal was to show that the number of crossing edges is at least $|S|$; thus, we are still short of where we wish to be.

   But there are a still edges we have not accounted for — namely, those in E_S which cross between the 0- and 1-subcubes. Since there is an edge between every vertex of the form $0x$ and the corresponding vertex $1x$, we conclude there are at least $|S_0| - |S_1|$ edges in E_S that cross between the two subcubes. Thus, the total number of edges crossing is at least $2^k - |S_0| + |S_1| + |S_0| - |S_1| = 2^k \geq |S|$, as desired.

□

4 Practice Problems

1. A *de Bruijn sequence* is a 2^n-bit circular sequence such that every string of length n occurs as a contiguous substring of the sequence exactly once. For example, the following is a de Bruijn sequence for the case $n = 3$:

   Notice that there are eight substrings of length three, each of which corresponds to a binary number from 0 to 7 such as 000, 001, 010, etc. It turns out that such sequences can be generated from the *de*

Bruijn graph, which is a directed graph $G = (V, E)$ on the vertex set $V = \{0,1\}^{n-1}$, i.e., the set of all $n-1$ bit strings. Each vertex $a_1a_2...a_{n-1} \in V$ has two outgoing edges:

$$(a_1a_2...a_{n-1},\ a_2a_3...a_{n-1}0) \in E \quad \text{and} \quad (a_1a_2...a_{n-1},\ a_2a_3...a_{n-1}1) \in E.$$

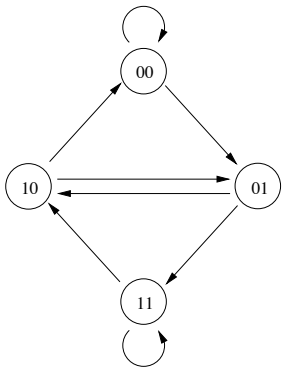
Therefore, each vertex also has two incoming edges:

$$(0a_1a_2...a_{n-2},\ a_1a_2...a_{n-1}) \in E \quad \text{and} \quad (1a_1a_2...a_{n-2},\ a_1a_2...a_{n-1}) \in E.$$

For example, for $n = 4$, the vertex 110 has two outgoing edges directed toward 100 and 101, and two incoming edges from 011 and 111. Note that these are directed edges, and self-loops are permitted.

The de Bruijn sequence is generated by an Eulerian tour in the de Bruijn graph. Euler's theorem (Theorem 5.1) can be modified to work for directed graphs — all we need to modify is the second condition, which should now say: "For every vertex v in V, the in-degree of v equals the out-degree of v." Clearly, the de Bruijn graph satisfies this condition, and therefore it has an Eulerian tour.

To actually generate the sequence, starting from any vertex, we walk along the tour and add the corresponding bit which was shifted in from the right as we traverse each edge. Here is the de Bruijn graph for $n = 3$.



Find the Eulerian tour of this graph that generates the de Bruijn sequence given above.

2. In this question, we complete the induction component of the proof of Theorem 5.2.

   (a) Suppose in the proof that G' has two distinct connected components G'_1 and G'_2. Complete the inductive step to show that G is connected and has $n-1$ edges. (Hint: Argue that you can apply the induction hypothesis to G'_1 and G'_2 separately. Note that this requires strong induction!)

   (b) More generally, G' may have $t \geq 2$ distinct connected components G'_1 through G'_t — generalize your argument above to this setting in order to complete the proof of Theorem 5.2.

Life lesson. This question teaches you a general paradigm for solving problems, be it in computer science research or everyday life. Specifically, when faced with a difficult problem (such as the proof of Theorem 5.2), first try to solve it in the simplest case possible (such as when G' is connected). Then, gradually extend your solution to handle more difficult cases until you establish the general claim (i.e., G' has t connected components).

---

3. Prove using induction on the number of vertices n that any connected graph must have at least $n-1$ edges.

1 Modular Arithmetic

Suppose you go to bed at 23:00 o'clock and want to get 8 hours of sleep. What time should you set your alarm for? Clearly, the answer is not $(23+8=)$ 31:00 o'clock, but rather $(31-24=)$ 7:00 o'clock. This is because in the 24-hour clock system the numbers "wrap around" back to 0:00 once we hit 24:00, so the naive answer 31:00 means the same thing as the correct answer 7:00, namely, 7 hours after midnight.

The clock system is an example of a very useful type of arithmetic known as *modular arithmetic*, in which we perform all arithmetic operations relative to a fixed number n called the *modulus*. In the 24-hour clock system the modulus is $n = 24$, and we can write the example above as

$$31 \pmod{24} \;=\; 7$$

which is read "31 *modulo* 24 is equal to 7." Here the function "$(\text{mod } 24)$" is another name for remainder when divided by 24. So $31 \pmod{24}$ is equal to 7 because $31 = 1 \cdot 24 + 7$ and $0 \leq 7 \leq 23$.

There are many other ways we use modular arithmetic in everyday life. As another example, consider the 7 days in a week. Suppose instead of calling them $\{\text{Monday, Tuesday, } \ldots, \text{ Sunday}\}$, we label them with $\{0, 1, \ldots, 6\}$. Then we can describe the 7-day system with arithmetic modulo $n = 7$. For instance, the sentence "3 days after Sunday is Wednesday" can be translated into the statement

$$6+3 \pmod{7} \;=\; 2$$

and this holds because $6+3=9$ and when we divide 9 by 7 we get a remainder of 2. You can also see this by imagining the numbers wrapped around a circle where 7 is identified with 0 (see Figure 1), so adding or subtracting numbers correspond to walking around the circle forward or backward.

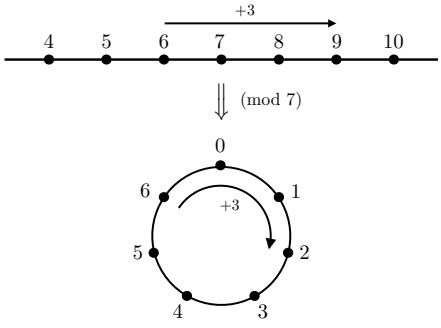


Figure 1: Addition by 3 on the integers (top) and modulo 7 (bottom).

Convince yourself that you can do the same with the 30 days in a month, 12 months in a year, etc.

So far these are simple examples from everyday life. But modular arithmetic is a very useful tool that lies at the heart of many powerful results in computer science, such as error-correcting codes and cryptography, including public-key cryptosystem such as RSA. To understand these results, let us first familiarize ourselves with modular arithmetic and explore its properties.

1.1 Addition, Subtraction, Multiplication

One of the most important properties of numbers modulo n is that the familiar arithmetic operations give robust results. The best way to illustrate this is by example. Suppose we wish to compute $42+35 \pmod{24}$. Here are two different ways of doing so. One way is to first compute the addition, then reduce $\pmod{24}$:

$$42+35 \pmod{24} \;=\; 77 \pmod{24} \;=\; 77-72 \;=\; 5.$$

The second way is to first apply $\pmod{24}$ to each term, add the results, and finally apply $\pmod{24}$ again:

$$42+35 \pmod{24} \;=\; [42 \pmod{24}]+[35 \pmod{24}] \pmod{24} \;=\; 18+11 \pmod{24} \;=\; 29 \pmod{24} \;=\; 5.$$

You see that we still get the same answer as before. This is no coincidence. We can perform any sequence of arithmetic operations $\pmod{24}$ and the result is robust — it remains unchanged whether we reduce $\pmod{24}$ only once at the end of all operations, or we reduce each intermediate result $\pmod{24}$, or even if we reduce some intermediate results $\pmod{24}$ at whim as long as we reduce the final answer $\pmod{24}$. Why is this the case? To see this we must view numbers modulo n in a completely different way.

For two integers a and b, let us define a to be *congruent* to b modulo n, written

$$a \;\equiv_n\; b$$

if and only if $a \pmod{n} = b \pmod{n}$, i.e., a and b give the same remainder when divided by n. In the above scenario, this captures the property that a and b would result in the same answer when used in the arithmetic operations. Here is another way of saying the same thing:

$$a \equiv_n b \;\;\Leftrightarrow\;\; n \,|\, (a-b)$$

(recall that $p \,|\, q$ means p divides q, i.e., q/p is an integer). Equivalently, we can write $a = b+k\cdot n$ for some integer k. Make sure you understand and can show that these conditions are all saying the same thing.

So when $n = 24$, all these numbers are congruent to each other:

$$\ldots \;\equiv_{24}\; -48 \;\equiv_{24}\; -24 \;\equiv_{24}\; 0 \;\equiv_{24}\; 24 \;\equiv_{24}\; 48 \;\equiv_{24}\; \ldots$$

Similarly,

$$\ldots \;\equiv_{24}\; -47 \;\equiv_{24}\; -23 \;\equiv_{24}\; 1 \;\equiv_{24}\; 25 \;\equiv_{24}\; 49 \;\equiv_{24}\; \cdots$$

and so on. Indeed, you can see that there are 24 such sets where the numbers within each set are all congruent to each other.

The reason that arithmetic modulo n is robust is that all the arithmetic operations — plus, minus, times — respect these sets. This is proved in the following theorem:

Theorem 6.1. *For all $n \geq 1$ and $a,b,c,d \in \mathbb{Z}$, the following are true:*

1. *If $a \equiv_n b$ and $c \equiv_n d$, then $a+c \equiv_n b+d$*
2. *If $a \equiv_n b$ and $c \equiv_n d$, then $a\cdot c \equiv_n b\cdot d$*

Proof. We prove part (1). The proof of (2) is left as an exercise.

