CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes Note 0

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. There is a unique such set, 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 the 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 the 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, or the **set difference** between B and A, written as $B - A$ or $B \setminus A$, is the set of elements in B, but not in A: $B \setminus A = \{x \in B \mid x \notin A\}$. For example, if $B = \{1,2,3\}$ and $A = \{3,4,5\}$, then $B \setminus A = \{1,2\}$. For another example, if $\mathbb{R}$ is the set of real numbers and $\mathbb{Q}$ is the set of rational numbers, then $\mathbb{R} \setminus \mathbb{Q}$ is the set of irrational numbers. Here are some important properties of complements:

- $A \setminus A = \emptyset$
- $A \setminus \emptyset = A$
- $\emptyset \setminus A = \emptyset$

Significant Sets

In mathematics, some sets are referred to so commonly that they are denoted by special symbols. These 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.

Products and Power Sets

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 $A \times B = \{(1,u),(1,v),(2,u),(2,v),(3,u),(3,v)\}$. And $\mathbb{N} \times \mathbb{N} = \{(0,0),(1,0),(0,1),(1,1),(2,0),\ldots\}$ is the set of all pairs of natural numbers. Given a set S, 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: $\mathscr{P}(S) = \{\{\},\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$. Note that, if $|S| = k$, then $|\mathscr{P}(S)| = 2^k$. [Why?]

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 $\sum_{i=1}^{n} i$. More generally we can write the sum $f(m)+f(m+1)+\cdots+f(n)$ as $\sum_{i=m}^{n} f(i)$. Thus, for example, $\sum_{i=5}^{n} i^2 = 5^2+6^2+\cdots+n^2$.

Analogously, to write the product $f(m)f(m+1)\cdots f(n)$ we use the notation $\prod_{i=m}^{n} f(i)$. For example, $\prod_{i=1}^{n} i = 1 \cdot 2 \cdots n$ is the product of the first n positive integers.

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 quantifier* $\exists$ ("there exists") is used in the following statement: $(\exists x \in \mathbb{Z})(x < 2 \text{ and } x^2 = 4)$. 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 (take $x = -2$).

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
Fall 2024 Course Notes Note 1

1 Propositional Logic

In order to be fluent in working with mathematical statements, you need to understand the basic framework of the language of mathematics. This first lecture, we will start by learning about what logical forms mathematical theorems may take, and how to manipulate those forms to make them easier to prove. In the next few lectures, we will learn several different methods of proving things.

Our first building block is the notion of a **proposition**, which is simply a statement which is either true or false.

These statements are all propositions:

(1) $\sqrt{3}$ is irrational.

(2) $1+1=5$.

(3) Julius Caesar had 2 eggs for breakfast on his 10^{th} birthday.

These statements are clearly not propositions:

(4) $2+2$.

(5) $x^2+3x=5$. [What is x?]

These statements aren't propositions either (although some books say they are). Propositions should not include fuzzy terms.

(6) Arnold Schwarzenegger often eats broccoli. [What is "often?"]

(7) Henry VIII was unpopular. [What is "unpopular?"]

Propositions may be joined together to form more complex statements. Let P, Q, and R be variables representing propositions (for example, P could stand for "3 is odd"). The simplest way of joining these propositions together is to use the connectives "and", "or" and "not."

(1) **Conjunction**: $P \wedge Q$ ("P and Q"). True only when both P and Q are true.

(2) **Disjunction**: $P \vee Q$ ("P or Q"). True when at least one of P and Q is true.

(3) **Negation**: $\neg P$ ("not P"). True when P is false.

Statements like these, with variables, are called *propositional forms*.

A fundamental principle known as the **law of the excluded middle** says that, for any proposition P, either P is true or $\neg P$ is true (but not both). Thus $P \vee \neg P$ is always true, regardless of the truth value of P. A propositional form that is always true regardless of the truth values of its variables is called a *tautology*. Conversely, a statement such as $P \wedge \neg P$, which is always false, is called a *contradiction*.

Concept check! If we let P stand for the proposition "3 is odd", Q stand for "4 is odd", and R for "$4+5=49$", what are the values of $P \wedge R$, $P \vee R$ and $\neg Q$?

A useful tool for describing the possible values of a propositional form is a **truth table**. Truth tables are the same as function tables: you list all possible input values for the variables, and then list the outputs given those inputs. (The order does not matter.)

Here are truth tables for conjunction and negation:

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

P	$\neg P$
T	F
F	T

Concept check! Write down the truth table for disjunction (OR).

The most important and subtle propositional form is an **implication**:

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

Here, P is called the *hypothesis* of the implication, and Q is the *conclusion*.[1]

We encounter implications frequently in everyday life; here are a couple of examples:

> If you stand in the rain, then you'll get wet.
>
> If you passed the class, you received a certificate.

An implication $P \implies Q$ is false only when P is true and Q is false. For example, the first statement above would be false only if you stood in the rain but didn't get wet. The second statement would be false only if you passed the class but didn't receive a certificate.

[1] P is also sometimes called the *antecedent* and Q the *consequent*.

Here is the truth table for $P \implies Q$ (along with an extra column that we'll explain in a moment):

P	Q	$P \implies Q$	$\neg P \vee Q$
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

Note that $P \implies Q$ is always true when P is false. This means that many statements that sound nonsensical in English are true, mathematically speaking. Examples are statements like: "If pigs can fly, then horses can read" or "If 14 is odd then $1+2=18$." (These are statements that we never make in everyday life, but are perfectly natural in mathematics.) When an implication is stupidly true because the hypothesis is false, we say that it is *vacuously true*.

Note also that $P \implies Q$ is **logically equivalent** to $\neg P \vee Q$, as can be seen by comparing the last two columns of the above truth table: for all possible truth values of P and Q, $P \implies Q$ and $\neg P \vee Q$ take the same values (i.e., they have the same truth table). We write this as $(P \implies Q) \equiv (\neg P \vee Q)$.

$P \implies Q$ is the most common form mathematical theorems take. Here are some of the different ways of saying it:

(1) if P, then Q;

(2) Q if P;

(3) P only if Q;

(4) P is sufficient for Q;

(5) Q is necessary for P;

(6) Q unless not P.

If both $P \implies Q$ and $Q \implies P$ are true, then we say "P if and only if Q" (abbreviated "P iff Q"). Formally, we write $P \iff Q$. Note that $P \iff Q$ is true only when P and Q have the same truth values (both true or both false).

For example, if we let P be "3 is odd," Q be "4 is odd," and R be "6 is even," then the following are all true: $P \implies R$, $Q \implies P$ (vacuously), and $R \implies P$. Because $P \implies R$ and $R \implies P$, we also see that $P \iff R$ is true.

Given an implication $P \implies Q$, we can also define its

(a) **Contrapositive**: $\neg Q \implies \neg P$

(b) **Converse**: $Q \implies P$

The contrapositive of "If you passed the class, you received a certificate" is "If you did not get a certificate, you did not pass the class." The converse is "If you got a certificate, you passed the class." Does the contrapositive say the same thing as the original statement? Does the converse?

Let's look at the truth tables:

P	Q	$\neg P$	$\neg Q$	$P \implies Q$	$Q \implies P$	$\neg Q \implies \neg P$	$P \iff Q$
T	T	F	F	T	T	T	T
T	F	F	T	F	T	F	F
F	T	T	F	T	F	T	F
F	F	T	T	T	T	T	T

Note that $P \implies Q$ and its contrapositive have the *same* truth values everywhere in their truth tables, so they are logically equivalent: $(P \implies Q) \equiv (\neg Q \implies \neg P)$. Many students confuse the contrapositive with the converse: note that $P \implies Q$ and $\neg Q \implies \neg P$ are logically equivalent, but $P \implies Q$ and $Q \implies P$ are not!

When two propositional forms are logically equivalent, we can think of them as "meaning the same thing." We will see in the next lecture how useful this can be for proving theorems.

2 Quantifiers

The mathematical statements you'll see in practice will not be made up of simple propositions like "3 is odd." Instead you'll see statements like:

(1) For all natural numbers n, n^2+n+41 is prime.

(2) If n is an odd integer, so is n^3.

(3) There is an integer k that is both even and odd.

In essence, these statements assert something about lots of simple propositions (even infinitely many!) all at once. For instance, the first statement is asserting that 0^2+0+41 is prime, 1^2+1+41 is prime, and so on. The last statement is saying that, as k ranges over all possible integers, we will find at least one value of k for which the statement is satisfied.

Why are the above three examples considered to be propositions, while earlier we claimed that "$x^2+3x=5$" was not? The reason is that in these examples, there is an underlying "universe" that we are working in, and the statements are *quantified* over that universe. To express these statements mathematically we need two **quantifiers**: The *universal quantifier* $\forall$ ("for all") and the *existential quantifier* $\exists$ ("there exists"). Examples:

(1) "Some mammals lay eggs." Mathematically, "some" means "at least one," so the statement is saying "There exists a mammal x such that x lays eggs." If we let our universe U be the set of mammals, then we can write: $(\exists x \in U)(x \text{ lays eggs})$. (Sometimes, when the universe is clear, we omit U and simply write $\exists x(x \text{ lays eggs})$.)

(2) "For all natural numbers n, n^2+n+41 is prime," can be expressed by taking our universe to be the set of natural numbers, denoted as $\mathbb{N}$: $(\forall n \in \mathbb{N})(n^2+n+41 \text{ is prime})$.

We refer to a statement which refers to a variable as a *predicate* or as a *propositional formula* when replacing the variable with a value makes the statement either true or false. For example, the statement "n^2+n+41 is prime" is a predicate or propositional formula with variable n. The value of the statement on natural numbers, n, is either true or false depending on the value of n. That is, replacing the variable with a value makes the predicate into a proposition. In particular, for $n=1$, we have "43 is prime" which is true. Where for $n=41$, the statement "$41^2+41+41$ is prime" is false as one can factor $41^2+41+41$ into 41×43.

Note that in a *finite* universe, we can express existentially and universally quantified propositions without quantifiers, using disjunctions and conjunctions respectively. For example, if our universe U is $\{1,2,3,4\}$, then $(\exists x \in U)P(x)$ is logically equivalent to $P(1)\vee P(2)\vee P(3)\vee P(4)$, and $(\forall x \in U)P(x)$ is logically equivalent to $P(1)\wedge P(2)\wedge P(3)\wedge P(4)$. However, in an infinite universe, such as the natural numbers, this is not possible.

Concept 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".

Some statements can have multiple quantifiers. As we will see, however, quantifiers do not commute. You can see this just by thinking about English statements. Consider the following (rather gory) example:

> "Every time I ride the subway in New York, somebody gets stabbed."
>
> "There is someone, such that every time I ride the subway in New York, that someone gets stabbed."

The first statement is saying that every time I ride the subway someone gets stabbed, but it could be a different person each time. The second statement is saying something truly horrible: that there is some poor guy Joe with the misfortune that every time I get on the New York subway, there is Joe, getting stabbed again. (Poor Joe will run for his life the second he sees me.)

Mathematically, we are quantifying over two universes: $T = \{\text{times when I ride on the subway}\}$ and $P = \{\text{people}\}$. The first statement can be written: $(\forall t \in T)(\exists p \in P)(p \text{ gets stabbed at time } t)$. The second statement says: $(\exists p \in P)(\forall t \in T)(p \text{ gets stabbed at time } t)$.

Let's look at a more mathematical example:

Consider

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

The first statement says that, given an integer, I 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.

3 Much Ado About Negation

What does it mean for a proposition P to be false? It means that its negation $\neg P$ is true. It's helpful to have some rules for working with negation, as will become more obvious next lecture when we look at proofs.

First, let's look at how to negate conjunctions and disjunctions:

$$\neg(P \wedge Q) \equiv (\neg P \vee \neg Q)$$

$$\neg(P \vee Q) \equiv (\neg P \wedge \neg Q)$$

These two equivalences are known as *De Morgan's Laws*, and they are quite intuitive: for example, if it is not the case that $P \wedge Q$ is true, then either P or Q must be false (and vice versa).

Concept check! Verify both of De Morgan's Laws by writing down the appropriate truth tables.

Negating propositions involving quantifiers actually follows analogous laws. Let's start with a simple example. Assume that the universe is $\{1,2,3,4\}$ and let $P(x)$ denote the propositional formula "$x^2 > 10$." Check that $\exists x P(x)$ is true but $\forall x P(x)$ is false. Observe that both $\neg(\forall x P(x))$ and $\exists x \neg P(x)$ are true because $P(1)$ is false. Also note that both $\forall x \neg P(x)$ and $\neg(\exists x P(x))$ are false, since $P(4)$ is true. The fact that each pair of statements had the same truth value is no accident, as the equivalences

$$\neg(\forall x P(x)) \equiv \exists x \neg P(x)$$

$$\neg(\exists x P(x)) \equiv \forall x \neg P(x)$$

are laws that hold for any proposition P quantified over any universe (including infinite ones).

It is helpful to think of English sentences to convince yourself (informally) that these laws are true. For example, assume that we are working within the universe $\mathbb{Z}$ (the set of all integers), and that $P(x)$ is the proposition "x is odd." We know that the statement $(\forall x P(x))$ is false, since not every integer is odd. Therefore, we expect its negation, $\neg(\forall x P(x))$, to be true. But how would you say the negation in English? Well, if it is not true that every integer is odd, then there must exist some integer which is not odd (i.e., even). How would this be written in propositional form? That's easy, it's just: $(\exists x \neg P(x))$.

To see a more complex example, fix some universe and propositional formula $P(x,y)$. Assume we have the proposition $\neg(\forall x \exists y P(x,y))$ and we want to push the negation operator inside the quantifiers. By the above laws, we can do it as follows:

$$\neg(\forall x \exists y P(x,y)) \equiv \exists x \neg(\exists y P(x,y)) \equiv \exists x \forall y \neg P(x,y).$$

Notice that we broke the complex negation into a smaller, easier problem as the negation propagated itself through the quantifiers. Note also that the quantifiers "flip" as we go.

4 Trickier Quantifier Examples

Let's look at a trickier set of examples:

Write the sentence "there are **at least** three distinct integers x that satisfy $P(x)$" as a proposition using quantifiers! One way to do it is

$$\exists x \exists y \exists z (x \neq y \wedge y \neq z \wedge z \neq x \wedge P(x) \wedge P(y) \wedge P(z)).$$

(Here all quantifiers are over the universe $\mathbb{Z}$ of integers.) Now write the sentence "there are **at most** three distinct integers x that satisfy $P(x)$" as a proposition using quantifiers. One way to do it is

$$\exists x \exists y \exists z \forall d (P(d) \implies d = x \vee d = y \vee d = z).$$

If you're not sure how to interpret this propositional logic statement, you can read it as: "There exist 3 three specific integers x, y, and z, such that if P(d) is true, then d has to be one of those three integers." There are many other ways to express this same statement, for example:

$$\forall x \forall y \forall v \forall z ((x \neq y \wedge y \neq v \wedge v \neq x \wedge x \neq z \wedge y \neq z \wedge v \neq z) \implies \neg(P(x) \wedge P(y) \wedge P(v) \wedge P(z))).$$

[Check that you understand both of the above alternatives.]

Note: We can also generate even more exactly identical alternatives, e.g. writing the contrapositives, propagating negations, etc. In general, you want to pick the propositional logic statement that most clearly expresses the idea.

Finally, what if we want to express the sentence "there are **exactly** three distinct integers x that satisfy $P(x)$"? This is now easy: we can just use the *conjunction* of the two propositions above.

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes 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 $a \mid b$) iff there exists an integer q such that $b = aq$. For example, $2 \mid 10$ because there exists an integer $q = 5$ such that $10 = 5 \cdot 2$. We say a natural number $p \geq 2$ 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 $a \mid b$ and $a \mid c$, then $a \mid (b+c)$.*

Concept check! Let $P(x,y)$ denote "$x \mid y$". 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 $a \mid b$ and $a \mid c$, 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 $a \mid (b+c)$, 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.

Concept check! Give a direct proof of the following statement: For any $a, b, c \in \mathbb{Z}$, if $a \mid b$ and $a \mid c$, then $a \mid (b-c)$.

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

$$(\forall n \in \mathbb{Z}^+)(n < 1000) \implies (\text{sum of } n\text{'s digits divisible by 9} \implies n \text{ divisible by 9}),$$

where $\mathbb{Z}^+$ 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.

$$\begin{aligned}
n \text{ is divisible by } 9 &\implies n = 9l \quad \text{for } l \in \mathbb{Z} \\
&\implies 100a + 10b + c = 9l \\
&\implies 99a + 9b + (a + b + c) = 9l \\
&\implies a + b + c = 9l - 99a - 9b \\
&\implies a + b + c = 9(l - 11a - b) \\
&\implies a + b + c = 9k \quad \text{for } k = l - 11a - b \in \mathbb{Z}
\end{aligned}$$

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.

Concept 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 $d \mid n$, then $n = dl$ 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 is 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!)

As another illustration of proof by contraposition, we will prove a famous theorem called the Pigeonhole Principle. Although the statement of the theorem may seem simple, it has surprising consequences.

Theorem 2.5 (Pigeonhole Principle)**.** *Let n and k be positive integers. Place n objects into k boxes. If $n > k$, then at least one box must contain multiple objects.*

The name of the theorem comes from imagining that the n objects are pigeons and we are trying to place them in pigeonholes.

Proof of Theorem 2.5. We proceed by contraposition. If all boxes contain at most one object, then the number of objects is at most the number of boxes, i.e., $n \leq k$. □

The utility of the theorem stems from the fact that it holds regardless of the *configuration* of the objects in the box. In situations where the objects are placed in the boxes in a complicated way, the conclusions of the theorem can be non-trivial.

Here is an example. A quick search on the Internet reveals that the number of hairs on the human head is roughly, on average, 100000. So, we may be reasonably confident that no human has more than 500000 hairs on his or her head. On the other hand, the population of San Francisco (as of 2016) exceeds 800000. If we think of the residents of San Francisco as "pigeons" and the number of hairs on a resident's head as the "box" into which he or she falls, then the Pigeonhole Principle allows us to conclude the intriguing fact that *there are two people in San Francisco with exactly the same number of hairs on their heads*!

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.

Concept 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 2.6. *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 2.6 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 2.6. Its proof will be deferred to a future lecture in which we learn about induction.

[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 millennia? Music? Fashion? All of these are quickly outdated with time. But mathematics is, in a sense, timeless.

Lemma 2.1. *Every natural number greater than one is either prime or has a prime divisor.*

Proof of Theorem 2.6. We proceed by contradiction. Suppose that Theorem 2.6 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, $p \mid q$ and $p \mid r$, implying $p \mid (q - r)$. 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 2.7. *$\sqrt{2}$ is irrational.*

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 2.6 and Theorem 2.7 share something fundamental in common — in both cases, we wish to show that something *doesn't* exist. For example, for Theorem 2.6, we wished to show that a largest prime doesn't exist, and for Theorem 2.7, 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 2.7, 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 2.7. 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 $R \wedge \neg R$ 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 2.8. *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 2.8 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 2.8? 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 **non-constructive** 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 $P \implies$ 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 $4 \leq 1$. ♠

Concept 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?)

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes

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:

Concept 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 also holds 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 it holds for $k+1$, the inductive step implies that it holds for $k+2$; and so on. 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 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.

Induction Hypothesis: For arbitrary $n = k \geq 0$, assume that $\sum_{i=0}^{k} i = \frac{k(k+1)}{2}$. In words, the Induction 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 $\sum_{i=0}^{k+1} i = \frac{(k+1)(k+2)}{2}$:

$$\sum_{i=0}^{k+1} i = \left(\sum_{i=0}^{k} i\right) + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}, \tag{3}$$

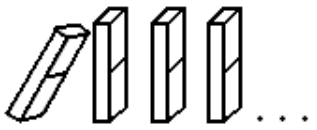
where the second equality follows from the Induction Hypothesis. By the principle of mathematical induction, the claim follows. □

Concept 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. **Induction Hypothesis**: For arbitrary $k \geq 0$, assume that $P(k)$ is true.
3. **Inductive Step**: With the assumption of the Induction Hypothesis in hand, show that $P(k+1)$ is true.

Let us visualize how these three steps fit together using dominoes. Let the statements $P(i)$ be represented by a sequence of dominoes, numbered $0, 1, 2, \ldots, n$, 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 $k+1^{\text{st}}$. 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, it in turn knocks over the $k+1^{\text{st}}$ domino. The base case ($n = 0$) knocks over the 0^{th} 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 $a \mid b$, if and only if there exists an integer q satisfying $b = aq$.

Theorem 3.2. *For all $n \in \mathbb{N}$, $n^3 - n$ is divisible by 3.*

Proof. We proceed by induction over n. Let $P(n)$ denote the statement that $n^3 - n$ is divisible by 3.

Base case ($n = 0$): $P(0)$ asserts that $3 \mid (0^3 - 0)$ or $3 \mid 0$, which is true since any non-zero integer divides 0.

Induction Hypothesis: Assume for $n = k \geq 0$ that $P(k)$ is true. That is, $3 \mid (k^3 - k)$, or equivalently $k^3 - k = 3q$ for some integer q.

Inductive Step: We show that $P(k+1)$ is true, i.e., that $3 \mid ((k+1)^3 - (k+1))$. To show this, we expand the number $(k+1)^3 - (k+1)$ as follows:

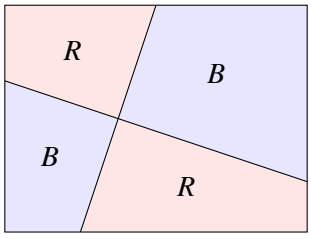
$$\begin{aligned}(k+1)^3 - (k+1) &= k^3 + 3k^2 + 3k + 1 - (k+1) \\ &= (k^3 - k) + 3k^2 + 3k \\ &= 3q + 3(k^2 + k) \text{ for some } q \in \mathbb{Z} \qquad \text{(by the Induction Hypothesis)} \\ &= 3(q + k^2 + k)\end{aligned}$$

We conclude that $3|((k+1)^3 - (k+1))$. Thus, by the principle of induction, $\forall n \in \mathbb{N}$, $3|(n^3 - n)$. □

Concept check! How exactly did the Induction Hypothesis help us show the third equality in the displayed derivation above?

Two Color Theorem. We now consider a more advanced proof by induction, which establishes 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[1] 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 by Appel and Haken. (For an interesting history of the problem and a 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 straight lines, such that each line divides the rectangle into two regions. Can we color such a simplified 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 of the above form 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 we can color the entire map using a single color.

Induction Hypothesis: For some arbitrary $n = k \geq 0$, assume $P(k)$.

[1] Countries are defined to be *adjacent* if they share any non-trivial part of a border, i.e., anything more than a point.

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 valid coloring. With this in mind, let us place back the line we removed, and leave the colors on one side of the line unchanged. On the other side of the line, swap red $\leftrightarrow$ blue. This is illustrated in Figure 1. We claim that this is a valid two-coloring of the map with $k+1$ lines.

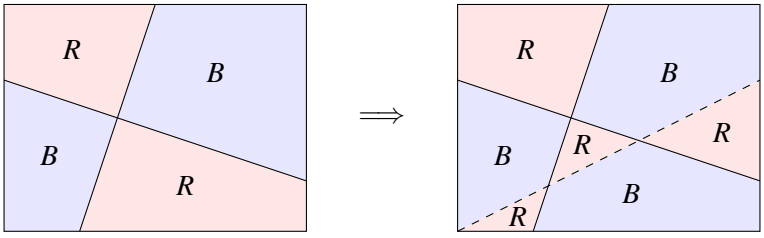


Figure 1: (Left) The map with the $(k+1)$st line removed. (Right) The map with the $(k+1)$st line shown as a dashed line.

Why does this work? Consider any 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 first 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: for all $n \geq 1$, 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.

Induction 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 equal to 1^2.

Induction 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 a 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: for all $n \geq 1$, $\sum_{i=1}^{n} \frac{1}{i^2} \leq 2$.

Concept 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 $\sum_{i=1}^{k} \frac{1}{i^2} + \frac{1}{(k+1)^2} \leq 2$? (Hint: Is it possible that $\sum_{i=1}^{k} \frac{1}{i^2}$ actually *equals* 2?)

Now let's again do something counterintuitive — let's prove the following *stronger* statement, i.e., let's strengthen our induction hypothesis.

Theorem 3.5. *For all $n \geq 1$,* $\displaystyle\sum_{i=1}^{n} \frac{1}{i^2} \leq 2 - \frac{1}{n}$.

Proof. We proceed by induction on n.

Base Case ($n = 1$): We have $\displaystyle\sum_{i=1}^{1} \frac{1}{i^2} = 1 \leq 2 - \frac{1}{1}$, as required.

Induction 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

$$2-\frac{1}{k}+\frac{1}{(k+1)^2}\leq 2-\frac{1}{k+1} \implies -\frac{1}{k}+\frac{1}{(k+1)^2}\leq -\frac{1}{k+1}. \tag{4}$$

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 induction 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 dominoes 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 dominoes 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[2] 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 $15 = 4\cdot 0+5\cdot 3$.

Induction 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. □

[2] This problem also goes under the name of the Postage Stamp Problem, which states that every amount of postage of 12 cents or more can be paid using only 4- and 5-cent stamps.

Concept 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 one or more 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.

Induction Hypothesis: Assume $P(n)$ 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 since $k+1$ is trivially the product of one prime (itself). 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 $x,y \leq k$). But this implies that $k+1$ can also be written as a product of primes. □

Concept check! Why does this proof fail if we instead use simple induction? (Hint: Suppose $k+1 = 42$. 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. Try to convince yourself of this claim.

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 there lived a famous Italian mathematician known as *Fibonacci*[3], who in 1202 considered the following 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 productive, we also have $F(2) = 1$.

[3]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.

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. The following exercise gives a lower bound on this exponential rate of growth. (The true rate of growth is rather larger, around 1.6^n.)

Exercise. Use induction to show that the Fibonacci numbers satisfy $F(n) \geq 2^{(n-1)/2}$ for all $n \geq 3$. Note that you will need *two* base cases: $n = 3$ and $n = 4$. (Why?)

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 $F(n)$:

```
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? You should be able to show by induction that the number of calls is at least $F(n)$ itself!

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 first word of that page comes after W, we recurse on the first half of the dictionary (before the middle page); else, we recurse on the second half of the dictionary (from the middle page onwards). (When the dictionary has an even number $2m$ of pages, we define the middle page to be the $(m+1)$st page.) 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
2   //     dictionary with at least 1 page.
3   // postcondition: Either the definition of W is returned,
4   //     or "W not found" is returned.
5   findWord(W, D) {
6       // Base case
7       if (D has precisely one page)
8           Look for W in D by brute force.
9           If found, return its definition; else, return "W not found".
10
11      // Recursive case
12      Let W' be the first word on the middle page of D.
13      if (W comes before W')
14          return findWord(first half of D)
15      else
16          return findWord(second half of D)
17  }
```

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 8 searches via brute force for W. Thus, if W is present, it is found and returned; else, we return "W not found", as desired.

Induction Hypothesis: Assume that `findWord()` is correct for all $1 \leq n \leq k$.

Inductive Step: We prove `findWord()` is correct for $n = k+1$. After line 12, 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 14, and in the second case we recurse on the second half of D in line 16. By the Induction Hypothesis, the recursive call correctly finds W in the first or second half, or returns "W not found", since both of these two halves contain at most k pages. Since we return the recursive call's answer, we conclude that findWord() correctly finds W in D of size $n = k+1$. By induction, findWord() is therefore correct. □

Concept 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 renowned mathematician George Pólya, 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.

Induction Hypothesis: Assume $P(n)$ holds for some arbitrary $n \geq 1$.

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 induction 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 $1^2 + 2^2 + 3^2 + \cdots + n^2 = \frac{1}{6}n(n+1)(2n+1)$.

2. 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 $1+x > 0$.

3. 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: For all $n \in \mathbb{N}$, $n > 1 \implies n! < n^n$.

   Prove this theorem using induction. (Hint: In the Inductive Step, write $(n+1)! = (n+1) \cdot n!$, and use the Induction Hypothesis.)

4. 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 if there is one, and that the number of questions is at most $3n-4$.

5. Use the proof of Theorem 3.6 to design an algorithm that, given any amount of postage of at least 12c, outputs the numbers of 4c and 5c stamps that make up this postage. What's the largest number of 5c stamps your algorithm will ever use?

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes Note 4

1 The Stable Matching Problem

In the previous two notes, we discussed several proof techniques. In this note, we apply some of these techniques to analyze the solution to an important problem known as the *Stable Matching Problem*, which we now introduce. The Stable Matching Problem is one of the highlights of the field of algorithms.

Suppose you run an employment system[1], and your task is to match up n jobs and n candidates. Each job has an ordered *preference list* of the n candidates, and each candidate has a similar list of the n jobs. For example, consider the case of $n = 3$, i.e., three jobs at Approximation Inc., Basis Co., and Control Corp., and three candidates[2] Anita, Bridget, and Christine, with the following preference lists (lists are ordered from most favorable to least favorable):

Jobs	Candidates		
Approximation Inc.	Anita	Bridget	Christine
Basis Co.	Bridget	Anita	Christine
Control Corp.	Anita	Bridget	Christine

Candidates	Jobs		
Anita	Basis Co.	Approximation Inc.	Control Corp.
Bridget	Approximation Inc.	Basis Co.	Control Corp.
Christine	Approximation Inc.	Basis Co.	Control Corp.

What you would like to do as head of the employment system is to match each job with a candidate. For example, two possible matchings are {(Approximation Inc., Anita), (Basis Co., Bridget), (Control Corp., Christine)} and {(Approximation Inc., Bridget), (Basis Co., Christine), (Control Corp., Anita)}. However, you don't want just any old matching! In order for your employment system to be successful, you wish the matching to make "everyone happy", in that nobody can realistically hope to benefit by switching jobs. 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 *Propose-and-Reject algorithm* (a.k.a. the Gale-Shapley algorithm), and we present it now.

2 The Propose-and-Reject Algorithm

We think of the algorithm as proceeding in "days" to have a clear unambiguous sense of discrete time.

[1] If you'd like, think of a hypothetical system that could be run by the EECS department for student internships.

[2] For clarity of exposition in written English, we will use female pronouns for candidates and neutral pronouns for jobs. If we were to use neutral pronouns for both, there might be confusion between inanimate jobs and living candidates.

Every Morning: Each job proposes (i.e. makes an offer) to the most preferred candidate on its list who has not yet rejected this job.

Every Afternoon: Each candidate collects all the offers she received in the morning; to the job offer she likes best among these, she responds "maybe" (she now has it *in hand* or *on a string*), and to the other offers she says "no" (i.e., she rejects them). (This is just a way for us to virtually model that there are no "exploding offers" and a job can't withdraw an offer once an offer is made.)

Every Evening: Each rejected job crosses off the candidate who rejected its offer from its list.

The above loop is repeated each successive day until there are no offers rejected. At that point, each candidate has a job offer in hand (i.e. on a string); and on this day, each candidate accepts their offered job (i.e. the job offer she has in hand) and the algorithm terminates.

Let's understand the algorithm by running it on our example above. The following table shows which jobs make offers to which candidates on the given day (the jobs in bold are "on a string"):

Days	Candidates	Offers
1	Anita	**Approximation Inc.**, Control Corp.
	Bridget	**Basis Co.**
	Christine	—
2	Anita	**Approximation Inc.**
	Bridget	**Basis Co.**, Control Corp.
	Christine	—
3	Anita	**Approximation Inc.**
	Bridget	**Basis Co.**
	Christine	**Control Corp.**

Thus, the algorithm outputs the matching: {(Approximation Inc., Anita), (Basis Co., Bridget), (Control Corp., 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 matchings 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 Matching 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 matching 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[3]! 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 Propose-and-Reject Algorithm, resulting in a stable matching.

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 Propose-and-Reject Algorithm. (The moral of the story? Careful modeling with appropriate abstractions pays off.)

4 Properties of the Propose-and-Reject Algorithm

There are two properties we wish to show about the propose-and-reject algorithm: First, that it doesn't run forever, i.e., it halts; and second, that it outputs a "good" (i.e., stable) matching. The former is easy to show; let us do it now.

Lemma 4.1. *The propose-and-reject algorithm always halts.*

Proof. The argument is simple: On each day that the algorithm does not halt, at least one job must eliminate some candidate from its 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 iterations (days). □

Next, we'd like to show that the algorithm finds a "good" matching. Before we do so, let's clarify what we mean by "good".

4.1 Stability

What properties should a good matching 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 much more basic criterion that is rooted in the idea of autonomy, namely *stability*. A matching is **unstable**[4] if there is a job and a candidate who both prefer working with each other over their current matchings. We will call[5] such a pair a *rogue couple*. So a matching of n jobs and n candidates is **stable** if it has no rogue couples. Let us now recall our example from earlier:

[3] Any resemblance to the frustration experienced by Berkeley students seeking internships in the summer and having to interview in the fall is not coincidental.

[4] Why is this called "unstable?" Because in such a situation, the rogue candidate could just renege on their official offer and the rogue job/employer could just fire the person that they officially hired to hire their preferred rogue candidate instead. Then one job is suddenly empty and one innocent person just got fired. This is not what we want to see happening. We want everyone to be happy enough that they all want to follow through on their final accepted offers.

[5] The choice of words for definitions is something that requires a balance of considerations. On the one hand, definitions in mathematical English have to be precise and unambiguous, and so to a computer, it doesn't matter what specific English words

Jobs	Candidates		
Approximation Inc.	Anita	Bridget	Christine
Basis Co.	Bridget	Anita	Christine
Control Corp.	Anita	Bridget	Christine

Candidates	Jobs		
Anita	Basis Co.	Approximation Inc.	Control Corp.
Bridget	Approximation Inc.	Basis Co.	Control Corp.
Christine	Approximation Inc.	Basis Co.	Control Corp.

Concept check! Consider the following matching for the example above: {(Approximation Inc., Christine), (Basis Co., Bridget), (Control Corp., Anita)}. Why is this matching unstable? (Hint: Approximation Inc. and Bridget are a rogue couple — why?)

Here is a stable matching for our example: {(Basis Co., Anita), (Approximation Inc., Bridget), (Control Corp., Christine)}. Why is (Approximation Inc., Anita) *not* a rogue couple here? It's certainly true that Approximation Inc. would rather work with Anita than its current employee Bridget. Unfortunately for it, however, Anita would rather be with her current employer (Basis Co.) than with Approximation Inc.. Note also that both Control Corp. and Christine are paired with their *least* favorite choice in this matching. Nevertheless, this does not violate stability since none of their more preferred choices would rather work with them than who they are matched with.

Before we discuss how to find a stable matching, let us ask a more basic question: Do stable matchings always exist? Surely the answer is yes, since we could start with any matching and seemingly make it more and more stable as follows: If there is a rogue couple, modify the current matching so that this couple is matched up. Repeat. Surely this procedure must result in a stable matching? 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. So it is not at all clear that this procedure will terminate!

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 our view of stable matching with asymmetric partners of different types, a person can be paired with any of the other $2n-1$ people, i.e., there is no asymmetry in types). Now, suppose our approach of iteratively matching up rogue couples indeed *did* work. Since this approach does not exploit the asymmetry, we can apply it to the roommates problem. Thus, by the (flawed) reasoning above, we could conclude that there must always exist a stable matching 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 matching!

---

we use for the definition. But our definitions are going to be used by humans, and we would like to allow people to leverage their intuition, etc. At the same time, we want the definitions to not be too cumbersome to use. Here, there is often a balancing act that we must follow between over-triggering the incredibly nuanced set of human intuitions and having concise wording. In this case, human intuition would suggest *potentially rogue couple* instead of *rogue couple* to be more accurate, since they aren't actually matched up to each other. However, since "potentiality" isn't something we want to model more fully, we just go with the more concise wording.

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 matching. For example, the matching {(A,B), (C,D)} contains the rogue couple B and C. How about {(B,C), (A,D)}? This matching is unstable because now A and C are a rogue couple.

Concept check! Verify that in this example there is *no* stable matching!

We conclude that any proof that there always exists a stable matching in the stable matching problem *must* use the fact that there are two different types[6] in an essential way. In the next section we give such a proof, and a comforting one at that: We prove that there must exist a stable matching because the propose-and-reject algorithm always outputs one!

4.2 Analysis

We now prove that the propose-and-reject algorithm always outputs a stable matching. Why should this be the case? Consider the following intuitive observation:

Observation 4.1. *Each job begins the algorithm with its first choice being a possibility; as the algorithm proceeds, however, its best available option can only get* worse *over time. In contrast, each candidate's offers can only get* better *with time.*

Thus, as the algorithm progresses, each job's options get worse, while each candidate's improves; at some point, the jobs and the candidates must "meet" in the middle, and intuitively such a matching should be stable.

Let us formalize this intuition beginning with a formal statement of Observation 4.1 via the following lemma.

Lemma 4.2 (Improvement Lemma)**.** *If job J makes an offer to candidate C on the kth day, then on every subsequent day C has a job offer in hand (on a string) which she likes at least as much as J.*

Proof. We proceed by induction on the day i, with $i \geq k$.

Base case $(i = k)$: On day k, C receives at least one offer (from J). At the end of day k, she will therefore have an offer in hand either from J or a job she likes more than J, since she chooses the best among her offers.

Inductive Hypothesis: Suppose the claim is true for some arbitrary $i \geq k$.

Inductive Step: We prove the claim for day $i+1$. By the Induction Hypothesis, on day i, C had an offer from job J' on a string which she likes at least as much as J. (Note that J' may be J.) According to the algorithm, J' proposes again to C again on day $i+1$ (since this job offer hasn't yet been rejected by her and

[6] For historical reasons that relate to how cable connectors and fasteners are described in mechanical systems, as well as how linguistics denotes grammatical types on nouns, this category of types is sometimes referred to using the technical term *gender*. You might see references to gender in the literature if you look up stable matchings. However, what matters here is that we need to match up two different kinds of things. It doesn't have to be jobs and candidates. It could packets and network ports, computational jobs and servers, threads and cores, students and slots in classes, etc. As long as the two things are different in nature.

is not allowed to explode). Therefore, at the end of day $i+1$, C will have on a string either J' or an offer from a job she likes more than J'; in both cases, she likes this job at least as much as J. Hence the claim holds for day $i+1$, completing the inductive step. □

Detour: The Well-Ordering Principle. Let's take a moment to consider an alternate proof of Lemma 4.2; in the process, we'll uncover a fundamental principle which is equivalent to induction.