Since $a \equiv_n b$, there exists $k \in \mathbb{Z}$ such that $a = b+k\cdot n$. Similarly, since $c \equiv_n d$, there exists $\ell \in \mathbb{Z}$ such that $c = d+\ell\cdot n$. Then we can write

$$a+c = (b+d) + (k+\ell)\cdot n$$

which means $(a+c)-(b+d)$ is divisible by n. This shows that $a+c \equiv_n b+d$. □

Remark: In our notation so far, $(\text{mod } n)$ is a function that maps an integer a to an element in the set $\{0,1,\ldots,n-1\}$ by computing the remainder when dividing a by n. The notation $\equiv_n$ says that two integers give the same remainder when divided by n. These two concepts are related, but quite different from each other. Once we get used to the two concepts, it is customary to blur the distinction between them and use the notation $(\text{mod } n)$ interchangeably: to denote the remainder, but also to denote equivalence as in

$$a \equiv b \,(\text{mod } n)$$

to denote $a \equiv_n b$.

2 Division and Multiplicative Inverses

In Section 1.1 we discussed addition, subtraction, and multiplication in the setting of modular arithmetic. Curiously, we left out division. The reason is that division is a bit more tricky in this setting, and thus warrants its own discussion. To begin, let's recall how division is done over the rational numbers $\mathbb{Q}$. Here, if we want to divide $x \in \mathbb{Q}$ by $y \in \mathbb{Q}$, we can equivalently multiply by y^{-1}, where if $y = a/b$, then $y^{-1} = b/a$. The term y^{-1} has a special name — it is a *multiplicative inverse* of y, i.e. a number such that $y \cdot y^{-1} = 1$. This suggests the following approach for dividing number x by y in modular arithmetic: Simply multiply x by y^{-1}.

But what is the multiplicative inverse of y modulo n? Is it unique as in the case of working over the rationals? Even more troubling — is it clear the inverse even *exists*? Remarkably, both scenarios are possible in modular arithmetic: Sometimes the inverse doesn't exist, and sometimes it does. When it *does* exist, however, it turns out to be unique. Let's explore these ideas further.

Consider the case of $x = 8$. Does x have a multiplicative inverse modulo $n = 15$? Yes! Note that $2x \equiv 16 \equiv 1 \,(\text{mod } 15)$. Thus, 2 is a multiplicative inverse of x modulo 15.

Sanity check! Consider now $x = 12$ and $n = 15$. Does x have a multiplicative inverse modulo m? (Hint: Since we are working modulo 15, there are only 15 possible choices for the inverse. Try plugging in $a = 0,1,2,\ldots$ into $ax \,(\text{mod } n)$ and seeing if you get the answer 1. Do you see a pattern forming?)

So sometimes the inverse exists, and sometimes it does not. Is there a way to distinguish between the two cases? Yes — it turns out that x has a multiplicative inverse modulo n if and only if the greatest common divisor of n and x is 1. Here, the *greatest common divisor* of two natural numbers x and y, denoted $\gcd(x,y)$, is the largest natural number that divides them both. For example, $\gcd(30,24) = 6$. If $\gcd(x,y) = 1$, it means that x and y share no common factors (except 1); in this case we call x and y *relatively prime*.

Sanity check! What is $\gcd(21,49)$? How about $\gcd(12,13)$?

We can show that the multiplicative inverse exists precisely when the number x is relatively prime to the modulus n. The proof is left as an exercise.

Theorem 6.2. *Let n,x be positive integers. Then x has a multiplicative inverse modulo n if and only if $\gcd(n,x) = 1$. Moreover, if it exists, then the multiplicative inverse is unique.*

2.1 Computing the Multiplicative Inverse

Theorem 6.2 gives us a sufficient condition for determining if a multiplicative inverse of x modulo n *exists*. But it doesn't tell us how to *find* such an inverse! In this section, we discuss an efficient algorithm for computing the inverse itself whenever it exists. Interestingly, this task is closely related to computing the greatest common divisor of two numbers, and both go via variants of *Euclid's algorithm*. It is worth noting that the latter is one of the oldest algorithms still in use, dating back to around 300 BC!

To begin, let us see how computing the multiplicative inverse of x modulo m is related to finding $\gcd(x,m)$. Suppose that for any pair of numbers x,y, we can not only compute $\gcd(x,y)$, but also a pair of integers a,b satisfying

$$d = \gcd(x,y) = ax + by. \tag{1}$$

(Note that this is not a modular equation, and the integers a,b can be zero or negative.) For example, we can write $1 = \gcd(35,12) = -1 \cdot 35 + 3 \cdot 12$, so here $a = -1$ and $b = 3$ are possible values for a,b.

If we can do this, then we can compute the multiplicative inverse of x modulo n as follows. First, compute integers a and b such that

$$1 = \gcd(n,x) = an + bx.$$

This implies, however, that $bx \equiv 1 \pmod{n}$. Thus, b must be the multiplicative inverse of x modulo n! Reducing b modulo n hence gives us the *unique* inverse we are looking for.

Sanity check! In the example $1 = \gcd(35,12) = -1 \cdot 35 + 3 \cdot 12$, what is the unique multiplicative inverse 12^{-1} of 12? Verify your answer by explicitly checking that indeed $12 \cdot 12^{-1} \equiv 1 \pmod{35}$.

We conclude that we have reduced the problem of computing inverses to the problem of finding integers a,b satisfying Equation (1). Remarkably, Euclid's algorithm for computing greatest common divisor *also* allows us to find the integers a and b described above, which we shall see now.

2.1.1 Euclid's Algorithm

We begin by describing the "basic" version of Euclid's algorithm, which only computes the gcd, and not the integers a and b as described in Equation (1). In the next section, we describe the *extended* version of Euclid's algorithm, which completes both tasks, as promised.

To begin, suppose we wish to compute the gcd of two numbers x and y. An easy "base case" is when either x or y is 0; for example, if $x = 0$, then $\gcd(x,y) = y$, since 0 is divisible by everything. In a nutshell, Euclid's algorithm works by reducing the general case of computing $\gcd(x,y)$ down to this base case. To accomplish this, we need the following theorem that allows us to reduce the size of the integers that we are working with, thereby getting us closer to the base case.

Theorem 6.3. *Let $x \geq y$ and let q,r be natural numbers such $x = yq + r$ and $r < y$. Then* $\gcd(x,y) = \gcd(y,r)$.

Proof. We give a direct proof. Note that the claim will follow if we can show that any common divisor d of x and y is also a common divisor of r and y, and vice versa. To show the forward direction, suppose $d \mid x$ and $d \mid y$. This means $z = x/d$ and $w = y/d$ are integers. Then we see that

$$r = x - yq = zd - wdq = (z - wq)d$$

implying $d \mid r$, as desired. The converse direction follows similarly. □

Exercise. Show the converse direction of the proof of Theorem 6.3, i.e., show that $d \mid y$ and $d \mid r$ imply $d \mid x$.

Given this theorem, let's see how to compute $\gcd(16,10)$. We begin by writing

$$16 = 10 \times 1 + 6.$$

Here $x = 16$, $y = 10$, $q = 1$, and $r = 6$. By Theorem 6.3, we know that $\gcd(16,10) = \gcd(10,6)$. Therefore, we can simplify the problem by now focusing on $\gcd(10,6)$, and writing

$$10 = 6 \times 1 + 4.$$

As before, Theorem 6.3 says $\gcd(10,6) = \gcd(6,4)$. You can probably see a pattern forming now. We apply the same idea on $\gcd(6,4)$, and so forth, obtaining the sequence of equations:

$$\begin{aligned} 6 &= 4 \times 1 + 2 \\ 4 &= 2 \times 2 + 0 \\ 2 &= 0 \times 0 + 2. \end{aligned}$$

Note that the last line is precisely the base case we discussed at the start of this section — we are looking for $\gcd(2,0)$, which is trivially 2. We conclude that $\gcd(16,10) = 2$, as desired.

Sanity check! Apply the algorithm sketched above to compute $\gcd(24,12)$.

Now that you've had a chance to get your hands dirty by trying out the algorithm, let's describe it formally using recursion as follows.

```
// precondition:  Assumes x ≥ y ≥ 0 and x > 0.
// postcondition:  Outputs gcd(x,y).
algorithm gcd(x,y)
  if y = 0 then return x
  else return gcd(y, x (mod y))
```

You should run through a second quick example to convince yourself that this formal description really does capture the algorithm we sketched previously.

Sanity check! Compute $\gcd(32,10)$ using the formal algorithm above.

2.1.2 Extended Euclid's Algorithm

We finally give an extended version of Euclid's algorithm which computes $\gcd(x,y)$, along with integers a and b satisfying $\gcd(x,y) = ax + by$, allowing us to finally compute multiplicative inverses. The algorithm is as follows.

```
//precondition: Assumes x ≥ y ≥ 0 and x > 0.
```

```
//postcondition: Outputs (d,a,b) where d = gcd(x,y) and a,b ∈ ℤ with d = ax+by.
algorithm extended-gcd(x,y)
  if y = 0 then return (x,1,0)
  else
    let (d,a,b) := extended-gcd(y, x (mod y))
    return (d, b, a − ⌊x/y⌋b)
```

Here the *floor* $\lfloor z \rfloor$ is the largest integer c such that $c \leq z$.

Note that the extended gcd algorithm has the same form as the basic gcd algorithm we saw earlier; the only difference is that we now also carry around the values a, b. Let's make sure you believe this algorithm does what we claim via an example.

Sanity check! Run extended-gcd(x,y) on inputs $x = 16$ and $y = 10$. Check that the resulting values of a and b indeed satisfy the postcondition that $d = ax + by$.

Finally, to understand *why* this algorithm works, let us prove its correctness!