Alternate proof of Lemma 4.2. As in our original proof, the claim certainly holds on day $i = k$. Suppose now, for the sake of contradiction, that the ith day for $i > k$ is the first counterexample where C has either no offer or an offer from some J^* inferior to J on a string. On day $i-1$, she had an offer from some job J' on a string and liked J' at least as much as J. According to the algorithm, J' still has an offer out to C on the ith day. (i.e. offers don't explode and can't be withdrawn) So C has the choice of at least one job (J') on the ith day; consequently, her best choice must be at least as good as J', and therefore certainly better than J^* or nothing. 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 is *also* a proof by induction! Why? Well, we began by establishing the base case of $i = k$. Next, instead of proving that for all i, $P(i) \implies P(i+1)$, we showed that the *negation* of this statement is false, i.e., that "there exists i such that $\neg(P(i) \implies P(i+1))$" does not hold; thus, by the law of the excluded middle, it must indeed hold that for all i, $P(i) \implies P(i+1)$.

Concept 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 says that any non-empty 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.*

Concept check! Consider the following subsets of the natural numbers: $S_1 = \{5, 2, 11, 7, 8\}$, $S_2 = \{n \in \mathbb{N} : n \text{ is odd}\}$, $S_3 = \{n \in \mathbb{N} : n \text{ is prime}\}$. Does each of these sets have a smallest element?

Where did we use the well-ordering principle in our proof? In the line "Suppose…that the ith day for $i > k$ is the first counterexample…" — specifically, if the natural numbers did not obey this principle, then the notion of a *first* counterexample would not be valid. Note also that induction relies on this principle: The statement "for all i, $P(i) \implies P(i+1)$" 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.

Concept check! Do the integers satisfy the well-ordering principle? How about the reals? How about the non-negative reals?

Back to our analysis of the Propose-and-Reject Algorithm. Returning to our analysis, we now prove that when the algorithm terminates, all n candidates have job offers. 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 propose-and-reject algorithm always terminates with a matching.*

Proof. We proceed by contradiction. Suppose that there is a job J that is left unpaired when the algorithm terminates. It must have made an offer to all n of the candidates on its list and been rejected by all of them. By the Improvement Lemma, since its offer was rejected, each of these n candidates has had a better offer in hand since J made an offer to her. Thus, when the algorithm terminates, n candidates have n jobs on a string not including J. So there must be at least $n+1$ jobs. But this is a contradiction, since there are only n jobs by assumption. □

To complete our analysis of the propose-and-reject algorithm, we need to verify the key property that the matching produced by the algorithm is stable.

Theorem 4.1. *The matching produced by the algorithm is always stable.*

Proof. We give a direct proof that, in the matching output by the algorithm, no job can be involved in a rogue couple. Consider any couple (J,C) in the final matching and suppose that J prefers some candidate C^* to C. We will argue that C^* prefers her job to J, so that (J,C^*) cannot be a rogue couple. Since C^* occurs before C in J's list, J must have made an offer to C^* before it made an offer to C. Therefore, by the Improvement Lemma, C^* likes her final job at least as much as J, and therefore prefers it to J. Thus no job J can be involved in a rogue couple, and the matching is stable. □

Notice that we did this proof of stability from the perspective of the jobs and not from the perspective of the candidates.

5 Optimality

In Section 4.2, we showed that the propose-and-reject algorithm always outputs a stable matching. But is this so impressive? Will your employment system really be successful outputting matches which are just good enough? To offer the best service (and to displace the current approach), you would ideally strive for some notion of *optimality* in the solutions you obtain.

Consider, for example, the case of 4 jobs and 4 candidates with the following preference lists (for simplicity, we use numbers and letters below to label the jobs and candidates):

Jobs	Candidates			
1	A	B	C	D
2	A	D	C	B
3	A	C	B	D
4	A	B	C	D

Candidates	Jobs			
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 matchings: $S = \{(1,A), (2,D), (3,C), (4,B)\}$ and $T = \{(1,A), (2,C), (3,D), (4,B)\}$. The fact that there are *two* stable matchings raises the questions: What is the best possible job for each candidate? What is the best possible candidate for each job? For example, let us consider job 2. The trivial guess for defining the best partner for 2 is its first choice, candidate A; unfortunately, A is just

not a realistic possibility for taking this job, as matching this job with A would not be stable, since this job is so low on her preference list and other jobs also prefer A. Indeed, there is no stable matching in which 2 is paired with A. Examining the two stable matchings, we find that the best possible *realistic* outcome for job 2 is to be paired with candidate 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, let's make a more precise and tenable definition of optimality.

Definition 4.2 (Optimal candidate for a job)**.** *For a given job J, the* optimal *candidate for J is the highest rank candidate on J's preference list that J could be paired with in any stable matching.*

In other words, the optimal candidate is the best that a job can do in a matching, *given that the matching is stable*.

Concept check! Who are the optimal candidates for each job A, B, C, and D in our example above? (Hint: The optimal candidate for job 2, as discussed above, is D.)

By definition, the best that each job can hope for is to be paired with its optimal candidate. But can all jobs achieve this optimality *simultaneously*? In other words, is there a stable matching such that each and every job is paired with its optimal candidate? If such a matching exists, we will call it a *job/employer optimal* matching. Returning to the example above, S is a job/employer optimal matching since A is 1's optimal candidate, D is 2's optimal candidate, C is 3's optimal candidate, and B is 4's optimal candidate.

Similarly, we can define an optimal job for a candidate.

Definition 4.3 (Optimal job for a candidate)**.** *For a given candidate C, the* optimal *job for C is the highest-ranked job on C's preference list that C could be paired with in any stable matching.*

In other words, the optimal job is the best job that a candidate could get in a matching, *given that the matching is stable*.

We can define a *candidate optimal* matching, which is the matching in which each candidate is paired with her optimal job.

Concept check! Check that T is a candidate optimal matching.

We can also go in the opposite direction and define the *pessimal* candidate for a job to be the lowest ranked candidate that it is ever paired with in some stable matching. This leads naturally to the notion of a *employer pessimal* matching — can you define it, and also a candidate pessimal matching?

Now, we get to the heart of the matter: Who is better off in the Propose-and-Reject Algorithm — jobs/employers or candidates?

Theorem 4.2. *The matching output by the Propose-and-Reject algorithm is job/employer optimal.*

Proof. Suppose for sake of contradiction that the matching is *not* employer optimal. Then, there exists a day on which some job had its offer rejected by its optimal candidate; let day k be the first such day. On this day, suppose J was rejected by C^* (its optimal candidate) in favor of an offer from J^*. By the definition of optimal candidate, there must exist a stable matching T in which J and C^* are paired together. Suppose

T looks like this: $\{\ldots,(J,C^*),\ldots,(J^*,C'),\ldots\}$. We will argue that (J^*,C^*) is a rogue couple in T, thus contradicting stability.

First, it is clear that C^* prefers J^* to J, since she rejected an offer from J in favor of an offer from J^* during the execution of the propose-and-reject algorithm. Moreover, since day k was the first day when some job had an offer rejected by its optimal candidate, before day k, job J^* had not had its offer rejected by its optimal candidate. Since J^* made an offer to C^* on day k, this implies that J^* likes C^* at least as much as its optimal candidate, and therefore at least as much as C' (its partner in the stable matching T). Therefore, (J^*,C^*) form a rogue couple in T, and so T is not stable. Thus, we have a contradiction, implying the matching output by the propose-and-reject algorithm must be employer/job[7] optimal. □

What proof techniques did we use to prove this Theorem 4.2? Again, it is a proof by induction, structured as an application of the well-ordering principle.

Concept check! Where did we use the well-ordering principle in the proof of Theorem 4.2?

Exercise. Can you rewrite the proof of Theorem 4.2 as a regular induction proof? (Hint: The proof is really showing by induction on k the following statement: For every k, no job gets rejected by its optimal candidate on the kth day.)

We conclude that employers would fare very well by following this algorithm. How about the candidates? The following theorem confirms a sad truth:

Theorem 4.3. *If a matching is employer/job optimal, then it is also candidate pessimal.*

Proof. We proceed by contradiction. Let $T=\{\ldots,(J,C),\ldots\}$ be the employer optimal matching (which we know from Theorem 4.2 is output by the algorithm). Suppose for the sake of contradiction that there exists a stable matching $S=\{\ldots,(J^*,C),\ldots,(J,C'),\ldots\}$ such that job J^* is lower-ranked on C's preference list than job J (i.e., J is not her pessimal job). We will argue that S cannot be stable by showing that (J,C) is a rogue couple in S. By assumption, C prefers job J to J^* since J^* is lower on her list. And J prefers candidate C to its partner C' in S because C is its partner in the employer optimal matching T. Contradiction. Therefore, the employer optimal matching is candidate pessimal. □

In light of these findings, what does the propose-and-reject algorithm teach us? Institutional structure matters greatly in the distributional quality of outcomes, and institutional structure reflects real-world power and its history.

Exercise. What simple modification would you make to the propose-and-reject algorithm so that it always outputs a *candidate optimal* matching?

Let us close with some final comments about the National Residency Matching Program. Until recently the propose-and-reject algorithm was run[8] with the hospitals doing the proposing, and so the matchings

[7] What we have actually proved is that the propose-and-reject algorithm returns a matching that is optimal for the "proposers." In our case, the employers/jobs are the ones that make offers, so it is employer/job optimal. If we ran propose-and-reject with candidates proposing, then it would be candidate-optimal.

[8] Everything happens inside a computer. None of the intermediate offers are actually made to humans, only the final stable offer.

produced were hospital optimal. In the nineties, the roles were reversed[9] 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 matching 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 matchings and related variants remains an active topic of research in EECS (and economics!). Although it is by now 30 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.

[9] This role reversal was possible precisely because the algorithm was already running inside a computer. Switching who makes offers in the outside world would have been far harder to do credibly given the difference in real-world negotiating *power* between medical students and hospitals. This is an important lesson in the ordering of improvements as you try to work towards change in the real world. A non-hospital-optimal automated matching system would never have been adopted in the first place, and then an intern-optimal matching system would have always remained an idealistic fantasy.

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes 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.

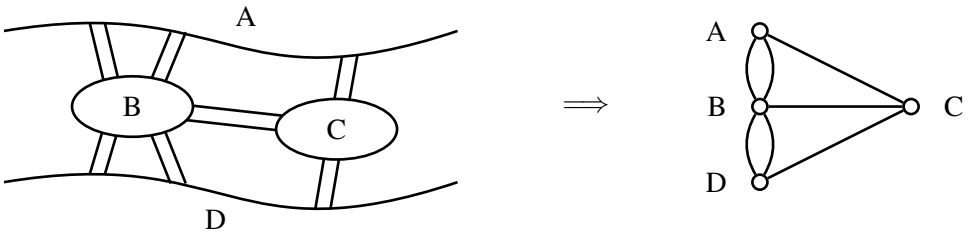


Figure 1: (Left) The city of Königsberg. (Right) The (multi-)graph modeling the bridge connections in Königsberg.

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.

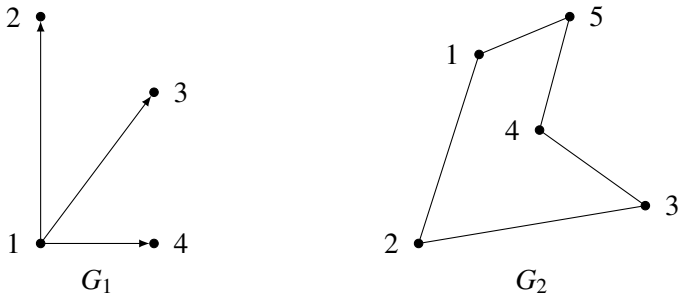


Figure 2: Examples of directed and undirected graphs, respectively.

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.

Concept 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*.

Concept 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., $\text{degree}(u) = |\{v \in V : \{u,v\} \in E\}|$. 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.

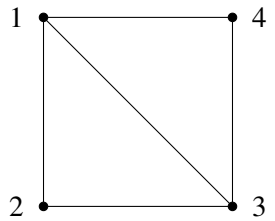
Concept check! What does the degree of a vertex represent in our *undirected* social network in which an edge $\{u,v\}$ means u and v know each other? How should we interpret an isolated vertex?

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.

Concept 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

Concept 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?

In this class, 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.

Concept 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 possibly 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.

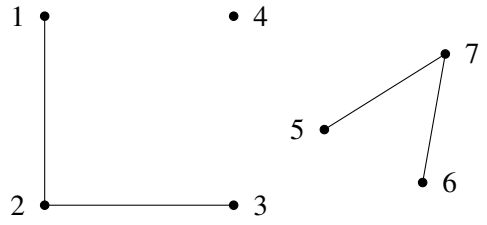
A quick summary of all of the terms:

	no repeated vertices	no repeated edges	start = end
Walk			
Path	✓	✓	
Tour			✓
Cycle	✓*	✓	✓

(*except for start and end vertices)

Remark: In this class, we will do our best to remind you of the subtleties in the definitions of the terms; don't worry too much about the formal definitions, as long as you understand the main idea.

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*.



Concept 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,...,G_k be the connected components
        when the edges in T are removed from G
```

```
    Let s_i be the first vertex in T that intersects G_i
    Output Splice(T, Euler(G_1, s_1), ..., Euler(G_k, s_k))
end EULER
```

We prove by induction on the size of G that $\text{EULER}(G,s)$ 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: $\text{EULER}(G,s)$ 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 $T = \text{FINDTOUR}(G,s)$ 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 $\text{EULER}(G_i, s_i)$ 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. □

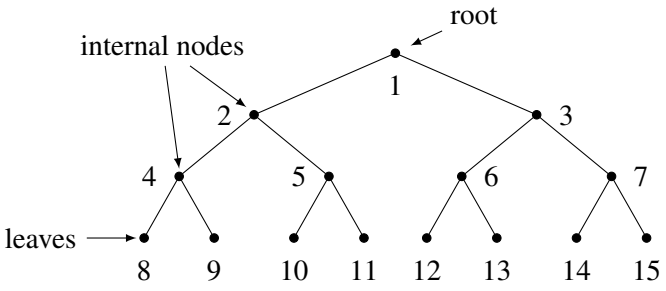
Concept check! Why does Theorem 5.1 imply the answer to the Seven Bridges of Königsberg is no?

---

3 Planarity, Euler's Formula, Coloring.

Trees

We need to discuss trees briefly here, though we will discuss them more later. A graph is a *tree* if it is connected and acyclic (contains no cycles). There are many other equivalent definitions. For example, a tree is a connected graph where the number of vertices is one more than the number of edges. Or, a tree is a connected graph such that if you delete any edge it becomes disconnected.



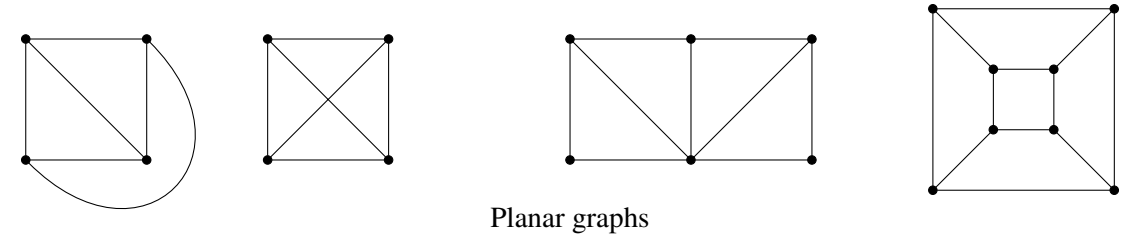
Planar Graphs

A graph is *planar* if it can be drawn on the plane without crossings. For example, the first four graphs shown below are planar. Notice that the first and second graphs are the same, but drawn differently. Even though

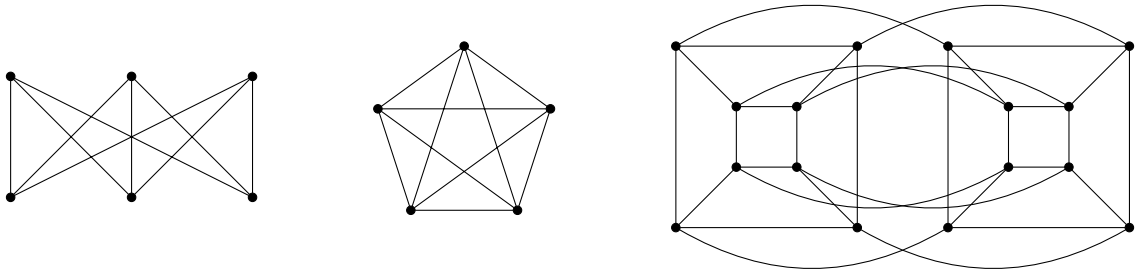
the second drawing has crossings, the graph is still considered planar since it is possible to draw it without crossings.

The other three graphs are not planar. The first one of them is the infamous "three houses-three wells graph," also called $K_{3,3}$, (This notation says there are two sets of vertices, each of size three and all edges between the two sets of vertices are present.) The second is the "complete" graph (every edge is present) with five nodes, or K_5. The third is the four-dimensional cube. We shall soon see how to prove that all three graphs are non-planar.

A useful concept is the notion of a *bipartite* graph, $G = (V,E)$ which is a graph where the vertices are split into two groups and edges only go between groups. More formally, we have $V = L \cup R$ and $E \subseteq L \times R$. Clearly, $K_{3,3}$ is bipartite. Moreover, the four dimensional cube is bipartite: this follows from the fact that every cycle in the cube has even length. We will show the equivalence a bit later.



Planar graphs



Non-planar graphs

When a planar graph is drawn on the plane, one can distinguish, besides its vertices (their number will be denoted v here) and edges (their number is e), the *faces* of the graph (more precisely, of the drawing). The faces are the regions into which the graph subdivides the plane. One of them is infinite, and the others are finite. The number of faces is denoted f. For example, for the first graph shown $f = 4$, and for the fourth (the cube) $f = 6$.

One basic and important fact about planar graphs is *Euler's formula*, $v + f = e + 2$ (check it for the graphs above). It has an interesting story. The ancient Greeks knew that this formula held for all polyhedra (check it for the cube, the tetrahedron, and the octahedron, for example), but could not prove it. How do you do induction on polyhedra? How do you apply the induction hypothesis? What is a polyhedron minus a vertex, or an edge? In the 18th century Euler realized that this is an instance of the inability to prove a theorem by induction *because it is too weak*, something that we saw time and again when we were studying induction. To prove the theorem, one has to generalize polyhedra. And the right generalization is *planar graphs*.

Can you see why planar graphs generalize polyhedra? Why are all polyhedra (without "holes") planar graphs?

Theorem 5.2. *(Euler's formula) For every connected planar graph, $v + f = e + 2$*

Proof. By induction on e. It certainly holds when $e = 0$, and $v = f = 1$. Now take any connected planar graph. Two cases:

- If it is a tree, then $f = 1$ (drawing a tree on the plane does not subdivide the plane), and $e = v - 1$ (check homework).
- If it is not a tree, find a cycle and delete any edge of the cycle. This amounts to reducing both e and f by one. By induction the formula is true in the smaller graph, and so it must be true in the original one.

□

What happens when the graph is not connected? How does the number of connected components enter the formula?

Take a planar graph with f faces, and consider one face. It has a number of *sides*, that is, edges that bound it clockwise. Note that an edge may be counted twice, if it has the same face on both sides, as it happens for example in a tree (such edges are called bridges). Denote by s_i the number of sides of face i. Now, if we add the s_i's we are going to get $2e$, because each edge is counted twice, once for the face on its right and once for the face on its left (they may coincide if the edge is a bridge). We conclude that, in any planar graph,

$$\sum_{i=1}^{f} s_i = 2e \tag{1}$$

Now notice that, since we don't allow parallel edges between the same two nodes, and if we assume that there are at least two edges (so there are at least three vertices), every face has at least three sides, or $s_i \geq 3$ for all i. It follows that $3f \leq 2e$. Solving for f and plugging into Euler's formula we get

$$e \leq 3v - 6.$$

This is an important fact. First it tells us that planar graphs are *sparse*, they cannot have too many edges. A 1,000-vertex connected graph can have anywhere between a thousand and half a million edges. This inequality tells us that for planar graphs the range is very small, between 999 and $2,994$.

It also tells us that K_5 is not planar: Just notice that it has five vertices and ten edges.

$K_{3,3}$ has $v = 6, e = 9$ so it passes the planarity test with flying colors. We must think a little harder to show that $K_{3,3}$ is non-planar. Notice that, if we had drawn it on the plane, there would be no triangles. Because a triangle means that two wells or two houses are connected together, which is false. So, Equation (1) now gives us $4f \leq 2e$, and solving for f and plugging into Euler's formula, $e \leq 2v - 4$, which shows that $K_{3,3}$ is non-planar.

So, we have established that K_5 and $K_{3,3}$ are both non-planar. There is something deeper going on: In some sense, these are *the only non-planar graphs*. This is made precise in the following famous result, due to the Polish mathematician Kuratowski (this is what "K" stands for).

Theorem 5.3. *A graph is non-planar if and only if it contains K_5 or $K_{3,3}$.*

"Contains" here means that one can identify nodes in the graph (five in the case of K_5, six in the case of $K_{3,3}$) which are connected as the corresponding graph through paths (possibly single edges), and such that no two of these paths share no vertex. For example, the 4-cube shown below is non-planar, because it contains $K_{3,3}$, as shown.

Can you find K_5 in the same graph?

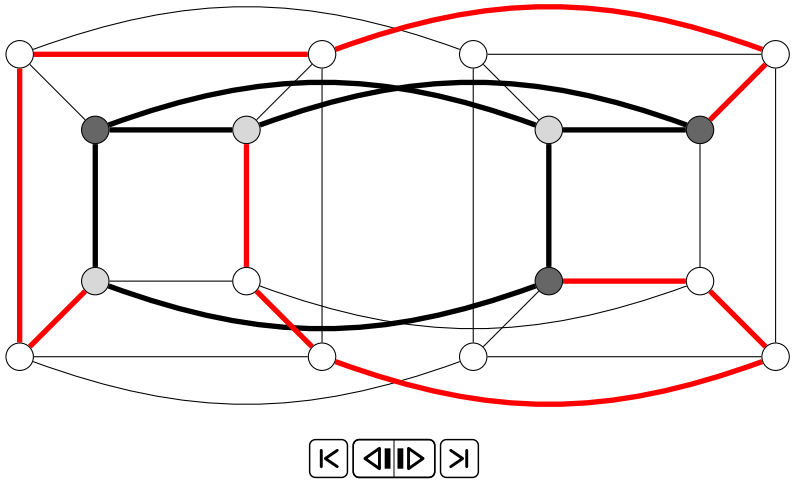


Figure 3: $K_{3,3}$ in a 4-cube (to view the animation, use Adobe Acrobat or another PDF reader supporting PDF animations)

One direction of Kuratowski's theorem is obvious: If a graph contains one of these two non-planar graphs, then of course it is itself non-planar. The other direction, namely that in the absence of these graphs we can draw any graph on the plane, is difficult. For a short proof you may want to type "proof of Kuratowski's theorem" in your favorite search engine.

Duality and Coloring

There is an interesting *duality* between planar graphs. For example, the Greeks knew that the octahedron and the cube are "dual" to each other, in the sense that the faces of one can be put in correspondence with the vertices of the other (think about it). The tetrahedron is self-dual. And the dodecahedron and the icosahedron (look for images in the web if you don't know these pretty things) are also dual to one another.

What does this mean? Take a planar graph G, and assume it has no bridges and no degree-two nodes. Draw a new graph G^*: Start by placing a node on each face of G. Then draw an edge between two faces if they touch at an edge — draw the new edge so that it crosses that edge. The result is G^*, also a planar graph. Notice now that, if you construct the dual of G^*, it is the original graph: $(G^*)^* = G$.

Duality is a convenient consideration when thinking about planar graphs. Also, it tells us that "coloring a political map so that no two countries who share a border have the same color" is the same problem as "coloring the vertices of a planar graph (the dual of the political map) so that no two adjacent vertices have the same color."

Returning for a moment to the concept of a bipartite graph in the context of coloring, one sees that a two colorable graph is equivalent to a bipartite graph as the coloring splits the vertices into two sets where edges only connect vertices in the two sets(colors). Here, with this observation, we can see that a graph that does not contain any odd length cycles is bipartite as we can greedily color any such graph with two colors as follows for a connected graph. Color one vertex, say x red, color its neighbors blue, and then continue by coloring their uncolored neighbors red, and continue alternatively assigning colors to uncolored neighbors. If this does not work, then some edge (u,v) must connect two vertices of the same color, in which case we can identify an odd cycle by using the path p_1 from x to u, the edge (u,v) and the path p_2 from v to x and removing common portions of p_1 and p_2. The lengths of p_1 and p_2 have the same parity (even or odd) as u and v have the same color and the total length of the cycle must therefore be odd as it includes the additional

edge (u, v).

Back to planar graphs, a famous theorem states that four colors are always enough! (Search for "four color theorem".)

We shall prove something weaker:

Theorem 5.4. *Every planar graph can be colored with five colors.*

Before proceeding with the proof, we consider alternative legal colorings of a graph. We first take the subset of vertices of two colors, say 1 and 2, and compute the connected components. We note that we can switch the two colors in a single connected component; consider any edge that is not in the connected component is clearly fine, any edge in the connected component has both endpoints switched which is also fine.

Proof. Induction on v. The base case is not worth talking about, so we go directly to the inductive step. Let G be a planar graph. I claim there is a node of degree five or less. In proof, consider the inequality $e \leq 3v - 6$. If all vertices had degree six or more, then e would be at least $3v$.

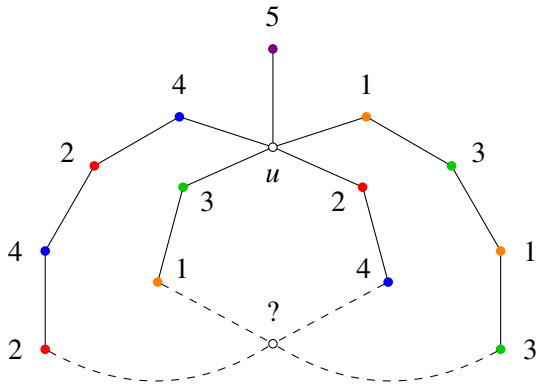
So, consider a node u of degree five or less. If it has degree four or less, we are done: Remove u, color the remaining graph with 5 colors (by induction), and then put u back in and color it by a color that is missing from its neighbors.

So, u must have 5 vertices, and in the coloring of $G - u$ they all got different colors. Look at them clockwise around u, and call them u_1, u_2, u_3, u_4, u_5, and their colors $1, 2, 3, 4, 5$.

Now try to change the color of u_2 to color 4 by determining the connected component containing u_2 and consisting of vertices that are colored 2 or 4. If u_4 is in the connected component, switching 2 and 4 is not helpful. But, we do know that there is a path between u_2 and u_4 colored 2 and 4.

Similarly, we can try to change the color of u_1 to 3, and this will succeed unless there is a path between u_1 to u_3 colored 1 and 3.

If both of these attempts fail, then we get two paths: one from u_1 to u_3 colored 1 and 3, and the other from u_2 to u_4 colored 2 and 4. But planarity says that these two paths must intersect at some vertex. What is the color of this vertex? □



4 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,

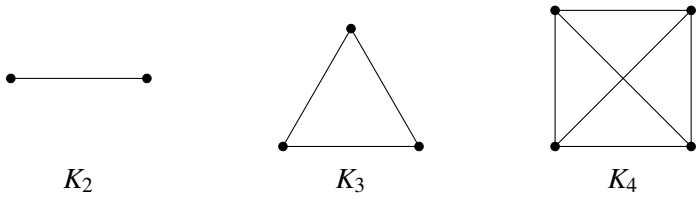
imagine that our graph represents the interconnection between routers on the internet. To send a packet from one node to another, we need to find a path in this graph from our source and destination. Therefore, to be able to send a packet from any node to any other node, we only need the graph to be 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 quadratically 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.

4.1 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 $E = \big\{\{v_i,v_j\} \mid v_i \neq v_j \text{ and } v_i,v_j \in V\big\}$.

Concept 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?

Concept check! What is the minimum number of edges which must be removed from K_n to obtain an isolated vertex?

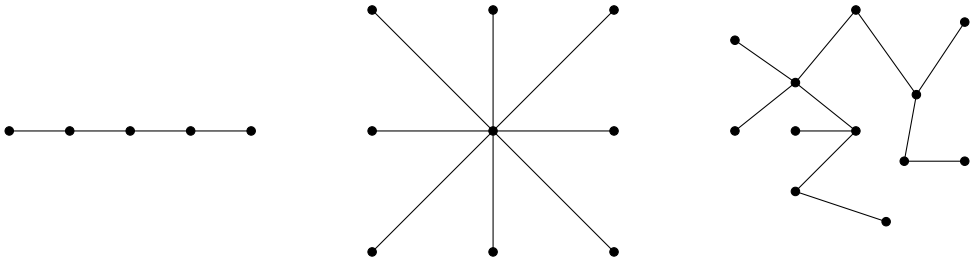
Finally, we can also discuss complete graphs for *directed* graphs, which are defined as you might expect: For any pair of vertices u and v, both $(u,v),(v,u) \in E$.

4.2 Trees

In this section, we return to 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.

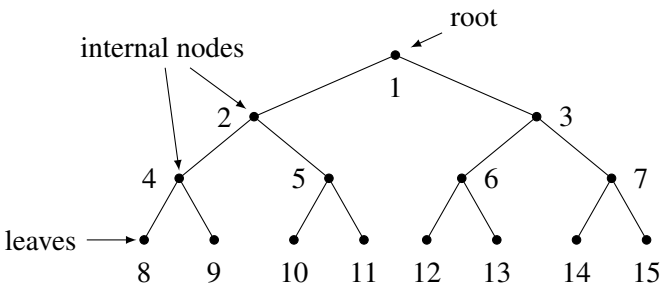
Here are three examples of trees:



Concept 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*. Note that in a rooted tree, a root is never a leaf. In particular, this differs from an unrooted tree; in an unrooted tree, *leaves* are any vertex of degree 1. 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.

Concept 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 5.5. *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.

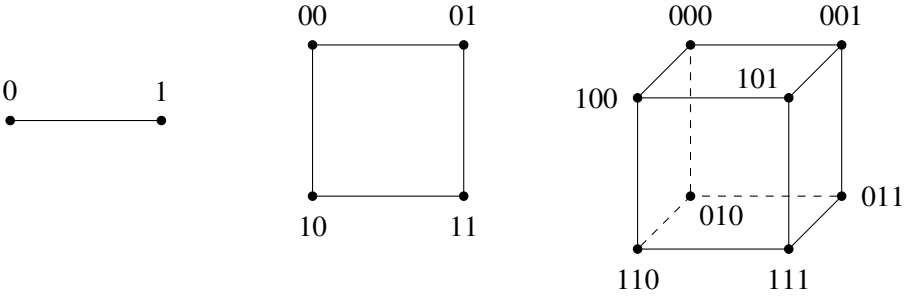
So, assume G' is connected. But now G' is a connected graph with no cycles on k vertices, so we can apply the induction hypothesis to G' to conclude that G' is connected and has $k-1$ edges. Let us now add v back to G' to obtain G. How many edges can be incident on v? Well, since G' is connected, then 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. □

4.3 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.

Concept 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 $V - S = \{v \in V \mid v \notin S\}$ is the set of vertices that are not in S.

Theorem 5.6. *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.,*

$$E_S \coloneqq \{\{u,v\} \in E \mid u \in S \text{ and } v \in V - S\}.$$

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 $V - S$.

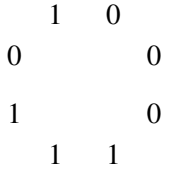
   What about the 0-subcube? Here, we cannot apply the induction hypothesis directly, but there is a way to apply it after a little massaging. Consider the set $V_0 - S_0$, where V_0 is the set of vertices in the 0-subcube. Note that $|V_0| = 2^k$ and $|V_0 - S_0| = |V_0| - |S_0| = 2^k - |S_0| < 2^k - 2^{k-1} = 2^{k-1}$. 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.

□

5 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:

$$(a_1a_2\ldots a_{n-1},\ a_2a_3\ldots a_{n-1}0) \in E \quad \text{and} \quad (a_1a_2\ldots a_{n-1},\ a_2a_3\ldots a_{n-1}1) \in E.$$

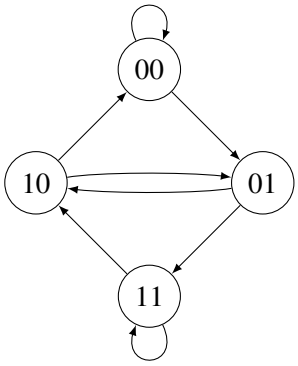
   Therefore, each vertex also has two incoming edges:

$$(0a_1a_2\ldots a_{n-2},\ a_1a_2\ldots a_{n-1}) \in E \quad \text{and} \quad (1a_1a_2\ldots a_{n-2},\ a_1a_2\ldots a_{n-1}) \in E.$$

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 5.5.

   (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 5.5.

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 5.5), 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

In several settings, such as error-correcting codes and cryptography, we sometimes wish to work over a smaller range of numbers. Modular arithmetic is useful in these settings, since it limits numbers to a predefined range $\{0, 1, \ldots, N-1\}$, and wraps around whenever you try to leave this range — like the hand of a clock (where $N = 12$) or the days of the week (where $N = 7$).

Example: Calculating the time When you calculate the time, you automatically use modular arithmetic. For example, if you are asked what time it will be 13 hours from 1 pm, you say 2 am rather than 14. Let's assume our clock displays 12 as 0. This is limiting numbers to a predefined range, $\{0, 1, 2, \ldots, 11\}$. Whenever you add two numbers in this setting, you divide by 12 and provide the remainder as the answer.

If we wanted to know what the time would be 24 hours from 2 pm, the answer is easy. It would be 2 pm. This is true not just for 24 hours, but for any multiple of 12 hours (ignoring the detail of am/pm). What about 25 hours from 2 pm? Since the time 24 hours from 2 pm is still 2 pm, after 25 hours it would be 3 pm. Another way to say this is that we add 1 hour, which is the remainder when we divide 25 by 12.

This example shows that under certain circumstances it makes sense to do arithmetic within the confines of a particular number (12 in this example). That is, we only keep track of the remainder when we divide by 12, and when we need to add two numbers, instead we just add the remainders. This method is quite efficient in the sense of keeping intermediate values as small as possible, and we shall see in later lectures how useful it can be.

More generally, we can define $x \pmod m$ (in words: "x modulo m") to be the remainder r when we divide x by m. I.e., if $x \pmod m \equiv r$, then $x = mq + r$ where $0 \leq r \leq m-1$ and q is an integer. Thus $29 \pmod{12} \equiv 5$ and $13 \pmod 5 \equiv 3$.

Computation

If we wish to calculate $x + y \pmod m$, we would first add $x + y$ and then calculate the remainder when we divide the result by m. For example, if $x = 14$ and $y = 25$ and $m = 12$, we would compute the remainder when we divide $x + y = 14 + 25 = 39$ by 12, and get the answer 3. Notice that we would get the same answer if we first computed $2 \equiv x \pmod{12}$ and $1 \equiv y \pmod{12}$ and added the results modulo 12 to get 3. The same holds for subtraction: $x - y \pmod{12}$ is $-11 \pmod{12}$, which is 1. Again, we could have obtained this directly by simplifying first, i.e., $(x \pmod{12}) - (y \pmod{12}) \equiv 2 - 1 \equiv 1$.

This idea saves us even more effort with multiplication: to compute $xy \pmod{12}$, we could first compute $xy = 14 \times 25 = 350$ and then compute the remainder when we divide by 12, which is 2. Notice that we get the same answer if we first compute $2 \equiv x \pmod{12}$ and $1 \equiv y \pmod{12}$ and simply multiply the results modulo 12.

More generally, while carrying out any sequence of additions, subtractions or multiplications mod m, we get the same answer if we reduce any intermediate results mod m. This can considerably simplify the calculations.

Set Representation

There is an alternative view of modular arithmetic which helps understand all this better. For any integer m, we say that x and y are *congruent modulo m* if they differ by a multiple of m or, in symbols,

$$x \equiv y \pmod{m} \iff m \text{ divides } (x-y).$$

For example, 29 and 5 are congruent modulo 12 because 12 divides $29-5$. We can also write $22 \equiv -2 \pmod{12}$. Notice that x and y are congruent modulo m iff they have the same remainder modulo m; in other words,

$$x \equiv y \pmod{m} \iff x \pmod{m} = y \pmod{m}.$$

What is the set of numbers that are congruent to $0 \pmod{12}$? These are all the multiples of 12: $\{\ldots, -36, -24, -12, 0, 12, 24, 36, \ldots\}$. What about the set of numbers that are congruent to $1 \pmod{12}$? These are all the numbers that give a remainder 1 when divided by 12: $\{\ldots, -35, -23, -11, 1, 13, 25, 37, \ldots\}$. Similarly the set of numbers congruent to $2 \pmod{12}$ is $\{\ldots, -34, -22, -10, 2, 14, 26, 38, \ldots\}$. Notice in this way we get 12 such sets of integers, and every integer belongs to one and only one of these sets.

In general, if we work modulo m, then we get m such disjoint sets whose union is the set of all integers: these are often called *residue classes mod m*. We can think of each set as represented by the unique element it contains in the range $(0, \ldots, m-1)$. The set represented by element i would be all numbers z such that $z = mx + i$ for some integer x. Observe that all of these numbers have remainder i when divided by m; they are therefore congruent modulo m.

We can understand the operations of addition, subtraction and multiplication in terms of these sets. When we add two numbers, say $x \equiv 2 \pmod{12}$ and $y \equiv 1 \pmod{12}$, it does not matter which x and y we pick from the two sets, since the result is always an element of the set that contains 3. The same is true about subtraction and multiplication. It should now be clear that the elements of each set are interchangeable when computing modulo m, and this is why we can reduce any intermediate results modulo m.

Here is a more formal way of stating this observation:

Theorem 6.1. *If $a \equiv c \pmod{m}$ and $b \equiv d \pmod{m}$, then $a+b \equiv c+d \pmod{m}$ and $a \cdot b \equiv c \cdot d \pmod{m}$.*

Proof. We know that $c = a + k \cdot m$ and $d = b + \ell \cdot m$ for integers k, ℓ, so $c + d = a + k \cdot m + b + \ell \cdot m = a + b + (k + \ell) \cdot m$, which means that $a + b \equiv c + d \pmod{m}$. The proof for multiplication is similar and left as an exercise. □

Exercise. Complete the proof of Theorem 6.1 for multiplication.

What this theorem tells us is that we can always reduce any arithmetic expression modulo m to a number in the range $\{0, 1, \ldots, m-1\}$. As an example, consider the expression $(13+11) \cdot 18 \pmod{7}$. Using the above theorem several times we can write:

$$\begin{aligned}
(13+11) \cdot 18 &\equiv (6+4) \cdot 4 \pmod{7} \\
&= 10 \cdot 4 \pmod{7} \\
&\equiv 3 \cdot 4 \pmod{7} \\
&= 12 \pmod{7} \\
&\equiv 5 \pmod{7}.
\end{aligned}$$

In summary, we can always do basic arithmetic (multiplication, addition, subtraction) calculations modulo m by reducing intermediate results modulo m. (Note that we haven't mentioned division: much more on that later!)

2 Exponentiation

Another standard operation in arithmetic algorithms (used heavily, e.g., in primality testing and RSA) is raising one number to a power modulo another number. I.e., how do we compute $x^y \pmod m$, where x, y, m are natural numbers and $m > 0$? A naïve approach would be to compute the sequence $x \pmod m, x^2 \pmod m, x^3 \pmod m, \ldots$ up to y terms, but this requires time exponential in the number of bits in y. We can do much better using the trick of *repeated squaring*:

```
algorithm mod-exp(x, y, m)
  if y = 0 then return (1)
  else
    z = mod-exp(x, y div 2, m)
    if y (mod 2) ≡ 0 then return (z * z (mod m))
    else return (x * z * z (mod m))
```

This algorithm uses the fact that any $y > 0$ can be written as $y = 2a$ or $y = 2a+1$, where $a = \lfloor \frac{y}{2} \rfloor$ (which we have written as `y div 2` in the above pseudo-code), plus the facts

$$x^{2a} = (x^a)^2; \qquad \text{and}$$
$$x^{2a+1} = x \cdot (x^a)^2.$$

Exercise. Use the above facts to prove by induction on y that the algorithm always returns the correct value.

What is its running time? The main task here, as is usual for recursive algorithms, is to figure out how many recursive calls are made. But we can see that the second argument, y, is being (integer) divided by 2 in each call, so the number of recursive calls is exactly equal to the number of bits, n, in y. (The same is true, up to a small constant factor, if we let n be the number of decimal digits in y.) Thus, if we charge only constant time for each arithmetic operation (`div`, `mod` etc.) then the running time of `mod-exp` is $O(n)$. Note that this is *very* efficient: it means that we can handle exponents with (at least) thousands of bits!

In a more realistic model (where we count the cost of operations at the bit level), we would need to look more carefully at the cost of each recursive call. Note first that the test on y in the `if`-statement just involves looking at the least significant bit of y, and the computation of $\lfloor \frac{y}{2} \rfloor$ is just a shift in the bit representation. Hence each of these operations takes only constant time. The cost of each recursive call is therefore dominated by the mod operation[1] in the final result. A fuller analysis of such algorithms is performed in CS170.

3 Bijections

Before talking about division, we are going to have to take a little detour to talk about inverses. From linear algebra and calculus, you already have a lot of intuition about when inverses do and do not exist. Recall

[1] You may want to analyze grade-school long-division for binary numbers to understand how long a mod operation would take. Since all arithmetic is being done mod m, the cost of this operation depends only on the number of bits in m and x (and not on y).

that a square matrix has an inverse only if it does not have a non-trivial nullspace. Why? Because if it has a non-trivial nullspace, it maps lots of vectors to the zero vector and there is no way to recover the information that was lost. The concept of bijection just allows us to formalize the mathematical intuition that we already have.

A function is a mapping from a set (called the *domain*) of inputs A to a set of outputs B: for input $x \in A$, $f(x)$ must be in the set B. To denote such a function, we write $f : A \to B$.

Consider the following examples of functions, where both functions map $\{0, \ldots, m-1\}$ to itself:

$$f(x) \equiv x+1 \pmod{m}$$
$$g(x) \equiv 2x \pmod{m}$$

A *bijection* is a function for which every $b \in B$ has a unique *pre-image* $a \in A$ such that $f(a) = b$. Note that this consists of two conditions:

1. f is *onto*: every $b \in B$ has a pre-image $a \in A$.
2. f is *one-to-one*: for all $a, a' \in A$, if $f(a) = f(a')$ then $a = a'$.

Looking back at our examples, we can see that f is a bijection; the unique pre-image of y is $y-1$. However, g is only a bijection if m is odd. Otherwise, it is neither one-to-one nor onto. The following lemma can be used to prove that a function is a bijection:

Lemma: For a finite set A, $f : A \to A$ is a bijection if there is an *inverse* function $g : A \to A$ such that $\forall x \in A$ $g(f(x)) = x$.

Proof. If $f(x) = f(x')$, then $x = g(f(x)) = g(f(x')) = x'$. Therefore, f is one-to-one. Since f is one-to-one, there must be $|A|$ elements in the range of f. This implies that f is also onto. (Can you see why? Prove this last fact for yourself. What kind of proof should you try?) □

The finiteness of the sets involved here make our life easier.

4 Inverses

We have so far discussed addition, multiplication and exponentiation. Subtraction is the inverse of addition and just requires us to notice that subtracting b modulo m is the same as adding $-b \equiv m - b \pmod{m}$.

What about division? This is a bit harder. Over the reals dividing by a number x is the same as multiplying by $y = 1/x$. Here y is that number such that $x \cdot y = 1$. Of course we have to be careful when $x = 0$, since such a y does not exist. Similarly, when we wish to divide by $x \pmod{m}$, we need to find $y \pmod{m}$ such that $x \cdot y \equiv 1 \pmod{m}$; then dividing by x modulo m will be the same as multiplying by y modulo m. Such a y is called the *multiplicative inverse* of x modulo m. In our present setting of modular arithmetic, can we be sure that x has an inverse mod m, and if so, is it unique (modulo m) and can we compute it?

As a first example, take $x = 8$ and $m = 15$. Then $2x = 16 \equiv 1 \pmod{15}$, so 2 is a multiplicative inverse of 8 mod 15. As a second example, take $x = 12$ and $m = 15$. Then the sequence $\{ax \pmod{m} : a = 1, 2, 3, \ldots\}$ is periodic, and takes on the values $(12, 9, 6, 3, 0, 12, 9, 6 \ldots)$. [Exercise: check this!] Thus 12 *has no multiplicative inverse mod 15* since the number 1 never appears in this sequence.

This is the first warning sign that working in modular arithmetic might actually be very different from grade-school arithmetic. Two weird things are happening. First, no multiplicative inverse seems to exist for

a number that isn't zero. (In normal arithmetic, the only number that has no inverse is zero.) Second, the "times table" for a number that isn't zero has zero showing up in it! So, e.g., 12 times 5 is equal to zero when we are considering numbers modulo 15. (In normal arithmetic, zero never shows up in the multiplication table for any number other than zero.)

So, when *does* x have a multiplicative inverse modulo m? The answer is: if and only if the greatest common divisor of m and x is 1. Moreover, when the inverse exists it is unique. Recall that 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 $\gcd(x,y)$ is 1, it means that x and y share no common factors (except 1). This is often expressed by saying that x and y are *relatively prime* or *coprime*.

Theorem 6.2. *Let m,x be positive integers such that $\gcd(m,x) = 1$. Then x has a multiplicative inverse modulo m, and it is unique (modulo m).*

Proof. Consider the sequence of m numbers $0, x, 2x, \ldots (m-1)x$. We claim that these are all distinct modulo m. Since there are only m distinct values modulo m, it must then be the case that $ax \equiv 1 \pmod{m}$ for exactly one a (modulo m). This a is the unique multiplicative inverse of x.

To verify the above claim, suppose for contradiction that $ax \equiv bx \pmod{m}$ for two distinct values a, b in the range $0 \leq b \leq a \leq m-1$. Then we would have $(a-b)x \equiv 0 \pmod{m}$, or equivalently, $(a-b)x = km$ for some integer k (possibly zero or negative).

However, x and m are relatively prime, so x cannot share any factors with m. This implies that $a-b$ must be an integer multiple of m. This is not possible, since $a-b$ ranges between 1 and $m-1$. □

Actually it turns out that $\gcd(m,x) = 1$ is also a *necessary* condition for the existence of an inverse: i.e., if $\gcd(m,x) > 1$ then x has no multiplicative inverse modulo m.

Exercise. Verify this claim. [Hint: Assume for contradiction that x does have an inverse, say a. Then write down a (non-modular) equation that x, m and a must satisfy. Finally, derive a contradiction from the fact that x and m have a common factor $d > 1$.]

Since we know that multiplicative inverses are unique when $\gcd(m,x) = 1$, we shall write the inverse of x as $x^{-1} \pmod{m}$. Being able to compute the multiplicative inverse of a number is crucial to many applications, so ideally the algorithm used should be efficient. It turns out that we can use an extended version of Euclid's algorithm, which computes the gcd of two numbers, to compute the multiplicative inverse.

5 Computing Inverses: Euclid's Algorithm

Let us first discuss how computing the multiplicative inverse of x modulo m is related to finding $\gcd(x,m)$. For any pair of numbers x, y, suppose we could not only compute $\gcd(x,y)$, but also find integers a, b such that

$$d = \gcd(x,y) = ax + by. \tag{1}$$

(Note that this is not a modular equation; and the integers a, b could 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 could do this then we'd be able to compute inverses, as follows. We first find integers a and b such that

$$1 = \gcd(m,x) = am + bx.$$

But this means that $bx \equiv 1 \pmod{m}$, so b is a multiplicative inverse of x modulo m. Reducing b modulo m gives us the unique inverse we are looking for. In the above example, we see that 3 is the multiplicative inverse of 12 mod 35. So, we have reduced the problem of computing inverses to that of finding integers a, b that satisfy equation (1). Remarkably, Euclid's algorithm for computing gcd's also allows us to find integers a and b as described above. So computing the multiplicative inverse of x modulo m is as simple as running Euclid's gcd algorithm on input x and m!

Euclid's algorithm

If we wish to compute the gcd of two numbers x and y, how would we proceed? If x or y is 0, then computing the gcd is easy; it is simply the other number, since 0 is divisible by everything (although of course it divides nothing). The algorithm for other cases is ancient, and although associated with the name of Euclid, is almost certainly a folk algorithm invented by craftsmen (the engineers of their day) because of its intensely practical nature[2].

This algorithm exists in cultures throughout the globe.

The algorithm for computing $\gcd(x,y)$ uses the following theorem to eventually reduce to the case where one of the numbers is 0.

Theorem 6.3. *Let $x \geq y > 0$. Then* $\gcd(x,y) = \gcd(y, x \pmod{y})$.

Proof. The theorem follows immediately from the fact that a number d is a common divisor of x and y *if and only if* d is a common divisor of y and $x \pmod{y}$. To see this, write $x = qy + r$ where q is an integer and $r = x \pmod{y}$. Then, if d divides x and y then it also divides x and qy, and thus it also divides their difference $r = x - qy$ (as we proved in Note 1). Conversely, if d divides y and r then it also divides qy and r and thus also their sum $x = qy + r$. □

Given this theorem, let's see how to compute $\gcd(16,10)$:

$$\begin{aligned}\gcd(16,10) &= \gcd(10,6)\\ &= \gcd(6,4)\\ &= \gcd(4,2)\\ &= \gcd(2,0) = 2\end{aligned}$$

In each line, we replace the pair of arguments (x,y) with $(y, x \pmod{y})$, until the second argument becomes 0. At this point the gcd is just the first argument. By the theorem, each of these substitions preserves the gcd.

This algorithm can be written recursively as follows. The algorithm assumes that the inputs are natural numbers x, y satisfying $x \geq y \geq 0$ and $x > 0$.

```
algorithm gcd(x, y)
  if y = 0 then return(x)
  else return(gcd(y, x (mod y)))
```

[2]This algorithm is used for figuring out a common unit of measurement for two lengths. You can imagine how this is extremely important for building something up from a scale model. Different lengths in a design can be expressed as integer multiples of a common length, and then a new measuring stick can be found for the scaled-up design. The fact that Euclid's algorithm deals naturally with real numbers is also important in understanding the topic of continued fractions — a topic important in the understanding of approximations and numerical computing.

Theorem 6.4. *The algorithm above correctly computes the gcd of x and y.*

Proof. Correctness is proved by (strong) induction on y, the smaller of the two input numbers. For each $y \geq 0$, let $P(y)$ denote the proposition that the algorithm correctly computes $\gcd(x,y)$ for all values of x such that $x \geq y$ (and $x > 0$). Certainly $P(0)$ holds, since $\gcd(x,0) = x$ and the algorithm correctly computes this in the `if`-clause. For the inductive step, we may assume that $P(z)$ holds for all $z < y$ (the inductive hypothesis); our task is to prove $P(y)$. The key observation here is that $\gcd(x,y) = \gcd(y, x \pmod y)$ — that is, replacing x by $x \pmod y$ does not change the gcd. This was proved in Theorem 6.3. Hence the `else`-clause of the algorithm will return the correct value provided the recursive call `gcd(y,x mod y)` correctly computes the value $\gcd(y, x \pmod y)$. But since $x \pmod y < y$, we know this is true by the inductive hypothesis! This completes our verification of $P(y)$, and hence the induction proof. □

What is the running time of this algorithm? We shall see that, in terms of arithmetic operations on integers, it takes time $O(n)$, where n is the total number of bits in the input (x,y). This is again very efficient. The argument for this fact will be similar to the one we used earlier for exponentiation, but slightly trickier: it is obvious that the arguments of the recursive calls become smaller and smaller (because $y \leq x$ and $x \pmod y < y$). The question is, how fast?

The key point we will prove is that, in the computation of $\gcd(x,y)$, after *two* recursive calls the first (larger) argument is smaller than x by at least a factor of two (assuming $x > 0$). (Note that we can't argue much about what happens in just one call.) There are two cases:

1. $y \leq \frac{x}{2}$. Then the first argument in the next recursive call, y, is already smaller than x by a factor of 2, and thus in the next recursive call it will be even smaller.
2. $x \geq y > \frac{x}{2}$. Then in two recursive calls the first argument will be $x \pmod y$, which is smaller than $\frac{x}{2}$.

So, in both cases the first argument decreases by a factor of at least two every two recursive calls. Thus after at most $2n$ recursive calls, where n is the number of bits in x, the recursion must stop. (Note that the first argument is always a natural number.)

Note that the above argument only shows that the *number of recursive calls* in the computation is $O(n)$. We can make the same claim for the running time if we assume that each call only requires constant time. Since each call involves one integer comparison and one mod operation, it is reasonable to claim that its running time is constant. In a more realistic model of computation, however, we should really make the time for these operations depend on the size of the numbers involved. This will be discussed in CS170.

Extended Euclid's algorithm

Recall that, in order to compute the multiplicative inverse, we need an algorithm which also returns integers a and b such that:

$$\gcd(x,y) = ax + by. \tag{2}$$

Then, in particular, when $\gcd(x,y) = 1$ we can deduce that b is an inverse of $y \bmod x$.

Now since this problem is a generalization of the basic gcd, it is perhaps not too surprising that we can solve it with a fairly straightforward extension of Euclid's algorithm.

The following recursive algorithm `extended-gcd` follows the same recursive structure as Euclid's original algorithm, but keeps track of the required coefficients a, b in equation (2) as the recursion unwinds. Specifically, the algorithm takes as input a pair of natural numbers $x \geq y$ as in Euclid's algorithm, and returns a triple of integers (d,a,b) such that $d = \gcd(x,y)$ and $d = ax + by$:

```
algorithm extended-gcd(x, y)
  if y = 0 then return(x, 1, 0)
  else
    (d, a, b) := extended-gcd(y, x (mod y))
    return((d, b, a - (x div y) * b))
```

In this algorithm, x div y denotes the usual truncated integer division, written mathematically as $\lfloor x/y \rfloor$.

Here is the sequence of recursive calls (top row) along with the sequence of triples that they return (bottom row) for the same input $(x,y) = (16,10)$ as in our previous gcd example:

$$\begin{array}{ccccccccc} \text{egcd}(16,10) & \longrightarrow & \text{egcd}(10,6) & \longrightarrow & \text{egcd}(6,4) & \longrightarrow & \text{egcd}(4,2) & \longrightarrow & \text{egcd}(2,0) \\ (2,2,-3) & \longleftarrow & (2,-1,2) & \longleftarrow & (2,1,-1) & \longleftarrow & (2,0,1) & \longleftarrow & (2,1,0) \end{array}$$

Exercise. Check the above execution sequence.

The final triple returned is $(d,a,b) = (2,2,-3)$, meaning that the gcd of $(x,y) = (16,10)$ is $d = 2$, and that it can be expressed as $d = ax + by = 2 \cdot 16 - 3 \cdot 10$, which we can easily see is correct. (In fact, the sequence of triples (d,a,b) returned each expresses $d = 2$ as a linear combination of the corresponding inputs to that recursive call: so we can check that $d = 1 \cdot 2 + 0 \cdot 0 = 0 \cdot 4 + 1 \cdot 2 = 1 \cdot 6 - 1 \cdot 4 = -1 \cdot 10 + 2 \cdot 6 = 2 \cdot 16 - 3 \cdot 10$.) Since $\gcd(16,10) \neq 1$, 10 doesn't have an inverse mod 16, so the coefficients a, b returned by the algorithm are not useful to us for computing inverses. However, the next exercise suggests an example where they allow us to read off the inverse, as described earlier.

Exercise. Hand-turn the algorithm yourself on the input $(x,y) = (35,12)$ and verify that it returns the triple $(1,-1,3)$. Deduce that the inverse of 12 mod 35 is 3.

Let's now reverse-engineer the algorithm and understand why it works and why it was designed this way. In the base case ($y = 0$), the algorithm returns the gcd value $d = x$ as before, together with coefficients $a = 1$ and $b = 0$; clearly these satisfy $ax + by = d$, as required.

When $y > 0$, the algorithm first recursively computes values (d,a,b) such that $d = \gcd(y, x \pmod y)$ and

$$d = ay + b(x \pmod y). \tag{3}$$

It then returns the triple (d,A,B), where $A = b$ and $B = a - \lfloor x/y \rfloor b$. Just as in our earlier analysis of the vanilla gcd algorithm, we know that the value d computed recursively will be equal to $\gcd(x,y)$. So the first component of the triple returned by the algorithm is correct.

What about the other two components, A and B? From the specification of the algorithm, they should be integers that satisfy

$$d = Ax + By. \tag{4}$$

To figure out what A and B should be in terms of the previously returned values a and b, we can rearrange equation (3), as follows:

$$\begin{aligned} d &= ay + b(x \pmod y) && (5)\\ &= ay + b(x - \lfloor x/y \rfloor y) && (6)\\ &= bx + (a - \lfloor x/y \rfloor b)y. && (7) \end{aligned}$$

(In the second line here, we have used the fact that $x \pmod{y} = x - \lfloor x/y \rfloor y$ — check this!) Now compare this last equation (7) with equation (4): comparing coefficients of x and y, we see that we need to take $A = b$ and $B = a - \lfloor x/y \rfloor b$. But this is exactly what the last line of the algorithm does, and this is why it works!

Exercise. Turn the above argument into a formal proof by induction that the algorithm extended-gcd is correct.

Since the extended gcd algorithm has exactly the same recursive structure as the vanilla version, its running time will be the same up to constant factors (reflecting the increased time per recursive call). So once again the running time on n-bit numbers will be $O(n)$ arithmetic operations. This means that we can find multiplicative inverses very efficiently.

Remark.

We note that a different version of the algorithm which is easier to run by hand can be developed as follows. The goal is to find $ax + by = d = gcd(x,y)$. We first observe if $d|x$ (or $x = id$) and $d|y$ (or $y = jd$), we have

$$ax + by = a(id) + b(jd) = (ai + bj)d,$$

or the expression $ax + by$ must be a multiple of the $gcd(x,y)$. Thus, we simply want to find the equation of this form with the smallest strictly positive right hand side.

We begin with the following identities of this form as follows.

$$(1)x + (0)y = x$$
$$(0)x + (1)y = y$$

Again the right hand sides are multiples of $d = gcd(x,y)$ as are any sum of integer multiples of the two equations. We wish to make the right hand sides "smaller": one way to do this is to subtract the second equation from the first, or more aggressively subtracting a multiple of the second from the first. In an example with $x = 16$ and $y = 6$, we have

$$(1)16 + (0)6 = 16$$
$$(0)16 + (1)6 = 6$$

and now subtracting two ($\lfloor 16/6 \rfloor$) times the second from the first, we obtain

$$(1)16 + (-2)6 = 4.$$

We can then subtract this equation from the previous, and obtain

$$(-1)16 + (3)6 = 2.$$

Now, we have obtained a solution as the right hand side is $gcd(16,6)$. In general, if one observes that if the previous equations have a right hand sides of x (e.g., 16) and y (e.g., 6) , the successive equation has a right hand side of $x - \lfloor x/y \rfloor y$ (e.g., $16 - (2)(6) = 4$) akin to the recursive call in the gcd algorithm. Moreover, the coefficients of the left hand side are the A, B pair that are returned by the e-gcd algorithm called with x and y. This method is essentially an iterative version of the e-gcd algorithm.

Division in modular arithmetic

Now that we know how to compute the inverse of x modulo m (assuming that x and m are coprime), how can we use it to do arithmetic? The simplest scenario is solving a modular equation such as the following:

$$8x \equiv 9 \pmod{15}. \tag{8}$$

To solve the analogous equation $8x = 9$ over the rational numbers, we would multiply both sides by 8^{-1} to get $x = 9/8$. Let's do the same thing in arithmetic mod 15. Recall that the inverse of $8 \pmod{15}$ is 2 (since $2 \cdot 8 = 16 \equiv 1 \pmod{15}$). Hence we can multiply both sides of equation (8) by $8^{-1} \equiv 2$ to get

$$x \equiv 18 \equiv 3 \pmod{15}.$$

i.e., the solution to the modular equation (8) is $x = 3$, and this solution is unique modulo 15.

6 Fundamental Theorem of Arithmetic

You may recall that any positive integer greater than 1 can be expressed uniquely as a product of primes, up to a reordering of the factors. This is known as the Fundamental Theorem of Arithmetic. For example, $12 = 2 \cdot 2 \cdot 3$. How do we know that this is true? It turns out that Extended Euclid's Algorithm is the key to proving this!

We'll start by proving an important fact about divisibility.

Claim: Let x, y, and z be positive integers such that $\gcd(x,y) = 1$. If $x \mid yz$, then $x \mid z$.

Proof: By Extended Euclid's algorithm, since $\gcd(x,y) = 1$, there exists integers a and b such that

$$1 = \gcd(x,y) = ax + by.$$

Multiplying both sides by z gives

$$z = axz + byz.$$

We know that $x \mid axz$, and $x \mid byz$ because $x \mid yz$. Thus, x divides their sum, or in other words, $x \mid axz + byz$. Since $axz + byz = z$, we conclude that $x \mid z$. □

Recall that a prime number p is a number whose only positive factors are 1 and p.

Fundamental Theorem of Arithmetic: Every positive integer $n > 1$ can be expressed uniquely in the form $p_1 p_2 \cdots p_k$, where each p_i is a (not necessarily unique) prime number, up to reordering of the prime factors.

For example, we can write $12 = 2 \cdot 2 \cdot 3$. Any other prime factorization of 12 is some reordering of $2, 2$, and 3.

Proof: We have shown in lecture that every positive integer n can be expressed in the form $p_1 p_2 \cdots p_k$, where each p_i is a (not necessarily unique) prime number. Thus, it suffices to show that such a form is unique up to reordering the factors.

In other words, suppose we can express n with two prime factorizations

$$n = p_1 p_2 \cdots p_k = q_1 q_2 \cdots q_l,$$

where the p_i and q_j are prime numbers. We need to show that $k = l$, and the p_i's are some reordering of the q_j's.

Without loss of generality, assume that $k \leq l$. (If $k \geq l$, we can simply swap the factorizations.) Our goal is to show that each p_i is one of the q_j's.

First, consider p_1. Since $p_1 \mid n$, we must have $p_1 \mid q_1 q_2 \cdots q_l$. If p_1 is one of $q_1, q_2, \ldots, q_{l-1}$, then we are done. Otherwise, if $p_1 \neq q_j$, for $1 \leq j \leq l-1$, then $\gcd(p_1, q_j) = 1$ for $1 \leq j \leq l-1$, as the only factor they share in common is 1. Applying the previous claim, we conclude that $p_1 \mid q_l$. Since 1 is not prime, p_1 is not 1, so $p_1 = q_l$.

Thus, we conclude that $p_1 = q_j$ for some j. We can thus divide both sides by p_1, reducing the number of primes on both sides by 1.

We now repeat the process for p_2, p_3, all the way to p_k. In the end, the left hand side will be 1, and the right hand side will be a product of $l-k$ primes. Every prime number is at least 2, so for the two sides to be equal, we must have $l-k=0$, or $k=l$. Moreover, in this process, we have matched each p_i with a unique q_j. Thus, the p_i's are some reordering of the q_j's, as desired. □

What was the key to the above proof? Well, to prove that a prime factorization existed (something we showed in lecture), we used strong induction. To prove that such a factorization is unique, we used a corollary of the Euclidean algorithm. The key to the Euclidean algorithm is the existence and uniqueness of division with remainder; in other words, for integers x and $m \neq 0$, there exists unique integers q and r such that $x = mq + r$ with $0 \leq r \leq m-1$. If we have other arithmetic systems that emulate this type of division algorithm, we can follow the same type of proof to get a form of unique prime factorization. This is an important generalization in number theory! [3]

7 Chinese Remainder Theorem.

With our understanding of inverses, we can now present an interesting aspect of modular arithmetic which is embodied in the Chinese Remainder Theorem, named thusly since its earliest known statement was by the Chinese mathematician Sunzi in the 3rd century AD.

The simplest case of the theorem is as follows:

Claim: For m, n with $gcd(m,n) = 1$ that there is exactly one $x \pmod{mn}$ that satisfies the equations:

$$x \equiv a \pmod{n} \text{ and } x \equiv b \pmod{m}.$$

The proof follows from the existence of inverses of n and m respectively modulo m and n, which holds when $\gcd(n,m) = 1$.

Proof: Let $u \equiv m(m^{-1} \pmod{n})$ and $v \equiv n(n^{-1} \pmod{m})$.

Note that

$$u \equiv 1 \pmod{n} \quad \text{and} \quad u \equiv 0 \pmod{m}$$

since $m(m^{-1} \pmod{n}) \equiv mm^{-1} \equiv 1 \pmod{n}$ and $mx \equiv 0 \pmod{m}$ for any x, including $x \equiv m^{-1} \pmod{n}$.

Similarly, we have

$$v \equiv 0 \pmod{n} \quad \text{and} \quad v \equiv 1 \pmod{m}$$

Thus, $x = ua + vb \equiv a(1) + b(0) \equiv a \pmod{n}$, and similarly $x \equiv b \pmod{m}$. That is we have an x that satisfies the equations.

To argue that there is a unique solution, consider two solutions x and y in $\{0, \ldots, mn-1\}$. Now $x - y \equiv 0 \pmod{n}$ and $x - y \equiv 0 \pmod{m}$, which implies that $x - y$ is a multiple of m and n. Since $gcd(m,n) = 1$, we have that $x - y = kmn$ for some integer k, which implies that $|x-y| = 0$ or $|x-y| \geq mn$. The former implies

[3] If you are interested in seeing more details, take a look at https://en.wikipedia.org/wiki/Euclidean_domain.

that $x = y$, the latter implies that one of x or y is not in $\{0, \ldots, mn-1\}$. That is, there is at most one solution in $\{0, \ldots, mn-1\}$. □

A quick review is that we formed a solution by finding u where $u \equiv 0 \pmod{n}$, and a v where $v \equiv 0 \pmod{m}$, which can then be added to together to get a solution for each modulus as the 0 doesn't affect the equations in the other modulus. The uniqueness follows since the difference of any two solutions has a difference which is a multiple of mn.

This can be extended to the full Chinese remainder theorem which states.

Chinese Remainder Theorem: Let $n_1, n_2, \ldots, n_k$ be positive integers that are coprime to each other. Then, for any sequence of integers a_i there is a unique integer x between 0 and $N = \prod_{i=1}^{k} n_i$ that satisfies the congruences:

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ \vdots \equiv \vdots \\ x \equiv a_i \pmod{n_i} \\ \vdots \equiv \vdots \\ x \equiv a_k \pmod{n_k} \end{cases}$$

Moreover,

$$x \equiv \left(\sum_{i=1}^{k} a_i b_i \right) \pmod{N} \tag{9}$$

where $b_i = \frac{N}{n_i} \left(\frac{N}{n_i} \right)_{n_i}^{-1}$ where $\left(\frac{N}{n_i} \right)_{n_i}^{-1}$ denotes the multiplicative inverse $\pmod{n_i}$ of the integer $\frac{N}{n_i}$.

Here, each b_i is $1 \pmod{n_i}$ and $0 \pmod{n_j}$ for $i \neq j$, and thus a multiple of b_i can be used to satisfy the equation $\pmod{n_i}$ while not affecting the solution of the equations $\pmod{n_j}$ for $i \neq j$. It may be useful to think of each x as the k-dimensional vector where the ith entry is $x \pmod{n_i}$. With this view, b_i is the vector consisting of 1 in the ith entry and 0 elsewhere. Then one can obtain the vector for x by adding multiples of the "basis" vectors, b_i.

The uniqueness follows again from the fact that the difference of two solutions, x and y must be a multiple of $N = n_1 \times \cdots \times n_k$ since they are relatively prime and therefore there cannot be two of them in the range $\{0, \ldots, \prod_{i=0}^{k} n_i - 1\}$. Or briefly, notice that for any two solutions, x, y, have $(x-y) \equiv 0 \pmod{n_i}$ for all i which implies the difference is a multiple of all the n_i and therefore larger than $\prod_{i=0}^{k} n_i$ which is the desired contradiction.

Finally, we note that x can be represented by (a, b) where $x \equiv a \pmod{n}$ and $x \equiv b \pmod{m}$, and given x and y with representations (x_a, x_b) and (y_a, y_b), that $x + y$ has the representation $(x_a + y_a, x_b + y_b)$. A similar observation works for multiplication, which implies that arithmetic $\pmod{mn}$ acts in the same manner as arithmetic over the pairs of numbers viewed in modulo m and n respectively. Thus, in addition to a bijection between these sets, this mapping between $\{0, \ldots, mn-1\}$ and $\{0, \ldots, n-1\} \times \{0, \ldots, m-1\}$ are considered isomorphic under these operations as well. This relationship extends in the same manner to a case where there are more than two relatively prime moduli.

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes Note 7

This note is partly based on Section 1.4 of "Algorithms," by S. Dasgupta, C. Papadimitriou and U. Vazirani, McGraw-Hill, 2007.

Public Key Cryptography

In this note, we discuss a very nice and important application of modular arithmetic: the *RSA public-key cryptosystem*, named after its inventors Ronald Rivest, Adi Shamir and Leonard Adleman.

Cryptography is an ancient subject that really blossomed into its modern form at the same time[1] as the other great revolutions in the general fields of information science/engineering. The basic setting for cryptography is typically described via a cast of three characters: Alice and Bob, who with to communicate confidentially over some (insecure) link, and Eve, an eavesdropper who is listening in and trying to discover what they are saying. Let's assume that Alice wants to transmit a message x (written in binary) to Bob. She will apply her *encryption function* E to x and send the encrypted message $E(x)$ over the link; Bob, upon receipt of $E(x)$, will then apply his *decryption function* D to it and thus recover the original message: i.e., $D(E(x)) = x$.

Since the link is insecure, Alice and Bob have to assume that Eve may get hold of $E(x)$. (Think of Eve as being a "sniffer" on the network.) Thus ideally we would like to know that the encryption function E is chosen so that just knowing $E(x)$ (without knowing the decryption function D) doesn't allow one to discover anything about the original message x.

For centuries cryptography was based on what are now called *private-key* protocols. In such a scheme, Alice and Bob meet beforehand and together choose a secret codebook, with which they encrypt all future correspondence between them. (This codebook plays the role of the functions E and D above.) Eve's only hope then is to collect some encrypted messages and use them to at least partially figure out the codebook.

Public-key schemes such as RSA, first invented in the 1970s, are significantly more subtle and tricky: they allow Alice to send Bob a message without ever having met him before! This almost sounds impossible, because in this scenario there is a symmetry between Bob and Eve: why should Bob have any advantage over Eve in terms of being able to understand Alice's message? The central idea behind the RSA cryptosystem is that Bob is able to implement a *digital lock*, to which only he has the key. Now by making this digital lock public, he gives Alice (or, indeed, anybody else) a way to send him a secure message which only he can open.

Here is how the digital lock is implemented in the RSA scheme. Each person has a *public key* known to the whole world, and a *private key* known only to him- or herself. When Alice wants to send a message x to Bob, she encodes it using Bob's public key. Bob then decrypts it using his private key, thus retrieving x. Eve is welcome to see as many encrypted messages for Bob as she likes, but she will not be able to decode them (under certain basic assumptions explained later in this Note).

[1] And in reality, involving some of the same cast of characters. Both Alan Turing and Claude Shannon were active in codebreaking during WW2 and Shannon had a classic paper that is considered the birth of the information-theoretic understanding of secrecy and dovetails with his other more famous paper that gave rise to the modern information-theoretic view of communication. Both cryptography and communication weave together information, computation, and randomness in surprising ways. In 70, you just get a tiny taste of these spectacularly beautiful fields.

When it comes to something like RSA, there are different ways that help different people understand the scheme. In this note, we will first just walk through the scheme and its analysis. Along the way, we'll discuss what happens when one raises numbers to powers in modulo arithmetic. However, at the end of this note, we will talk about how anyone could have come up with the RSA scheme. After all, the understanding of exponentiating in modulo arithmetic predates RSA historically. It turns out that with the background understanding and perspective provided by the "vector view" of the Chinese Remainder Theorem and Fermat's Little Theorem's understanding of exponentiation, the RSA scheme is quite natural.

The RSA scheme is based heavily on modular arithmetic. Let p and q be two large primes (typically having, say, 512 bits each), and let $N = pq$. We will think of messages to Bob as numbers modulo N, excluding the trivial values 0 and 1. (Larger messages can always be broken into smaller pieces and sent separately.)

Also, let e be any number that is relatively prime to $(p-1)(q-1)$. (Typically e is chosen to be a small value such as 3.) Then Bob's *public key* is the pair of numbers (N,e). This pair is published to the whole world. (Note, however, that the numbers p and q are *not* public; this point is crucial and we will return to it below.)

What is Bob's private key? This will be the number d, which is the *inverse* of e mod $(p-1)(q-1)$. (This inverse is guaranteed to exist because e and $(p-1)(q-1)$ are coprime.)

We are now in a position to describe the encryption and decryption functions:

- **[Encryption]:** When Alice wants to send a message x (assumed to be an integer mod N) to Bob, she computes the value $E(x) \equiv x^e \pmod{N}$ and sends this to Bob.
- **[Decryption]:** Upon receiving the value $y = E(x)$, Bob computes $D(y) \equiv y^d \pmod{N}$; this will be equal to the original message x.

Example: Let $p = 5$, $q = 11$, and $N = pq = 55$. (In practice, p and q would be much larger.) Then we can choose $e = 3$, which is relatively prime to $(p-1)(q-1) = 40$. Thus Bob's public key is $(55,3)$. His private key is $d \equiv 3^{-1} \pmod{40} \equiv 27$. For any message x that Alice (or anybody else) wishes to send to Bob, the encryption of x is $y \equiv x^3 \pmod{55}$, and the decryption of y is $x \equiv y^{27} \pmod{55}$. So, for example, if the message is $x = 13$, then the encryption is $y = 13^3 \equiv 52 \pmod{55}$, and this is decrypted as $52^{27} \equiv 13 \pmod{55}$.

How do we know that this scheme works? We need to check that Bob really does recover the original message x. The following theorem establishes this fact.

Theorem 7.1: Under the above definitions of the encryption and decryption functions E and D, we have $D(E(x)) \equiv x \pmod{N}$ for every possible message $x \in \{0,1,\ldots,N-1\}$.

The proof of this theorem makes use of a standard fact from number theory known as *Fermat's Little Theorem*, which tells us that exponentiation is periodic when done modulo a prime, and that period is one less than the prime in question. Precisely, it says:

Theorem 7.2: **[Fermat's Little Theorem]** For any prime p and any $a \in \{1,2,\ldots,p-1\}$, we have $a^{p-1} \equiv 1 \pmod{p}$.

Proof. Let S denote the set of non-zero integers mod p, i.e., $S = \{1,2,\ldots,p-1\}$. Consider the sequence of numbers $a, 2a, 3a, \ldots, (p-1)a \bmod p$. We already saw in the previous Lecture Note that, whenever $\gcd(p,a) = 1$ (i.e., p,a are coprime, which certainly holds here since p is prime) these numbers are all distinct. Therefore, since none of them is zero, and there are $p-1$ of them, they must include each element of S exactly once. Therefore, the set of numbers

$$S' = \{a \pmod{p}, 2a \pmod{p}, \ldots, (p-1)a \pmod{p}\}$$

is exactly the same as S (just in a different order)!

Now suppose we take the product of all numbers in S, mod p. Clearly, this product is

$$1 \times 2 \times \cdots \times (p-1) \equiv (p-1)! \pmod{p}. \tag{1}$$

On the other hand, what if we take the product of all the numbers in S'? Clearly this is

$$a \times 2a \times \cdots \times (p-1)a \equiv a^{p-1}(p-1)! \pmod{p}. \tag{2}$$

But from our observation in the previous paragraph that the sets of numbers in S and in S' are exactly the same (mod p), the products in (1) and (2) must in fact be equal mod p. Hence we have

$$(p-1)! \equiv a^{p-1}(p-1)! \pmod{p}. \tag{3}$$

Finally, since p is prime, we know that every non-zero integer has an inverse mod p, and therefore $(p-1)!$ has an inverse mod p. Hence we can multiply both sides of (3) by the inverse of $(p-1)!$ to get $a^{p-1} \equiv 1 \pmod{p}$, as required. □

Armed with Fermat's Little Theorem, we can now go back and prove the correctness of RSA.

Proof of Theorem 7.1. To prove the statement, we have to show that

$$(x^e)^d \equiv x \pmod{N} \qquad \text{for every } x \in \{0,1,\ldots,N-1\}. \tag{4}$$

Let's consider the exponent, which is ed. By definition of d, we know that $ed \equiv 1 \pmod{(p-1)(q-1)}$; hence we can write $ed = 1 + k(p-1)(q-1)$ for some integer k, and therefore

$$x^{ed} - x = x^{1+k(p-1)(q-1)} - x = x(x^{k(p-1)(q-1)} - 1). \tag{5}$$

Looking back at equation (4), our goal is to show that this last expression in equation (5) is equal to 0 mod N for every x.

Now we claim that the expression $x(x^{k(p-1)(q-1)} - 1)$ in (5) is divisible by p. To see this, we consider two cases:

Case 1: *x is not a multiple of p.* In this case, since $x \not\equiv 0 \pmod{p}$, we can use Fermat's Little Theorem to deduce that $x^{p-1} \equiv 1 \pmod{p}$, and hence $x^{k(p-1)(q-1)} - 1 \equiv 0 \pmod{p}$, as required.

Case 2: *x is a multiple of p.* In this case the expression in (5), which has x as a factor, is clearly divisible by p.

By an entirely symmetrical argument, $x(x^{k(p-1)(q-1)} - 1)$ is also divisible by q. Therefore, it is divisible by both p and q, and since p and q are primes it must be divisible by their product, $pq = N$. But this implies that the expression is equal to 0 mod N, which is exactly what we wanted to prove. □

CRT Proof of Theorem 7.1. A closely related proof can be obtained using the Chinese remainder theorem. That is,

$$x^{ed} = x^{k(p-1)(q-1)+1} = (x^{k(q-1)})^{(p-1)} x \equiv x \pmod{p}$$

for both $x \not\equiv 0 \pmod{p}$ by Fermat's Theorem and for $x = 0 \pmod{p}$ by inspection. Similarly, $x^{ed} \equiv x \pmod{q}$. By the uniqueness property established in Chinese remainder theorem, only $x \pmod{pq}$ satisfies these two equations. □

So we have seen 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 get any useful information by observing the encrypted messages? The security of RSA hinges upon the following basic assumption:

$$\text{Given } N, e \text{ and } y \equiv x^e \pmod{N}, \text{ there is no efficient algorithm for determining } x.$$

This assumption is quite plausible. How might Eve try to guess x? She could experiment with all possible values of x, each time checking whether $x^e \equiv y \pmod{N}$; but she would have to try on the order of N values of x, which is completely unrealistic if N is a number with (say) 512 bits. Alternatively, she could try to factor N to retrieve p and q, and then figure out d by computing the inverse of e mod $(p-1)(q-1)$; but this approach requires Eve to be able to *factor* N into its prime factors, a problem which is believed to be impossible to solve efficiently for large values of N. We should point out that the security of RSA has not been formally proved: it rests on the assumptions that breaking RSA is essentially tantamount to factoring N, and that factoring is hard.

We close this note with a brief discussion of implementation issues for 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 $x^e \pmod{N}$, and Bob has to compute $y^d \pmod{N}$.)

We briefly discuss the implementation of each of these tasks in turn.

To find large prime numbers, we use the fact that, given a positive integer n, there is an efficient algorithm that determines whether or not n is prime. (Here "efficient" means a running time of $O((\log n)^k)$ for some small k, i.e., a low-degree power of the *number of bits in* n. Notice the dramatic contrast here with factoring: we can tell efficiently whether or not n is prime, but in the case that it is not prime we cannot efficiently find its factors. The success of RSA hinges crucially on this distinction.) Given that we can test for primes, Bob just needs to generate some random integers n with the right number of bits, and test them until he finds two primes p and q. This works because of the following basic fact from number theory (which we will definitely not prove[2]), which says that a reasonably large fraction of positive integers are prime:

Theorem 7.3: **[Prime Number Theorem]** Let $\pi(n)$ denote the number of primes that are less than or equal to n. Then for all $n \geq 17$, we have $\pi(n) \geq \frac{n}{\ln n}$. (And in fact, $\lim_{n\to\infty} \frac{\pi(n)}{n/\ln n} = 1$.)

Setting $n = 2^{512}$, for example, the Prime Number Theorem says that roughly one in every 355 of all 512-bit numbers are prime. Therefore, if we keep picking random 512-bit numbers and testing them, we would expect to have to try only about 355 numbers until we find a prime.

We turn now to the second task above: modular exponentiation. This is actually something we have already discussed in the previous Lecture Note. Recall from that Note that we can compute an exponential expression $x^y \pmod{N}$ by repeated squaring, using a number of multiplications that is only $O(n)$, where n is the

[2]Why don't we prove this? The reason is that standard proofs of this fact rely on complex analysis (i.e. calculus involving complex functions), and need to lean on the study of more advanced relatives of the transfer functions and s-impedances that you saw in EECS16B. Is it surprising that you need this kind of machinery to study prime numbers? This is simply another aspect of the interconnected beauty of mathematics. We need complex numbers to study the behavior of differential equations and electrical circuits — why should it be any surprise that they are needed to properly understand the prime numbers?

number of *bits* in y (i.e., the length of the binary representation of y). Since in the RSA application the exponents in the expressions that Alice and Bob need to compute (e, d, respectively) are both less than N, the number of multiplications needed is $O(\log N)$, since the number of bits in N is $O(\log N)$. Furthermore, since all multiplication is being done mod N, all multiplications involve numbers with at most $O(\log N)$ bits; and, as is well known from elementary school long multiplication, multiplying m-bit numbers takes $O(m^2)$ operations (and can actually done even faster: see CS170). Hence the total cost of Alice's and Bob's exponential computations is only $O((\log N)^3)$. Thus it is possible to work with very large numbers (having hundreds of bits) in practical implementations of RSA.

To summarize, then, in the RSA protocol Bob need only perform simple calculations such as multiplication, exponentiation and primality testing to implement his digital lock. Similarly, Alice and Bob need only perform simple calculations to lock and unlock the 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 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.

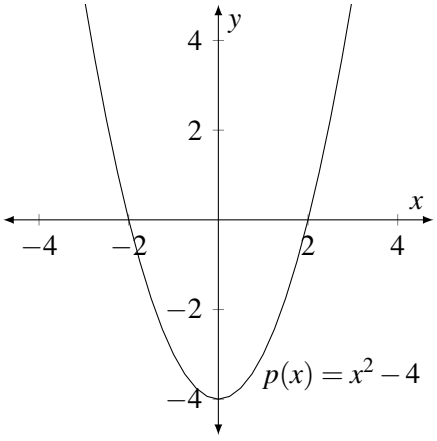
1 Polynomials

Polynomials constitute a rich class of functions which are both easy to describe and widely applicable in topics ranging from Fourier analysis, cryptography and communication, to control and computational geometry. In this note, we will discuss further properties of polynomials which make them so useful. The key idea here is to extend what you already know about polynomials over the real and complex numbers to modulo arithmetic. We will then describe how to take advantage of these properties to develop a secret-sharing scheme.

Recall that a *polynomial* in a single variable is an expression that has an associated function.[1] The polynomial expression $p(x) = a_d x^d + a_{d-1}x^{d-1} + \cdots + a_1 x + a_0$. 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.

The polynomial function can be viewed as a function in some domain with multiplication and addition operations, by evaluating the expression in the natural manner. For example, $p(x) = 5x^3 + 2x + 1$ can be evaluated at $x = 3$ and to obtain $p(3) = 5(3)^3 + 2(3) + 1 = 142$.

To proceed, a familiar 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:



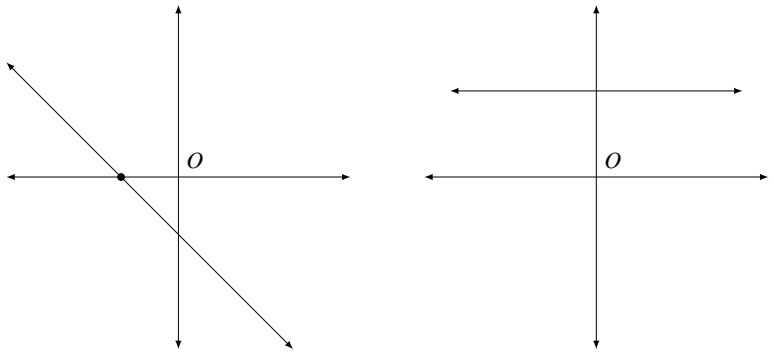
We now state two fundamental properties of polynomials that we will prove in due course.

[1] In the reals, the association is unique. In modular arithmetic modulo a prime p, it isn't necessarily unique as modulo p, by Fermat's Little Theorem, $x^p = x \pmod p$ for a prime p. Still, there is a uniquely associated polynomial expression to a function where the expression does not have terms with exponents larger than $p-1$ under arithmetic modulo a prime p. Our applications generally assume that this is the expression we are working with.

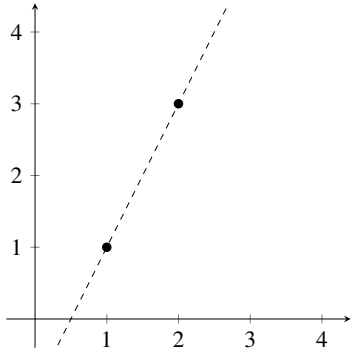
Property 1: A non-zero polynomial of degree d has at most d roots.

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 plot over the reals of a linear (degree 1) polynomial $y = a_1 x + a_0$ is a line that may not go through the origin. 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.



2 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 $p(x) = a_d x^d + \cdots + a_1 x + a_0$ such that $p(x_i) = y_i$ for $i = 1$ to $d+1$? We will give an algorithm for reconstructing the coefficients $a_0, \ldots, a_d$, and therefore the polynomial $p(x)$. Another standard algorithm that is more clearly linear-algebraic is described in the next note, because it will be important as a stepping stone.

The method here is called *Lagrange interpolation* and it should remind you of the Chinese Remainder Theorem: Let us start by solving an easier problem. Suppose that we are told that $y_1 = 1$ and $y_j = 0$ for $2 \leq j \leq d+1$. 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 $2 \leq j \leq d+1$. 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 (since x_1 is not equal to any of the other x_i). 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: denote by $\Delta_i(x)$ the degree d polynomial that goes through these $d+1$ points. Then $\Delta_i(x) = \dfrac{\prod_{j \neq i}(x - x_j)}{\prod_{j \neq i}(x_i - x_j)}$.

We 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 $\Delta_1(x), \ldots, \Delta_{d+1}(x)$ as described above. Now we claim that 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.

In the above construction, we can think of the polynomials $\Delta_i(x)$ as a "natural basis" for all polynomials whose values are specified at the points $\{x_j\}$. Note that these basis polynomials depend only on the x_j, and not on the values y_j at the points. We then sum the basis polynomials Δ_i, with coefficients equal to the values y_i, to construct the desired polynomial $p(x)$.

As an example, suppose we want to find the degree-2 polynomial $p(x)$ that passes through the three points $(x_1, y_1) = (1,1)$, $(x_2, y_2) = (2,2)$ and $(x_3, y_3) = (3,4)$. The three polynomials Δ_i are as follows:

$$\Delta_1(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)} = \frac{(x-2)(x-3)}{2} = \frac{1}{2}x^2 - \frac{5}{2}x + 3;$$
$$\Delta_2(x) = \frac{(x-1)(x-3)}{(2-1)(2-3)} = \frac{(x-1)(x-3)}{-1} = -x^2 + 4x - 3;$$
$$\Delta_3(x) = \frac{(x-1)(x-2)}{(3-1)(3-2)} = \frac{(x-1)(x-2)}{2} = \frac{1}{2}x^2 - \frac{3}{2}x + 1.$$

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.

2.1 Proof of Property 2

We are now in a position to prove Property 2 stated earlier, namely:

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 above 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 $q(x_i) = y_i$ 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 (to be proved below) implies it can have at most d roots. But on the other hand $r(x_i) = p(x_i) - q(x_i) = 0$ at $d+1$ distinct points x_i, so $r(x)$ has at least $d+1$ roots. 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 $q(x)$. We can compute the quotient and remainder using long division for polynomials, as illustrated in the following example.

Example. We wish to divide $p(x) = x^3 + x^2 - 1$ by $q(x) = x - 1$:

- First we subtract a factor $x^2(x-1)$ to write: $p(x) = x^2(x-1) + (2x^2 - 1)$.
- Then we subtract a factor $2x(x-1)$ to write the remainder as $2x^2 - 1 = 2x(x-1) + (2x-1)$.
- Then we subtract a factor $2(x-1)$ to write the remainder as $2x - 1 = 2(x-1) + 1$.
- Finally, putting the above three lines together, we get that $p(x) = (x^2 + 2x + 2)(x-1) + 1$.

Written out using notation that you might be familiar with:

$$\begin{array}{r|l} & x^2+2x+2 \\ \hline x-1 & x^3 + x^2 - 1 \\ & \underline{-x^3 + x^2} \\ & \quad 2x^2 \\ & \quad \underline{-2x^2 + 2x} \\ & \qquad 2x - 1 \\ & \qquad \underline{-2x + 2} \\ & \qquad\quad 1 \end{array}$$

Therefore, the quotient is $q'(x) = x^2 + 2x + 2$, and the remainder is $r(x) = 1$.

2.2 Proof of Property 1

Now let us prove Property 1, which we restate from earlier:

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 two claims:

Claim 1 If a is a root of a polynomial $p(x)$ with degree $d \geq 1$, 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 $p(x) = c(x - a_1) \cdots (x - a_d)$, where c is a real number.

Note that Claim 2 immediately implies Property 1: we just need to 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 (of degree $d-1$) 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 $p(x) = (x-a)q(x)$.

Proof of Claim 2

Proof by induction on d.

- Base Case: $d = 0$. If $p(x)$ is a polynomial of degree 0 then it is simply a constant c, so it can trivially be written in the desired form.
- Inductive Hypothesis: For some $d \geq 0$, any polynomial of degree d with distinct roots $a_1, \ldots, a_d$ can be written as $p(x) = c(x-a_1)\cdots(x-a_d)$.
- Inductive Step: Let $p(x)$ be a polynomial of degree $d+1$ with distinct roots $a_1, \cdots, a_{d+1}$. By Claim 1, $p(x) = (x-a_{d+1})q(x)$ for some polynomial $q(x)$ of degree d. Since $0 = p(a_i) = (a_i - a_{d+1})q(a_i)$ for all $i \neq d+1$, and $a_i - a_{d+1} \neq 0$, $q(a_i)$ must be equal to 0. Thus $q(x)$ is a polynomial of degree d with distinct roots $a_1, \ldots, a_d$. We can now apply the inductive hypothesis to $q(x)$ to write $q(x) = c(x-a_1)\cdots(x-a_d)$. Substituting in $p(x) = (x-a_{d+1})q(x)$, we obtain $p(x) = c(x-a_1)\cdots(x-a_{d+1})$, as desired.

3 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, or indeed the rational numbers, rather than from the real numbers. However, the proofs[2] do not go through if the values are restricted to being natural numbers or integers. Let us try to understand these facts a little more closely. The only properties of numbers that we used in polynomial interpolation and in the proofs of Properties 1 and 2 are that we can add, subtract, multiply and divide any pair of numbers as long as we are not dividing by 0. (You should go back and check this claim!) Therefore, everything holds just as well for the complex numbers and the rational numbers. On the other hand, we cannot subtract two natural numbers and guarantee that the result is a natural number, and dividing two integers does not generally result in an integer, so everything falls apart for natural numbers and integers.

However, if we work with numbers modulo a prime[3] 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

[2] However, the result of property 1 does hold for polynomials with integer or natural number coefficients with the domain restricted to integers or natural numbers. That is because we have proved something *stronger* by going to the rationals or reals. If a polynomial has no more than d real roots, it certainly has no more than d integer roots. After all, every integer is a real number!

Meanwhile, property 2 does not hold at all if we restrict all the numbers involved to be integers. Try finding a straight line that goes through $(0,0)$ and $(2,1)$. It doesn't have integer coefficients.

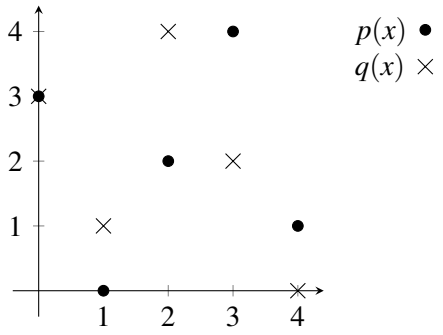
[3] Again, it is useful to consider what happens if the m is not a prime. What happens to property 1? Consider $m = 8$ and the equation $x^3 = 0$. How many roots? 0 is a root for sure, but so are $2, 4, 6$. That is 4 distinct roots to a third-degree polynomial. This means that property 1 does not hold and neither does property 2.

Unlike in the case of the integers, we cannot simply embed the numbers mod 8 into a larger field for which the result holds. This goes to show that we must be careful. It turns out that there is a way construct a finite field with 8 elements, but it is not simply the integers mod 8.

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 $2x+3 \equiv 0 \pmod 5$ holds. Solving for x, we get that $2x = -3 \equiv 2 \pmod 5$, and thus $x = 1 \pmod 5$. Note that this is consistent with Property 1 since we got only one root of a degree-1 polynomial.

Now consider the polynomials $p(x) = 2x+3 \pmod 5$ and $q(x) = 3x-2 \pmod 5$. We can plot the values y of each polynomial 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, $(0,3)$, 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 Properties 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). Therefore, they hold whenever we work with integers 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 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: https://en.wikipedia.org/wiki/Field_(mathematics). While you are there, if you are interested, 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 course: http://en.wikipedia.org/wiki/Galois_field. (Take Math 113 and Math 114 to really get into this stuff, or take EECS 229B which also covers many of these properties as it builds the fundamentals of error-correcting codes.)

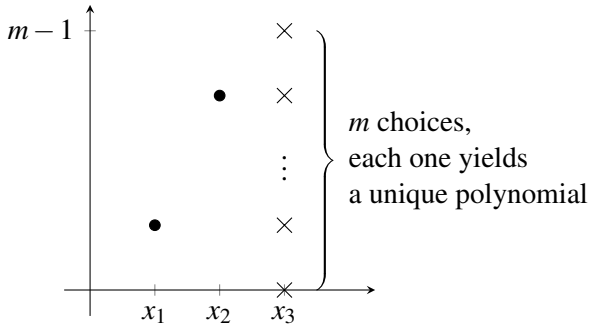
4 Counting

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 $p(x) = a_d x^d + a_{d-1}x^{d-1} + \cdots + a_0$ by specifying its $d+1$ coefficients a_i is known as the *coefficient representation* of $p(x)$. Is there any other way to specify $p(x)$?

Yes, 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, the polynomial can take any one of m values at each of these three points, 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 $0, 1, \ldots, d$, for a total of m^{d+1} possibilities. 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, Lagrange interpolation can be used to 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, suppose 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 two 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 two 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 reals, there would be infinitely many polynomials of degree d that can go through d points! (Think of a line, which has degree 1. If you were just given one point, there would be infinitely many possibilities for the second point, each of which uniquely defines a line.)

5 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 serious 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 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 both of the following properties hold:

1. Any group of k of these officials can pool their information to figure out the launch code and initiate the strike.

2. No group of $k-1$ or fewer officials 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[4]?

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 we have:

1. 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 $P(0) = s$.

2. Any group of $k-1$ (or fewer) officials has no information about s! To see this, observe that they know only $k-1$ points through which $P(x)$, an unknown polynomial of degree $k-1$, passes. They wish to reconstruct $P(0) = s$. 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 that the officials have 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[5] about s.

[4]This is a note where polynomials are playing a starring role and so we are going to use polynomials to do this in a nicely structured way. However, your background with solving systems of linear equations should give you a hint as to how you could have come up with this. You know that you generically need k linear equations to solve for k unknowns. When you have fewer than k equations, in general you just get a solution for some unknowns in terms of other unknowns. You can't actually figure out exact values for any of the unknowns themselves. This tells you that a way to protect a secret is to accompany it with another $k-1$ pieces of sacrificial information that you don't care about, and then to distribute linear equations to participants. If k get together, and the equations have linear independence, then the group can solve for all the unknowns, including the secret. If there are fewer than k equations, you want it to be the case that you can't solve for any of the unknowns exactly. The sacrificial information protects the secret.

The role of polynomials here is simply to give us a structured way to get systems of linear equations that have nice properties.

[5]Note that this is one major 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. Another major reason is to keep the computations exact using integer arithmetic in a bounded range, rather than floating point operations over the reals that can suffer from numerical instability.

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 basis functions:

$$\Delta_3(x) = \frac{(x-4)(x-5)}{(3-4)(3-5)} = \frac{(x-4)(x-5)}{2} = 4(x-4)(x-5);$$
$$\Delta_4(x) = \frac{(x-3)(x-5)}{(4-3)(4-5)} = \frac{(x-3)(x-5)}{-1} = 6(x-3)(x-5);$$
$$\Delta_5(x) = \frac{(x-3)(x-4)}{(5-3)(5-4)} = \frac{(x-3)(x-4)}{2} = 4(x-3)(x-4).$$

(Note that all divisions above are to be interpreted as multiplication by the corresponding inverse mod 7; e.g., in the first line for Δ_3, division by 2 is really multiplication by $2^{-1} = 4 \pmod 7$.) 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 this computation!) Finally, 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.

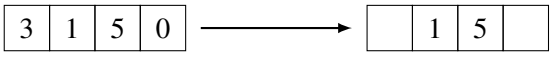
1 Error Correcting Codes

In this note, we will discuss the problem of transmitting messages across an unreliable communication channel. The channel may cause some parts of the message ("packets") to be lost, or dropped; or, more seriously, it may cause some packets to be corrupted. We will learn how to "encode" the message by introducing redundancy into it in order to protect against both of these types of errors. Such an encoding scheme is known as an "error correcting code." Error correcting codes are a major object of study in mathematics, computer science and electrical engineering; they belong to a field known as "Information Theory," one of the core computer sciences[1], along with the theory of computation, control theory, communication theory, and estimation/learning/signal-processing theory. In addition to the beautiful theory underlying them (which we will glimpse in this note), they are of great practical importance: every time you use your cellphone, satellite TV, DSP, cable modem, disk drive, CD-ROM, DVD player etc., or send and receive data over the internet, you are using error correcting codes to ensure that information is transmitted reliably. Error-correcting codes are also used in data centers and to speed up parallel computations.

There are, very roughly speaking, (at least) two distinct flavors of error correcting codes: algebraic[2] codes, which are based on polynomials over finite fields, and combinatorial codes, which are based on graph theory. In this note we will focus on algebraic codes, and in particular on so-called Reed-Solomon codes (named after two of their inventors). In doing so, we will be making essential use of the properties we learned about polynomials in the last lecture. These error-correcting codes also have a fundamentally linear-algebraic flavor to them as well, as will become clear throughout this note. Being based in finite fields allows the "information as degrees of freedom" perspective from linear-algebraic modeling to be made literal in terms of discrete symbols, bit-rates, etc.

1.1 Erasure Errors

We will consider two situations in which we wish to transmit information over 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 during transmission, as shown below:



We refer to such errors as *erasure errors*. Suppose that the message consists of n packets and suppose that

[1] This is also reflected in an interesting history. The IEEE was formed by the merger of two societies — the Institute of Radio Engineers and the American Institute of Electrical Engineers. The Radio Engineers were intimately involved with communication, cryptography, radar, etc. The IRE and AIEE merged to form the IEEE in 1963, but the IRE was already bigger than the AIEE by 1957. This merger was natural because electronics was becoming an important implementation substrate for both sets of engineers and the modern theory of electrical systems used the same mathematics as radio and signal processing (as well as control). This is why the word "EE" is often used to describe these computer sciences. At Berkeley, of course, this is all just a historical footnote since you simply experience EECS as a single entity, but the history remains in some of the course numbers.

[2] The ones we study are representative of what are called algebraic-geometry codes, because of their connections to algebraic geometry — the branch of mathematics that studies the roots of polynomial equations.

at most k packets are lost during transmission. We will show how to encode the initial message consisting of n packets into a redundant encoding consisting of $n+k$ packets such that the recipient can reconstruct the message from *any* n received packets. Note that in this setting the packets are labeled with headers, and thus the recipient knows exactly which packets were dropped during transmission.

We can assume without loss of generality that the content of each packet is a number modulo q, where q is a prime. For example, the content 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 perfectly suited[3] to solve this problem and are the backbone of this error-correcting scheme.

To see this, let us denote the message to be sent by $m_1, \ldots, m_n$, where each m_i is a number in $GF(q)$, 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 $1 \leq i \leq n$ (i.e., $P(x)$ contains all of the information about the message, and evaluating $P(i)$ gives the intended 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 points $n+j$. (Recall that our transmitted codeword should be redundant, i.e., it should contain more packets than the original message to account for the lost packets. This is the distinction between a codeword and a message. A codeword is what is transmitted and has redundancy by construction while the message is something that the user gives us and does not necessarily contain any redundancy.) Thus the transmitted codeword is $c_1 = P(1), c_2 = P(2), \ldots, c_{n+k} = P(n+k)$. Since we are working modulo q, we must make sure that $n+k \leq q$, but this condition does not impose a serious constraint since q is assumed to be very large.

3. 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 (not just the original n packets). Once we have reconstructed the polynomial P, we can evaluate $P(x)$ at $x = 1, \ldots, n$ to recover the original message $m_1, \ldots, m_n$.

Example

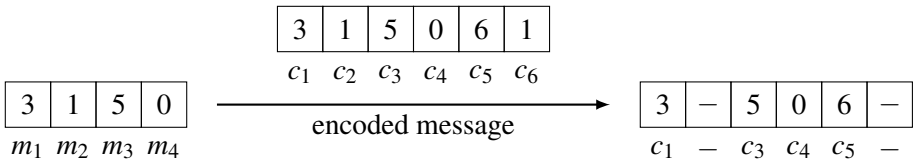
Suppose Alice wants to send Bob a message of $n = 4$ packets and she wants to guard against $k = 2$ lost packets. Then, assuming the packets can be coded up as integers between 0 and 6, Alice can work over $GF(7)$ (since $7 \geq n+k = 6$; of course, in real applications, we would be working over a much larger field!). Suppose the message that Alice wants to send to Bob is $m_1 = 3$, $m_2 = 1$, $m_3 = 5$, and $m_4 = 0$. The unique polynomial of degree $n-1 = 3$ described by these 4 points is $P(x) = x^3 + 4x^2 + 5$.

Exercise. We derived this polynomial using Lagrange interpolation mod 7. Check this derivation, and verify also that indeed $P(i) = m_i$ 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 codeword which consists of $n+k = 6$ packets, where $c_j = P(j)$ for $1 \leq j \leq 6$. So Alice will send $c_1 = P(1) = 3$, $c_2 = P(2) = 1$, $c_3 = P(3) = 5$, $c_4 = P(4) = 0$, $c_5 = P(5) = 6$, and $c_6 = P(6) = 1$.

Now suppose packets 2 and 6 are dropped, in which case we have the following situation:

[3] In real-world implementations, we do not do this. Instead, we work directly in finite fields that have size 2^{32} because that is a prime power and working with fields that are a power-of-two in size is convenient for computer operations. However, the efficient construction of such fields is beyond the scope of 70. A taste can be had in Math 114 and in EE229B.



From the values that Bob received (3, 5, 0, and 6), he uses Lagrange interpolation and computes the following basis polynomials (where everything should be interpreted mod 7):

$$\Delta_1(x) = \frac{(x-3)(x-4)(x-5)}{-24} \equiv 2(x-3)(x-4)(x-5) \pmod 7$$
$$\Delta_3(x) = \frac{(x-1)(x-4)(x-5)}{4} \equiv 2(x-1)(x-4)(x-5) \pmod 7$$
$$\Delta_4(x) = \frac{(x-1)(x-3)(x-5)}{-3} \equiv 2(x-1)(x-3)(x-5) \pmod 7$$
$$\Delta_5(x) = \frac{(x-1)(x-3)(x-4)}{8} \equiv (x-1)(x-3)(x-4) \pmod 7.$$

(Note that we have used the fact here that the inverses of -24, 4, -3 and 8 (mod 7) are 2, 2, 2 and 1 respectively.) He then reconstructs the polynomial $P(x) = 3 \cdot \Delta_1(x) + 5 \cdot \Delta_3(x) + 0 \cdot \Delta_4(x) + 6 \cdot \Delta_5(x) = x^3 + 4x^2 + 5 \pmod 7$. Bob then evaluates $m_2 = P(2) = 1$, which is the packet that was lost from the original codeword. More generally, no matter which two packets were dropped, following exactly the same method Bob can always reconstruct $P(x)$ and thus the original underlying message.

Exercise. Check Bob's calculation above, and verify that he really does reconstruct the correct polynomial, as claimed. Remember that all arithmetic must be done mod 7.

Let us consider what would happen if Alice sent one fewer packet. If Alice only sent c_j for $1 \le j \le n+k-1$, 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 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 smaller number of characters is impossible.

1.2 Polynomial Interpolation Revisited

Let us take a brief digression to discuss another method of polynomial interpolation (different from Lagrange interpolation discussed in the previous note) which will be useful in handling general errors. We need to make the underlying linear-algebra here more explicit so that we can better lean on it going forward.

Again, 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 $p(x) = a_d x^d + \cdots + a_1 x + a_0$ 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 variables are 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, and saying that the value of the polynomial is y_i: $a_d x_i^d + a_{d-1} x_i^{d-1} + \cdots + a_0 = y_i$.

Now solving these equations gives the coefficients of the polynomial $p(x)$. For example, suppose we are given the three pairs $(-1,2)$, $(0,1)$, and $(2,5)$; our goal is to 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 such systems of equations always have a solution and that it is unique. Fortunately, the work that we did in the previous note via Lagrange Interpolation has already established this. (**Can you see why?**) This is one of the advantages of being able to look at the same problem from different angles — this theme will become more prominent in later parts of the course.

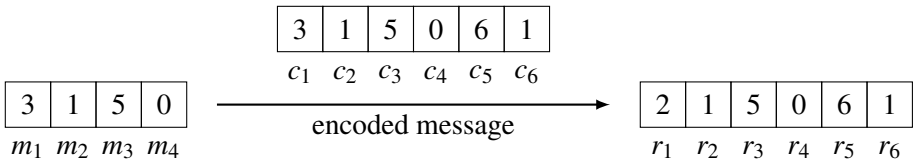
1.3 General Errors

Let us now return to our main topic of error correction, and consider a much more challenging scenario. Suppose that Alice wishes to communicate with Bob over a noisy channel (say, via a modem). Her message is $m_1, \ldots, m_n$, where we may think of the m_i's as characters (either bytes or characters in the English alphabet). The problem now is that some of the characters are *corrupted* during transmission due to channel noise. Thus Bob receives exactly as many characters as Alice transmits, but k of them are corrupted, and Bob has no idea which k these are! If you'd like, you can think of these as being corrupted by a malicious adversary that is only limited by being able to corrupt no more than k characters. Recovering from such general errors is much more challenging than recovering from erasure errors, though once again polynomials hold the key. As we shall see, Alice can still guard against k general errors, at the expense of transmitting only $2k$ additional characters (twice as many as in the erasure case we saw above).

We will again think of each character as a number modulo q for some prime q. (For the English alphabet, q is some prime larger than 26, say $q = 29$.) As before, we can describe the message by a polynomial $P(x)$ of degree $n-1$ over $GF(q)$, such that $P(1) = m_1, \ldots, P(n) = m_n$. As before, to cope with the errors Alice will transmit additional characters obtained by evaluating $P(x)$ at additional points. As mentioned above, in order to guard against k general errors, Alice must transmit $2k$ additional characters: thus the encoded codeword is $c_1, \ldots, c_{n+2k}$ where $c_j = P(j)$ for $1 \leq j \leq n+2k$, and $n+k$ of these characters that Bob receives are uncorrupted. As before, we must put the mild constraint on q that it be large enough so that $q > n+2k$.

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 (3150) as in our erasure example above, and that c_1 is corrupted from 3 to 2. This scenario can be visualized in the following figure:

From Bob's viewpoint, the problem of reconstructing Alice's message is the same as reconstructing the polynomial $P(x)$ from the $n+2k$ received characters $r_1, r_2, \ldots, r_{n+2k}$. In other words, Bob is given $n+2k$



values, $r_1, r_2, \ldots, r_{n+2k}$ modulo q, with the promise that there is a polynomial $P(x)$ of degree $n-1$ over $GF(q)$ such that $P(i) = r_i$ for $n+k$ distinct values of i between 1 and $n+2k$. Bob must reconstruct $P(x)$ from this data (in the above example, $n+k=5$, and $r_2 = P(2) = 1$, $r_3 = P(3) = 5$, $r_4 = P(4) = 0$, $r_5 = P(5) = 6$, and $r_6 = P(6) = 1$). Note, however, that Bob does not know *which* of the $n+k$ values are correct!

Does Bob even have sufficient information to reconstruct $P(x)$? Our first observation is that there is at most one polynomial $P(x)$ of degree at most $n-1$ such that $P(i) = r_i$ for at least $n+k$ of the values i between 1 and $n+2k$. To see this, suppose that $P(i) = r_i$ for $n+k$ points, and $P'(i) = r_i$ on $n+k$ (possibly different) points. As there are only $n+2k$ points in total, these sets must overlap on n points and thus $P(i) = r_i = P'(i)$ for n points which implies they must be the same degree $n-1$ polynomial. Furthermore, we observe that the original polynomial has $P(i) = r_i$ for $n+k$ points as there are at most k errors. Thus, $P(x)$ is the one and only polynomial consistent with Bob's information.

To summarize, then, Bob's task is to find a polynomial $P(x)$ of degree $n-1$ such that $P(i) = r_i$ for at least $n+k$ values of i. If he can do this, he can be sure that the polynomial he has found is in fact the original polynomial $P(x)$.

But how can Bob efficiently find such a polynomial? The issue at hand is the locations of the k errors. Let $e_1, \ldots, e_k$ be the k locations at which errors occurred (so that $P(i) = r_i$ for all $i \notin \{e_1, \ldots, e_k\}$). Bob doesn't know where these errors are.

Now Bob could try to guess where the k errors lie, but this would take too long (it would take exponential time, in fact). Instead, Bob will employ a clever trick based on the following definition. Consider the so-called *error-locator polynomial*

$$E(x) = (x - e_1)(x - e_2) \cdots (x - e_k).$$

Note that $E(x)$ is a polynomial of degree k (since x appears k times). Note also that Bob does not know this polynomial explicitly, because he does not know the positions e_i of the errors. However, he will use the polynomial symbolically, and will eventually compute the values e_i that appear in it!

Let us make a simple but crucial observation about this polynomial:

$$P(i)E(i) = r_i E(i) \qquad \text{for } 1 \le i \le n+2k. \tag{1}$$

To see this, note that it holds at points i at which no error occurred since at those points $P(i) = r_i$; and it is trivially true at points i at which an error occurred since then $E(i) = 0$.

This observation forms the basis of a very clever algorithm invented by Berlekamp and Welch. Looking more closely at the equalities in (1), we will show that they can be made to correspond to $n+2k$ linear equations in $n+2k$ unknowns, from which the locations of the errors and coefficients of $P(x)$ can be easily deduced (in analogous fashion to the interpolation scheme we saw in the previous section).

Define the polynomial $Q(x) \coloneqq P(x)E(x)$, which has degree $n+k-1$ and is therefore described by $n+k$ coefficients $a_0, a_1, \ldots, a_{n+k-1}$. The error-locator polynomial $E(x) = (x-e_1)\cdots(x-e_k)$ has degree k and is described by $k+1$ coefficients $b_0, b_1, \ldots, b_k$ but the leading coefficient (the coefficient b_k of x^k) is always 1.

So we can write:

$$Q(x) = a_{n+k-1}x^{n+k-1} + a_{n+k-2}x^{n+k-2} + \cdots + a_1x + a_0;$$
$$E(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0.$$

(Remember that we don't yet know any of the coefficients a_i or b_i.)

Now notice that, substituting $Q(x) = P(x)E(x)$ in (1), we get the $n+2k$ equations

$$Q(i) = r_iE(i) \qquad \text{for } 1 \le i \le n+2k.$$

Writing out the ith equation using the coefficients of $Q(x)$ and $E(x)$, we get

$$a_{n+k-1}i^{n+k-1} + a_{n+k-2}i^{n+k-2} + \cdots + a_1i + a_0 = r_i(i^k + b_{k-1}i^{k-1} + \cdots + b_1i + b_0) \pmod q.$$

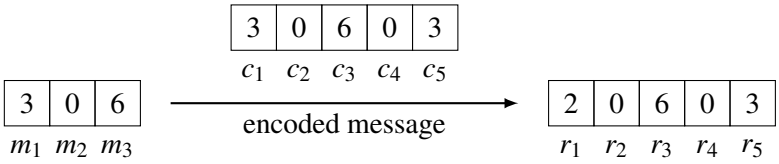
This is a set of $n+2k$ linear equations, one for each value of i, in the $n+2k$ unknowns $a_0, a_1, \ldots, a_{n+k-1}$, $b_0, b_1, \ldots, b_{k-1}$. We can solve the systems of linear equations and get $E(x)$ and $Q(x)$. We can then compute the ratio $\frac{Q(x)}{E(x)}$ to obtain $P(x)$. This is best illustrated by an example.

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 $a = 0, \ldots, g = 6$. So the message which Alice wishes for Bob to receive is "dag". Then Alice interpolates (e.g., using Lagrange: exercise!) 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 Alice knows that up to $k = 1$ character could be corrupted; then 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 actually corrupted, and Bob receives the character 2 instead of 3 (i.e., Alice sends the encoded codeword "dagad" but Bob instead receives "cagad"). Summarizing, we have the following situation: Let $E(x) = (x - e_1) = x + b_0$ be the



error-locator polynomial—remember, Bob doesn't know what $e_1 = -b_0$ is yet since he doesn't know where the error occurred—and let $Q(x) = a_3x^3 + a_2x^2 + a_1x + a_0$ (where again the coefficients a_i are unknown). Now Bob just substitutes $i = 1$, $i = 2$, …, $i = 5$ into the equations $Q(i) = r_iE(i)$ from (1) above and simplifies to get five linear equations in five unknowns (recall that we are working modulo 7 and that r_i is the value Bob received for the i-th character):

$$\begin{aligned}
a_3 + a_2 + a_1 + a_0 + 5b_0 &= 2 \\
a_3 + 4a_2 + 2a_1 + a_0 &= 0 \\
6a_3 + 2a_2 + 3a_1 + a_0 + b_0 &= 4 \\
a_3 + 2a_2 + 4a_1 + a_0 &= 0 \\
6a_3 + 4a_2 + 5a_1 + a_0 + 4b_0 &= 1
\end{aligned}$$

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 $P(x) = \frac{Q(x)}{E(x)} = \frac{x^3+6}{x-1} = x^2 + x + 1 \pmod 7$. 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".

Exercise. Verify the derivation of the above system of equations from the equalities $Q(i) = r_i E(i)$ for $i = 1, 2, 3, 4, 5$. Also, solve the equations (this is a little tedious, but not hard!) and check that your solution agrees with the one above. Remember that all the arithmetic should be done mod 7.

Exercise. Redo the example above for the case where the second character is corrupted from a "0" to a "5", and all other characters are uncorrupted.

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 actual 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, suppose that the solution we construct is $Q'(x), E'(x)$; how do we know that this solution satisfies the property that $E'(x)$ divides $Q'(x)$ and that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$?

To see that this is true, we note first that, based on our method for calculating $Q'(x), E'(x)$, we know that $Q'(i) = r_i E'(i)$ for $1 \le i \le n + 2k$; and of course we also have, by definition, $Q(i) = r_i E(i)$ for the same values of i. Multiplying the first of these equations by $E(i)$ and the second by $E'(i)$, we get

$$Q'(i)E(i) = Q(i)E'(i) \qquad \text{for } 1 \le i \le n + 2k, \tag{2}$$

since both sides are equal to $r_i E(i)E'(i)$. Equation (2) tells us that the two polynomials $Q(x)E'(x)$ and $Q'(x)E(x)$ are equal at $n + 2k$ points. But these two polynomials both have degree $n + 2k - 1$, so they are completely determined by their values at $n + 2k$ points. Therefore, since they agree at $n + 2k$ points, they must be the same polynomial, i.e., $Q(x)E'(x) = Q'(x)E(x)$ for all[4] x. Now we may divide through by the polynomial $E(x)E'(x)$ (which by construction is not the zero polynomial) to obtain $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$, which is what we wanted. Hence we can be sure that any solution we find is correct.

Exercise. (Trickier). Is the solution $Q'(x), E'(x)$ always unique? The above analysis tells us that the ratio must always satisfy $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$, but does not guarantee that $Q'(x) = Q(x)$ and $E'(x) = E(x)$. Hint: What happens in our example above when in fact none of the characters is corrupted (although Alice still assumes that $k = 1$ character may be corrupted)? Try writing out and solving the equations in this case. You should get a whole family of solutions, one for each possible value $b_0 \in GF(7)$. (Since in this scenario there is no error, b_0 is undetermined.) The other values can then be written in terms of b_0: $a_3 = 1$; $a_2 = 1 + b_0$;

[4] Note that this is a much stronger statement than equation (2). Equation (2) says that the *values* of the polynomials agree at certain points i; this statement says that the two polynomials are equal everywhere, i.e., they are the same polynomial.

$a_1 = 1 + b_0$; $a_0 = b_0$. But, no matter what the value of b_0, the quotient $\frac{Q(x)}{E(x)} = \frac{x^3+(1+b_0)x^2+(1+b_0)x+b_0}{x+b_0} = x^2+x+1$, so we again recover $P(x)$ (as we know we must).

1.4 (Optional) Distance properties

So far, we have talked about the Reed-Solomon codes in a way that constantly references their polynomial-evaluation based nature. However, it is useful to step back and ask ourselves what generic features of the codewords enabled us to recover from erasures and general errors.

It is useful here to treat a codeword as a string/vector of some fixed length, say L characters long. To protect a message of length n against k erasures, we saw that $L \geq n+k$ was required. To protect a message of length n against k general errors introduced by a malicious adversary, we saw that $L \geq n+2k$ was required. It is a natural question to wonder whether this is merely a feature of the Reed-Solomon codes or whether it is more fundamental.

To answer this, we define a kind of "distance" on strings of length L. We say that the *Hamming distance* between strings $\vec{s} = (s_1, s_2, \ldots, s_L)$ and $\vec{r} = (r_1, r_2, \ldots, r_L)$ is just the count of the number of positions in which the two strings differ. In mathematical notation:

$$d(\vec{s}, \vec{r}) = \sum_{i=1}^{L} \mathbf{1}(r_i \neq s_i) \tag{3}$$

where the notation $\mathbf{1}(r_i \neq s_i)$ denotes a function that returns a 1 if the condition inside is true and 0 if it is false.

The error and erasure correcting properties of a code are determined by the distance properties of the codewords $\vec{c}(m)$, Intuitively, if the codewords are too close together, then the code is more sensitive to errors and erasures. Having codewords far apart from each other allows a code, in principle, to tolerate more erasures and errors.

To make this precise, the *Minimum Distance* of a code is defined as the distance between the two closest codewords. Let m and $\widetilde{m}$ be two distinct messages $m \neq \widetilde{m}$. Then the minimum distance of the code is $\min_{m \neq \widetilde{m}} d(\vec{c}(m), \vec{c}(\widetilde{m}))$.

If the messages themselves are the set of strings of length n, then the minimum distance of the set of messages is just 1 because two distinct strings must differ somewhere. So clearly, when the minimum distance is 1, there is no protection against errors or erasures.

When the minimum distance is larger than 1, then there is some protection against erasures and errors. Suppose that the minimum distance was 2 and one position was erased in a codeword. In principle, cycling through all the possible characters for that position would certainly yield at least 1 codeword because the true codeword could be obtained that way. But notice that no other codeword could be obtained that way since if one were to be obtained, it would have a Hamming distance of just 1 — and we stipulated that the minimum distance was 2.

It is a simple exercise to generalize the above argument to show that when the minimum distance is $k+1$ or better, then the code can in principle recover from k erasure errors. If the minimum distance is k or less, then there is clearly a codeword pair for which erasing k positions would make the pair ambiguous. The resulting string could have come from either of these codewords. So we can't hope for anything better.

For general errors, the situation is a little trickier. To get an intuition for what the corresponding story should be, imagine that you are an attacker and want to confuse the decoder. You see the encoded codeword. What

are you going to do? It is intuitively clear that you will want to make the received string look like it came from another codeword. What codeword would you choose to impersonate? Intuitively, it makes sense to look for the closest codeword in the neighborhood. Suppose it was at a Hamming distance of d. That means that if you were to change d positions, then you could make the received string look exactly like this other codeword. Clearly, the decoder is pretty much guaranteed to make an error at this point.

But what if you only changed $d-1$ positions to partially impersonate the other codeword? At this point, the decoder is facing a choice. It could decode to this other codeword and chalk the single-position discrepancy up to general errors, or it could decode to the true codeword and think that $d-1$ positions have been changed. What choice will it make? In general, the decoder will be perfectly confused between these two choices if exactly $\frac{d}{2}$ errors have been made in a malicious way designed to partially impersonate this other codeword. However, if the number of general errors are strictly less than half the minimum distance of the code, in principle we should be able to decode the unique codeword that is less than $\frac{d}{2}$ from the received string.

The distance properties of Reed-Solomon Codes

It turns out that Reed-Solomon codes have the best-possible distance properties. (Of course, that is only a part of their attraction. Their other attraction is that their algebraic structure gives us a very nice efficient way of decoding these codes by solving systems of linear equations instead of brute-force searching through all nearby codewords.)

Theorem: The Reed Solomon code that takes n message characters to a codeword of size $n+2k$ has minimum distance $2k+1$.

Proof: We prove this using the two claims:

Claim 1) The minimum distance is $\leq 2k+1$

Claim 2) The minimum distance is $\geq 2k+1$

If we show that both (1) and (2) are true, then the minimum distance must be $2k+1$.

Proof of Claim (1):

We prove Claim (1) by constructing an example. If we can show there exist two codewords $\vec{c}_a$ and $\vec{c}_b$ such that $d(\vec{c}_a, \vec{c}_b) \leq 2k+1$, then the minimum over all distances between codewords must be $\leq 2k+1$.

Consider $\vec{m}_a = m_1 m_2 \cdots m_n$. Also consider $\vec{m}_b = m_1 m_2 \cdots \bar{m}_n$, where the two messages are identical in the first $n-1$ positions but differ in the last position. (All strings of length n are valid messages.) So $m_n \neq \bar{m}_n$. The Hamming distance between these two messages is 1. We use these messages to generate polynomials $P_a(x)$ and $P_b(x)$ and these are evaluated at i such that $1 \leq i \leq n+2k$ to generate the codewords $\vec{c}_a$ and $\vec{c}_b$ of length $n+2k$. We use value/interpolation encoding, so $P_a(i) = m_i = P_b(i)$ for $1 \leq i \leq n-1$. So the first $n-1$ positions of $\vec{c}_a$ and $\vec{c}_b$ are identical. So they can differ in at most $n+2k-(n-1) = 2k+1$ places. But this means the Hamming distance between them is $\leq 2k+1$. So we have constructed two codewords that have distance at most $2k+1$. Then the minimum distance between all pairs of codewords must be less than or equal to this. Hence, the minimum distance $\leq 2k+1$.

Proof of Claim (2):

Assume it were possible that the minimum distance between two distinct codewords could be $\leq 2k$. We use this to reach a contradiction. Let these two distinct codewords be $\vec{c}_a$ and $\vec{c}_b$, corresponding to distinct message polynomials $P_a(x)$ and $P_b(x)$. (Note: these have nothing to do with the $\vec{c}_a$ and $\vec{c}_b$ used in the Proof for Claim (1), we are just using the same variable names.) By the construction of Reed-Solomon codes, $P_a(x)$ and $P_b(x)$ have degree at most $n-1$ since the message is of size n.

Then, $d(\vec{c}_a, \vec{c}_b) \leq 2k$. So $\vec{c}_a$ and $\vec{c}_b$ must be identical in $\geq n+2k-2k = n$ positions. But $\vec{c}_a$ and $\vec{c}_b$ are just the evaluations of the message polynomials $P_a(x)$ and $P_b(x)$ at $1 \leq i \leq n+2k$. So $P_a(x)$ and $P_b(x)$ are identical on at least n points. But this means they must be the same polynomial, since they are of degree at most $n-1$. Which is a contradiction, since we assumed that $\vec{c}_a \neq \vec{c}_b$ were distinct codewords. So our assumptions must be false, and the minimum distance between two distinct codewords is $\geq 2k+1$.