Theorem 6.4. *If x and y satisfy the preconditions of extended-gcd, then the output (d,a,b) of extended-gcd(x,y) satisfy its postconditions.*

Proof. We proceed by strong induction on y. The base case is $y = 0$, in which case the algorithm returns $(d,a,b) = (x,1,0)$. This is correct since $x = \gcd(x,0)$ and $x = 1 \cdot x + 0 \cdot y$, as desired. Assume now the claim is true for all $0 < y \leq k$. We prove the claim for $y = k+1$. Let x and y be the input to extended-gcd. Then by the induction hypothesis, the first line in the `else` clause returns (d,a,b) such that $d = \gcd(y, x \pmod y)$ and $d = ay + b(x \pmod y)$. Thus, our job is to check that the last line of the algorithm returns values satisfying the postcondition, i.e., that $d = \gcd(x,y)$ and that $d = bx + (a - \lfloor x/y \rfloor b)y$. The first of these is correct by Theorem 6.3. To see the second, note that we can write $x \pmod y = x - \lfloor x/y \rfloor y$ (check this!), so

$$d = ay + b(x \pmod y) = ay + b(x - \lfloor x/y \rfloor y) = bx + (a - \lfloor x/y \rfloor b)y.$$

This concludes the proof. □

3 Practice Problems

1. Complete the proof of Theorem 6.1.
2. Prove Theorem 6.2.
3. Prove that the product of any $k \geq 1$ consecutive integers is divisible by k.

CS 70 Discrete Mathematics and Probability Theory
Spring 2015 Vazirani Note 7

1 Bijections

The notion of a mathematical *function*, i.e. a mapping f from an input set A to an output set B, is ubiquitous in our everyday lives. For example, your professor might assign you a letter grade of A through F based on a function which maps your numerical grade, i.e. an integer from 0 to 100, to the set {"A", "B", "C", "D", "F"}. Or consider the process of paying federal taxes, in which your income level in dollars is mapped via a function to a tax bracket which dictates the percentage of tax you must pay. Both of these are examples of functions, which most generally can be described using the notation $f : A \mapsto B$. Here, A is a set called the *domain* of f, and B is a set known as the *range* of f. Of all the possible functions in the wild, there is a special type which turns out to be particularly useful in computer science, specifically in the study of cryptography — namely, the notion of a *bijective* function. It is this class of functions which we study in this lecture.

Sanity check! Consider the "birthday function" f, which given a person's birth date, outputs the age of that person. What are the domain A and range B of f?

Intuitively, a *bijection* is a function with the property that any output of the function can be uniquely mapped back to some input. More formally, a function $f : A \mapsto B$ is a bijection iff for all $b \in B$, there exists a unique *pre-image* $a \in A$ such that $f(a) = b$. Let us demonstrate with the following example of a function $f : \{0, \ldots, m-1\} \mapsto \{0, \ldots, m-1\}$:

$$f(x) = x + 1 \bmod m.$$

Here, f is a bijection since the unique pre-image of any $y \in \{0, \ldots, m-1\}$ is $y - 1$. The special case of $m = 4$ is depicted in Figure 1; note the special property that each element on the right side has precisely one matching element on the left side.

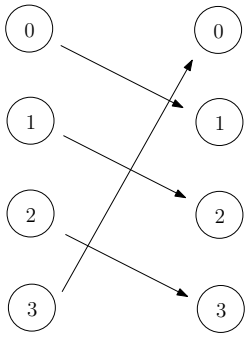


Figure 1: The bijection $f(x) = x + 1 \bmod 4$ with domain $A = \{0, \ldots, 3\}$ and range $B = \{0, \ldots, 3\}$.

Sanity check! Consider the same function $f(x) = x + 1 \bmod 4$, except now with $A = \{0, \ldots, 4\}$ and range $B = \{0, \ldots, 3\}$. Is f still a bijection? Why not? (Give two elements of A which map to the same element of B.)

If you stare at Figure 1 long enough, you might realize that the property of being a bijection is actually equivalent to having two separate properties: (1) Each element on the right has *some* pre-image on the left, and (2) if two arrows are incident on an element on the right, then both arrows must originate from the same element on the left. More formally, these properties can be stated as:

1. f is *onto* or *surjective*: Any $b \in B$ has a pre-image in A, i.e. for all $b \in B$ there exists an $a \in A$ such that $f(a) = b$.
2. f is *one-to-one* or *injective*: For all $a, a' \in A$, if $f(a) = f(a')$ then $a = a'$.

To help solidify our understanding of bijections, let us consider two more functions. First, recall the example given at the start of this lecture about the function mapping a numerical grade in the set $\{0, 1, \ldots, 100\}$ to a letter grade $\{$"*A*", "*B*", "*C*", "*D*", "*F*"$\}$. Is this a bijection? *No*, since each letter grade has many numerical grades mapped to it. (In fact, it is worth noting that there *cannot* exist a bijection between these two sets, since they have different sizes. This connection between bijections and set sizes is the key to Cantor's celebrated idea that there is more than one notion of infinity!) Next, consider the following function g mapping $\{0, \ldots, m-1\}$ to itself:

$$g(x) = 2x \bmod m. \tag{1}$$

It turns out that g is only a bijection if m is odd.

Sanity check!

1. Show that for the value $m = 4$, the function g is not a bijection since it is not one-to-one. Can you generalize your proof to arbitrary m?
2. Draw a diagram analogous to Figure 1 to show that for $m = 5$, the function g is a bijection.

A nice alternate way of thinking about bijectivity is the following, which will prove useful in our discussion on cryptography in this lecture.

Lemma 7.1. *A function $f : A \to A$ is a bijection iff there is an inverse function $g : A \to A$ such that $g(f(x)) = x$ and $f(g(y)) = y$ for all $x, y \in A$.*

Proof. We give a direct proof. First assume there exists an inverse function g as described in the claim. Then, to see that f is one-to-one, note that whenever $f(x) = f(x')$, we have that $x = g(f(x)) = g(f(x')) = x'$, as desired. To see that f is onto, note that for any $y \in A$ we have $f(g(y)) = y$, so $g(y) \in A$ is the pre-image of y.

Conversely, assume f is bijective. This means every $y \in A$ has a unique pre-image $x \in A$ such that $f(x) = y$. Define $g \colon A \to A$ by $g(y) = x$. Then we see that by construction, we have $f(g(y)) = y$, and similarly, $g(f(x)) = g(y) = x$. This shows that g is the inverse function of f, as desired. □

Sanity check! Show that the function g from Equation (2) is a bijection for the case $m = 5$ by explicitly describing an inverse function as in Lemma 7.1.

2 Fermat's Little Theorem

In 1640, Pierre de Fermat stated one of the fundamental results of elementary number theory, nowadays known as *Fermat's Little Theorem*. Ultimately, our goal in this lecture is to study the use of bijections in cryptography, and it turns out Fermat's Little Theorem will be a crucial tool in this venture. Thus, we state and prove it now. Its proof also uses the concept of a bijection.

Theorem 7.1. [Fermat's Little Theorem] *For any prime p and any $a \in \{1,2,\ldots,p-1\}$, we have $a^{p-1} \equiv 1 \bmod p$.*

Proof. Let $S = \{1,2,\ldots,p-1\}$. We give a direct proof consisting of two parts. First, we show that the function $f : S \to S$ such that $f(x) \equiv ax \bmod p$ is a bijection. Second, we use the fact that f is a bijection to prove the claim.

Sanity check! Verify that for $a = 3, p = 7$, the function $f(x) \equiv ax \bmod p$ is a bijection corresponding to the image below.

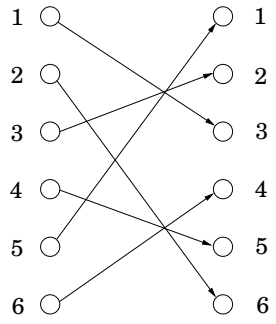


Figure 2: Multiplication by (3 mod 7)

To prove that f is indeed a bijection, since the domain and range of f are the same set S, it suffices to prove that $ax \bmod p$ and $ax' \bmod p$ are distinct if $x \neq x'$. To prove the latter, assume for sake of contradiction that there exist distinct $x, x' \in S$ such that

$$a \cdot x \equiv a \cdot x' \pmod{p}.$$

Then, since by definition a is non-zero, and since a and p are co-prime, then by Theorem 6.2, we know a has a multiplicative inverse a^{-1} modulo p. Multiplying both sides of the equation above by a^{-1} thus yields $x \equiv x' \pmod{p}$, which contradicts our assumption that x and x' are distinct. We conclude that f is a bijection.

Let us now prove our main claim. Since f is a bijection, its image is S; this implies that the product of all elements in S equals the product of all elements in the image of f. Specifically, we have

$$(p-1)! \equiv a^{p-1} \cdot (p-1)! \pmod{p}.$$

Since $(p-1)!$ is relatively prime to p, it again follows by Theorem 6.2 that we can divide both sides of this equation by $(p-1)!$, yielding the desired statement. □

3 Public Key Encryption and RSA

One of the oldest and most fundamental problems facing mankind has been: How to send a message securely from a sender, say Alice, to a receiver Bob, so that an eavesdropper Eve gains little to no information about the message? As you might imagine, this task has a host of applications in areas ranging from banking to the military. In fact, Julius Caesar himself was known to use an encryption scheme nowadays known as the *Caesar cipher* in order to protect messages of military significance. Unfortunately, Caesar's cipher had a significant drawback: In order to encode and decode messages, one needed to know a *secret key*. This led to a chicken-and-the-egg scenario — how does Alice share her secret key with Bob without Eve learning anything about it to begin with? In the 1970s, a breakthrough solution to this problem was discovered in the form of *public key encryption*.