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.

1 Counting Sequences

We start by considering a simple scenario. We pick $k \leq n$ elements from an n-element set $S = \{1, 2, \ldots, n\}$ one at a time while removing the sampled element from S. This type of sampling is called *sampling without replacement*. 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?

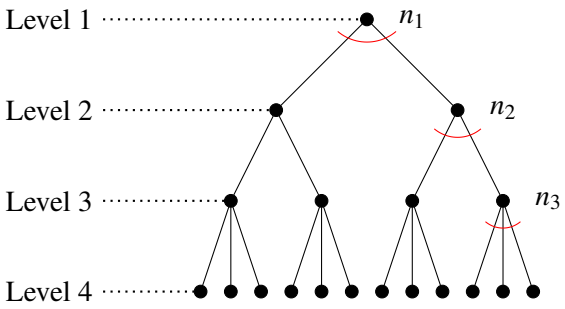
If we were dealing cards, the set would be $S = \{1, \ldots, 52\}$, where each number represents a card in a deck of 52 cards. Picking an element of S in this case refers to dealing one card. Note that once a card is dealt, it is no longer in the deck and so it cannot be dealt again. So the hand of k 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, regardless of 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 $52 \times 51 \times \cdots \times [52 - (k-1)]$.

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 $n_1 \times n_2 \times n_3 \times \cdots \times n_k$.

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 vertex at the second level,..., n_k at every vertex at the k-th level. Each path from the root to a vertex at level $k+1$ (i.e., a leaf vertex) 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 $n_1 \times n_2 \times n_3 \times \cdots \times n_k$. For example, if $n_1 = 2$, $n_2 = 2$, and $n_3 = 3$, then there are 12 leaves (i.e., outcomes).

2 Counting Sets

Consider a slightly different question. We would like to pick k distinct elements of $S = \{1, 2, \ldots, n\}$ (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 to obtain the same outcome?

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 have seen in Section 1, 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 $\frac{52!}{47!}$ distinct 5-card sequences, and there are 5! sequences placed in each bin. The total number of bins is therefore $\frac{52!}{47!5!}$.

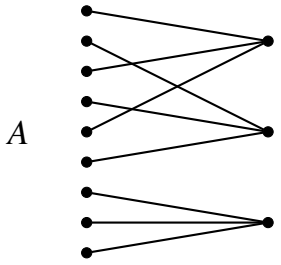
This quantity $\frac{n!}{(n-k)!k!}$ is used so often that there is special notation for it: $\binom{n}{k}$, pronounced "*n choose k*," and it's known as the *binomial coefficient*. This is the number of ways of picking k distinct elements from S, where the order of placement does not matter. Equivalently, it's the number of ways of choosing k objects out of a total of n distinct 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 are made does not matter. Let A be the set of ordered objects and let B be the set of unordered objects. If there exists an m-to-1 function f from A to B, we can count the number of ordered objects (pretending that the order matters) and divide by m (the number of ordered objects per unordered objects) to

obtain $|B|$, the number of unordered objects.

Note that we are assuming that 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:



$$|B| = \frac{|A|}{m}$$

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 $\frac{52!}{47!5!}$.

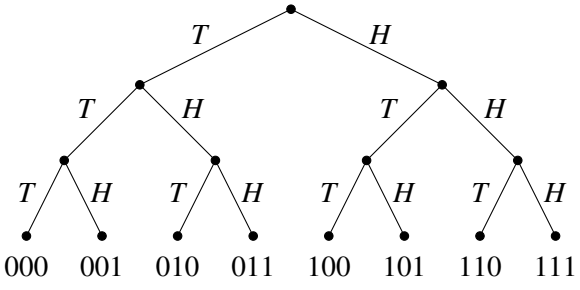
3 Sampling with Replacement

Sometimes we wish to consider a different scenario of sampling where after we choose an element from $S = \{1, 2, \ldots, n\}$, we throw it back into S so that we can choose it again. Assume that we are still picking k elements out of S one at a time and that order matters. How many distinct sequences of k elements can we obtain under this new sampling scheme? We can use the First Rule of Counting. Since we have n choices in each trial, $n_1 = n_2 = \cdots = n_k = n$. 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).

3.1 Coin Tosses

Let us return to coin tosses. How many different outcomes are there if we toss a coin k times? By an outcome, we mean a length-k sequence consisting of heads and tails. Suppose we represent each outcome by a binary string of length k where 0 corresponds to a tail (T) and 1 a head (H). For example, the string 001 would represent an outcome of 2 tails followed by a head. The following tree, in which the leaves are in 1-to-1 correspondence with the set of possible outcomes, illustrates the experiment for $k = 3$:



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 the set S in multiple trials). Order also matters; e.g., 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 $n = 2$.

3.2 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 $n^2 = 6^2 = 36$.

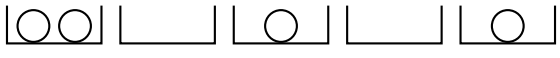
3.3 Sampling with Replacement, but where Order Does Not Matter

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 trickier to analyze. It may seem natural to apply the Second Rule of Counting because order does not matter. Let's consider this method. We first pretend that order matters and observe 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 $S = \{1,2,\ldots,n\}$ 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, as described below.

Assume we have one bin for each element of S, so n bins in total. 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. The following picture illustrates an example of placement where $S = \{1,\ldots,5\}$ and $k = 4$:



001 101 10

Counting the number of multisets 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 Section 2. 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!

In a sense, we have found the zeroth rule of counting:

Zeroth Rule of Counting: If a set A can be placed into a one-to-one correspondence with a set B (i.e. you can find a bijection between the two — an invertible pair of maps that map elements of A to elements of B and vice-versa), then $|A| = |B|$.

This is the very heart of what it means to count and is key to many combinatorial arguments as we will explore further in the next section.

4 Combinatorial Proofs

Combinatorial proofs are interesting because they rely on intuitive counting arguments rather than tedious algebraic manipulation. They often feel like proofs by stories — the same story, told from multiple points of view.

As a warm-up, let's prove the Binomial Theorem using counting arguments:

Theorem 10.1 (Binomial Theorem)**.** *For all $n \in \mathbb{N}$,*

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}.$$

Proof. The left-hand side can be written as the n-fold product $(a+b)(a+b)\cdots(a+b)$. Label these n factors as $f_1, f_2, \ldots, f_n$. When $(a+b)(a+b)\cdots(a+b)$ is expanded out via multiplication, it will result in a sum over 2^n monomials, and to each monomial in the sum, each factor f_i contributes either an a or a b. For example, in $(a+b)(a+b) = a^2+ab+ba+b^2$, a^2 is formed by multiplying a from f_1 and a from f_2; ab is formed by multiplying a from f_1 and b from f_2; ba is formed by multiplying b from f_1 and a from f_2; b^2 is formed by multiplying b from f_1 and b from f_2. In general, the monomial $a^k b^{n-k}$ is obtained when k copies of a's and $n-k$ copies of b's are multiplied. There are exactly $\binom{n}{k}$ ways to choose k copies of a's and $n-k$ copies of b's from the n factors $f_1, f_2, \ldots, f_n$. □

This well-known result has numerous applications. For example, the following combinatorial identity can be easily derived using Theorem 10.1:

Corollary 10.1. *For all $n \in \mathbb{N}$,*

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = 0.$$

Proof. Set $a = -1$ and $b = 1$ in Theorem 10.1. □

In what follows, we will consider more sophisticated versions of combinatorial proofs. The key idea is to show that the two sides of a given equation correspond to two different ways to count the same kind of objects. In more advanced settings, the goal is to establish a bijection between the set of objects counted on the left-hand side of an identity with the set of objects counted on the right-hand side. We illustrate these concepts below.

4.1 Combinatorial Identities

Using combinatorial arguments, we can prove complex facts, such as

$$\binom{n}{k+1} = \binom{n-1}{k} + \binom{n-2}{k} + \cdots + \binom{k}{k}, \tag{1}$$

which is known as the *hockey-stick identity*. 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 of (1) is just the number of ways of choosing a subset X with $k+1$ elements from a set $S = \{1, \ldots, n\}$. Let us think about a different process that results in the choice of a subset with $k+1$ elements. We start by picking the lowest-numbered element of X. If we picked 1, to finish constructing X, we must now choose k elements out of the remaining $n-1$ elements of S greater than 1; this can be done in $\binom{n-1}{k}$ ways. If instead the lowest-numbered element we picked is 2, then we have to choose k elements from the remaining $n-2$ elements of S greater than 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 1. 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:

$$\textbf{First element selected is} \begin{cases} \text{element } 1, & \text{leading to } \binom{n-1}{k} \text{ remaining ways,} \\ \text{element } 2, & \text{leading to } \binom{n-2}{k} \text{ remaining ways,} \\ \text{element } 3, & \text{leading to } \binom{n-3}{k} \text{ remaining ways,} \\ \quad\vdots & \quad\vdots \\ \text{element } (n-k), & \text{leading to } \binom{k}{k} \text{ remaining ways.} \end{cases}$$

These factors correspond to the terms on the right-hand side of (1). (Note that the lowest-numbered object we select cannot be higher than $n-k$ as we have to select $k+1$ distinct objects.)

Another combinatorial identity we will prove is

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n. \tag{2}$$

This identity actually follows immediately from setting $a = 1$ and $b = 1$ in the Binomial Theorem above, but we will provide a more direct combinatorial proof. Suppose we have a set S with n distinct elements. On the left-hand side of (2), $\binom{n}{i}$ 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 of (2) 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 j-th 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).

Table 1: Permutations of $\{1,2,3\}$ and the number of fixed points.

(π_1,π_2,π_3)	# fixed points
$(1,2,3)$	3
$(1,3,2)$	1
$(2,1,3)$	1
$(2,3,1)$	0
$(3,1,2)$	0
$(3,2,1)$	1

Let us look at an example, where $S = \{1,2,3\}$ (so $n = 3$). Enumerate all $2^3 = 8$ possible subsets of S: $\varnothing, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}$. The eight 3-bit vectors corresponding to these subsets are $(0,0,0), (1,0,0), (0,1,0), (0,0,1), (1,1,0), (1,0,1), (0,1,1), (1,1,1)$, respectively. On the left-hand side of equation (2), 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 $\varnothing = \{\}$. 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 every element of S). Summing, we get $1+3+3+1 = 8$, as expected.

4.2 Permutations and Derangements

Question: Suppose we collect the homeworks of n students, randomly shuffle them, and return them to the students. If we repeated this experiment many times, then on average how many students would receive their own homework?

At first this may seem like a rather challenging problem, but you will be able to answer this question in a few lines later in the course once you have some basic tools of probability theory under your belt. For now, let's translate the question into a mathematical statement and consider a related counting problem.

Label the homeworks as $1,2,\ldots,n$ and let π_i denote the homework that is returned to the i-th student. Then, the vector $\pi = (\pi_1,\ldots,\pi_n)$ corresponds to a permutation of the set $\{1,\ldots,n\}$. 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 total, there are $n!$ distinct permutations of $\{1,\ldots,n\}$.

In this setting, the i-th student receives their 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 π. To illustrate the idea concretely, let us consider the example $n = 3$. Table 1 shows a complete list of all $3! = 6$ permutations of $\{1,2,3\}$, together with the corresponding number of fixed points.

We now consider an interesting counting problem concerning derangements, which are defined as follows.

Definition 10.1 (Derangement)**.** *A permutation with no fixed points is called a* derangement*.*

Let D_n denote the number of derangements of $\{1,2,\ldots,n\}$. As can be seen in Table 1, $D_3 = 2$; specifically, the derangements are $(2,3,1)$ and $(3,1,2)$. Our goal is to find a formula for D_n as a function of n for all $n \in \mathbb{Z}^+$. We will learn two different ways to solve this problem.

The first method involves a combinatorial proof. For concreteness, let's consider the case of $n = 4$. Table 2a lists all derangements of $\{1,2,3,4\}$, with each row corresponding to a derangement. In the first derangement, note that the entries shown in blue corresponds to a derangement of $\{1,2\}$. In the second and third derangements, the entries shown in red are in 1-to-1 correspondence with the derangements of $\{1,2,3\}$,

shown in Table 2b. Note that the red 4 in Table 2a satisfies the same constraint as the number 3 in Table 2b; specifically, $\pi_3 \neq 4$ in the former and $\pi_3 \neq 3$ in the latter. In summary, the above discussion suggests that D_4 is related to D_2 and D_3. The following theorem formalizes this relationship.

Table 2: List of derangements. (a) $n = 4$. (b) $n = 3$.

(a)

π_1	π_2	π_3	π_4
2	**1**	4	3
2	**4**	**1**	3
4	**1**	**2**	3
4	3	2	1
3	4	2	1
2	3	4	1
3	4	1	2
4	3	1	2
3	1	4	2

(b)

π_1	π_2	π_3
2	3	1
3	1	2

Theorem 10.2. *For an arbitrary positive integer $n \geq 3$, the number D_n of derangements of $\{1,\ldots,n\}$ satisfies*

$$D_n = (n-1)(D_{n-1} + D_{n-2}). \tag{3}$$

Proof. We provide a combinatorial proof by establishing relevant bijections. In a derangement $(\pi_1,\ldots,\pi_n)$ of $\{1,\ldots,n\}$, suppose $\pi_n = j \in \{1,\ldots,n-1\}$; i.e., j is what n maps to under the permutation. There are $n-1$ choices for j, and this gives rise to the multiplicative factor $(n-1)$ in (3). (The case of $j = n$ is excluded, since a derangement cannot have $\pi_n = n$.) For a given value of $j \in \{1,\ldots,n-1\}$, there are two cases to consider:

Case 1: If $\pi_j = n$, then j and n get swapped, and there is a 1-to-1 correspondence between the set of derangements of the remaining elements and the set of derangements of $\{1,\ldots,n-2\}$. This gives rise to D_{n-2}.

Case 2: If $\pi_j \neq n$, then n satisfies the same constraint as j, so there is a 1-to-1 correspondence between the set of derangements of $\{1,\ldots,j-1,j+1,\ldots,n\}$ and the set of derangements of $\{1,\ldots,n-1\}$. This gives rise to D_{n-1}.

This completes our derivation of (3). □

Together with the boundary conditions $D_1 = 0$ and $D_2 = 1$ (check this), D_n can be determined efficiently (in linear time in n) using the recursion in (3) and memoization. In fact, the recursion can be solved analytically to yield (see Exercise 2) an explicit formula:

$$D_n = n! \sum_{k=0}^{n} \frac{(-1)^k}{k!}. \tag{4}$$

Coming up with the recursion in (3) and solving it require some creativity. As mentioned earlier, there is an alternate way to tackle this counting problem, which turns out to be a more straightforward approach. This alternate method utilizes a powerful tool called the principle of inclusion-exclusion, which we now describe.

5 The Principle of Inclusion-Exclusion

In this section we will learn a useful formula that can be employed to count objects without over-counting. We start with a familiar example. Let A_1 and A_2 be two subsets of the same finite set A, and suppose we want to count the number of elements in $A_1 \cup A_2$. If A_1 and A_2 are disjoint, then clearly $|A_1 \cup A_2| = |A_1| + |A_2|$. If A_1 and A_2 have common elements, however, then those elements would be counted twice in the previous formula. To correct for this overcounting, we need to subtract $|A_1 \cap A_2|$ from $|A_1| + |A_2|$, thus yielding

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|.$$

The principle of inclusion-exclusion generalizes this concept to an arbitrary number of subsets.

Theorem 10.3 (Inclusion-Exclusion)**.** *Let $A_1, \ldots, A_n$ be arbitrary subsets of the same finite set A. Then,*

$$|A_1 \cup \cdots \cup A_n| = \sum_{k=1}^{n} (-1)^{k-1} \sum_{S \subseteq \{1,\ldots,n\}: |S|=k} |\cap_{i \in S} A_i|. \tag{5}$$

Remarks:

1. The inner summation in (5) is over all size-k subsets of $\{1, 2, \ldots, n\}$. More explicitly, (5) can be written as

$$|A_1 \cup \cdots \cup A_n| = \sum_{i=1}^{n} |A_i| - \sum_{i<j} |A_i \cap A_j| + \sum_{i<j<k} |A_i \cap A_j \cap A_k| - \sum_{i<j<k<l} |A_i \cap A_j \cap A_k \cap A_l| + \cdots$$
$$+ (-1)^{n-1} |A_1 \cap A_2 \cap \cdots \cap A_n|, \tag{6}$$

   where $\sum_{i<j}$ denotes summing over all $i, j \in \{1, \ldots, n\}$ such that $i < j$, and so on.

2. The standard way to prove Theorem 10.3 is via induction on n. Here, we will provide a combinatorial proof.

Proof. First, note that if $a \notin A_1 \cup \cdots \cup A_n$, then $a \notin A_i$ for all $i \in \{1, \ldots, n\}$ and hence $a \notin \cap_{i \in S} A_i$ for any $S \subseteq \{1, \ldots, n\}$, so it is not counted by the right-hand side (RHS) of (5). We now show that if $a \in A_1 \cup \cdots \cup A_n$, then it is counted exactly once by the RHS. Define the index set $M \subseteq \{1, \ldots, n\}$ as

$$M = \{i \in \{1, \ldots, n\} \mid a \in A_i\};$$

that is, $a \in A_i$ if and only if $i \in M$. Note that $m := |M| \geq 1$, since $a \in A_1 \cup \cdots \cup A_n$. Clearly, if $S \not\subseteq M$, then $a \notin \cap_{i \in S} A_i$ (i.e., if S contains an index i not in M, then $a \notin A_i$, so $a \notin \cap_{i \in S} A_i$). Furthermore, $a \in \cap_{i \in S} A_i$ for any non-empty subset $S \subseteq M$. Therefore, the number of times that a is counted on the RHS of (5) is

$$\sum_{k=1}^{m} (-1)^{k-1} \sum_{S \subseteq M: |S|=k} 1 \;=\; \sum_{k=1}^{m} (-1)^{k-1} \binom{m}{k} \;=\; 1,$$

where the first equality follows from the fact that the number of distinct k-element subsets of M is $\binom{m}{k}$, while the second equality follows from Corollary 10.1. □

Application: Derangements Revisited

We now illustrate the power of Theorem 10.3 by applying it to derive the formula shown in (4) for the number D_n of derangements of $\{1,\ldots,n\}$.

Let A_i denote the set of all permutations of $\{1,\ldots,n\}$ with i being a fixed point. Then, since there are $n!$ distinct permutations of $\{1,\ldots,n\}$ and $|A_1 \cup A_2 \cup \cdots \cup A_n|$ counts the number of permutations with at least one fixed point, we have

$$D_n = n! - |A_1 \cup A_2 \cup \cdots \cup A_n|. \tag{7}$$

What is the cardinality of A_i? We know that i maps to i, while the remaining $n-1$ elements can be arranged in an arbitrary way. So, $|A_i| = (n-1)!$. Similarly, for $i \neq j$, i maps to i and j maps to j, while the remaining $n-2$ elements can be arranged in an arbitrary way, so we conclude $|A_i \cap A_j| = (n-2)!$. In general, for any subset $S \subseteq \{1,\ldots,n\}$ of size $|S| = k$, $|\cap_{i\in S} A_i| = (n-k)!$, while there are $\binom{n}{k}$ such subsets. Thus, the inclusion-exclusion formula (5) gives

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{k=1}^{n} (-1)^{k-1} \binom{n}{k} (n-k)! = \sum_{k=1}^{n} (-1)^{k-1} \frac{n!}{k!(n-k)!} (n-k)! = n! \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k!}.$$

Substituting this result into (7) gives

$$D_n = n! - n! \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k!} = n! \sum_{k=0}^{n} \frac{(-1)^k}{k!}.$$

6 Exercises

1. Provide a direct combinatorial proof of Corollary 10.1.

2. Using (3) and the boundary conditions $D_1 = 0$ and $D_2 = 1$, derive (4). [Hint: For $n \geq 2$, define $Q_n = D_n - nD_{n-1}$ and obtain a recursion for Q_n.]

3. Prove Theorem 10.3 using induction.

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes **Note 11**

To Infinity And Beyond: Countability and Computability

This note ties together two topics that might seem like they have nothing to do with each other. The nature of infinity (and more particularly, the distinction between different levels of infinity) and the fundamental nature of computation and proof. This note can only scratch the surface — if you want to understand this material more deeply, there are wonderful courses in the Math department as well as CS172 and graduate courses like EECS229A that will connect this material to the nature of information and compression as well.

1 Bijections

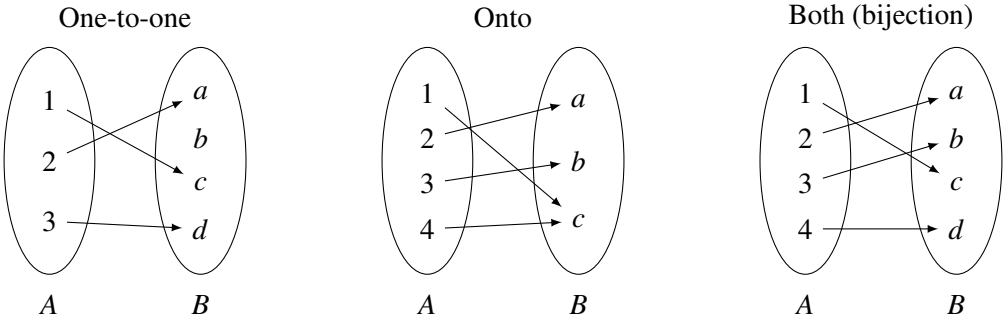
Two finite sets have the same size if and only if their elements can be paired up, so that each element of one set has a unique partner in the other set, and vice versa. We formalize this through the concept of a *bijection*.

Consider a function (or mapping) f that maps elements of a set A (called the *domain* of f) to elements of set B (called the *range* of f). Since f is a function, it must specify, for each element $x \in A$ ("input"), exactly one element $f(x) \in B$ ("output"). Recall that we write this as $f : A \to B$. We say that f is a *bijection* if every element $a \in A$ has a unique *image* $b = f(a) \in B$, and every element $b \in B$ has a unique *pre-image* $a \in A : f(a) = b$.

f is a *one-to-one function* (or an *injection*) if f maps distinct inputs to distinct outputs. More rigorously, f is one-to-one if the following holds: $x \neq y \implies f(x) \neq f(y)$.

f is *onto* (or *surjective*) if it "hits" every element in the range (i.e., each element in the range has at least one pre-image). More precisely, a function f is onto if the following holds: $(\forall y \; \exists x)(f(x) = y)$.

Here are some simple examples to help visualize one-to-one and onto functions, and bijections:



Note that, according to the above definitions, f is a bijection if and only if it is both one-to-one and onto.

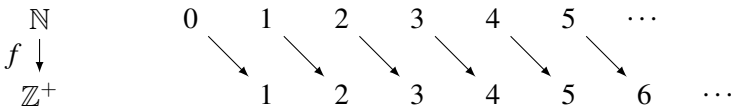
Exercise. Let $f : A \to B$ be a bijection. Show that f has an *inverse* $f^{-1} : B \to A$ that satisfies $f^{-1}(f(a)) = a$ for all $a \in A$, and that f^{-1} is also a bijection.

---

2 Cardinality

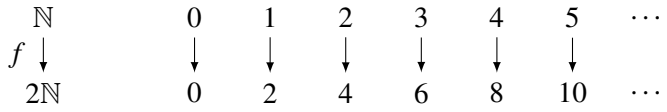
How can we determine whether two sets have the same *cardinality* (or "size")? The answer to this question, reassuringly, lies in early grade school memories: by demonstrating a *pairing* between elements of the two sets. More formally, we need to demonstrate a *bijection* f between the two sets. The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

Our first question about infinite sets is the following: 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 define a mapping between the natural numbers and the positive integers as follows:



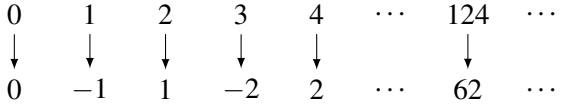
Why is this mapping a bijection? Clearly, the function $f : \mathbb{N} \to \mathbb{Z}^+$ is onto because every positive integer is hit. And it is also one-to-one because no two natural numbers have the same image. (The image of n is $f(n) = n+1$, so if $f(n) = f(m)$ then we must have $n = m$.) Since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

What about the set of *even* natural numbers $2\mathbb{N} = \{0,2,4,6,\ldots\}$? 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:



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 infinitely many 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 f:



In other words, our function is defined as follows:

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

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,

$$f(x) = f(y) \implies \frac{x}{2} = \frac{y}{2} \implies x = y.$$

Hence $x = y$. In the second case,

$$f(x) = f(y) \implies -\frac{x+1}{2} = -\frac{y+1}{2} \implies x = y.$$

So in both cases $f(x) = f(y) \implies x = y$, 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.

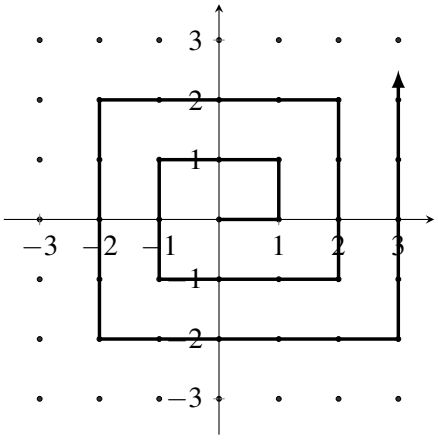
Since f is a bijection, this tells us that $\mathbb{N}$ and $\mathbb{Z}$ have the same size.

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.) It should be clear that this mapping maps every pair of integers injectively to a natural number, because each pair occupies a unique position along the spiral.

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.

Exercise. Show that the set $\mathbb{N} \times \mathbb{N}$ of all ordered pairs of natural numbers is countable.

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: $(101, 10, 111, 0, 100, 110)$. 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:

$$5x^5 + 2x^4 + 7x^3 + 4x + 6$$
$$\downarrow$$
$$101210211122100 2110$$

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.

3 Cantor's Diagonalization

We have established that $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ all have the same cardinality. What about $\mathbb{R}$, the set of real numbers? Surely they are countable too? After all, the rational numbers, like the real numbers, are dense (i.e., between any two rational numbers a, b there is a rational number, namely $\frac{a+b}{2}$). 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. We shall now prove this, using a beautiful argument due to Cantor that is known as *diagonalization*. In fact, we will show something even stronger: the real numbers in the interval $[0,1]$ are uncountable!

Exercise. Show how to find a rational number between any two (distinct) real numbers.

In preparation for the proof, recall that any real number can be written out uniquely as an infinite decimal with no trailing zeros. In particular, a real number in the interval $[0,1]$ can be written as $0.d_1d_2d_3\ldots$. In this representation, we write for example 1 as $0.999\ldots$[1], and 0.5 as $0.4999\ldots$. (Thus rational numbers will always be represented as recurring decimals, while irrational ones will be represented as non-recurring ones. Importantly for us, all of these expressions will be infinitely long and unique.)

Theorem: The real interval $\mathbb{R}[0,1]$ (and hence also the set of real numbers $\mathbb{R}$) is uncountable.

Proof: Suppose towards a contradiction that there is a bijection $f : \mathbb{N} \to \mathbb{R}[0,1]$. Then, we can enumerate the real numbers in an infinite list $f(0), f(1), f(2), \ldots$ as follows:

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

$$
\begin{aligned}
f(0) &= 0\ .\textcircled{5}2\ 1\ 4\ 9\ 3\ 5\ 6\ \cdots \\
f(1) &= 0\ .\ 1\textcircled{4}1\ 6\ 2\ 9\ 8\ 5\ \cdots \\
f(2) &= 0\ .\ 9\ 4\textcircled{7}8\ 2\ 7\ 1\ 2\ \cdots \\
f(3) &= 0\ .\ 5\ 3\ 0\textcircled{9}8\ 1\ 7\ 5\ \cdots \\
&\vdots \qquad\qquad\quad \vdots
\end{aligned}
$$

The number circled in the diagonal can be viewed as some real number $r = 0.5479\ldots$, since it is an infinite decimal expansion. Now consider the real number s obtained by modifying every digit of r, say by replacing each digit d with $d+2 \pmod{10}$; 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, $f(n)$. But by construction s differs from $f(n)$ in the $(n+1)$th digit, so these two numbers cannot be equal! So we have constructed 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 $2 \pmod{10}$ as opposed to adding 1 is that the same real number can have two decimal expansions; for example $0.999\ldots = 1.000\ldots$. 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. (An alternative way of avoiding this potential pitfall is to replace each digit by some different digit chosen from the range $\{1,2,\ldots,8\}$.)

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 contradiction that our bijective function $f : \mathbb{N} \to \mathbb{Q}[0,1]$ produces the following mapping:

$$
\begin{aligned}
f(0) &= 0\ .\textcircled{1}4\ 0\ 0\ 0\ \cdots \\
f(1) &= 0\ .\ 5\textcircled{9}2\ 4\ 5\ \cdots \\
f(2) &= 0\ .\ 2\ 1\textcircled{4}2\ 1\ \cdots \\
&\vdots \qquad\qquad\quad \vdots
\end{aligned}
$$

This time, let us consider the number q obtained by modifying every digit of the diagonal, say by replacing each digit d with $d+2 \pmod{10}$. 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 that 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.

4 The Cantor Set

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 (open) interval $(\frac{1}{3},\frac{2}{3})$, leaving $[0,\frac{1}{3}] \cup [\frac{2}{3},1]$. We then proceed to remove the middle third of each of these two remaining intervals, and so on. The first three iterations are illustrated below:

The Cantor set contains all points that have *not* been removed: $C = \{x : x \text{ not removed}\}$. How much of the original unit interval is left after this process is repeated infinitely? Well, we start with an interval of length 1, and after the first iteration we remove $\frac{1}{3}$ of it, 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 these iterations infinitely often, we are left with:

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

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... 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.

5 Power Sets and Higher Orders of Infinity

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 $\mathscr{P}(S) = \{\{\}, \{1\}, \{2\}, \{3\}, \{1,2\},$

[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 $\mathbb{R}$).

$\{1,3\},\{2,3\},\{1,2,3\}\}$.

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, where a 1 in the ith position indicates that the ith element of S is in the subset, and a 0 indicates that it is not. In the example above, the subset $\{1,3\}$ corresponds to the string 101. 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: $|\mathscr{P}(\mathbb{N})| > |\mathbb{N}|$.

Proof: Suppose towards a contradiction that there is a bijection $f : \mathbb{N} \to \mathscr{P}(\mathbb{N})$. Recall that we can represent a subset by a binary string, with one bit for each element of $\mathbb{N}$. (So, since $\mathbb{N}$ is infinite, the string will be infinitely long. Contrast the case of $\{0,1\}^*$ discussed earlier, which consists of all binary strings of *finite* length.) Consider the following diagonalization picture in which the function f maps natural numbers x to binary strings which correspond to subsets of $\mathbb{N}$:

	0	1	2	3	4	5	$\cdots$
0	(1)	0	0	0	0	0	$\cdots$
1	0	(0)	0	0	0	0	$\cdots$
2	1	0	(1)	0	0	0	$\cdots$
$\vdots$							

In this example, we have assigned the following mapping: $0 \to \{0\}$, $1 \to \{\}$, $2 \to \{0,2\}$, $\ldots$ (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 $\mathbb{N}$. Clearly it is, since b is an infinite binary string which corresponds to a subset of $\mathbb{N}$. 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 $\mathbb{N}$. Thus we have a contradiction, implying that $\mathscr{P}(\mathbb{N})$ 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 for the limits of computation (the Halting Problem) and the foundations of logic in mathematics (Gödel's incompleteness theorem).

1 The Liar's Paradox

Recall that propositions are statements that are either true or false. We saw in an earlier lecture 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, assuming that the barber's statement is true, 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!

2 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 this 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*. The recursion theorem states that given any program $P(x,y)$, we can always convert it to another

[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.

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 a "self-aware" version of P, since Q essentially has access to its own description.

3 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.

Theorem: *The Halting Problem is uncomputable; i.e., there does not exist a computer program TestHalt with the behavior specified above on all inputs* (P,x).

Proof: Assuming for contradiction the existence of the program `TestHalt`, use it to construct the following 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):

	P_0	P_1	P_2	P_3	P_4	$\cdots$
P_0	(H)	L	H	L	H	$\cdots$
P_1	L	(L)	H	L	L	$\cdots$
P_2	H	H	(H)	H	L	$\cdots$
P_3	H	H	H	(H)	H	$\cdots$
$\vdots$						

The $(i, j)^{\text{th}}$ entry in the table above is H if program P_i halts on input P_j, and L (for "Loops") 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 questions we would like to answer about programs 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. And we cannot check to see if a given 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 ran into trouble when deciding 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 solve the Halting Problem?

For example, given a program P, what if we ask instead the question: "Does P halt 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 (such as P itself). However, it turns out this seemingly easier problem is still uncomputable! We prove this claim by showing that if we could solve the Easy Halting Problem, then we could also solve the Halting Problem itself; 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 solves 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)
  construct a program P' that, on input 0, returns P(x)
  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 $P(x)$. Such a program P' can be constructed very simply as follows:

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

That is, the new program P' ignores its input y and always returns $P(x)$. (Note that the descriptions of P and of x are "hard-wired" into P'.) 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 solve 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?" to another problem "Does P' halt on 0?", in the sense that if we know how to solve 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.

4 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:

$$(\forall n \geq 3)\ \neg(\exists x, y, z \in \mathbb{Z}^+)(x^n + y^n = z^n).$$

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 arithmetic is either inconsistent (there are false statements that can be proved) or incomplete

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

(there are true statements that cannot be proved). Gödel proved his result by exploiting the deep connection between proofs and arithmetic. Actually Gödel's theorem also embodies a deep connection between proofs and computation, which was illuminated after Turing formalized the definition of computation in 1936 via the notion of Turing machines and computability.

In the rest of this note, we will first sketch the essence of Gödel's proof, and then we will outline an easier proof of the theorem using what we know about the Halting Problem.

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 all of arithmetic.

Now suppose we can write the following statement:

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

Once we have this statement, there are two possibilities:

1. Case 1: $S(F)$ is provable. Then the statement $S(F)$ is true, but by inspecting the content of the statement 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 statement $S(F)$ 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 statement $S(F)$. This is the difficult part of Gödel's proof, which requires a clever encoding (a so-called "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 Halting Problem, which we have seen is impossible.

Recall that in the Halting Problem we want to decide whether a given program P halts on a given input x. For fixed P and x, let $S_{P,x}$ denote the proposition "P halts on input x." The key observation is that this proposition can be phrased as a statement in arithmetic. The form of the statement $S_{P,x}$ will be

$$\exists z(z \text{ encodes a valid halting execution sequence of } P \text{ on input } x).$$

Although the details require some work, your programming intuition should hopefully convince you that such a statement can be written, in a fairly mechanical way, using only the language of standard arithmetic, with the usual operators, connectives and quantifiers: basically the statement just has to check, step by step, that the string z (encoded as a very long integer in binary) lists out the sequence of states that a computer would go through when running program P on input x.

Now let us assume, for contradiction, that arithmetic is both consistent and complete. This means that, for any (P,x), the statement $S_{P,x}$ is either true or false, and that there must exist a proof in arithmetic of either $S_{P,x}$ or its negation, $\neg S_{P,x}$ (and not both). But now recall that a proof is simply a finite binary string. Therefore,

there are only countably many possible proofs, so we can enumerate them one by one and search for a proof of $S_{P,x}$ or $\neg S_{P,x}$. The following program performs this task:

```
Search(P, x)
  for every proof q:
    if q is a proof of S_{P,x} then output "yes"
    if q is a proof of ¬S_{P,x} then output "no"
```

The program Search takes as input the program P, and proceeds to check every possible proof until it finds either one that proves $S_{P,x}$, or one that proves $\neg S_{P,x}$. By assumption, we know that one of these proofs always exists, so the program Search will terminate in finite time, and it will correctly solve the Halting Problem. On the other hand, since we have already established that the Halting Problem is uncomputable, such a program Search cannot exist. Therefore, our initial assumption must be wrong, so it is not true that arithmetic is both consistent and complete.

Note that in the argument above we rely on the fact that, given a proof, we can construct a program that mechanistically checks whether it is a valid proof of a given proposition. This is a manifestation of the intimate connection between proofs and computation.

CS 70 Discrete Mathematics and Probability Theory
Fall 2024 Course Notes Note 13

Introduction to Discrete Probability

Probability theory has its origins in gambling—analyzing card games, dice, roulette wheels. Today it is an essential tool in engineering and the sciences. No less so in EECS, where its use is widespread in algorithms, systems, signal processing, learning theory and control/AI.

Here are some typical statements that you might see concerning probability:

1. The chance of getting a flush in a 5-card poker hand is about 2 in 1000.
2. The chance that this randomized primality testing algorithm outputs "prime" when the input is not prime is at most one in a trillion.
3. In this load-balancing scheme, the probability that any processor has to deal with more than 12 requests is negligible.
4. The average time between system failures is about 3 days.
5. There is a 30% chance of a magnitude 8.0 earthquake in Northern California before 2040.

Implicit in all such statements is the notion of an underlying *probability space*. This may be the result of a random experiment that we have ourselves constructed (as in 1, 2 and 3 above), or some model we build of the real world (as in 4 and 5 above). None of these statements makes sense unless we specify the probability space we are talking about: for this reason, statements like 5 (which are typically made without this context) are almost content-free.

In this note, we will try to understand all this more clearly. The first important notion here is that of a *random 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 the outcome is to occur when we actually perform the experiment.

1 Random Experiments

Typically, a random 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 Note 10. Recall from Note 10 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 random experiments. The outcome of a random experiment is called a *sample point*, and the *sample space* (often denoted by Ω) is the set of all possible outcomes of the experiment.

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 Ω has 16 elements, as illustrated in the following picture:

$$\Omega = \begin{array}{cccc} HHHH & HTHH & THHH & TTHH \\ HHHT & HTHT & THHT & TTHT \\ HHTH & HTTH & THTH & TTTH \\ HHTT & HTTT & THTT & TTTT \end{array}$$

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

2 Probability Spaces

A probability space is a sample space Ω, together with a probability $\mathbb{P}[\omega]$ (often also denoted as $\Pr[\omega]$) for each sample point ω, such that

- (Non-negativity): $0 \leq \mathbb{P}[\omega] \leq 1$ for all $\omega \in \Omega$.
- (Sum to 1): $\sum_{\omega \in \Omega} \mathbb{P}[\omega] = 1$, i.e., the sum of the probabilities over all outcomes is 1.

The easiest way to assign probabilities to sample points is to do it *uniformly*: if $|\Omega| = N$, then $\mathbb{P}[\omega] = \frac{1}{N}$, $\forall \omega \in \Omega$. 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 spaces soon.

After performing an experiment, we are often interested in knowing whether a certain event occurred. For example, we 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? The answer is to identify the event "exactly 2 H's" with the set of all those outcomes in which there are exactly two H's, namely: $\{HHTT, HTHT, HTTH, THHT, THTH, TTHH\} \subset \Omega$. Hence, formally an event A is just a subset of the sample space Ω, i.e., $A \subseteq \Omega$.

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

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

Note that $0 \leq \mathbb{P}[A] \leq 1$ for all $A \subseteq \Omega$, and $\mathbb{P}[\Omega] = 1$. The probability of getting exactly two H's in four coin tosses can be calculated using this definition as follows. Event 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 since there are six sample points in A, we obtain $\mathbb{P}[A] = 6 \cdot \frac{1}{16} = \frac{3}{8}$.

3 Examples

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

3.1 Coin Tosses

Suppose we have a coin with $\mathbb{P}[H] = p$ and $\mathbb{P}[T] = 1 - p$, 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 previous 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 $p = \frac{2}{3}$? Then the probabilities are different. For example, $\mathbb{P}[HHHH] = \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} = \frac{16}{81}$, while $\mathbb{P}[TTHH] = \frac{1}{3} \times \frac{1}{3} \times \frac{2}{3} \times \frac{2}{3} = \frac{4}{81}$. (Note: We have simply multiplied probabilities here; we will explain later why this is the correct thing to do for this example. It is NOT always OK to multiply probabilities like this.)

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

Next, consider the event B that there are exactly two heads. The probability of any particular outcome with two heads (such as $HTHT$) is $\left(\frac{2}{3}\right)^2 \left(\frac{1}{3}\right)^2$. 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, $\mathbb{P}[B] = 6\left(\frac{2}{3}\right)^2 \left(\frac{1}{3}\right)^2 = \frac{24}{81} = \frac{8}{27}$.

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 length-n sequences of H's and T's. If the coin has $\mathbb{P}[H] = p$, and if we consider any sequence of n coin flips with exactly r H's, then 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 and each has probability $p^r(1-p)^{n-r}$. So, $\mathbb{P}[C] = \binom{n}{r} p^r(1-p)^{n-r}$.

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

3.2 Rolling Dice

Consider rolling two fair dice. In this experiment, $\Omega = \{(i, j) : 1 \leq i, j \leq 6\}$. The probability space is uniform, i.e., all sample points have the *same* probability $\frac{1}{|\Omega|} = \frac{1}{36}$. Hence, the probability of any event A is

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

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 enumerating the sample points contained in each event, it can be easily shown that $|A| = 6$ and $|B| = 11$. Then, by the observation above, it follows that $\mathbb{P}[A] = \frac{6}{36} = \frac{1}{6}$ and $\mathbb{P}[B] = \frac{11}{36}$.

3.3 Card Shuffling

Consider the random experiment of shuffling a deck of standard playing cards. Here, Ω is equal to the set of all 52! permutations of the deck. We assume that the probability space is uniform. (Note that we are 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.) We can generalize this experiment to randomly "shuffling" n items, for arbitrary n. (For card shuffling, $n = 52$.) For example,

suppose we collect the homeworks of n students in a class, then redistribute them randomly, one per student. The result is a random *permutation* of the homeworks, with each outcome having probability $\frac{1}{n!}$.

3.4 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 Note 10, $|\Omega| = \binom{52}{5} = \frac{52\times51\times50\times49\times48}{5\times4\times3\times2\times1} = 2,598,960$. Assuming that the deck is well 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 familiar with poker, a *flush* is a hand in which all cards have the same suit, say Hearts.) Since the probability space is uniform, computing $\mathbb{P}[A]$ reduces to simply computing $|A|$, the number of poker hands that 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 $4 \times \binom{13}{5}$. Then we have

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

3.5 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: 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 $\{(b_1, b_2, \ldots, b_{20}) : 1 \leq b_i \leq 10 \text{ for each } i = 1, \ldots, 20\}$, where the component b_i denotes the bin in which ball i lands. The cardinality $|\Omega|$ of the sample space is equal to 10^{20}, since 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, $\mathbb{P}[A] = \frac{9^{20}}{10^{20}} = \left(\frac{9}{10}\right)^{20} \approx 0.12$.

Let B be the event that bin 1 contains at least one ball. This event is the *complement* $\overline{A}$ of A, i.e., it consists of precisely those sample points which are not in A. So $\mathbb{P}[B] = 1 - \mathbb{P}[A] \approx 0.88$. More generally, if we throw m balls into n bins, we have:

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

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 4 times is the special case in which the number of balls (m) is 4 and the number of bins (n) is 2. Rolling two dice is the special case in which $m = 2$ and $n = 6$.

3.6 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 $S = \{1, \ldots, 365\}$, and the random experiment consists of drawing a sample of n elements from S with replacement, 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 a pair of people have the same birthday. If we want to determine $\mathbb{P}[A]$, it might be simpler to instead compute the probability of the complement of A; i.e., $\mathbb{P}[\overline{A}]$, where $\overline{A}$ is the event that no two people have the same birthday. Since $\mathbb{P}[A] = 1 - \mathbb{P}[\overline{A}]$, we can then easily compute $\mathbb{P}[A]$.

We are again working in a uniform probability space, so we just need to determine $|\overline{A}|$. Equivalently, we are computing the number of ways 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 nth person, for a total of $365 \times 364 \times \cdots \times (365 - n + 1)$. This is simply an application of the First Rule of Counting from Note 10; we are sampling without replacement (since we need to avoid collisions) and the order matters.

In summary, we have

$$\mathbb{P}[\overline{A}] = \frac{|\overline{A}|}{|\Omega|} = \frac{365 \times 364 \times \cdots \times (365 - n + 1)}{365^n},$$

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

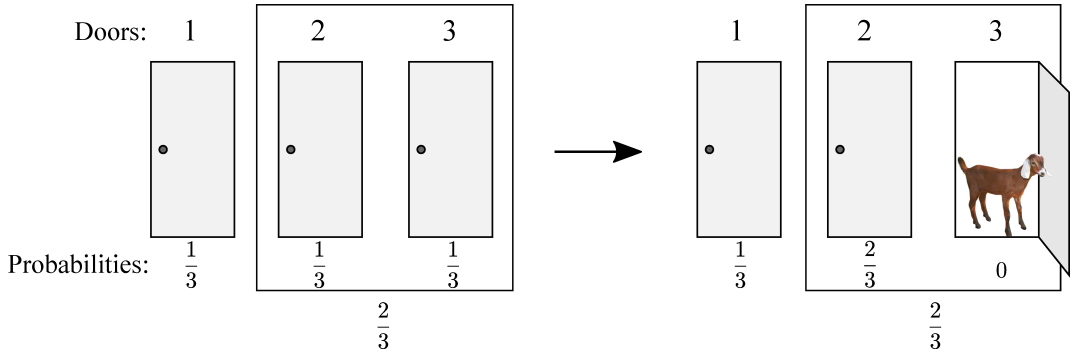
3.7 The Monty Hall Problem

This example is loosely based on an (in)famous 1970s TV game show hosted by Monty Hall. A contestant is shown three doors; behind one of the doors is a valuable prize (a car), and behind the other two are goats. The contestant picks a door (but does not 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/her current door, or switching to the other unopened one. The contestant wins the prize if and only if their chosen door is the correct one. The question is: Does the contestant have a better chance of winning if he/she switches doors?

Intuitively, it seems obvious that since there are only two remaining doors after Carol 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. We will see that actually, the contestant has a better chance of picking the car if he or she uses the switching strategy. We will first give an intuitive pictorial argument, and then take a more rigorous probability approach to the problem.

To see why it is in the contestant's best interests to switch, consider the following. Initially when the contestant chooses the door, he or she 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, the door the contestant originally chose still has a $\frac{1}{3}$ chance of winning, and 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:



What is the sample space here? Well, we can describe the outcome of the game (up to the point where the contestant makes his/her final decision) using a triple of the form (i, j, k), where $i, j, k \in \{1, 2, 3\}$. The values i, j, k respectively specify the location of the prize, the initial door chosen by the contestant, and the door opened by Carol. Note that some triples are not possible: e.g., $(1, 2, 1)$ is not, because Carol never opens the prize door. Thinking of the sample space as a tree structure, in which first i is chosen, then j, and finally k (depending on i and j), we see that there are exactly 12 sample points.

Assigning probabilities to the sample points here requires pinning down some assumptions:

- The prize is equally likely to be behind any of the three doors.
- Initially, the contestant is equally likely to pick any of the three doors.
- If the contestant happens to pick the prize door (so there are two possible doors for Carol to open), Carol is equally likely to pick either one. (Actually our calculation will have the same result no matter how Carol picks the door.)

From this, we can assign a probability to every sample point. For example, the point $(2, 1, 3)$ corresponds to the prize being placed behind door 2 (with probability $\frac{1}{3}$), the contestant picking door 1 (with probability $\frac{1}{3}$), and Carol opening door 3 (with probability 1, because she has no choice). So

$$\mathbb{P}[(2, 1, 3)] = \frac{1}{3} \times \frac{1}{3} \times 1 = \frac{1}{9}.$$

[Note: Again we are multiplying probabilities here, without proper justification!] Note that there are six outcomes of this type, characterized by having $i \neq j$ (and hence k must be different from both). On the other hand, we have

$$\mathbb{P}[(1, 1, 2)] = \frac{1}{3} \times \frac{1}{3} \times \frac{1}{2} = \frac{1}{18}.$$

And there are six outcomes of this type, having $i = j$. These are the only possible outcomes, so we have completely defined our probability space. Just to check our arithmetic, we note that the sum of the probabilities of all outcomes is $(6 \times \frac{1}{9}) + (6 \times \frac{1}{18}) = 1$.

Let's return to the Monty Hall problem. Recall that we want to investigate the relative merits of the "sticking" strategy and the "switching" strategy. Let's suppose the contestant decides to switch doors. The event W we are interested in is the event that the contestant wins. Which sample points (i, j, k) are in W? Well, since the contestant is switching doors, their initial choice j cannot be equal to the prize door, which is i. And all outcomes of this type correspond to a win for the contestant, because Carol must open the second non-prize

door, leaving the contestant to switch to the prize door. So W consists of all outcomes of the first type in our earlier analysis; recall that there are six of these, each with probability $\frac{1}{9}$. So $\mathbb{P}[W] = \frac{6}{9} = \frac{2}{3}$. That is, using the switching strategy, the contestant wins with probability $\frac{2}{3}$. It should be intuitively clear (and easy to check formally—try it!) that under the sticking strategy their probability of winning is $\frac{1}{3}$. (In this case, the contestant is really just picking a single random door.) So by switching, the contestant actually improves their odds by a huge amount!

4 Summary

The Monty Hall example well illustrates 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 contained in 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 people with PhDs to newspapers at the time, of the claim that the switching strategy in the Monty Hall problem does not improve the odds.

Conditional Probability, Independence, and Combinations of Events

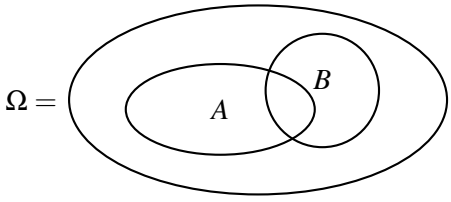
One of the key properties of coin flips is *independence*: if you flip a fair coin ten times and get ten T's, this does not make it more likely that the next coin flip will be H's. It still has exactly 50% chance of being H.

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? 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.

1 Conditional Probability

Let's consider an example with a smaller sample space. Suppose we toss $m = 4$ labeled balls into $n = 3$ labeled bins; this is a uniform sample space with $3^4 = 81$ sample points. From the previous note, we already know that the probability the first bin is empty is $(1 - \frac{1}{3})^4 = (\frac{2}{3})^4 = \frac{16}{81}$. What is the probability of this event *given* that the second bin is empty? Let A denote the event that the first bin is empty, and B the event that the second bin is empty. In the language of conditional probability, we wish to compute the probability $\mathbb{P}[A \mid B]$, which we read as "the conditional probability of A given B."

How should we compute $\mathbb{P}[A \mid B]$? 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 be the probability of each sample point $\omega \in B$ given that the event B occurs? If they all simply inherited their probabilities from Ω, then the sum of these probabilities would be $\sum_{\omega \in B} \mathbb{P}[\omega] = \mathbb{P}[B]$, which in general is less than 1. So, to get the correct normalization, we need to *scale* the probability of each sample point by $\frac{1}{\mathbb{P}[B]}$. That is, for each sample point $\omega \in B$, the new probability becomes

$$\mathbb{P}[\omega \mid B] = \frac{\mathbb{P}[\omega]}{\mathbb{P}[B]}.$$

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

$$\mathbb{P}[A \mid B] = \sum_{\omega \in A \cap B} \mathbb{P}[\omega \mid B] = \sum_{\omega \in A \cap B} \frac{\mathbb{P}[\omega]}{\mathbb{P}[B]} = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]}.$$

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

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]}.$$

Returning to our example of balls and bins, to compute $\mathbb{P}[A \mid B]$ we need to figure out $\mathbb{P}[A \cap B]$. But $A \cap B$ is the event that both the first two bins are empty, i.e., all four balls fall in the third bin. So $\mathbb{P}[A \cap B] = \frac{1}{81}$ (why?). Therefore,

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]} = \frac{1/81}{16/81} = \frac{1}{16}.$$

Not surprisingly, $\mathbb{P}[A \mid B] = \frac{1}{16} = 0.0625$ is quite a bit less than $\mathbb{P}[A] = \frac{16}{81} \approx 0.1975$; 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.

To compute $\mathbb{P}[A \mid B]$, we need to figure out $\mathbb{P}[A \cap B]$. 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, $\mathbb{P}[A \cap B] = \frac{12}{52 \cdot 51}$, while $\mathbb{P}[B]$, the probability of drawing an ace in the first trial, is $\frac{4}{52}$. Therefore,

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]} = \frac{3}{51},$$

which is less than $\mathbb{P}[A] = \frac{4}{52} = \frac{1}{13}$ (see Exercise 1). Hence, if the first card is an ace, it makes it less likely that the second card is also an ace.

2 Bayesian Inference

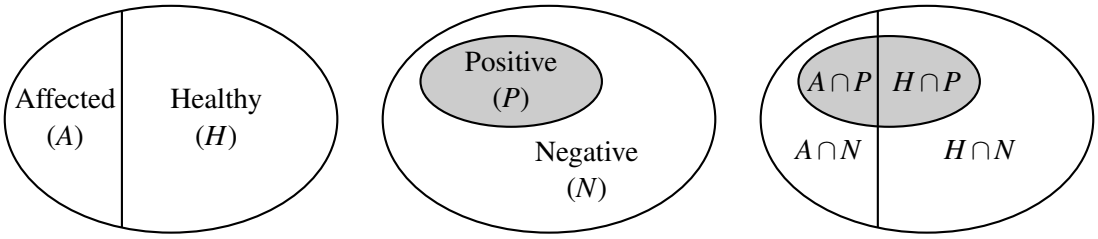
Now that we have introduced the notion of conditional probability, we can see how it is used in real world settings. Conditional probability is at the heart of a subject called *Bayesian inference*, used extensively in fields such as machine learning, communications and signal processing. Bayesian inference 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 $\mathbb{P}[A \mid B]$. In this interpretation, $\mathbb{P}[A]$ 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. $\mathbb{P}[A \mid B]$ can be interpreted as the *posterior* probability of A after the observation. It reflects our updated 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 the probability that the person has the disorder? (Note that this is presumably *not* the same as the simple probability that a random person in the population has the disorder, which is just $\frac{1}{20}$.) The implicit probability space here is the entire US population with equal probabilities.



The sample space here consists of all people in the US — denote their number by N (so $N \approx 330$ 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 as:

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

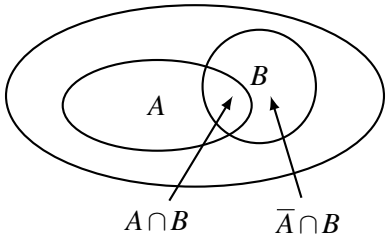
We want to calculate $\mathbb{P}[A \mid B]$. We can proceed as follows:

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]} = \frac{\mathbb{P}[B \mid A]\mathbb{P}[A]}{\mathbb{P}[B]}. \tag{1}$$

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

$$\mathbb{P}[B \mid A] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[A]}.$$

To evaluate (1), we need to compute $\mathbb{P}[B]$. This is the probability that a random person tests positive. To compute this, we can sum two values: the probability $\mathbb{P}[\overline{A} \cap B]$ that a healthy person tests positive and the probability $\mathbb{P}[A \cap B]$ that an affected person tests positive. We can sum because the events $\overline{A} \cap B$ and $A \cap B$ do not intersect:



By again applying the definition of conditional probability, we have:

$$\begin{aligned}\mathbb{P}[B] = \mathbb{P}[A \cap B] + \mathbb{P}[\overline{A} \cap B] &= \mathbb{P}[B \mid A]\mathbb{P}[A] + \mathbb{P}[B \mid \overline{A}]\mathbb{P}[\overline{A}] \\ &= \mathbb{P}[B \mid A]\mathbb{P}[A] + \mathbb{P}[B \mid \overline{A}](1 - \mathbb{P}[A]).\end{aligned} \tag{2}$$

Combining (1) and (2), we have expressed $\mathbb{P}[A \mid B]$ in terms of $\mathbb{P}[A], \mathbb{P}[B \mid A]$ and $\mathbb{P}[B \mid \overline{A}]$:

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[B \mid A]\mathbb{P}[A]}{\mathbb{P}[B \mid A]\mathbb{P}[A] + \mathbb{P}[B \mid \overline{A}](1 - \mathbb{P}[A])}. \tag{3}$$

By plugging in the values written above, we obtain $\mathbb{P}[A \mid B] = \frac{9}{47} \approx 0.19$.

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

Of course, (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.

3 Bayes' Rule and Total Probability Rule

Equations (1) and (2) are very useful in their own right. The former is called **Bayes' Rule** and the latter is called the **Total Probability Rule**. Bayes' Rule is useful when one wants to calculate $\mathbb{P}[A \mid B]$ but one is given $\mathbb{P}[B \mid A]$ 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 $\mathbb{P}[B \mid A]$. If A does not happen, the probability that B happens is $\mathbb{P}[B \mid \overline{A}]$. If we know or can easily calculate these two probabilities and also $\mathbb{P}[A]$, then the Total Probability Rule yields the probability of event B.

3.1 Examples

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 0.7. If person Y is chosen, you will win with probability 0.3. Your opponent is chosen by flipping a biased coin such that the probability of choosing X is 0.6.

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

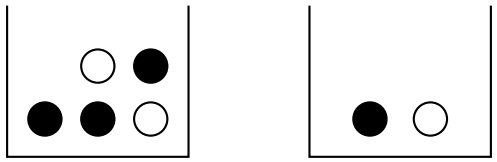
- $\mathbb{P}[A \mid B_X] = 0.7$, (if person X is chosen, you win with probability 0.7)
- $\mathbb{P}[A \mid B_Y] = 0.3$, (if person Y is chosen, you win with probability 0.3)
- $\mathbb{P}[B_X] = 0.6$, (person X is chosen with probability 0.6)

- $\mathbb{P}[B_Y] = 0.4$, (person Y is chosen with probability 0.4)

By using the Total Probability Rule, we have $\mathbb{P}[A] = \mathbb{P}[A \mid B_X]\mathbb{P}[B_X] + \mathbb{P}[A \mid B_Y]\mathbb{P}[B_Y]$, and plugging in the known values gives $\mathbb{P}[A] = (0.7 \times 0.6) + (0.3 \times 0.4) = 0.54$.

Balls and Bins

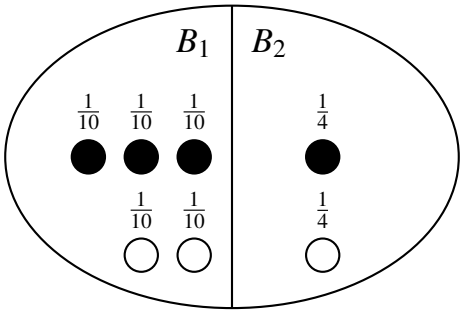
Imagine we have the following two bins each containing some number of black and white balls:



Bin 1 Bin 2

Suppose one of the two bins is chosen with equal probability and a ball is drawn from the chosen bin uniformly at random. What is the probability that we picked Bin 1 given that a white ball was drawn, i.e., $\mathbb{P}[\text{Bin 1} \mid \bigcirc]$?

Is the answer $\frac{2}{3}$, since we know that there are a total of three white balls, two of which are in Bin 1? This reasoning is *incorrect*. Instead, what we should do is appropriately scale each sample point as the following picture shows:



This diagram shows that the sample space Ω consists of the outcomes in event B_1 (corresponding to Bin 1) and event B_2 (corresponding to Bin 2), i.e., $\Omega = B_1 \cup B_2$. We can use the definition of conditional probability to see that

$$\mathbb{P}[\text{Bin 1} \mid \bigcirc] = \frac{\frac{1}{10} + \frac{1}{10}}{\frac{1}{10} + \frac{1}{10} + \frac{1}{4}} = \frac{\frac{2}{10}}{\frac{9}{20}} = \frac{4}{9}.$$

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

$$\mathbb{P}[\bigcirc] = \mathbb{P}[\bigcirc \mid \text{Bin 1}] \times \mathbb{P}[\text{Bin 1}] + \mathbb{P}[\bigcirc \mid \text{Bin 2}] \times \mathbb{P}[\text{Bin 2}] = \frac{2}{5} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{9}{20}.$$

Observe that we can apply the Total Probability Rule here because $\mathbb{P}[\text{Bin 1}]$ is the complement of $\mathbb{P}[\text{Bin 2}]$. Finally, upon plugging the above values into Bayes' Rule, we obtain the probability that we picked Bin 1 given that we picked a white ball:

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

All we have done above is to combine Bayes' Rule and the Total Probability Rule; this is also how we obtained (3). Equivalently, we could have plugged in the appropriate values to (3).

3.2 Generalization

We now consider Bayes' Rule and the Total Probability Rule in a more general context. First, we define a *partition* of an event as follows.

Definition 14.2 (Partition of an event)**.** *We say that an event A is partitioned into n events $A_1, \ldots, A_n$ if*

1. *$A = A_1 \cup A_2 \cup \cdots \cup A_n$, and*
2. *$A_i \cap A_j = \varnothing$ for all $i \neq j$ (i.e., $A_1, \ldots, A_n$ are mutually exclusive).*

In other words, each outcome in A belongs to exactly one of the events $A_1, \ldots, A_n$.

Now, let $A_1, \ldots, A_n$ be a partition of the sample space Ω. Then, the **Total Probability Rule** for any event B is

$$\mathbb{P}[B] = \sum_{i=1}^{n} \mathbb{P}[B \cap A_i] = \sum_{i=1}^{n} \mathbb{P}[B \mid A_i]\mathbb{P}[A_i], \tag{4}$$

while **Bayes' Rule**, assuming $\mathbb{P}[B] \neq 0$, is given by

$$\mathbb{P}[A_i \mid B] = \frac{\mathbb{P}[B \mid A_i]\mathbb{P}[A_i]}{\mathbb{P}[B]} = \frac{\mathbb{P}[B \mid A_i]\mathbb{P}[A_i]}{\sum_{j=1}^{n} \mathbb{P}[B \mid A_j]\mathbb{P}[A_j]}, \tag{5}$$

where the second equality follows from the Total Probability Rule.

4 Combinations of Events

In most applications of probability in Computer Science, we are interested in things like $\mathbb{P}[\bigcup_{i=1}^{n} A_i]$ and $\mathbb{P}[\bigcap_{i=1}^{n} A_i]$, where the A_i are simple events (i.e., we know or can easily compute $\mathbb{P}[A_i]$). 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.

4.1 Independent Events

Definition 14.3 (Independence)**.** *Two events A, B in the same probability space are said to be independent if $\mathbb{P}[A \cap B] = \mathbb{P}[A] \times \mathbb{P}[B]$.*

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

$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[A \cap B]}{\mathbb{P}[B]} = \frac{\mathbb{P}[A] \times \mathbb{P}[B]}{\mathbb{P}[B]} = \mathbb{P}[A].$$

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 $\mathbb{P}[B \mid A] = \mathbb{P}[B]$ provided $\mathbb{P}[A] > 0$.) For events A, B such that $\mathbb{P}[B] > 0$, the condition $\mathbb{P}[A \mid B] = \mathbb{P}[A]$ 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 of 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 14.4 (Mutual independence)**.** *Events $A_1, \ldots, A_n$ are said to be mutually independent if for **every** subset $I \subseteq \{1, \ldots, n\}$ with size $|I| \geq 2$,*

$$\mathbb{P}[\textstyle\bigcap_{i \in I} A_i] = \prod_{i \in I} \mathbb{P}[A_i]. \tag{6}$$

An equivalent definition of mutual independence is as follows.

Definition 14.5 (Mutual independence)**.** *Events $A_1, \ldots, A_n$ are said to be mutually independent if for all $B_i \in \{A_i, \overline{A}_i\}, i = 1, \ldots, n$,*

$$\mathbb{P}[B_1 \cap \cdots \cap B_n] = \prod_{i=1}^{n} \mathbb{P}[B_i]. \tag{7}$$

Remarks.

1. In Definition 14.4, (6) needs to hold for *every* subset I of $\{1, \ldots, n\}$ with size $|I| \geq 2$. The cases of $|I| = 0$ and $|I| = 1$ are omitted, as they impose no constraints.
2. Note that (6) imposes $2^n - n - 1$ constraints on the probability distribution, while (7) defines 2^n constraints. It turns out that exactly $n + 1$ constraints implied by (7) are actually redundant.

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

$$\mathbb{P}[A_i \mid \textstyle\bigcap_{j \in I} A_j] = \mathbb{P}[A_i].$$

Note that the independence of every pair of events (so-called *pairwise independence*) does *not* necessarily imply mutual independence. As illustrated in the following 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.

Example: Pairwise Independent but Not Mutually Independent Events

Suppose you toss a fair coin twice and let A be the event that the first flip is H and B be the event that the second flip is H. 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, the chance of the second flip being the same as the first is still $\frac{1}{2}$. Another way of saying this is that

$\mathbb{P}[A \cap C] = \frac{1}{4} = \mathbb{P}[A]\mathbb{P}[C]$ since $A \cap C$ is the event that the first flip is H and the second is also H. 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.

4.2 Intersections of Events

Computing the probability of an intersection of mutually independent events is easy; it follows from the definition. We simply multiply the probabilities of each event. How do we compute the probability of an intersection for events that are not mutually 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*

$$\mathbb{P}[A \cap B] = \mathbb{P}[A]\mathbb{P}[B \mid A].$$

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

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

Proof. The first assertion follows directly from the definition of $\mathbb{P}[B \mid A]$ (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 $\mathbb{P}[A] = \mathbb{P}[A]$, which is trivially true. For an arbitrary $n > 1$, assume (the inductive hypothesis) that

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

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

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

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.

Example: 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

$$\begin{aligned}
\mathbb{P}[A] &= \mathbb{P}[A_1] \times \mathbb{P}[A_2 \mid A_1] \times \mathbb{P}[A_3 \mid A_1 \cap A_2] \\
&= \mathbb{P}[A_1] \times \mathbb{P}[A_2] \times \mathbb{P}[A_3] \\
&= \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{8}.
\end{aligned}$$

The second line here follows from the fact that the tosses are mutually independent. Of course, we already know that $\mathbb{P}[A] = \frac{1}{8}$ from our definition of the probability space in the previous 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,

$$\mathbb{P}[A] = \mathbb{P}[A_1] \times \mathbb{P}[A_2] \times \mathbb{P}[A_3] = p^3.$$

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

Example: Monty Hall Revisited

Recall the Monty Hall problem from the previous note: 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 C_i be the event that the contestant chooses door i, let P_i be the event that the prize is behind door i, and let H_i be the event that the host chooses door i. Then, by the Product Rule,

$$\mathbb{P}[C_1 \cap P_2 \cap H_3] = \mathbb{P}[C_1] \times \mathbb{P}[P_2 \mid C_1] \times \mathbb{P}[H_3 \mid C_1 \cap P_2].$$

The probability of C_1 is $\frac{1}{3}$, since the contestant is choosing the door at random. The probability of P_2 given C_1 is still $\frac{1}{3}$ since they are independent. The probability of the host choosing door 3 given events C_1 and P_2 is 1; the host cannot choose door 1 since the contestant has already chosen it, and the host cannot choose door 2 since the host must reveal a goat (and not the prize). Therefore,

$$\mathbb{P}[C_1 \cap P_2 \cap H_3] = \frac{1}{3} \times \frac{1}{3} \times 1 = \frac{1}{9}.$$

Observe that we needed 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 H_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 3? We can use the same technique as above, but our final answer will be different. This is left as an exercise (see Exercise 4).

Now, noting that $\mathbb{P}[H_3 \mid C_1] = \frac{1}{2}$ (see Exercise 5) and $\mathbb{P}[C_1 \cap H_3] = \mathbb{P}[C_1]\mathbb{P}[H_3 \mid C_1] = \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$, we obtain

$$\mathbb{P}[P_2 \mid C_1 \cap H_3] = \frac{\mathbb{P}[C_1 \cap P_2 \cap H_3]}{\mathbb{P}[C_1 \cap H_3]} = \frac{\frac{1}{9}}{\frac{1}{6}} = \frac{2}{3},$$

which is an alternative derivation of the same conclusion described in the previous note.

Example: Poker Hands

Let's use the Product Rule to compute the probability of a flush in a different way. This is equal to $4 \times \mathbb{P}[A]$, 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

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

Clearly $\mathbb{P}[A_1] = \frac{13}{52} = \frac{1}{4}$. What about $\mathbb{P}[A_2 \mid A_1]$? 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 $\mathbb{P}[A_2 \mid A_1] = \frac{12}{51}$. Similarly, $\mathbb{P}[A_3 \mid A_1 \cap A_2] = \frac{11}{50}$, and so on. So we get

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

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

4.3 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 $\mathbb{P}[A_i] = \frac{1}{6}$ for each i. Therefore, they claim

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

Is this calculation correct? Well, suppose instead that the casino rolled six dice, and again you win if and only if 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 $\mathbb{P}[A_i]$, we are counting some sample points more than once.

Fortunately, in Note 10 we learned about the Principle of Inclusion-Exclusion which allows us to deal with this kind of situation. In the proof of Theorem 10.3, it was shown that every $\omega \notin A_1 \cup \cdots \cup A_n$ is not counted by the Inclusion-Exclusion formula, while every $\omega \in A_1 \cup \cdots \cup A_n$ is counted exactly once by the formula. Hence, rather than summing (with appropriate signs) the cardinalities of $\bigcap_{i \in S} A_i$ in the Inclusion-Exclusion formula, if we instead sum their probabilities $\mathbb{P}[\bigcap_{i \in S} A_i]$, we obtain the following useful formula for computing $\mathbb{P}[A_1 \cup \cdots \cup A_n]$:

Theorem 14.2 (Inclusion-Exclusion)**.** *Let $A_1, \ldots, A_n$ be events in some probability space, where $n \geq 2$. Then, we have*

$$\mathbb{P}[A_1 \cup \cdots \cup A_n] = \sum_{k=1}^{n} (-1)^{k-1} \sum_{S \subseteq \{1,\ldots,n\} : |S| = k} \mathbb{P}[\bigcap_{i \in S} A_i]. \tag{8}$$

The right hand side of (8) can be written as

$$\mathbb{P}[\bigcup_{i=1}^{n} A_i] = \sum_{i=1}^{n} \mathbb{P}[A_i] - \sum_{i<j} \mathbb{P}[A_i \cap A_j] + \sum_{i<j<k} \mathbb{P}[A_i \cap A_j \cap A_k] - \cdots + (-1)^{n-1} \mathbb{P}[A_1 \cap A_2 \cap \cdots \cap A_n],$$

where $\sum_{i<j}$ denotes summing over all $i, j \in \{1, \ldots, n\}$ such that $i < j$, and so on. That is, to compute $\mathbb{P}[\bigcup_i A_i]$, we start by summing the event probabilities $\mathbb{P}[A_i]$, then we *subtract* the probabilities of all pairwise intersections, then we *add* back in the probabilities of all three-way intersections, and so on. You might like to verify it for the special case $n = 3$ by drawing a Venn diagram. You might also like to prove the formula for general n by induction (in similar fashion to the proof of the Product Rule above).

Using (8), the probability we get lucky in the new game in Las Vegas is given by

$$\mathbb{P}[A_1 \cup A_2 \cup A_3] = \mathbb{P}[A_1] + \mathbb{P}[A_2] + \mathbb{P}[A_3] - \mathbb{P}[A_1 \cap A_2] - \mathbb{P}[A_1 \cap A_3] - \mathbb{P}[A_2 \cap A_3] + \mathbb{P}[A_1 \cap A_2 \cap A_3].$$

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 $\mathbb{P}[A_i \cap A_j] = \mathbb{P}[A_i]\mathbb{P}[A_j] = (\frac{1}{6})^2 = \frac{1}{36}$, and similarly $\mathbb{P}[A_1 \cap A_2 \cap A_3] = (\frac{1}{6})^3 = \frac{1}{216}$. So, we get

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

So your odds are quite a bit worse than what 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 summing the first k terms gives us an overestimate when k is odd and an underestimate when k is even, and both these estimates get better as k increases.

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

1. (**Mutually exclusive events**) If the events $A_1, \ldots, A_n$ are mutually exclusive (i.e., $A_i \cap A_j = \varnothing$ for all $i \neq j$), then
$$\mathbb{P}[\bigcup_{i=1}^n A_i] = \sum_{i=1}^n \mathbb{P}[A_i].$$
[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**) Let $A_1, \ldots, A_n$ be events in some probability space. Then, for all $n \in \mathbb{Z}^+$,
$$\mathbb{P}[\bigcup_{i=1}^n A_i] \leq \sum_{i=1}^n \mathbb{P}[A_i]. \tag{9}$$
This merely says that adding up the $\mathbb{P}[A_i]$ can only *over*estimate the probability of the union. Crude as it may seem, we will later see how to use the union bound effectively in Computer Science examples.

5 Exercises

1. Suppose five cards are dealt from a standard deck of playing cards. Let A_i be the event that the ith card is an ace. Show that $\mathbb{P}[A_i] = \frac{1}{13}$ for all $i = 1, \ldots, 5$.

2. Prove equations (4) and (5).

3. Show that Definitions 14.4 and 14.5 of mutual independence are equivalent.

4. In the Monty Hall problem, find $\mathbb{P}[C_1 \cap P_1 \cap H_3]$.
5. In the Monty Hall problem, show that $\mathbb{P}[H_3 \mid C_1] = \frac{1}{2}$.
6. Prove the union bound (9).

Random Variables: Distribution and Expectation

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 $X = 3$; whereas if we observe the outcome $\omega = HTHT$, then $X = 2$. 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 (# heads)	probability of 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* (abbreviated *r.v.*). As we see from the example above, a random variable X typically 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 summaries (also called *statistics*) of an experiment.

1 Random Variables

Before we formalize the above notions, let us consider another example to enforce our conceptual understanding of a random variable.

Example: Fixed Points of Permutations

Question: 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 ith student. We call i a *fixed point* of π if $\pi_i = i$, i.e., if student i receives their own homework.

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 us call the number of fixed points X_n, which is a random variable taking values in the set $\{0, 1, 2, \ldots, n\}$. (Actually the value $X_n = n - 1$ is not possible: why?)

Formal Definition of a Random Variable

We now formalize the concepts discussed above.

Definition 15.1 (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 our 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.

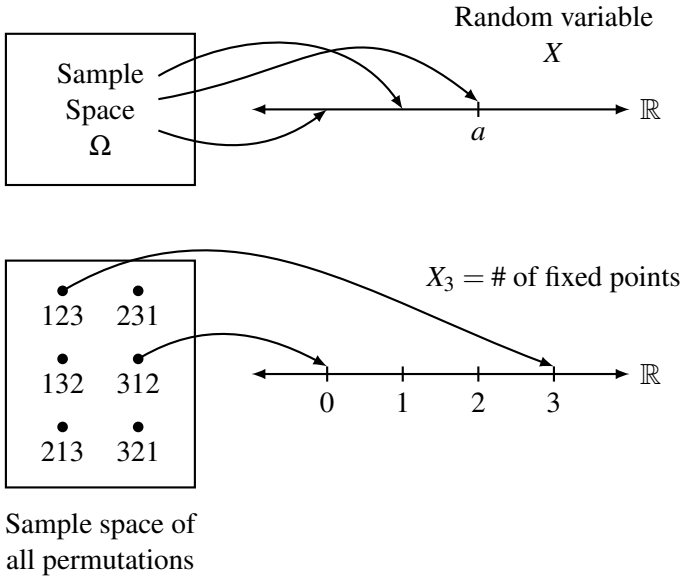


Figure 1: Visualization of how a random variable is defined on the sample space.

Exercise: By completing the lower picture in Figure 1, show that the only possible values for the r.v. X_3 are 0, 1 and 3, and that their probabilities are $\frac{1}{3}$, $\frac{1}{2}$ and $\frac{1}{6}$, respectively.

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

2 Probability Distribution

When we introduced the basic probability space in an earlier note, 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.

Analogously, there are two important things 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 (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, $\mathbb{P}[X = a]$. The collection of these probabilities, for all possible values of a, is known as the *distribution* of the random variable X.

Definition 15.2 (Distribution)**.** *The distribution of a discrete random variable X is the collection of values $\{(a, \mathbb{P}[X = a]) : a \in \mathscr{A}\}$, 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:

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

If needed, we may also write $\mathbb{P}[X = a] = 0$ 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 assume. The height of the bar at a value a is the probability $\mathbb{P}[X = a]$. Each of these probabilities can be computed by looking at the probability of the corresponding event in the sample space.

Note that the collection of events $X = a$, for $a \in \mathscr{A}$, 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. So, when we sum up the probabilities of the events $X = a$, we are really summing up the probabilities of all the sample points, giving us a total of exactly 1.

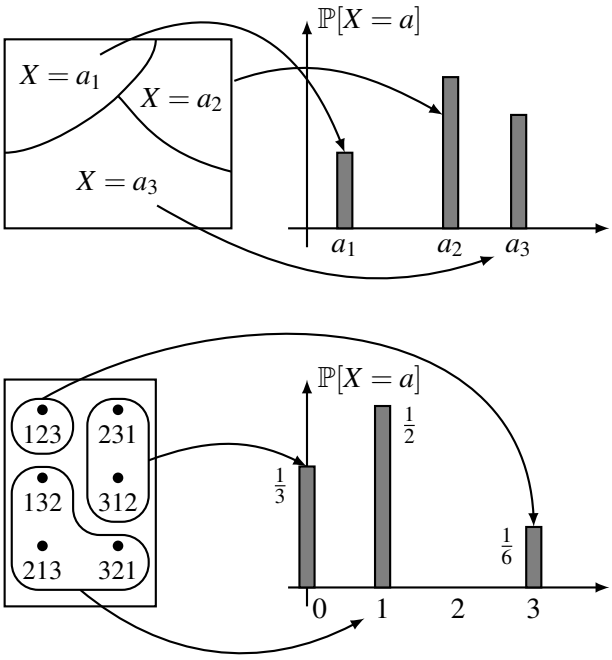


Figure 2: Visualization of how the distribution of a random variable is defined.

Bernoulli Distribution

A simple yet very useful probability distribution is the *Bernoulli* distribution of a random variable which takes value in $\{0,1\}$:

$$\mathbb{P}[X = i] = \begin{cases} p, & \text{if } i = 1, \\ 1-p, & \text{if } i = 0, \end{cases}$$

where $0 \leq p \leq 1$. We say that X is distributed as a *Bernoulli* random variable with parameter p, and write

$$X \sim \text{Bernoulli}(p) \qquad \text{or} \qquad x \sim \text{Ber}(p).$$

Binomial Distribution

Let us 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 that shows H with probability p. Each sample point ω is a sequence of tosses, and $X(\omega)$ 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 that X can take. 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 probability $p^i(1-p)^{n-i}$, since the coin flips are independent. There are exactly $\binom{n}{i}$ of these sample points. Hence,

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

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, and we write

$$X \sim \text{Bin}(n, p),$$

where n denotes the number of trials and p the probability of success (observing an H in the example). 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 $\sum_{i=0}^{n} \mathbb{P}[X = i] = 1$, 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 from an earlier note where we saw it combinatorially, for $a = p$ and $b = 1 - p$.

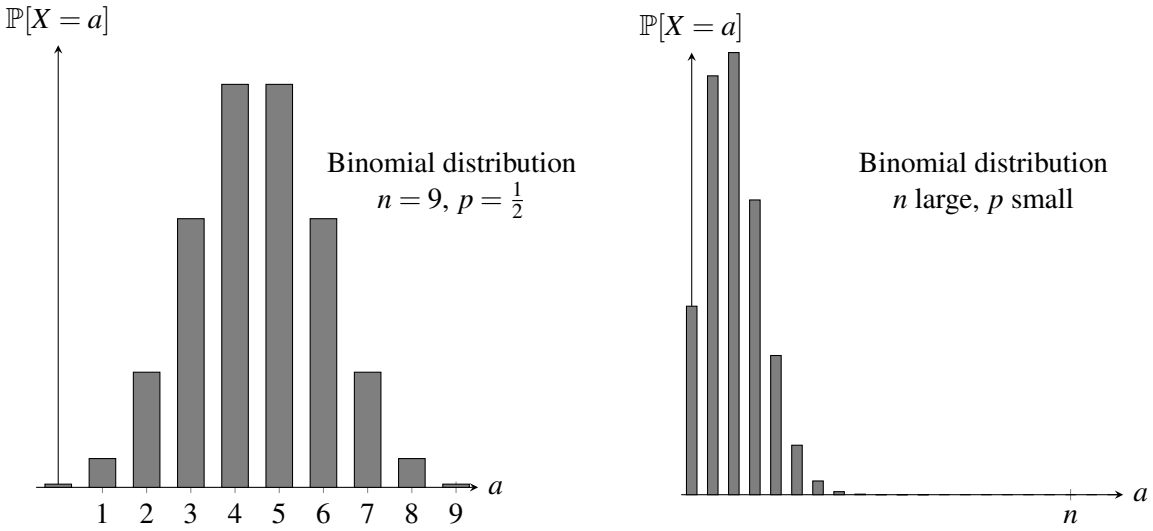


Figure 3: The binomial distributions for two choices of (n, p).

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 earlier. 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: $X \sim \text{Bin}(n+k, 1-p)$ (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:

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

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

Hypergeometric Distribution

Consider an urn containing $N = B + W$ balls, where B balls are black and W are white. Suppose you randomly sample $n \leq N$ balls from the urn *with* replacement, and let X denote the number of black balls in your sample. What is the probability distribution of X? Since the probability of seeing a black ball is

B/N for each draw, independently of all other draws, X follows the binomial distribution $\text{Bin}(n,p)$, where $p = B/N$.

What if you randomly sample $n \leq N$ balls from the urn *without* replacement? In this case, the probability of seeing a black ball in the i-th draw depends on the colors of the $i-1$ balls already drawn; that is, unlike in the case of sampling with replacement, the samples are not independent. The probability distribution of the number Y of black balls in this setting can be found as follows.

Since all ways of drawing n balls are equally likely, we are in a uniform probability space Ω, so we can compute probabilities using counting. Specifically, for any $k = 0, 1, \ldots, n$, we have

$$\mathbb{P}[Y = k] = \frac{|E_k|}{|\Omega|}, \tag{2}$$

where $E_k \subseteq \Omega$ is the set of outcomes that contain exactly k black balls. The total number of possible outcomes is $|\Omega| = \binom{N}{n}$. The total number of outcomes containing exactly k black balls is

$$|E_k| = \binom{B}{k}\binom{N-B}{n-k};$$

to see this, note that there are $\binom{B}{k}$ ways to choose the k black balls out of the B in the urn, and $\binom{N-B}{n-k}$ ways to choose the remaining $n-k$ balls out of the $N-B$ white balls in the urn. Plugging these calculations into (2), we conclude that

$$\mathbb{P}[Y = k] = \frac{\binom{B}{k}\binom{N-B}{n-k}}{\binom{N}{n}},$$

for $k \in \{0, 1, \ldots, n\}$. (Note that $\binom{m}{j} = 0$ if $j > m$, so $\mathbb{P}[Y = k] \neq 0$ only if $\max(0, n+B-N) \leq k \leq \min(n, B)$.)

This probability distribution is called the *hypergeometric distribution* with parameters N, B, n, and we write

$$Y \sim \text{Hypergeometric}(N, B, n).$$

3 Multiple Random Variables and Independence

Often one is interested in multiple random variables on the same sample space. Consider, for example, the sample space of flipping three coins. One could define many random variables: for example a random variable X indicating the number of heads, or a random variable Y indicating the number of tails, or a binary random variable Z indicating whether the first toss is H or not. Note that for each sample point, any random variable has a specific value: e.g., for $\omega = HTT$, we have $X(\omega) = 1$, $Y(\omega) = 2$, and $Z(\omega) = 1$.

The concept of a distribution can then be extended to probabilities for the combination of values for multiple random variables.

Definition 15.3. *The joint distribution of two discrete random variables X and Y is the collection of values $\{((a,b), \mathbb{P}[X = a, Y = b]) : a \in \mathscr{A},\ b \in \mathscr{B}\}$, where $\mathscr{A}$ is the set of all possible values taken by X and $\mathscr{B}$ is the set of all possible values taken by Y.*

Given a joint distribution of X and Y, the distribution $\mathbb{P}[X = a]$ of X is called the *marginal distribution* of X, and can be found by summing over the values of Y. That is,

$$\mathbb{P}[X = a] = \sum_{b \in \mathscr{B}} \mathbb{P}[X = a, Y = b].$$

The marginal distribution of Y is defined analogously.

A joint distribution over any set of random variables $X_1, \ldots, X_n$ (for example, X_i could be the value of the i-th roll of a sequence of n die rolls) is $\mathbb{P}[X_1 = a_1, \ldots, X_n = a_n]$, where $a_i \in \mathscr{A}_i$ and $\mathscr{A}_i$ is the set of possible values for X_i. The marginal distribution of X_i can be obtained by summing over all the possible values of the other variables.

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

Definition 15.4 (Independence)**.** *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*

$$\mathbb{P}[X = a, Y = b] = \mathbb{P}[X = a]\mathbb{P}[Y = b], \quad \forall a, b.$$

Mutual independence of more than two r.v.'s is defined similarly.

A very important example of independent random variables are indicator random variables for independent events. If I_i denotes the indicator r.v. for the i-th toss of a coin being H, then $I_1, \ldots, I_n$ are mutually independent random variables. This example motivates the commonly used phrase "*independent and identically distributed (i.i.d.)* set of random variables." In this example, $\{I_1, \ldots, I_n\}$ is a set of i.i.d. indicator random variables.

4 Expectation

The distribution of a r.v. contains *all* the 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 would 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 *summarize* 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 15.5 (Expectation)**.** *The expectation of a discrete random variable X is defined as*

$$\mathbb{E}[X] = \sum_{a \in \mathscr{A}} a \times \mathbb{P}[X = a], \tag{3}$$

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

Technical Note. Expectation is well defined provided that the sum on the right hand side of (3) is absolutely convergent, i.e., $\sum_{a \in \mathscr{A}} |a| \times \mathbb{P}[X = a] < \infty$. There are discrete random variables for which expectations do not exist, such as the r.v. X with distribution $\mathbb{P}[X = i] = \frac{6}{\pi^2 i^2}$ for all positive integers i. (The reason for the factor $\frac{6}{\pi^2}$ here is to make the probabilities sum to 1, since $\sum_{i=1}^{\infty} \frac{1}{i^2} = \frac{\pi^2}{6}$.)

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

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

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 $\mathbb{E}[X]$ may not actually be a value that X can take). 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.

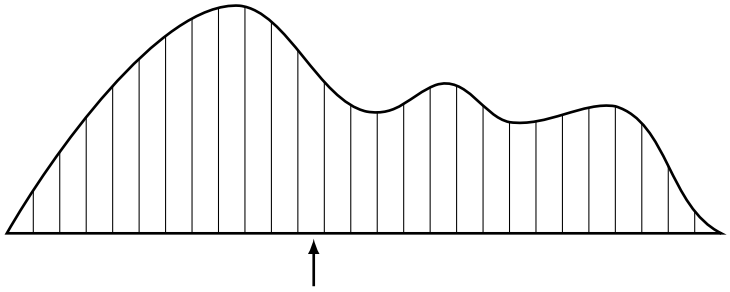


Figure 4: Physical interpretation of expected value as the balance point.

4.1 Examples

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

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

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

2. **Two dice.** Throw two fair dice and 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
$\mathbb{P}[X=a]$	$\frac{1}{36}$	$\frac{1}{18}$	$\frac{1}{12}$	$\frac{1}{9}$	$\frac{5}{36}$	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{1}{9}$	$\frac{1}{12}$	$\frac{1}{18}$	$\frac{1}{36}$

   The expectation is therefore

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

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 $\mathbb{P}[X=1] = \frac{18}{38}$, $\mathbb{P}[X=-1] = \frac{20}{38}$. Thus,

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

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

4.2 Linearity of Expectation

So far, we have 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 15.1. *For any two random variables X and Y on the same probability space, we have*

$$\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y].$$

Also, for any constant c, we have

$$\mathbb{E}[cX] = c\,\mathbb{E}[X].$$

Proof. We first rewrite the definition of expectation in a more convenient form. Recall from Definition 15.5 that

$$\mathbb{E}[X] = \sum_{a \in \mathscr{A}} a \times \mathbb{P}[X = a].$$

Consider a particular term $a \times \mathbb{P}[X = a]$ in the above sum. Notice that $\mathbb{P}[X = a]$, by definition, is the sum of $\mathbb{P}[\omega]$ over those sample points ω for which $X(\omega) = a$. Furthermore, 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

$$\mathbb{E}[X] = \sum_{\omega \in \Omega} X(\omega) \times \mathbb{P}[\omega]. \tag{4}$$

This equivalent definition of expectation will make the present proof much easier (though it is usually less convenient for actual calculations). Applying (4) to $\mathbb{E}[X+Y]$ gives:

$$\begin{aligned}
\mathbb{E}[X+Y] &= \sum_{\omega \in \Omega} (X+Y)(\omega) \times \mathbb{P}[\omega] \\
&= \sum_{\omega \in \Omega} (X(\omega) + Y(\omega)) \times \mathbb{P}[\omega] \\
&= \sum_{\omega \in \Omega} (X(\omega) \times \mathbb{P}[\omega]) + \sum_{\omega \in \Omega} (Y(\omega) \times \mathbb{P}[\omega]) \\
&= \mathbb{E}[X] + \mathbb{E}[Y]
\end{aligned}$$

In the last step, we used (4) 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 15.1 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 15.1 to conclude things like $\mathbb{E}[3X - 5Y] = 3\,\mathbb{E}[X] - 5\,\mathbb{E}[Y]$, regardless of whether or not X and Y are independent. This important property is known as linearity of expectation.