In public key encryption, there are *two* keys: One for encryption, E, which is public knowledge to everyone, and one for decryption, D, which is known only by the receiver, Bob. In this way, *anyone in the world* can send Bob a secret message: The sender encodes the message using E, sends the encrypted message to Bob, who then decrypts it using D. How could such a scheme be possible? It turns out the secret ingredient is a special bijection which is easy to compute (i.e. anyone in the world can do it), but believed to be very difficult to *invert* without knowledge of a secret key, which only Bob possesses. This special bijection is known as the RSA function, named after its inventors Ronald Rivest, Adi Shamir and Leonard Adleman, and is as follows:

$$E(x) \equiv x^e \bmod N,$$

where $N = pq$ for two large primes p and q, $E : \{0,\ldots,N-1\} \mapsto \{0,\ldots,N-1\}$, and e is relatively prime to $(p-1)(q-1)$. It may not be clear *a priori* that this is indeed a bijection; we shall prove this shortly. The inverse of the RSA function is given by the decryption operation:

$$D(x) \equiv x^d \bmod N$$

where d is the multiplicative inverse of e mod $(p-1)(q-1)$. RSA now works as follows: The public key is (N,e), known to everyone in the world, and the private key is d, known only to Bob. To send a secret message $x \in \{1,\ldots,N-1\}$ to Bob, Alice applies the *encryption function* E to x to obtain a *ciphertext* $E(x)$, which she then sends to Bob. Upon receipt of Alice's ciphertext $E(x)$, Bob applies his *decryption function* D to recover x. This should convince you that RSA allows you to indeed encrypt and decrypt a message x. But is it *secure*? We defer this discussion to Section 3.1. For now, let us prove that indeed $D(E(x)) = x$ for all $x \in \{1,\ldots,N-1\}$, which in turn (by Theorem 7.1) implies that E is a bijection.

Theorem 7.2. *For E and D as defined above, we have $D(E(x)) = x \bmod N$ for all $x \in \{0,1,\ldots,N-1\}$.*

Proof. To prove the statement, we have to show that

$$(x^e)^d = x \bmod N \qquad \text{for every } x \in \{0,1,\ldots,N-1\}, \tag{2}$$

where recall $N = pq$ for primes p and q, $\gcd(e,(p-1)(q-1)) = 1$, and d is the multiplicative inverse of e mod $(p-1)(q-1)$. Let's consider the exponent, which is ed. By definition of d, we know that $ed = 1 \bmod (p-1)(q-1)$; hence we can write $ed = 1 + k(p-1)(q-1)$ for some integer k, and therefore

$$x^{ed} - x = x^{1+k(p-1)(q-1)} - x = x(x^{k(p-1)(q-1)} - 1). \tag{3}$$

Looking back at Equation (2), our goal is to show that this last expression in Equation (3) is equal to 0 mod N for every x. To do this, we show that the expression is divisible both by p and q; thus, since p and q are primes, the expression must also be divisible by $N = pq$, which will yield our claim.

We begin by showing that $x(x^{k(p-1)(q-1)} - 1)$ in (3) is divisible by p. We have two cases to consider:

Case 1: (x *is a multiple of* p) In this case, p clearly divides the expression in (3) since $p \mid x$.

Case 2: (x *is not a multiple of* p) Since $x \neq 0 \bmod p$, we can use Fermat's Little Theorem to deduce that $x^{p-1} = 1 \bmod p$. Then $(x^{p-1})^{k(q-1)} \equiv 1^{k(q-1)} \bmod p$, which implies that $x^{k(p-1)(q-1)} - 1 = 0 \bmod p$, and so p divides the expression in (3).

The proof that $q \mid x(x^{k(p-1)(q-1)} - 1)$ is identical. This completes the proof. □

3.1 Security of RSA

We have thus far shown that the RSA protocol is *correct*, in the sense that Alice can encrypt messages in such a way that Bob can reliably decrypt them again. But how do we know that it is *secure*, i.e., that Eve cannot obtain any information about Alice's message x from the ciphertext $E(x)$? To be clear, there is no known formal proof of this fact. Rather, the security of RSA hinges upon the following widely held *assumption*:

Given N, e and $y = x^e \bmod N$, there is no efficient algorithm for determining x.

To help us appreciate why this assumption may be true, let us brainstorm how Eve might try to guess x:

- (Brute force) The naive approach would be via brute force — simply try all possible values of x, each time checking whether $x^e = y \bmod N$. But this approach is unbelievably inefficient; Eve would have to try $O(N)$ values of x, which if N is a 512-bit number, boils down to 2^{512} values of x to try — this number is larger than estimates for the age of the Universe in femtoseconds!
- (Reduction to factoring) Eve could be more clever and instead attempt to factor N to into its prime factors p and q; then, she could compute d by determining the multiplicative inverse of $e \bmod (p-1)(q-1)$. But this requires the ability to factor large numbers, a task which is *also* considered impossible to solve efficiently.
- (Computing $(p-1)(q-1)$ directly) Finally, Eve could try to compute $(p-1)(q-1)$ without factoring N; but it turns out this is equivalent to factoring N.

Thus, the hardness of RSA depends on the presumed difficulty of factoring large numbers, known as the *Factoring Problem*. In fact, the Factoring Problem occupies a special place in theoretical computer science. It is one of the few known problems which is neither known to be efficiently solvable (i.e. in the complexity class P), nor intractable (i.e. complete for the class NP). However, there *has* been a remarkable development over the last two decades; it turns out the Factoring Problem can be solved efficiently on a *quantum* computer!

Sanity check! Our discussion regarding the brute force approach above deserves a moment's reflection, in particular with respect to the following common fallacy: Namely, in order to try all possible factors x of N requires $O(N)$ iterations, which is often interpreted as "polynomial time" since the algorithm runs in time

linear in N. Why is this algorithm *not* "polynomial time"? (Hint: What is the relevant quantity with respect to which we typically measure complexity?)

3.2 Implementations of RSA

We close this note with a brief discussion of implementation issues regarding RSA. Since we have argued that breaking RSA is impossible because *factoring* would take a very long time, we should check that the computations that Alice and Bob themselves have to perform are much simpler, and can be done efficiently.

There are really only two non-trivial things that Alice and Bob have to do:

1. Bob has to find prime numbers p and q, each having many (say, 512) bits.
2. Both Alice and Bob have to compute exponentials mod N. (Alice has to compute $x^e \bmod N$, and Bob has to compute $y^d \bmod N$.)

Both of these tasks can be carried out efficiently. The first requires the implementation of an efficient test for primality as well as a rich source of primes. You will learn how to tackle each of these tasks in the algorithms course CS170. The second requires an efficient algorithm for modular exponentiation known as "repeated squaring", which is not very difficult, but will also be discussed in detail in CS170.

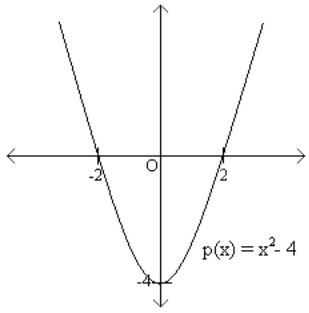
To summarize, then, in the RSA protocol Bob need only perform simple calculations such as multiplication, exponentiation and primality testing to determine the encryption and decryption keys. Similarly, Alice and Bob need only perform simple calculations to lock and unlock the secret message respectively — operations that any pocket computing device could handle. By contrast, to unlock the message without the key, Eve would have to perform operations like factoring large numbers, which (at least according to widely accepted belief) requires more computational power than all of the world's most sophisticated computers combined! This compelling guarantee of security without the need for private keys explains why the RSA cryptosystem is such a revolutionary development in cryptography.

Polynomials

Polynomials constitute a rich class of functions which are both easy to describe and widely applicable in topics ranging from Fourier analysis to computational geometry. In this note, we will discuss properties of polynomials which make them so useful. We will then describe how to take advantage of these properties to develop a secret sharing scheme.

Recall from your high school math that a *polynomial* in a single variable is of the form $p(x) = a_d x^d + a_{d-1}x^{d-1} + \ldots + a_0$. Here the *variable* x and the *coefficients* a_i are usually real numbers. For example, $p(x) = 5x^3 + 2x + 1$, is a polynomial of *degree* $d = 3$. Its coefficients are $a_3 = 5$, $a_2 = 0$, $a_1 = 2$, and $a_0 = 1$. Polynomials have some remarkably simple, elegant and powerful properties, which we will explore in this note.

First, a definition: we say that a is a *root* of the polynomial $p(x)$ if $p(a) = 0$. For example, the degree 2 polynomial $p(x) = x^2 - 4$ has two roots, namely 2 and -2, since $p(2) = p(-2) = 0$. If we plot the polynomial $p(x)$ in the x-y plane, then the roots of the polynomial are just the places where the curve crosses the x axis:

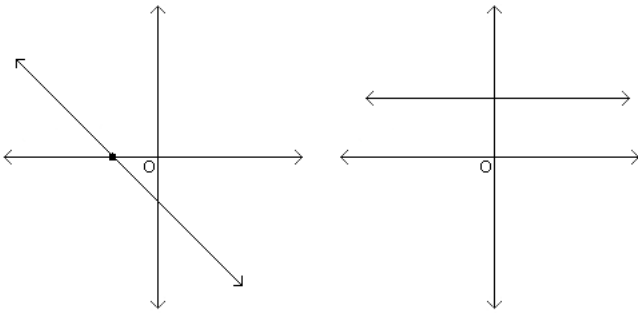


We now state two fundamental properties of polynomials that we will prove in due course.

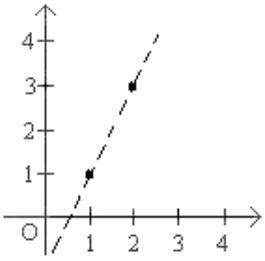
Property 1: A non-zero polynomial of degree d has at most d roots.

Property 2: Given $d+1$ pairs $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})$, with all the x_i distinct, there is a unique polynomial $p(x)$ of degree (at most) d such that $p(x_i) = y_i$ for $1 \leq i \leq d+1$.

Let us consider what these two properties say in the case that $d = 1$. A graph of a linear (degree 1) polynomial $y = a_1 x + a_0$ is a line. Property 1 says that if a line is not the x-axis (i.e. if the polynomial is not $y = 0$), then it can intersect the x-axis in at most one point.



Property 2 says that two points uniquely determine a line.



Polynomial Interpolation

Property 2 says that two points uniquely determine a degree 1 polynomial (a line), three points uniquely determine a degree 2 polynomial, four points uniquely determine a degree 3 polynomial, and so on. Given $d+1$ pairs $(x_1,y_1),\ldots,(x_{d+1},y_{d+1})$, how do we determine the polynomial $p(x) = a_d x^d + \ldots + a_1 x + a_0$ such that $p(x_i) = y_i$ for $i = 1$ to $d+1$? We will give an efficient algorithms for reconstructing the coefficients $a_0,\ldots,a_d$, and therefore the polynomial $p(x)$.

The method is called *Lagrange interpolation*: Let us start by solving an easier problem. Suppose that we are told that $y_1 = 1$ and $y_j = 0$ for $2 \le j \le d+1$. Now can we reconstruct $p(x)$? Yes, this is easy! Consider $q(x) = (x-x_2)(x-x_3)\cdots(x-x_{d+1})$. This is a polynomial of degree d (the x_i's are constants, and x appears d times). Also, we clearly have $q(x_j) = 0$ for $2 \le j \le d+1$. But what is $q(x_1)$? Well, $q(x_1) = (x_1-x_2)(x_1-x_3)\cdots(x_1-x_{d+1})$, which is some constant not equal to 0. Thus if we let $p(x) = q(x)/q(x_1)$ (dividing is ok since $q(x_1) \neq 0$), we have the polynomial we are looking for. For example, suppose you were given the pairs $(1,1)$, $(2,0)$, and $(3,0)$. Then we can construct the degree $d = 2$ polynomial $p(x)$ by letting $q(x) = (x-2)(x-3) = x^2 - 5x + 6$, and $q(x_1) = q(1) = 2$. Thus, we can now construct $p(x) = q(x)/q(x_1) = (x^2 - 5x + 6)/2$.

Of course the problem is no harder if we single out some arbitrary index i instead of 1: i.e. $y_i = 1$ and $y_j = 0$ for $j \neq i$. Let us introduce some notation: let us denote by $\Delta_i(x)$ the degree d polynomial that goes through these $d+1$ points. Then $\Delta_i(x) = \frac{\Pi j \neq i (x-x_j)}{\Pi j \neq i (x_i - x_j)}$.

Let us now return to the original problem. Given $d+1$ pairs $(x_1,y_1),\ldots,(x_{d+1},y_{d+1})$, we first construct the $d+1$ polynomials $\Delta_1(x),\ldots,\Delta_{d+1}(x)$. Now we can write $p(x) = \sum_{i=1}^{d+1} y_i \Delta_i(x)$. Why does this work? First notice that $p(x)$ is a polynomial of degree d as required, since it is the sum of polynomials of degree d. And when it is evaluated at x_i, d of the $d+1$ terms in the sum evaluate to 0 and the *i-th* term evaluates to y_i times 1, as required.

As an example, suppose we want to find the degree-2 polynomial $p(x)$ that passes through the three points

$(1,1)$, $(2,2)$ and $(3,4)$. The three polynomials Δ_i are as follows: If $d = 2$, and $x_i = i$, for instance, then

$$\Delta_1(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)} = \frac{(x-2)(x-3)}{2} = \frac{1}{2}x^2 - \frac{5}{2}x + 3;$$
$$\Delta_2(x) = \frac{(x-1)(x-3)}{(2-1)(2-3)} = \frac{(x-1)(x-3)}{-1} = -x^2 + 4x - 3;$$
$$\Delta_3(x) = \frac{(x-1)(x-2)}{(3-1)(3-2)} = \frac{(x-1)(x-2)}{2} = \frac{1}{2}x^2 - \frac{3}{2}x + 1.$$

The polynomial $p(x)$ is therefore given by

$$p(x) = 1 \cdot \Delta_1(x) + 2 \cdot \Delta_2(x) + 4 \cdot \Delta_3(x) = \frac{1}{2}x^2 - \frac{1}{2}x + 1.$$

You should verify that this polynomial does indeed pass through the above three points.

Proof of Property 2

We would like to prove property 2:

Property 2: Given $d+1$ pairs $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})$, with all the x_i distinct, there is a unique polynomial $p(x)$ of degree (at most) d such that $p(x_i) = y_i$ for $1 \leq i \leq d+1$.

We have shown how to find a polynomial $p(x)$ such that $p(x_i) = y_i$ for $d+1$ pairs $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})$. This proves part of property 2 (the existence of the polynomial). How do we prove the second part, that the polynomial is unique? Suppose for contradiction that there is another polynomial $q(x)$ such that $p(x_i) = y_i$ for all $d+1$ pairs above. Now consider the polynomial $r(x) = p(x) - q(x)$. Since we are assuming that $q(x)$ and $p(x)$ are different polynomials, $r(x)$ must be a non-zero polynomial of degree at most d. Therefore, property 1 implies it can have at most d roots. But on the other hand $r(x_i) = p(x_i) - q(x_i) = 0$ on $d+1$ distinct points. Contradiction. Therefore, $p(x)$ is the unique polynomial that satisfies the $d+1$ conditions.

Polynomial Division

Let's take a short digression to discuss polynomial division, which will be useful in the proof of property 1. If we have a polynomial $p(x)$ of degree d, we can divide by a polynomial $q(x)$ of degree $\leq d$ by using long division. The result will be:

$$p(x) = q(x)q'(x) + r(x)$$

where $q'(x)$ is the quotient and $r(x)$ is the remainder. The degree of $r(x)$ must be smaller than the degree of $p(x)$.

Example. We wish to divide $p(x) = x^3 + x^2 - 1$ by $q(x) = x - 1$:

$$\begin{array}{r|l} & X^2 + 2X + 2 \\ \hline X-1 & X^3 + X^2 \quad\quad - 1 \\ & \underline{-X^3 + X^2} \\ & \quad 2X^2 \\ & \quad \underline{-2X^2 + 2X} \\ & \quad\quad 2X - 1 \\ & \quad\quad \underline{-2X + 2} \\ & \quad\quad\quad 1 \end{array}$$

Now $p(x) = x^3 + x^2 - 1 = (x-1)(x^2 + 2x + 2) + 1$, $r(x) = 1$ and $q'(x) = x^2 + 2x + 2$.

Proof of Property 1

Now let us turn to property 1: a non-zero polynomial of degree d has at most d roots.The idea of the proof is as follows. We will prove the following claims:

> **Claim 1** If a is a root of a polynomial $p(x)$ with degree d, then $p(x) = (x-a)q(x)$ for a polynomial $q(x)$ with degree $d-1$.
>
> **Claim 2** A polynomial $p(x)$ of degree d with distinct roots $a_1, \ldots, a_d$ can be written as $p(x) = c(x-a_1)\cdots(x-a_d)$.

Claim 2 implies property 1. We must show that $a \neq a_i$ for $i = 1, \ldots d$ cannot be a root of $p(x)$. But this follows from claim 2, since $p(a) = c(a-a_1)\cdots(a-a_d) \neq 0$.

Proof of Claim 1

Dividing $p(x)$ by $(x-a)$ gives $p(x) = (x-a)q(x)+r(x)$, where $q(x)$ is the quotient and $r(x)$ is the remainder. The degree of $r(x)$ is necessarily smaller than the degree of the divisor $(x-a)$. Therefore $r(x)$ must have degree 0 and therefore is some constant c. But now substituting $x = a$, we get $p(a) = c$. But since a is a root, $p(a) = 0$. Thus $c = 0$ and therefore $p(x) = (x-a)q(x)$, thus showing that $(x-a)|p(x)$.

Claim 1 implies Claim 2

Proof by induction on d.

- Base Case: We must show that a polynomial $p(x)$ of degree 1 with root a_1 can be written as $p(x) = c(x-a_1)$. By Claim 1, we know that $p(x) = (x-a_1)q(x)$, where $q(x)$ has degree 0 and is therefore a constant.
- Inductive Hypothesis: A polynomial of degree $d-1$ with distinct roots $a_1, \ldots, a_{d-1}$ can be written as $p(x) = c(x-a_1)\cdots(x-a_{d-1})$.
- Inductive Step: Let $p(x)$ be a polynomial of of degree d with distinct roots $a_1, \cdots, a_d$. By Claim 1, $p(x) = (x-a_d)q(x)$ for some polynomial $q(x)$ of degree $d-1$. Since $0 = p(a_i) = (a_i-a_d)q(a_i)$ for all $i \neq d$ and $a_i - a_d \neq 0$ in this case, $q(a_i)$ must be equal to 0. Then $q(x)$ is a polynomial of degree $d-1$ with distinct roots $a_1, \ldots, a_{d-1}$. We can now apply the inductive assumption to $q(x)$ to write $q(x) = c(x-a_1)\cdots(x-a_{d-1})$. Substituting in $p(x) = (x-a_d)q(x)$, we finally obtain that $p(x) = c(x-a_1)\cdots(x-a_d)$.

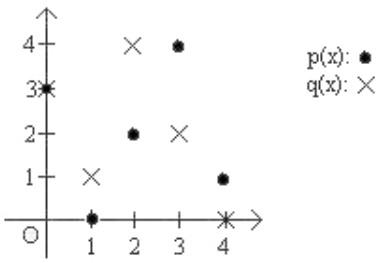
Finite Fields

Both property 1 and property 2 also hold when the values of the coefficients and the variable x are chosen from the complex numbers instead of the real numbers or even the rational numbers. They do not hold if the values are restricted to being natural numbers or integers. Let us try to understand this a little more closely. The only properties of numbers that we used in polynomial interpolation and in the proof of property 1 is that we can add, subtract, multiply and divide any pair of numbers as long as we are not dividing by 0. We cannot subtract two natural numbers and guarantee that the result is a natural number. And dividing two integers does not usually result in an integer.

But if we work with numbers modulo a prime m, then we can add, subtract, multiply and divide (by any non-zero number modulo m). To check this, recall that x has an inverse mod m if $\gcd(m,x) = 1$, so if m is prime *all* the numbers $\{1,\ldots,m-1\}$ have an inverse mod m. So both property 1 and property 2 hold if the coefficients and the variable x are restricted to take on values modulo m. This remarkable fact that these properties hold even when we restrict ourselves to a *finite* set of values is the key to several applications that we will presently see.

Let us consider an example of degree $d = 1$ polynomials modulo 5. Let $p(x) = 2x+3 \pmod 5$. The roots of this polynomial are all values x such that $2x+3 = 0 \pmod 5$ holds. Solving for x, we get that $2x = -3 = 2 \pmod 5$ or $x = 1 \pmod 5$. Note that this is consistent with property 1 since we got only 1 root of a degree 1 polynomial.

Now consider the polynomials $p(x) = 2x+3$ and $q(x) = 3x-2$ with all numbers reduced mod 5. We can plot the value of each polynomial y as a function of x in the x-y plane. Since we are working modulo 5, there are only 5 possible choices for x, and only 5 possible choices for y:



Notice that these two “lines" intersect in exactly one point, even though the picture looks nothing at all like lines in the Euclidean plane! Looking at these graphs it might seem remarkable that both property 1 and property 2 hold when we work modulo m for any prime number m. But as we stated above, all that was required for the proofs of property 1 and 2 was the ability to add, subtract, multiply and divide any pair of numbers (as long as we are not dividing by 0), and they hold whenever we work modulo a prime m.

When we work with numbers modulo a prime m, we say that we are working over a finite field, denoted by F_m or $GF(m)$ (for Galois Field). In order for a set to be called a field, it must satisfy certain axioms which are the building blocks that allow for these amazing properties and others to hold. If you would like to learn more about fields and the axioms they satisfy, you can visit Wikipedia's site and read the article on fields: `http://en.wikipedia.org/wiki/Field_\%28mathematics\%29`. While you are there, you can also read the article on Galois Fields and learn more about some of their applications and elegant properties which will not be covered in this lecture: `http://en.wikipedia.org/wiki/Galois_field`.

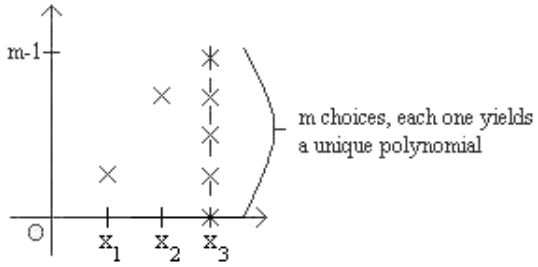
Counting

How many polynomials of degree (at most) 2 are there modulo m? This is easy: there are 3 coefficients, each of which can take on one of m values for a total of m^3. Writing $p(x) = a_d x^d + a_{d-1}x^{d-1} + \ldots + a_0$ by specifying its $d+1$ coefficients a_i is known as the coefficient representation of $p(x)$. Is there any other way to specify $p(x)$?

Sure, there is! Our polynomial of degree (at most) 2 is uniquely specified by its values at any three points, say $x = 0,1,2$. Once again each of these three values can take on one of m values, for a total of m^3 possibilities. In general, we can specify a degree d polynomial $p(x)$ by specifying its values at $d+1$ points, say $0,1,\ldots,d$. These $d+1$ values, $(y_0, y_1, \ldots, y_d)$ are called the value representation of $p(x)$. The coefficient representation

can be converted to the value representation by evaluating the polynomial at $0, 1, \ldots, d$. And as we've seen, polynomial interpolation can convert the value representation to the coefficient representation.

So if we are given three pairs $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ then there is a unique polynomial of degree 2 such that $p(x_i) = y_i$. But now, wuppose we were only given two pairs $(x_1, y_1), (x_2, y_2)$; how many distinct degree 2 polynomials are there that go through these two points? Notice that there are exactly m choices for y_3, and for each choice there is a unique (and distinct) polynomial of degree 2 that goes through the three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$. It follows that there are exactly m polynomials of degree at most 2 that go through 2 points $(x_1, y_1), (x_2, y_2)$, as shown below:



What if you were only given one point (x_1, y_1)? Well, there are m^2 choices for y_2 and y_3, yielding m^2 polynomials of degree at most 2 that go through the point given. A pattern begins to emerge, as is summarized in the following table:

Polynomials of degree $\leq d$ over F_m	
# of points	# of polynomials
$d+1$	1
d	m
$d-1$	m^2
$\vdots$	$\vdots$
$d-k$	m^{k+1}
$\vdots$	$\vdots$
0	m^{d+1}

Note that the reason that we can count the number of polynomials in this setting is because we are working over a finite field. If we were working over an infinite field such as the rationals, there would be infinitely many polynomials of degree d that can go through d points! Think of a line, which has degree one. If you were just given one point, there would be infinitely many possibilities for the second point, each of which uniquely defines a line.

Secret Sharing

In the late 1950's and into the 1960's, during the Cold War, President Dwight D. Eisenhower approved instructions and authorized top commanding officers for the use of nuclear weapons under very urgent emergency conditions. Such measures were set up in order to defend the United States in case of an attack in which there was not enough time to confer with the President and decide on an appropriate response. This would allow for a rapid response in case of a Soviet attack on U.S. soil. This is a perfect situation in which a secret sharing scheme could be used to ensure that a certain number of officials must come together in order to successfully launch a nuclear strike, so that for example no single person has the power and control

over such a devastating and destructive weapon. Suppose the U.S. government finally decides that a nuclear strike can be initiated only if at least $k > 1$ major officials agree to it. We want to devise a scheme such that (1) any group of k of these officials can pool their information to figure out the launch code and initiate the strike but (2) no group of $k-1$ or fewer have any information about the launch code, even if they pool their knowledge. For example, they should not learn whether the secret is odd or even, a prime number, divisible by some number a, or the secret's least significant bit. How can we accomplish this?

Suppose that there are n officials indexed from 1 to n and the launch code is some natural number s. Let q be a prime number larger than n and s. We will work over $GF(q)$ from now on.

Now pick a random polynomial $P(x)$ of degree $k-1$ such that $P(0) = s$ and give $P(1)$ to the first official, $P(2)$ to the second,..., $P(n)$ to the n^{th}. Then

- Any k officials, having the values of the polynomial at k points, can use Lagrange interpolation to find P, and once they know what P is, they can compute $P(0) = s$ to learn the secret. Another way to say this is that any k officials have between them a value representation of the polynomial, which they can convert to the coefficient representation, which allows them to evaluate $P(0) = s$.
- Any group of $k-1$ officials has no information about s. So they know only $k-1$ points through which $P(x)$, an unknown polynomial of degree $k-1$ passes. They wish to reconstruct $P(0)$. But by our discussion in the previous section, for each possible value $P(0) = b$, there is a unique polynomial of degree $k-1$ that passes through the $k-1$ points of the $k-1$ officials as well as through $(0,b)$. Hence the secret could be any of the q possible values $\{0,1,\ldots,q-1\}$, so the officials have—in a very precise sense—no information about s. Another way of saying this is that the information of the officials is consistent with q different value representations, one for each possible value of the secret, and thus the officials have no information about s. (Note that this is the main reason we choose to work over finite fields rather than, say, over the real numbers, where the basic secret-sharing scheme would still work. Because there are only finitely many values in our field, we can quantify precisely how many remaining possibilities there are for the value of the secret, and show that this is the same as if the officials had no information at all.)

Example. Suppose you are in charge of setting up a secret sharing scheme, with secret $s = 1$, where you want to distribute $n = 5$ shares to 5 people such that any $k = 3$ or more people can figure out the secret, but two or fewer cannot. Let's say we are working over $GF(7)$ (note that $7 > s$ and $7 > n$) and you randomly choose the following polynomial of degree $k-1 = 2 : P(x) = 3x^2 + 5x + 1$ (here, $P(0) = 1 = s$, the secret). So you know everything there is to know about the secret and the polynomial, but what about the people that receive the shares? Well, the shares handed out are $P(1) = 2$ to the first official, $P(2) = 2$ to the second, $P(3) = 1$ to the third, $P(4) = 6$ to the fourth, and $P(5) = 3$ to the fifth official. Let's say that officials 3, 4, and 5 get together (we expect them to be able to recover the secret). Using Lagrange interpolation, they compute the following delta functions:

$$\Delta_3(x) = \frac{(x-4)(x-5)}{(3-4)(3-5)} = \frac{(x-4)(x-5)}{2} = 4(x-4)(x-5);$$
$$\Delta_4(x) = \frac{(x-3)(x-5)}{(4-3)(4-5)} = \frac{(x-3)(x-5)}{-1} = 6(x-3)(x-5);$$
$$\Delta_5(x) = \frac{(x-3)(x-4)}{(5-3)(5-4)} = \frac{(x-3)(x-4)}{2} = 4(x-3)(x-4).$$

They then compute the polynomial over $GF(7)$: $P(x) = (1)\Delta_3(x) + (6)\Delta_4(x) + (3)\Delta_5(x) = 3x^2 + 5x + 1$ (verify the computation!). Now they simply compute $P(0)$ and discover that the secret is 1.

Let's see what happens if two officials try to get together, say persons 1 and 5. They both know that the polynomial looks like $P(x) = a_2x^2 + a_1x + s$. They also know the following equations:

$$P(1) = a_2 + a_1 + s = 2$$
$$P(5) = 4a_2 + 5a_1 + s = 3$$

But that is all they have two equations with three unknowns, and thus they cannot find out the secret. This is the case no matter which two officials get together. Notice that since we are working over $GF(7)$, the two people could have guessed the secret ($0 \leq s \leq 6$) and constructed a unique degree 2 polynomial (by property 2). But the two people combined have the same chance of guessing what the secret is as they do individually. This is important, as it implies that two people have no more information about the secret than one person does.

Error Correcting Codes

We will consider two situations in which we wish to transmit information on an unreliable channel. The first is exemplified by the internet, where the information (say a file) is broken up into packets, and the unreliability is manifest in the fact that some of the packets are lost (or erased) during transmission. Moreover the packets are labeled so that the recipient knows exactly which packets were received and which were dropped. We will refer to such errors as erasure errors. See the figure below:

In the second situation, some of the packets are corrupted during transmission due to channel noise. Now the recipient has no idea which packets were corrupted and which were received unmodified:

In the above example, packets 1 and 4 are corrupted. These types of errors are called general errors. We will discuss methods of encoding messages, called error correcting codes, which are capable of correcting both erasure and general errors.

Assume that the information consists of n packets. We can assume without loss of generality that the contents of each packet is a number modulo q (denoted by $GF(q)$), where q is a prime. For example, the contents of the packet might be a 32-bit string and can therefore be regarded as a number between 0 and $2^{32}-1$; then we could choose q to be any prime larger than 2^{32}. The properties of polynomials over $GF(q)$ (i.e., with coefficients and values reduced modulo q) are the backbone of both error-correcting schemes. To see this, let us denote the message to be sent by $m_1,\ldots,m_n$ and make the following crucial observations:

1) There is a unique polynomial $P(x)$ of degree $n-1$ such that $P(i)=m_i$ for $1\le i\le n$ (i.e., $P(x)$ contains all of the information about the message, and evaluating $P(i)$ gives the contents of the *i-th* packet).

2) The message to be sent is now $m_1=P(1),\ldots,m_n=P(n)$. We can generate additional packets by evaluating $P(x)$ at additional points $n+1,n+2,\ldots,n+j$ (remember, our transmitted message must be redundant, i.e., it must contain more packets than the original message to account for the lost or corrupted packets). Thus the transmitted message is $c_1=P(1),c_2=P(2),\ldots,c_{n+j}=P(n+j)$. Since we are working modulo q, we must make sure that $n+j\le q$, but this condition does not impose a serious constraint since q is very large.

Erasure Errors

Here we consider the setting of packets being transmitted over the internet. In this setting, the packets are labeled and so the recipient knows exactly which packets were dropped during transmission. One additional observation will be useful:

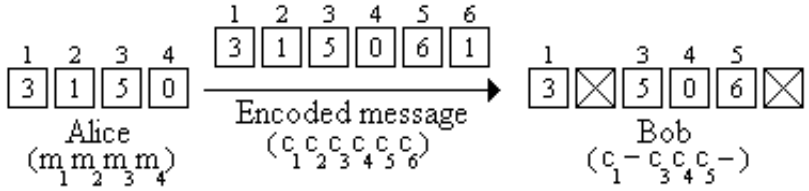
3) By Property 2 in Note 8, we can uniquely reconstruct $P(x)$ from its values at any n distinct points, since it has degree $n-1$. This means that $P(x)$ can be reconstructed from any n of the transmitted packets. Evaluating this reconstructed polynomial $P(x)$ at $x = 1, \ldots, n$ yields the original message $m_1, \ldots, m_n$.

Recall that in our scheme, the transmitted message is $c_1 = P(1), c_2 = P(2), \ldots, c_{n+j} = P(n+j)$. Thus, if we hope to be able to correct k errors, we simply need to set $j = k$. The encoded message will then consist of $n+k$ packets.

Example

Suppose Alice wants to send Bob a message of $n = 4$ packets and she wants to guard against $k = 2$ lost packets. Then, assuming the packets can be coded up as integers between 0 and 6, Alice can work over $GF(7)$ (since $7 \geq n+k = 6$). Suppose the message that Alice wants to send to Bob is $m_1 = 3$, $m_2 = 1$, $m_3 = 5$, and $m_4 = 0$. She interpolates to find the unique polynomial of degree $n-1 = 3$ described by these 4 points: $P(x) = x^3 + 4x^2 + 5$ (verify that $P(i) = m_i$ for $1 \leq i \leq 4$).

Since $k = 2$, Alice must evaluate $P(x)$ at 2 extra points: $P(5) = 6$ and $P(6) = 1$. Now, Alice can transmit the encoded message which consists of $n+k = 6$ packets, where $c_j = P(j)$ for $1 \leq j \leq 6$. So $c_1 = P(1) = 3$, $c_2 = P(2) = 1$, $c_3 = P(3) = 5$, $c_4 = P(4) = 0$, $c_5 = P(5) = 6$, and $c_6 = P(6) = 1$. Suppose packets 2 and 6 are dropped, in which case we have the following situation:



From the values that Bob received (3, 5, 0, and 6), he uses Lagrange interpolation and computes the following delta functions:

$$\Delta_1(x) = \frac{(x-3)(x-4)(x-5)}{-24}$$
$$\Delta_3(x) = \frac{(x-1)(x-4)(x-5)}{4}$$
$$\Delta_4(x) = \frac{(x-1)(x-3)(x-5)}{-3}$$
$$\Delta_5(x) = \frac{(x-1)(x-3)(x-4)}{8}.$$

He then reconstructs the polynomial $P(x) = (3)\Delta_1(x) + (5)\Delta_3(x) + (0)\Delta_4(x) + (6)\Delta_5(x) = x^3 + 4x^2 + 5$. Bob then evaluates $m_2 = P(2) = 1$, which is the packet that was lost from the original message. More generally, no matter which two packets were dropped, following the same method Bob could still have reconstructed $P(x)$ and thus the original message.

Let us consider what would happen if Alice sent one fewer packet. If Alice only sent c_j for $1 \leq j \leq n+k-1$, then with k erasures, Bob would only receive c_j for $n-1$ distinct values j. Thus, Bob would not be able to reconstruct $P(x)$ (since there are exactly q polynomials of degree at most $n-1$ that agree with the $n-1$ packets which Bob received). This error-correcting scheme is therefore optimal: it can recover the n characters of the transmitted message from any n received characters, but recovery from any fewer characters is impossible.

Polynomial Interpolation

Let us take a brief digression to discuss another method of polynomial interpolation which will be useful in handling general errors. The goal of the algorithm will be to take as input $d+1$ pairs $(x_1,y_1),\cdots,(x_{d+1},y_{d+1})$, and output the polynomial $p(x)=a_dx^d+\cdots+a_1x+a_0$ such that $p(x_i)=y_i$ for $i=1$ to $d+1$.

The first step of the algorithm is to write a system of $d+1$ linear equations in $d+1$ variables: the coefficients of the polynomial $a_0,\ldots,a_d$. Each equation is obtained by fixing x to be one of $d+1$ values: $x_1,\cdots,x_{d+1}$. Note that in $p(x)$, x is a variable and $a_0,\ldots,a_d$ are fixed constants. In the equations below, these roles are swapped: x_i is a fixed constant and $a_0,\ldots,a_d$ are variables. For example, the *i-th* equation is the result of fixing x to be x_i: $a_dx_i^d+a_{d-1}x_i^{d-1}+\ldots+a_0=y_i$.

Now solving these equations gives the coefficients of the polynomial $p(x)$. For example, given the 3 pairs $(-1,2)$, $(0,1)$, and $(2,5)$, we will construct the degree 2 polynomial $p(x)$ which goes through these points. The first equation says $a_2(-1)^2+a_1(-1)+a_0=2$. Simplifying, we get $a_2-a_1+a_0=2$. Similarly, the second equation says $a_2(0)^2+a_1(0)+a_0=1$, or $a_0=1$. And the third equation says $a_2(2)^2+a_1(2)+a_0=5$ So we get the following system of equations:

$$\begin{aligned} a_2-a_1+a_0&=2\\ a_0&=1\\ 4a_2+2a_1+a_0&=5 \end{aligned}$$

Substituting for a_0 and multiplying the first equation by 2 we get:

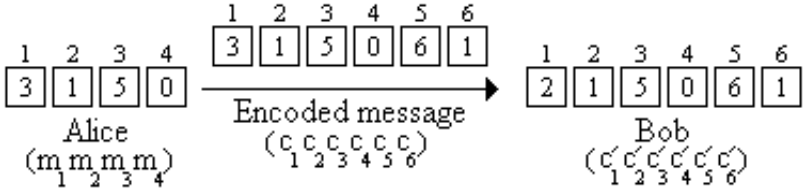
$$\begin{aligned} 2a_2-2a_1&=2\\ 4a_2+2a_1&=4 \end{aligned}$$

Then, adding the two equations we find that $6a_2=6$, so $a_2=1$, and plugging back in we find that $a_1=0$. Thus, we have determined the polynomial $p(x)=x^2+1$. To justify this method more carefully, we must show that the equations always have a solution and that it is unique. This involves showing that a certain determinant is non-zero, which we will leave as an exercise.

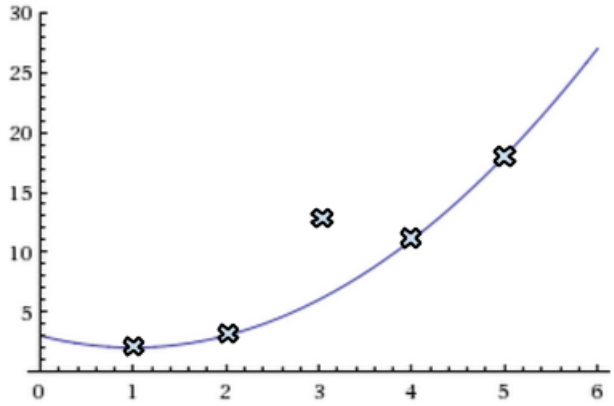
General Errors

Now let us return to general errors. General errors are much more challenging to correct than erasure errors. This is because packets are corrupted, not erased and Bob no longer knows which packets are correct. As we shall see shortly, Alice can still guard against k general errors, at the expense of transmitting only $2k$ additional packets or characters (only twice as many as in the erasure case). Thus the encoded message is $c_1,\ldots,c_{n+2k}$ where $c_j=P(j)$ for $1\le j\le n+2k$. This means that at least $n+k$ of these characters are received uncorrupted by Bob.

For example, if Alice wishes to send $n=4$ characters to Bob via a modem in which $k=1$ of the characters is corrupted, she must redundantly send an encoded message consisting of 6 characters. Suppose she wants to transmit the same message as above, and that c_1 is corrupted and changed to $r_1=2$. This scenario can be visualized in the following figure:



Bob's goal is to reconstruct $P(x)$ from the $n+2k$ received characters $r_1, \cdots, r_{n+2k}$. He knows that $P(i)$ must equal r_i on at least $n+k$ points (since only k points are corrupted), but he does not know which of the $n+k$ values are correct. As an example, consider a possible scenario depicted in the picture below- the points represent the message received from Alice, and the line represents $P(x)$. In this example, $n=3$, $k=1$, and the third packet is corrupted. Bob does not know the index at which the message and the polynomial deviate:



Bob attempts to construct $P(x)$ by searching for a polynomial $P'(x)$ with the following property: $P'(i) = r_i$ for at least $n+k$ distinct values of i between 1 and $n+2k$. Of course, $P(x)$ is one such polynomial. It turns out that $P(x)$ is actually the only polynomial with the desired property. Therefore, $P'(x)$ must equal $P(x)$.

Finding $P(x)$ efficiently requires a remarkable idea, which is just about simple enough to be described here. Suppose packets $e_1, \ldots, e_k$ are corrupted. Define the degree k polynomial $E(x)$ to be $(x-e_1)\cdots(x-e_k)$. Let us make a simple but crucial observation: $P(i)E(i) = r_iE(i)$ for $1 \leq i \leq n+2k$ (this is true at points i at which no error occurred since $P(i) = r_i$, and trivially true at points i at which an error occurred since $E(i) = 0$).

This observation forms the basis of a very clever algorithm invented by Berlekamp and Welch. Looking more closely at these equalities, we will show that they are $n+2k$ linear equations in $n+2k$ unknowns. The unknowns correspond to the coefficients of $E(x)$ and $Q(x)$ (where we define $Q(x) = P(x)E(x)$). Once $Q(x)$ and $E(x)$ are known, we can divide $Q(x)$ by $E(x)$ to obtain $P(x)$.

Since $Q(x)$ is a polynomial of degree $n+k-1$, it can be described by $n+k$ coefficients. $E(x)$ is a degree k polynomial, but its definition implies that its first coefficient must be 1. It can therefore be described by k coefficients:

$$Q(x) = a_{n+k-1}x^{n+k-1} + \cdots + a_1x + a_0$$
$$E(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0$$

As seen in the interpolation method above, once we fix a value i for x, $Q(i)$ and $E(i)$ are linear functions of $a_{n_k-1}, \cdots, a_0$ and $b_{k-1}, \cdots, b_0$ respectively. The received value r_i is also fixed. Therefore the equation $Q(i) = r_iE(i)$ is a linear equation in the $n+2k$ unknowns $a_{n+k-1}, \ldots, a_0$ and $b_{k-1}, \ldots, b_0$. We thus have $n+2k$ linear equations, one for each value of i, and $n+2k$ unknowns. We can solve these equations and get $E(x)$ and $Q(x)$. We can then compute the ratio $\frac{Q(x)}{E(x)}$ to obtain $P(x)$.

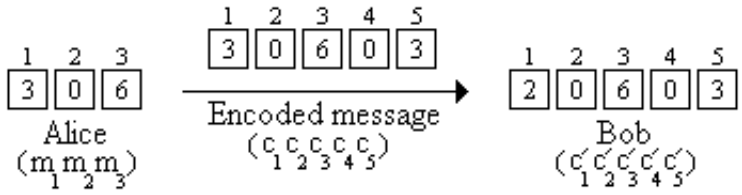
Example

Suppose we are working over $GF(7)$ and Alice wants to send Bob the $n = 3$ characters "3," "0," and "6" over a modem. Turning to the analogy of the English alphabet, this is equivalent to using only the first 7 letters of the alphabet, where $a = 0, b = 1, \ldots, g = 6$. So the message which Alice wishes for Bob to receive is "dag". Then Alice interpolates to find the polynomial

$$P(x) = x^2 + x + 1,$$

which is the unique polynomial of degree 2 such that $P(1) = 3$, $P(2) = 0$, and $P(3) = 6$.

Suppose that $k = 1$ character is corrupted, so she needs to transmit the $n + 2k = 5$ characters $P(1) = 3$, $P(2) = 0$, $P(3) = 6$, $P(4) = 0$, and $P(5) = 3$ to Bob. Suppose $P(1)$ is corrupted, so he receives 2 instead of 3 (i.e., Alice sends the encoded message "dagad" but Bob instead receives "cagad"). Summarizing, we have the following situation:



Let $E(x) = x + b_0$ be the error-locator polynomial—remember, Bob doesn't know what b_0 is yet since he doesn't know where the error occurred. Let $Q(x) = a_3x^3 + a_2x^2 + a_1x + a_0$. Now Bob just substitutes $x = 1, x = 2, \cdots, x = 5$ into $Q(x) = r_xE(x)$ and simplifies to get five linear equations in five unknowns. Recall that we are working modulo 7 and that $r_i = c_i'$ is the value Bob received for the *i-th* character.

The first equation will be $a_3 + a_2 + a_1 + a_0 = 2(1 + b_0)$, which simplifies to $a_3 + a_2 + a_1 + a_0 + 5b_0 = 2$. Bob can determine the remaining equations in the same manner, obtaining:

$$\begin{aligned}
a_3 + a_2 + a_1 + a_0 + 5b_0 &= 2 \\
a_3 + 4a_2 + 2a_1 + a_0 &= 0 \\
6a_3 + 2a_2 + 3a_1 + a_0 + b_0 &= 4 \\
a_3 + 2a_2 + 4a_1 + a_0 &= 0 \\
6a_3 + 4a_2 + 5a_1 + a_0 + 4b_0 &= 1
\end{aligned}$$

Bob then solves this linear system and finds that $a_3 = 1$, $a_2 = 0$, $a_1 = 0$, $a_0 = 6$, and $b_0 = 6$ (all mod 7). (As a check, this implies that $E(x) = x + 6 = x - 1$, so the location of the error is position $e_1 = 1$, which is correct since the first character was corrupted from a "d" to a "c".) This gives him the polynomials $Q(x) = x^3 + 6$ and $E(x) = x - 1$. He can then find $P(x)$ by computing the quotient $P(x) = \frac{Q(x)}{E(x)} = \frac{x^3+6}{x-1} = x^2 + x + 1$. Bob notices that the first character was corrupted (since $e_1 = 1$), so now that he has $P(x)$, he just computes $P(1) = 3 =$ "d" and obtains the original, uncorrupted message "dag".

Finer points

Two points need further discussion. How do we know that the $n + 2k$ equations are consistent? What if they have no solution? This is simple. The equations must be consistent since $Q(x) = P(x)E(x)$ together with the error locator polynomial $E(x)$ gives a solution.

A more interesting question is this: how do we know that the $n+2k$ equations are independent, i.e., how do we know that there aren't other spurious solutions in addition to the real solution that we are looking for? Put more mathematically, how do we know that the solution $Q'(x)$ and $E'(x)$ that we reconstruct satisfies the property that $E'(x)$ divides $Q'(x)$ and that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$?

We claim that $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$. Since the degree of both $Q(x)E'(x)$ and $Q'(x)E(x)$ is $n+2k-1$ and they are equal at $n+2k$ points, it follows from Property 2 of Note 7 that they are the same polynomial. Rearranging, we get $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$.

Why is the claim above true? Based on our method of obtaining $Q'(x)$ and $E'(x)$, we know that $Q'(i) = r_i E'(i)$ and $Q(i) = r_i E(i)$. Now assume $E(i)$ is 0. Then $Q(i)$ is also 0, so both $Q(i)E'(i)$ and $Q'(i)E(i)$ are 0 and the claim holds. The same reasoning applies when $E'(i)$ is 0. If both $E(i)$ and $E'(i)$ are not 0, we can rearrange the above equality to obtain $\frac{Q'(i)}{E'(i)} = \frac{Q(i)}{E(i)}$, which implies the claim.