Important caveat: Theorem 15.1 does ***not*** *say that $\mathbb{E}[XY] = \mathbb{E}[X]\,\mathbb{E}[Y]$, or that $\mathbb{E}\left[\frac{1}{X}\right] = \frac{1}{\mathbb{E}[X]}$, etc. These claims are not true in general. It is only* sums *and* differences *and* constant multiples *of random variables that behave so nicely.*

4.3 Applications of Linearity of Expectation

Now let us see some examples of Theorem 15.1 in action.

1. **Two dice again.** Here is a much less painful way of computing $\mathbb{E}[X]$, where X is the sum of the scores of the two dice. Note that $X = Y_1 + Y_2$, where Y_i is the score on die i. We know from example 1 in Section 4.1 that $\mathbb{E}[Y_1] = \mathbb{E}[Y_2] = \frac{7}{2}$. So, by Theorem 15.1, we have $\mathbb{E}[X] = \mathbb{E}[Y_1] + \mathbb{E}[Y_2] = 7$.

2. **More roulette.** Suppose we play the roulette game mentioned in Section 4.1 $n \geq 1$ times. Let X_n be our expected net winnings. Then $X_n = Y_1 + Y_2 + \cdots + Y_n$, where Y_i is our net winnings in the ith play. We know from earlier that $\mathbb{E}[Y_i] = -\frac{1}{19}$ for each i. Therefore, by Theorem 15.1, $\mathbb{E}[X_n] = \mathbb{E}[Y_1] + \mathbb{E}[Y_2] + \cdots + \mathbb{E}[Y_n] = -\frac{n}{19}$. For $n = 1000$, $\mathbb{E}[X_n] = -\frac{1000}{19} \approx -53$, so if you play 1000 games, you expect to lose about \$53.

3. **Fixed points of permutations.** Let us return to the homework permutation example with an arbitrary number n of students. Let X_n denote the number of students who receive their own homework after shuffling (or equivalently, the number of fixed points). To take advantage of Theorem 15.1, we need to write X_n as a *sum* of simpler r.v.'s. Since X_n *counts* the number of times something happens, we can write it as a sum using the following useful trick:

$$X_n = I_1 + I_2 + \cdots + I_n, \qquad \text{where } I_i = \begin{cases} 1, & \text{if student } i \text{ gets their own homework,} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

   [You should think about this equation for a moment. Remember that all the I_i's are random variables. What does an equation involving random variables mean? What we mean is that, *at every sample point* ω, we have $X_n(\omega) = I_1(\omega) + I_2(\omega) + \cdots + I_n(\omega)$. Why is this true?]

   A Bernoulli random variable such as I_i is called an indicator random variable of the corresponding event (in this case, the event that student i gets their own homework). For indicator r.v.'s, the expectation is particularly easy to calculate. Specifically,

$$\mathbb{E}[I_i] = (0 \times \mathbb{P}[I_i = 0]) + (1 \times \mathbb{P}[I_i = 1]) = \mathbb{P}[I_i = 1].$$

   In our case, we have

$$\mathbb{P}[I_i = 1] = \mathbb{P}[\text{student } i \text{ gets their own homework}] = \frac{1}{n}.$$

   We can now apply Theorem 15.1 to (5), yielding

$$\mathbb{E}[X_n] = \mathbb{E}[I_1] + \mathbb{E}[I_2] + \cdots + \mathbb{E}[I_n] = n \times \frac{1}{n} = 1.$$

   So, we see that the expected number of students who get their own homeworks in a class of size n is 1. That is, the expected number of fixed points in a random permutation of n items is always 1, regardless of n!

4. **Coin tosses.** Toss a fair coin $n \geq 1$ times. Let the r.v. X_n be the number of heads observed. As in the previous example, to take advantage of Theorem 15.1 we write

$$X_n = I_1 + I_2 + \cdots + I_n,$$

where I_i is the indicator r.v. of the event that the ith toss is H. Since the coin is fair, we have

$$\mathbb{E}[I_i] = \mathbb{P}[I_i = 1] = \mathbb{P}[i\text{th toss is } H] = \frac{1}{2}.$$

Using Theorem 15.1, we therefore get

$$\mathbb{E}[X_n] = \sum_{i=1}^{n} \frac{1}{2} = \frac{n}{2}.$$

More generally, in n tosses of a biased coin that comes up H with probability p, $\mathbb{E}[X_n] = np$. (Check this!) So the expectation of a binomial r.v. $X \sim \text{Bin}(n, p)$ is equal to np. Note that it would have been much messier (though possible) to reach the same conclusion by computing this directly from the definition of expectation in (3) and the distribution of a binomial r.v. in (1).

5. **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 m tosses of a biased coin with $\mathbb{P}[H] = \frac{1}{n}$ (why?). So, from the previous example, we get $\mathbb{E}[X] = \frac{m}{n}$. 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 would get into a messy calculation involving binomial coefficients.

   Here is another example on the same sample space. Let the r.v. Y_n be the number of empty bins. The distribution of Y_n is horrible to contemplate: to get a feel for this, you might like to write it down for $m = n = 3$ (i.e., 3 balls, 3 bins). However, computing the expectation $\mathbb{E}[Y_n]$ is easy using Theorem 15.1. As in the previous two examples, we write

$$Y_n = I_1 + I_2 + \cdots + I_n, \tag{6}$$

   where I_i is the indicator r.v. of the event "bin i is empty". The expectation of I_i is easy to find:

$$\mathbb{E}[I_i] = \mathbb{P}[I_i = 1] = \mathbb{P}[\text{bin } i \text{ is empty}] = \left(1 - \frac{1}{n}\right)^m,$$

   as discussed earlier. Applying Theorem 15.1 to (6), we therefore obtain

$$\mathbb{E}[Y_n] = \sum_{i=1}^{n} \mathbb{E}[I_i] = n\left(1 - \frac{1}{n}\right)^m,$$

   a simple formula, quite easily derived. Let us see how it behaves in the special case $m = n$ (same number of balls as bins). In this case we get $\mathbb{E}[Y_n] = n\left(1 - \frac{1}{n}\right)^n$. Now the quantity $\left(1 - \frac{1}{n}\right)^n$ can be approximated (for large enough values of n) by the number $\frac{1}{e}$.[2] So we see that, for large n,

$$\mathbb{E}[Y_n] \approx \frac{n}{e} \approx 0.368n.$$

   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,

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

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 $(1 - \frac{1}{n})^n \approx 0.358$, which is very close to $\frac{1}{e} \approx 0.368$. The approximation gets better and better for larger n.

Functions of Random Variables

In the previous note, we considered simple linear functions (sums, differences and constant multiples) of random variables. In this note, we will consider more general functions of a random variable.

Formally, let X be a random variable on a sample space Ω with probability distribution $\mathbb{P}_X$. Then for a function $f(\cdot)$ on the range of X, $f(X)$ is the random variable Y on the same sample space Ω, where $Y(\omega) = f(X(\omega))$ for $\omega \in \Omega$.

In terms of sample spaces, the event "$Y = y$" is equivalent to the event "$X \in f^{-1}(y)$". Here, we take $f^{-1}(y)$ to be the set $\{x \mid f(x) = y\}$. (When $f(\cdot)$ is one-to-one, it is a bit simpler, i.e., "$X = x$" is the same event as "$Y = f(x)$".)

With this view, the distribution of $Y = f(X)$, $\mathbb{P}_Y$, can be derived from the distribution of X as follows:

$$\mathbb{P}_Y[Y = y] = \sum_{x:f(x)=y} \mathbb{P}_X[X = x]. \tag{1}$$

From this, we observe that

$$\mathbb{E}[Y] = \sum_y y\, \mathbb{P}_Y[Y = y] = \sum_y \sum_{x:f(x)=y} y\, \mathbb{P}_X[X = x] = \sum_x f(x)\mathbb{P}_X[X = x].$$

The first equality is by the definition of expectation, the second is equation (1), and the last comes by observing that each value of x is included exactly once in the inner summation. Thus, we have the following identity:

$$\mathbb{E}[f(X)] = \sum_x f(x)\mathbb{P}_X[X = x].$$

In words: to compute the expectation of $f(X)$, we take the same weighted average as in the usual expectation of X, but replace the value x of X by $f(x)$. (This identity is sometimes called the Law of the Unconscious Statistician (LOTUS) as it is useful and used without much thought at times.)

An example that we will use extensively is the following. For any r.v. X, we have $\mathbb{E}\left[X^2\right] = \sum_x x^2 \mathbb{P}_X[X = x]$. **Warning:** Note that $\mathbb{E}\left[X^2\right]$ is **not** the same as $(\mathbb{E}[X])^2$; and, more generally, $\mathbb{E}[f(X)]$ is **not** the same as $f(\mathbb{E}[X])$.

Random Variables: Variance and Covariance

We have seen in the previous note that if we take a biased coin that comes up heads with probability p and toss it n times, 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?

1 Random Walk

Let us consider a simpler setting that is equivalent to tossing a fair coin n times, but is easier to picture. Suppose we have a particle that starts at position 0 and performs a random walk in one dimension. At each time step, the particle moves either one step to the right or one step to the left with equal probability (this kind of random walk is called *symmetric*), 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?

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. S_n denote the position of the particle (relative to our starting point 0) after n moves. Thus, we can write

$$S_n = X_1 + X_2 + \cdots + X_n, \tag{2}$$

where $X_i = +1$ if the ith move is to the right and $X_i = -1$ if the move is to the left.

The expectation of S_n can be easily computed as follows. Since $\mathbb{E}[X_i] = (\frac{1}{2} \times 1) + (\frac{1}{2} \times (-1)) = 0$, applying linearity of expectation immediately gives $\mathbb{E}[S_n] = \sum_{i=1}^{n} \mathbb{E}[X_i] = 0$. 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 $|S_n|$, the *distance* of the particle from 0?

Rather than consider the r.v. $|S_n|$, which is a little difficult to work with due to the absolute value operator, we will instead look at the r.v. S_n^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 squared distance after n steps is equal to n:

Proposition 16.1. *For the random variable S_n defined in (2), we have $\mathbb{E}\left[S_n^2\right] = n$.*

Proof. We use the expression (2) and expand the square:

$$\mathbb{E}\left[S_n^2\right] = \mathbb{E}\left[(X_1 + X_2 + \cdots + X_n)^2\right] = \mathbb{E}\left[\sum_{i=1}^{n} X_i^2 + 2\sum_{i<j} X_i X_j\right] = \sum_{i=1}^{n} \mathbb{E}\left[X_i^2\right] + 2\sum_{i<j} \mathbb{E}[X_i X_j]. \tag{3}$$

In the last equality we have used linearity of expectation. To proceed, we need to compute $\mathbb{E}\left[X_i^2\right]$ and $\mathbb{E}[X_i X_j]$ for $i \neq j$. Since X_i can take on only values ± 1, clearly $X_i^2 = 1$ always, so $\mathbb{E}\left[X_i^2\right] = 1$. To compute $\mathbb{E}[X_i X_j]$ for $i \neq j$, note $X_i X_j = +1$ when $X_i = X_j = +1$ or $X_i = X_j = -1$, and otherwise $X_i X_j = -1$. Therefore,

$$\begin{aligned}
\mathbb{P}[X_i X_j = 1] &= \mathbb{P}[(X_i = X_j = +1) \vee (X_i = X_j = -1)] \\
&= \mathbb{P}[X_i = X_j = +1] + \mathbb{P}[X_i = X_j = -1] \\
&= \mathbb{P}[X_i = +1] \times \mathbb{P}[X_j = +1] + \mathbb{P}[X_i = -1] \times \mathbb{P}[X_j = -1] \\
&= \frac{1}{4} + \frac{1}{4} = \frac{1}{2},
\end{aligned}$$

where the second equality follows from the fact that the events $X_i = X_j = +1$ and $X_i = X_j = -1$ are mutually exclusive, while the third equality follows from the independence of the events $X_i = +1$ and $X_j = +1$, and likewise for the events $X_i = -1$ and $X_j = -1$. Since the only other possible value for $X_i X_j$ is -1, we must also have $\mathbb{P}[X_i X_j = -1] = \frac{1}{2}$, and hence $\mathbb{E}[X_i X_j] = 0$. [**Note:** Later we will see a much simpler way to compute $\mathbb{E}[X_i X_j]$ using the fact that X_i, X_j are independent.]

Finally, plugging $\mathbb{E}\left[X_i^2\right] = 1$ and $\mathbb{E}[X_iX_j] = 0$, for $i \neq j$, into (3) gives $\mathbb{E}\left[S_n^2\right] = \sum_{i=1}^n 1 + 2\sum_{i<j} 0 = n$, as claimed. □

So, for the symmetric random walk example, we see that the 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 $\mathbb{E}[|S_n|] = \sqrt{\mathbb{E}[S_n^2]} = \sqrt{n}$. (Why not?) We will see later in the course how to make precise deductions about $|S_n|$ from knowledge of $\mathbb{E}\left[S_n^2\right]$. For the moment, however, let us agree to view $\mathbb{E}\left[S_n^2\right]$ as an intuitive measure of "spread" of the r.v. S_n.

For a more general r.v. X with expectation $\mathbb{E}[X] = \mu$, what we are really interested in is $\mathbb{E}\left[(X-\mu)^2\right]$, the expected squared distance *from the mean*. In our symmetric random walk example, we have $\mathbb{E}[S_n] = \mu = 0$, so $\mathbb{E}\left[(S_n-\mu)^2\right]$ just reduces to $\mathbb{E}\left[S_n^2\right]$.

Definition 16.1 (Variance)**.** *For a r.v. X with expectation $\mathbb{E}[X] = \mu$, the* variance *of X is defined to be*

$$\mathrm{Var}(X) = \mathbb{E}\left[(X-\mu)^2\right].$$

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

The point of taking the square root of variance is to put the standard deviation "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. We shall usually use the variance. For the random walk example above, Proposition 16.1 implies that $\mathrm{Var}(S_n) = n$, and the standard deviation is $\sigma(S_n) = \sqrt{n}$.

The following observation provides a slightly different way to compute the variance, which sometimes turns out to be simpler.

Theorem 16.1. *For a r.v. X with expectation $\mathbb{E}[X] = \mu$, we have* $\mathrm{Var}(X) = \mathbb{E}\left[X^2\right] - \mu^2$.

Proof. From the definition of variance, we have

$$\mathrm{Var}(X) = \mathbb{E}\left[(X-\mu)^2\right] = \mathbb{E}\left[X^2 - 2\mu X + \mu^2\right] = \mathbb{E}\left[X^2\right] - 2\mu\,\mathbb{E}[X] + \mu^2 = \mathbb{E}\left[X^2\right] - \mu^2.$$

In the third equality, we used linearity of expectation; note that, since $\mu = \mathbb{E}[X]$ is a constant, $\mathbb{E}[\mu X] = \mu\,\mathbb{E}[X] = \mu^2$ and $\mathbb{E}\left[\mu^2\right] = \mu^2$. □

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

$$\mathrm{Var}(cX) = c^2\mathrm{Var}(X). \tag{4}$$

Verifying this is straightforward and left as an **exercise**.

2 Variance Computation

Let us 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 $\mathbb{E}[X] = \frac{7}{2}$. So we just need to compute $\mathbb{E}\left[X^2\right]$, which is a routine calculation:

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

Thus, from Theorem 16.1,

$$\text{Var}(X) = \mathbb{E}\left[X^2\right] - (\mathbb{E}[X])^2 = \frac{91}{6} - \frac{49}{4} = \frac{35}{12}.$$

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

$$\mathbb{E}[X] = \frac{n+1}{2}, \qquad \text{Var}(X) = \frac{n^2-1}{12}, \qquad \sigma(X) = \sqrt{\frac{n^2-1}{12}}. \tag{5}$$

You should verify these as an **exercise**. [Recall that $\sum_{i=1}^n i = \frac{n(n+1}{2}$ and $\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$.]

3. **Fixed points of permutations.** Let X_n be the number of fixed points in a random permutation of n items (i.e., in the homework permutation example, X_n is the number of students in a class of size n who receive their own homework after shuffling). We saw in the previous note that $\mathbb{E}[X_n] = 1$, regardless of n. To compute $\mathbb{E}\left[X_n^2\right]$, write $X_n = I_1 + I_2 + \cdots + I_n$, where $I_i = 1$ if i is a fixed point, and $I_i = 0$ otherwise. Then as usual we have

$$\mathbb{E}\left[X_n^2\right] = \sum_{i=1}^n \mathbb{E}\left[I_i^2\right] + 2\sum_{i<j} \mathbb{E}[I_i I_j]. \tag{6}$$

Since I_i is an indicator r.v., we have that $\mathbb{E}\left[I_i^2\right] = \mathbb{P}[I_i = 1] = \frac{1}{n}$. For $i < j$, since both I_i and I_j are indicators, we can compute $\mathbb{E}[I_i I_j]$ as follows:

$$\mathbb{E}[I_i I_j] = \mathbb{P}[I_i I_j = 1] = \mathbb{P}[I_i = 1 \wedge I_j = 1] = \mathbb{P}[\text{both } i \text{ and } j \text{ are fixed points}] = \frac{1}{n(n-1)}.$$

Make sure that you understand the last step here! Plugging this into equation (6) we get

$$\mathbb{E}\left[X_n^2\right] = \sum_{i=1}^n \frac{1}{n} + 2\sum_{i<j} \frac{1}{n(n-1)} = \left(n \times \frac{1}{n}\right) + \left[2\binom{n}{2} \times \frac{1}{n(n-1)}\right] = 1 + 1 = 2.$$

Thus, $\text{Var}(X_n) = \mathbb{E}\left[X_n^2\right] - (\mathbb{E}[X_n])^2 = 2 - 1 = 1$. 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.

3 Sum of Independent Random Variables

One of the most important and useful facts about variance is that 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 $\text{Var}(X) = \sum_{i=1}^n \text{Var}(X_i) = n \cdot \text{Var}(X_1)$. This means that the standard deviation is $\sigma(X) = \sqrt{n} \cdot \sigma(X_1)$. Note that by contrast, the expected value is $\mathbb{E}[X] = n \cdot \mathbb{E}[X_1]$. 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 16.2. *For* independent *random variables X, Y, we have $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$.*

Proof. We have

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

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 16.3. *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 16.1 and linearity of expectation, we have

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

Since X, Y are independent, Theorem 16.2 implies that the final term in this expression is zero. □

It is very important to remember that **neither** of the above two results is true in general when X, Y are not independent. As a simple example, note that even for a $\{0,1\}$-valued r.v. X with $\mathbb{P}[X = 1] = p$, $\mathbb{E}\left[X^2\right] = p$ is not equal to $\mathbb{E}[X]^2 = p^2$ (because of course X and X are not independent!). This is in contrast to linearity of 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, regardless of whether or not the r.v.'s are independent.

Example

Let us return to our motivating example of a sequence of n coin tosses. Let X_n denote the number of Heads in n tosses of a biased coin with Heads probability p (i.e., $X_n \sim \text{Binomial}(n, p)$). As usual, we write $X_n = I_1 + I_2 + \cdots + I_n$, where $I_i = 1$ if the ith toss is H, and $I_i = 0$ otherwise.

We already know $\mathbb{E}[X_n] = \sum_{i=1}^n \mathbb{E}[I_i] = np$. We can compute $\mathrm{Var}(I_i) = \mathbb{E}\left[I_i^2\right] - \mathbb{E}[I_i]^2 = p - p^2 = p(1-p)$. Since the I_i's are independent, by Theorem 16.3 we get $\mathrm{Var}(X_n) = \sum_{i=1}^n \mathrm{Var}(I_i) = np(1-p)$.

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 (5), where the standard deviation $\sigma(X) = \sqrt{\frac{n^2-1}{12}} \approx \frac{n}{\sqrt{12}}$ (for large n) 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.

4 Covariance and Correlation

The expression $\mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$ in the proof of Theorem 16.3 is a measure of association between X, Y, and is called the *covariance*:

Definition 16.2 (Covariance)**.** *The* covariance *of random variables X and Y, denoted* $\mathrm{Cov}(X,Y)$*, is defined as*

$$\mathrm{Cov}(X,Y) = \mathbb{E}[(X-\mu_X)(Y-\mu_Y)] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y],$$

where $\mu_X = \mathbb{E}[X]$ and $\mu_Y = \mathbb{E}[Y]$.

Exercise: Check that the above two expressions for $\mathrm{Cov}(X,Y)$ are indeed equal.

Remarks. We note some important facts about covariance.

1. If X, Y are independent, then $\mathrm{Cov}(X,Y) = 0$. However, the converse is **not** true.
2. $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$.
3. Covariance is *bilinear*; i.e., for any collection of random variables $\{X_1,\ldots,X_n\}, \{Y_1,\ldots,Y_m\}$ and fixed constants $\{a_1\ldots,a_n\}, \{b_1,\ldots,b_m\}$,

$$\mathrm{Cov}(\textstyle\sum_{i=1}^n a_i X_i, \sum_{j=1}^m b_j Y_j) = \sum_{i=1}^n \sum_{j=1}^m a_i b_j \mathrm{Cov}(X_i, Y_j).$$

For general random variables X and Y,

$$\mathrm{Var}(X+Y) = \mathrm{Var}(X) + \mathrm{Var}(Y) + 2\mathrm{Cov}(X,Y).$$

While the *sign* of $\mathrm{Cov}(X,Y)$ (positive or negative) is informative of how X and Y are associated, its magnitude is difficult to interpret. A statistic that is easier to interpret is *correlation*:

Definition 16.3 (Correlation)**.** *Suppose X and Y are random variables with $\sigma(X) > 0$ and $\sigma(Y) > 0$. Then, the correlation of X and Y is defined as*

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sigma(X)\sigma(Y)}.$$

Correlation is more useful than covariance because the former always ranges between -1 and $+1$, as the following theorem shows:

Theorem 16.4. *For any pair of random variables X and Y with $\sigma(X) > 0$ and $\sigma(Y) > 0$,*

$$-1 \le \mathrm{Corr}(X,Y) \le +1.$$

Proof. Let $\mathbb{E}[X] = \mu_X$ and $\mathbb{E}[Y] = \mu_Y$, and define $\widetilde{X} = (X-\mu_X)/\sigma(X)$ and $\widetilde{Y} = (Y-\mu_Y)/\sigma(Y)$. Then, $\mathbb{E}[\widetilde{X}^2] = \mathbb{E}[\widetilde{Y}^2] = 1$, so

$$0 \le \mathbb{E}[(\widetilde{X}-\widetilde{Y})^2] = \mathbb{E}[\widetilde{X}^2] + \mathbb{E}[\widetilde{Y}^2] - 2\,\mathbb{E}[\widetilde{X}\widetilde{Y}] = 2 - 2\,\mathbb{E}[\widetilde{X}\widetilde{Y}]$$
$$0 \le \mathbb{E}[(\widetilde{X}+\widetilde{Y})^2] = \mathbb{E}[\widetilde{X}^2] + \mathbb{E}[\widetilde{Y}^2] + 2\,\mathbb{E}[\widetilde{X}\widetilde{Y}] = 2 + 2\,\mathbb{E}[\widetilde{X}\widetilde{Y}],$$

which implies $-1 \leq \mathbb{E}[\widetilde{X}\widetilde{Y}] \leq +1$. Finally, note that

$$\text{Corr}(X,Y) = \frac{\text{Cov}(X,Y)}{\sigma(X)\sigma(Y)} = \frac{\mathbb{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma(X)\sigma(Y)} = \mathbb{E}\left[\widetilde{X}\widetilde{Y}\right].$$

Therefore, we can deduce that $-1 \leq \text{Corr}(X,Y) \leq +1$. □

Note that the above proof shows that $\text{Corr}(X,Y) = +1$ if and only if $\mathbb{E}[(\widetilde{X}-\widetilde{Y})^2] = 0$, which implies $\widetilde{X} = \widetilde{Y}$ with probability 1. Similarly, $\text{Corr}(X,Y) = -1$ if and only if $\mathbb{E}[(\widetilde{X}+\widetilde{Y})^2] = 0$, which implies $\widetilde{X} = -\widetilde{Y}$ with probability 1. In terms of the original random variables X,Y, this means the following: if $\text{Corr}(X,Y) = \pm 1$, then there exist constants a and b such that, with probability 1,

$$Y = aX + b,$$

where $a > 0$ if $\text{Corr}(X,Y) = +1$ and $a < 0$ if $\text{Corr}(X,Y) = -1$. (The values a, b depend on $\mu_X, \mu_Y, \sigma(X), \sigma(Y)$; in particular, $a = \pm\frac{\sigma(Y)}{\sigma(X)}$ according to whether $\text{Corr}(X,Y) = \pm 1$.) Thus X and Y differ just by a scaling factor and a shift.

5 Exercises

1. For any random variable X and constant c, show that $\text{Var}(cX) = c^2\text{Var}(X)$.
2. For $X \sim \text{Uniform}\{1,\ldots,n\}$, show that $\mathbb{E}[X]$ and $\text{Var}(X)$ are as shown in (5).

Concentration Inequalities and the Laws of Large Numbers

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 $\mathbb{E}[S_n] = np$, we see that the estimator $\hat{p}$ always has the correct expected value: $\mathbb{E}[\hat{p}] = \frac{1}{n}\mathbb{E}[S_n] = p$. What is the role of n here? Why do we take the average of many samples, rather than just one? As we shall see, as n gets larger the estimate $\hat{p}$ gets closer to the desired value p. This is a manifestation of the *Law of Large Numbers*, which corresponds to our own empirical experience and which we shall prove formally 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* $1 - \delta$, namely, $\mathbb{P}[|\hat{p} - p| \leq \varepsilon] \geq 1 - \delta$. 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 $\mathbb{P}[|\hat{p} - p| \leq \varepsilon] \geq 1 - \delta$, it suffices to toss the coin n times where

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

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

1 Markov's Inequality

Before discussing Chebyshev's inequality, we first prove the following simpler bound, which applies only to *nonnegative* random variables (i.e., r.v.'s which take only values ≥ 0).

Markov's inequality is intuitively similar to the notion that not everyone can score better than average. More precisely, at most half the people can score at least twice the average (assuming scores are non-negative).

Theorem 17.1 (**Markov's Inequality**). *For a* nonnegative *random variable X (i.e., $X(\omega) \geq 0$ for all $\omega \in \Omega$) with finite mean,*

$$\mathbb{P}[X \geq c] \leq \frac{\mathbb{E}[X]}{c},$$

for any positive constant c.

Proof. Let $\mathscr{A}$ denote the range of X and consider any constant $c \in \mathscr{A}$. Then,

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{a \in \mathscr{A}} a \times \mathbb{P}[X = a] \\
&\geq \sum_{a \geq c} a \times \mathbb{P}[X = a] \\
&\geq \sum_{a \geq c} c \times \mathbb{P}[X = a] \\
&= c \sum_{a \geq c} \mathbb{P}[X = a] \\
&= c\,\mathbb{P}[X \geq c],
\end{aligned}$$

where the first line is just the definition of expectation, while the second line follows from the fact that X is a nonnegative random variable (i.e., $a \geq 0$ for all $a \in \mathscr{A}$). Rearranging the last inequality gives us the desired result. □

Setting $c = 2\,\mathbb{E}[x]$ in Markov's inequality, the statement is that at most $1/2$ the people can be at least twice the average. The proof itself is analogous to this observation.

A slicker proof can be provided using the indicator function $I\{\cdot\}$, defined for any event $\mathscr{E}$ as

$$I\{\mathscr{E}\} = \begin{cases} 1, & \text{if } \mathscr{E} \text{ is true,} \\ 0, & \text{if } \mathscr{E} \text{ is false.} \end{cases}$$

Alternative proof of Theorem 17.1. Since X is a nonnegative random variable and $c > 0$, we have

$$X \geq cI\{X \geq c\}. \tag{1}$$

since the right hand side is 0 if $X < c$ and is c if $X \geq c$. Taking the expectation of both sides gives

$$\mathbb{E}[X] \geq c\,\mathbb{E}[I\{X \geq c\}] = c\mathbb{P}[X \geq c],$$

where the first inequality follows from (1) and the last equality from the fact that $I\{X \geq c\}$ is an indicator random variable. □

There is an intuitive way to understand Markov's inequality through the analogy of balancing a seesaw, illustrated in Figure 1. Imagine that the probability distribution of a nonnegative random variable X is resting on a fulcrum at $\mu = \mathbb{E}[X]$. We are trying to find an upper bound on the percentage of the distribution which lies beyond $k\mu$, i.e., $\mathbb{P}[X \geq k\mu]$. In other words, we seek to add as much mass m_2 as possible on the seesaw at (or beyond) $k\mu$ while keeping it balanced. This mass will represent the upper bound we are searching for. To minimize the mass's effect, we must imagine that the mass of the distribution which lies beyond $k\mu$ is concentrated at exactly $k\mu$. However, to keep things balanced, we must add another mass m_1 as far to the left of 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 nonnegative. Moreover, the two masses m_1 and m_2 must add up to 1, since they represent the total mass of the probability distribution. Since the distances to the fulcrum are in the ratio $k-1$ to 1, a unit mass at $k\mu$ balances $k-1$ units of mass at 0. So the masses should be $\frac{k-1}{k}$ at 0 and $\frac{1}{k}$ at $k\mu$. This tells us that the maximum value of the mass m_2 is $\frac{1}{k}$, which is exactly Markov's bound with $\alpha = k\mu$.

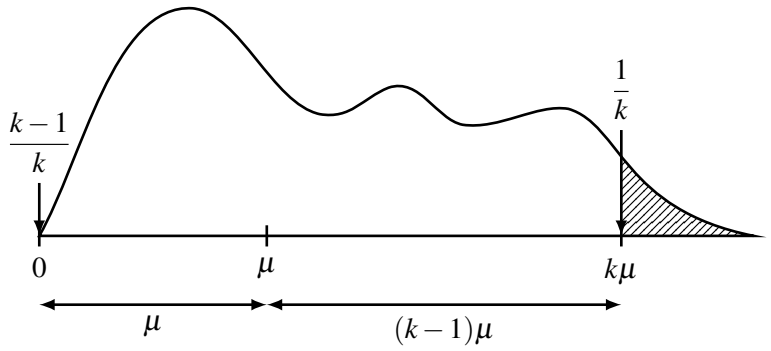


Figure 1: Markov's inequality interpreted as balancing a seesaw.

Example: Coin Tosses

Consider tossing a fair coin n times and let X denote the number of heads observed. What is the probability of observing more than $\frac{3}{4}n$ heads? Let's apply Markov's inequality to obtain an upper bound on $\mathbb{P}[X \geq \frac{3}{4}n]$. Since $X \sim \text{Binomial}(n, \frac{1}{2})$, we have $\mathbb{E}[X] = \frac{1}{2}n$, so

$$\mathbb{P}[X \geq \tfrac{3}{4}n] \leq \frac{\mathbb{E}[X]}{\frac{3}{4}n} = \frac{2}{3},$$

which does not depend on n. Is this a good upper bound? The exact answer for $\mathbb{P}[X \geq \frac{3}{4}n]$ is given by

$$\mathbb{P}[X \geq \tfrac{3}{4}n] = \sum_{k=\left\lceil \frac{3}{4}n \right\rceil}^{n} \binom{n}{k} \frac{1}{2^n},$$

which is $\approx 5.5 \times 10^{-2}$ for $n = 10$ and $\approx 2.8 \times 10^{-7}$ for $n = 100$. As these numerical values show, $\mathbb{P}[X \geq \frac{3}{4}n]$ is a decreasing function of n, and so Markov's inequality does not provide a very good upper bound for this example. In the next section, we will see how to obtain an improved bound.

2 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 17.2 (Chebyshev's Inequality). *For a random variable X with finite expectation $\mathbb{E}[X] = \mu$,*

$$\mathbb{P}[|X - \mu| \geq c] \leq \frac{\text{Var}(X)}{c^2}, \tag{2}$$

for any positive constant c.

Proof. Define $Y = (X - \mu)^2$ and note that $\mathbb{E}[Y] = \mathbb{E}\left[(X - \mu)^2\right] = \text{Var}(X)$. Also, notice that the event that we are interested in, $|X - \mu| \geq c$, is exactly the same as the event $Y = (X - \mu)^2 \geq c^2$. Therefore, $\mathbb{P}[|X - \mu| \geq c] = \mathbb{P}[Y \geq c^2]$. Moreover, Y is obviously nonnegative, so we can apply Markov's inequality in Theorem 17.1 to get

$$\mathbb{P}[|X - \mu| \geq c] = \mathbb{P}[Y \geq c^2] \leq \frac{\mathbb{E}[Y]}{c^2} = \frac{\text{Var}(X)}{c^2}.$$

This completes the proof. □

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

Corollary 17.1. *For any random variable X with finite expectation $\mathbb{E}[X] = \mu$ and finite standard deviation $\sigma = \sqrt{\text{Var}(X)}$,*

$$\mathbb{P}[|X - \mu| \geq k\sigma] \leq \frac{1}{k^2},$$

for any constant $k > 0$.

Proof. Plug $c = k\sigma$ 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 that only have means and variances, Chebyshev's inequality is essentially the only tool. Its power derives from the fact that it can be applied to *any* random variable with a mean and a variance.

Example: Coin Tosses Revisited

Let's revisit the coin toss example considered above and apply Chebyshev's inequality to obtain an upper bound on $\mathbb{P}[X \geq \frac{3}{4}n]$. Recalling $\mathbb{E}[X] = \frac{n}{2}$, we obtain

$$\mathbb{P}[X \geq \tfrac{3}{4}n] = \mathbb{P}[X - \tfrac{n}{2} \geq \tfrac{n}{4}] \leq \mathbb{P}[\left|X - \tfrac{n}{2}\right| \geq \frac{n}{4}] \leq \frac{\text{Var}(X)}{\left(\frac{n}{4}\right)^2}.$$

Since $X \sim \text{Binomial}(n, \frac{1}{2})$, we have $\text{Var}(X) = n\frac{1}{2}(1 - \frac{1}{2}) = \frac{n}{4}$, so we obtain

$$\mathbb{P}[X \geq \tfrac{3}{4}n] \leq \frac{\text{Var}(X)}{\left(\frac{n}{4}\right)^2} = \frac{4}{n},$$

which is much better than the constant bound of $\frac{2}{3}$ given by Markov's inequality.

3 Applications

In this section, we discuss applications of Chebyshev's inequality to estimation problems.

3.1 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 $X_i = 1$ if the i-th coin toss comes up Heads and $X_i = 0$ otherwise, and the random variables $X_1, \ldots, X_n$ are independent and identically distributed. Then

$\mathbb{E}[X_i] = \mathbb{P}[X_i = 1] = p$, so by linearity of expectation,

$$\mathbb{E}[\hat{p}] = \mathbb{E}\left[\tfrac{1}{n}S_n\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[X_i] = p.$$

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:

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

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

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

To make the right hand side less than the desired value δ, we need to set

$$n \geq \frac{\sigma^2}{\varepsilon^2\delta}. \tag{3}$$

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 $\mathbb{P}[X_i = 1] = p$, we have $\sigma^2 = p(1-p)$, and inequality (3) becomes $n \geq \frac{p(1-p)}{\varepsilon^2\delta}$. Since $p(1-p)$ is maximized[1] when $p = \frac{1}{2}$, we can conclude that it is sufficient to choose n such that:

$$n \geq \max_p \frac{p(1-p)}{\varepsilon^2\delta} = \frac{1}{4\varepsilon^2\delta}, \tag{4}$$

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. And the coin tosses are independent.[2] Our calculation above shows that to get an estimate $\hat{p}$ that is accurate to within an error of 0.01 with probability at least 95%, it suffices to sample $n = 50000$ people[3]. 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 fairly small number of people.

[1] Use calculus to prove this.

[2] We are assuming here that the sampling is done "with replacement"; i.e., we select each person in the sample from the entire population, including those we have already picked. So there is a small chance that we will pick the same person twice.

[3] Actually the sample size needed is quite a bit smaller than this, as you will see in later classes such as CS174 using more sophisticated concentration bounds. But this is certainly a valid upper bound.

3.2 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 $\mathbb{E}[X_i] = \mu$, 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 $\mathbb{E}[\hat{\mu}] = \mu$ 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 $\varepsilon\mu$:

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

Using equation (3), but substituting $\varepsilon\mu$ in place of ε, it is enough for the sample size n to satisfy

$$n \geq \frac{\sigma^2}{\mu^2} \times \frac{1}{\varepsilon^2\delta}. \tag{5}$$

Here ε and $1-\delta$ are the desired relative error and confidence level respectively. Now of course we don't know the other two quantities, μ and σ^2, appearing in equation (5). In practice, we would use a lower bound on μ and an upper bound on σ^2 (just as we used an upper bound on $p(1-p)$ in the coin tossing problem). Plugging these bounds into equation (5) 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) \$20,000 (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 in a population of size 325 million, then assuming a relatively small value of μ means that the variance must be at least about $\frac{(50\times 10^9)^2}{325\times 10^6} \approx 7.7 \times 10^{12}$. 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.

4 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 17.3 (Law of Large Numbers). *Let $X_1, X_2, \ldots,$ be a sequence of i.i.d. (independent and identically distributed) random variables with common finite expectation $\mathbb{E}[X_i] = \mu$ for all i. Then, their partial sums $S_n = X_1 + X_2 + \cdots + X_n$ satisfy*

$$\mathbb{P}\left[\left|\tfrac{1}{n}S_n - \mu\right| \geq \varepsilon\right] \to 0 \qquad \text{as } n \to \infty,$$

for every $\varepsilon > 0$, however small.

Proof. Let $\text{Var}(X_i) = \sigma^2$ be the common variance of the r.v.'s; we assume that σ^2 is finite.[4] With this (relatively mild) assumption, the LLN is an immediate consequence of Theorem 17.2. Since $X_1, X_2, \ldots$ are i.i.d. random variables with $\mathbb{E}[X_i] = \mu$ and $\text{Var}(X_i) = \sigma^2$, we have $\mathbb{E}\left[\frac{1}{n}S_n\right] = \mu$ and $\text{Var}(\frac{1}{n}S_n) = \frac{\sigma^2}{n}$, so by Chebyshev's inequality we have

$$\mathbb{P}\left[\left|\tfrac{1}{n}S_n - \mu\right| \geq \varepsilon\right] \leq \frac{\text{Var}(\frac{1}{n}S_n)}{\varepsilon^2} = \frac{\sigma^2}{n\varepsilon^2} \to 0 \qquad \text{as } n \to \infty.$$

□

Notice that the LLN says that the probability of *any* deviation ε 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.

[4] If σ^2 is not finite, the LLN still holds but the proof requires more advanced mathematical tools.

Three Killer Applications: Hashing, Coupon Collecting and Load Balancing

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

- How large can we choose m while ensuring that with probability at least $1/2$, no two balls land in the same bin?
- If $m = n$ (same number of balls as bins), 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 key insights into two important engineering applications: hashing and load balancing. These in turn arise in many contexts, such as the design of efficient data structures and resource allocation in networks, where demands can be assumed to be random (so-called "stochastic multiplexing.")

Our answers to these questions are based largely on the application of a simple technique called the union bound, discussed earlier. Given any two events A and B, it states that $\mathbb{P}[A \cup B] \leq \mathbb{P}[A] + \mathbb{P}[B]$. This observation plus some "back-of-the-envelope" calculations will get us a long way.

1 Hashing

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 325m 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. Given a key, it also reports whether or not the key belongs to 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 (much smaller than U). 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—i.e., 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.

1.1 Balls and Bins

Let us 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 from T, independently of all other keys. 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 $\mathbb{P}[A]$ will decrease as m increases (with n fixed). Our goal is to find the largest value of m such that $\mathbb{P}[A] \geq 1-\varepsilon$, where $\varepsilon \in (0,1)$ is a specified tolerance level of collision probability. [Note: Strictly speaking, we are looking at different sample spaces here, one for each value of m. So it would be more correct to write $\mathbb{P}_m$ rather than just $\mathbb{P}$, to make clear which sample space we are talking about. However, we will omit this detail.]

1.2 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 $\mathbb{P}[A]$. There are exactly $\binom{m}{2} = \frac{m(m-1)}{2}$ possible pairs among our m keys. Imagine these are numbered from 1 to $\binom{m}{2}$ (it does not matter how). Let C_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 $\overline{A} = \bigcup_{i=1}^{\binom{m}{2}} C_i$. What is $\mathbb{P}[C_i]$? We claim it is just $\frac{1}{n}$ for every i; this is the probability that two particular balls land in the same bin when there are n bins. (Check that you understand this!)

So, using the union bound from earlier, we have

$$\mathbb{P}[\overline{A}] = \mathbb{P}\left[\bigcup_{i=1}^{\binom{m}{2}} C_i\right] \leq \sum_{i=1}^{\binom{m}{2}} \mathbb{P}[C_i] = \binom{m}{2} \times \frac{1}{n} = \frac{m(m-1)}{2n} \approx \frac{m^2}{2n}.$$

This means that the probability of having a collision is less than ε if $\frac{m^2}{2n} \leq \varepsilon$; that is, if $m \leq \sqrt{2\varepsilon n}$. (For $\varepsilon = \frac{1}{2}$, this just says $m \leq \sqrt{n}$.) Thus, if we wish to suffer no collisions with high probability, the size of the hash table must be about the *square* of the cardinality of the set we are trying to store. (In the later section on load balancing, we will see what happens to the number of collisions if we decrease the size of the hash table and make it similar to the size of the set we are trying to store.)

As detailed in Section 1.4, it turns out that we can derive a slightly more accurate bound for m using techniques that are less crude than the union bound. However, although that alternate bound is a little better, both bounds are the same in terms of their dependence on n (both are of the form $m = O(\sqrt{n})$).

1.3 The Birthday Paradox Revisited

Recall the Birthday Paradox from an earlier lecture: how many people need to be in a room in order to ensure that two of them have the same birthday, with probability at least $\frac{1}{2}$? This is exactly the hashing problem considered in the previous section, with $n = 365$. (Why? Actually there is a difference of 1 in the answers, since the hashing problem wants to avoid collisions with probability at least $\frac{1}{2}$ while the birthday problem wants to get at least one collision with probability at least $\frac{1}{2}$). We noted in the earlier lecture, by an exact calculation, that the answer to the birthday problem is 23. This is in line with our approximate calculation in the previous section, since 23 is of a similar order to $\sqrt{365} \approx 19$.

1.4 A More Accurate Bound

In this section, we derive a more accurate bound on m for the collision problem, which will show that the square-root dependence $m = O(\sqrt{n})$ is not only sufficient but also necessary.

Main Idea

Let's fix the value of m and try to compute $\mathbb{P}[A]$ exactly, rather than approximating it using the union bound as we did above. (This exact calculation is what we did for the birthday problem in the earlier lecture.) 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 an earlier note 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 will assume that $m \leq n$.

Now we can calculate the probability of no collision:

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

Before going on, let's pause to observe that we could compute $\mathbb{P}[A]$ 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

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

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, $\mathbb{P}[A_3 \mid A_1 \cap A_2]$ 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 (1) gives an *exact* formula for the probability of no collision when m keys are hashed. All we need to do now is plug values $m = 1,2,3,\ldots$ into (1) until we find that $\mathbb{P}[A]$ drops below $1-\varepsilon$. The corresponding value of m (minus 1) is what we want. (Again, this is exactly what we did for the birthday problem in an earlier lecture.)

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 $\mathbb{P}[A]$ drops below $1-\varepsilon$.

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 $\mathbb{P}[A]$ 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 (1) around", so that it tells us the value of m at which $\mathbb{P}[A]$ 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

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

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

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

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 will 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 (2):

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

Note that we have 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 (which it will be, since we'll only need to take m up to about $\sqrt{n}$). Once we are done, we will see that the approximation is actually pretty good even for modest sizes of n.

Now we can undo the logs in (3) to get our expression for $\mathbb{P}[A]$:

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

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

$$-\frac{m^2}{2n} \geq \ln(1-\varepsilon) = -\ln\left(\frac{1}{1-\varepsilon}\right),$$

or equivalently

$$m \leq \sqrt{2\ln\left(\frac{1}{1-\varepsilon}\right)} \times \sqrt{n}. \tag{4}$$

For $\varepsilon = \frac{1}{2}$, the coefficient in front of $\sqrt{n}$ is $\sqrt{2\ln 2} \approx 1.177$. So the bottom line is that we can hash approximately $m = \lfloor 1.177\sqrt{n} \rfloor$ 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 (1) to compute the exact largest value $m = m_0$ for which $\mathbb{P}[A]$ remains above $\frac{1}{2}$, for a few sample values of n. Then we can compare these values with our estimate $m \approx 1.177\sqrt{n}$:

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. (Note in particular that when $n = 365$ we get $m_0 = 22$. This corresponds to the fact that, when there are 22 people in a room, the probability that any two share a birthday is less than $\frac{1}{2}$, but adding one more person pushes this probability above $\frac{1}{2}$.)

If instead we were interested in keeping the collision probability below (say) $\varepsilon = \frac{1}{20}$ (i.e., 5%), Equation (4) implies 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 $1.177\sqrt{n}$ 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).

2 Coupon Collecting

In the *coupon collecting* problem, our goal is 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?

This problem too can be modelled with balls and bins, where the question now is: "How many balls should one throw in order to have at least one ball in each of the n bins?"

This question can again be addressed using the union bound method on the experiment of throwing m balls into n bins. First we calculate the probability that the *first* bin is empty. A ball misses the bin with probability

$(1-\frac{1}{n})$ and, using the product rule for independent events, we can conclude that all m balls miss the bin with probability $(1-\frac{1}{n})^m$. For large n, we can approximate this value by $e^{-\frac{m}{n}}$, using the standard exponential approximation $(1-\frac{1}{n})^n \approx e^{-1}$. (This approximation is actually very good, even for moderate values of n: try it out yourself!)

Now we can use the union bound to bound the probability that *any* of the n bins is empty. Writing A for the event that any of the bins is empty, and A_i for the event that the ith bin is empty, we have

$$\mathbb{P}[A] = \mathbb{P}\Big[\bigcup_{i=1}^{n} A_i\Big] \leq n \times e^{-\frac{m}{n}}.$$

Finally, by choosing $m = n\ln n + n$, we see that the probability that any bin is empty is at most $1/e$. Thus, with probability at least $1 - 1/e$ (which is a bit larger than $\frac{1}{2}$), we will have collected all n coupons after buying at most $n\ln n + n$ cereal boxes.

This rough calculation actually gives us a pretty accurate answer: more sophisticated calculations can be used to show that, even for moderate values of n, the number of boxes required to collect all the coupons is close to $n\ln n$. To interpret this result, note that we certainly have to buy at least n boxes in order to collect all the coupons; but we would have to be incredibly lucky to get them all in only n boxes! We can view the $\ln n$ factor as the "overhead" in the number of boxes we have to buy in order to be pretty sure we've got all the coupons.

Despite its folksy description, coupon collecting arises frequently in the analysis of many randomized processes and algorithms: e.g., how long does it take for all n components in some system to fire, given that a random one fires each second?

3 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?

3.1 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 will not 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 $\mathbb{P}[A_k] \leq \frac{1}{2}$.

If we have such a value k, then we will know that, with high 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 are just using this for illustration. Essentially the same analysis can be used to find k such that $\mathbb{P}[A_k] \leq 0.05$ (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 will 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.

3.2 Applying the Union Bound

From the hashing application, we know[1] that we get collisions with high probability already when $m \approx 1.177\sqrt{n}$. So, when $m = n$, the maximum load will certainly be larger than 1 (with high 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 will 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 will do is to find the smallest k such that

$$\mathbb{P}[A_k(1)] \leq \frac{1}{2n}. \tag{5}$$

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

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

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:

$$\mathbb{P}[A_k] = \mathbb{P}\left[\bigcup_{i=1}^{n} A_k(i)\right] \leq \sum_{i=1}^{n} \mathbb{P}[A_k(i)] \leq n \times \frac{1}{2n} = \frac{1}{2}.$$

It is worth standing back to notice what we did here: we wanted to conclude that $\mathbb{P}[A_k] \leq \frac{1}{2}$. We could not 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 $A_k(1), \ldots, A_k(n)$, and all have the same probability, it is enough for us to show that $\mathbb{P}[A_k(i)] \leq \frac{1}{2n}$; the union bound then guarantees that $\mathbb{P}[A_k] \leq \frac{1}{2}$. This kind of reasoning is very common in applications of probability in engineering contexts like Computer Science.

Now all that is left to do is to find the smallest k that satisfies (5). That is, we wish to bound the probability that bin 1 has at least k balls (and find the smallest value of k so that this probability is smaller than $\frac{1}{2n}$). 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 equal to the union of the events B_S over all sets S of cardinality k. Thus we have:

$$\mathbb{P}[A_k(1)] = \mathbb{P}[\bigcup_{S} B_S].$$

[1] Note that this problem of finding the heaviest load in load balancing is identical to asking what the length of the longest linked-list in the hash-table would be, when we are storing n items in a hash table of size n.

We can use the union bound on $\mathbb{P}[\bigcup_S B_S]$:

$$\mathbb{P}\left[\bigcup_S B_S\right] \leq \sum_S \mathbb{P}[B_S].$$

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

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

Hence, to achieve our original goal of satisfying (5), we need to find the smallest k so that $\binom{n}{k}\frac{1}{n^k} \leq \frac{1}{2n}$. First note that

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

so $\binom{n}{k}\frac{1}{n^k} \leq \frac{1}{2n}$ if $\frac{1}{k!} \leq \frac{1}{2n}$. Taking logs, this is equivalent to $\ln k! \geq \ln(2n)$. Using a coarse form of Stirling's approximation $\ln k! \approx k\ln k - k$ for large k, we conclude that $\ln k! \approx \ln(2n)$ if k is chosen to be

$$k \approx \frac{\ln n}{\ln\ln n},$$

for large n. This approximation is actually in line with the value obtained by more detailed calculations, and for large n is a good estimate of the maximum load. In fact, a more careful analysis suggests that the value $k = \frac{2\ln n}{\ln\ln n}$ is a good upper bound on the maximum load for reasonable values of n. (The extra factor of 2 wipes out some lower order terms.) The following table shows the growth of this bound, giving a good idea of how the maximum load behaves as a function of n.

n	10	20	50	100	500	1000	10^4	10^5	10^6	10^7	10^8	10^{15}
$\frac{2\ln n}{\ln\ln n}$	5.5	5.5	5.7	6.0	6.8	7.2	8.2	9.4	10.6	11.6	12.6	20

Finally, here is one punchline from the above. Let's say the US population is about 350 million. Suppose we mail 350 million items of junk mail, each one with a random US address. Then with probability at least $\frac{1}{2}$, no one person anywhere will receive more than about a dozen items!

Geometric and Poisson 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 is the Binomial(n,p) distribution, introduced earlier. 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 $\mathbb{P}[S_n = k] = \binom{n}{k} p^k (1-p)^{n-k}$ for $k \in \{0,1,\ldots,n\}$. The expected value is $\mathbb{E}[S_n] = np$ and the variance is $\mathrm{Var}(S_n) = np(1-p)$. The binomial distribution is frequently used to model the number of successes in a repeated experiment.

1 Geometric Distribution

Consider repeatedly tossing a biased coin with Heads probability p. Let X denote the number of tosses *until the first Head appears*. Then X is a random variable that takes values in $\mathbb{Z}^+$, the set of positive integers. The event that $X = i$ is equal to the event of observing Tails for the first $i-1$ tosses and getting Heads in the ith toss, which occurs with probability $(1-p)^{i-1}p$. Such a random variable is called a *geometric* random variable. Note that X is unbounded: i.e., for any positive integer i (no matter how large), there is some positive probability that $X = i$.

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*

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

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

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

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

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

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

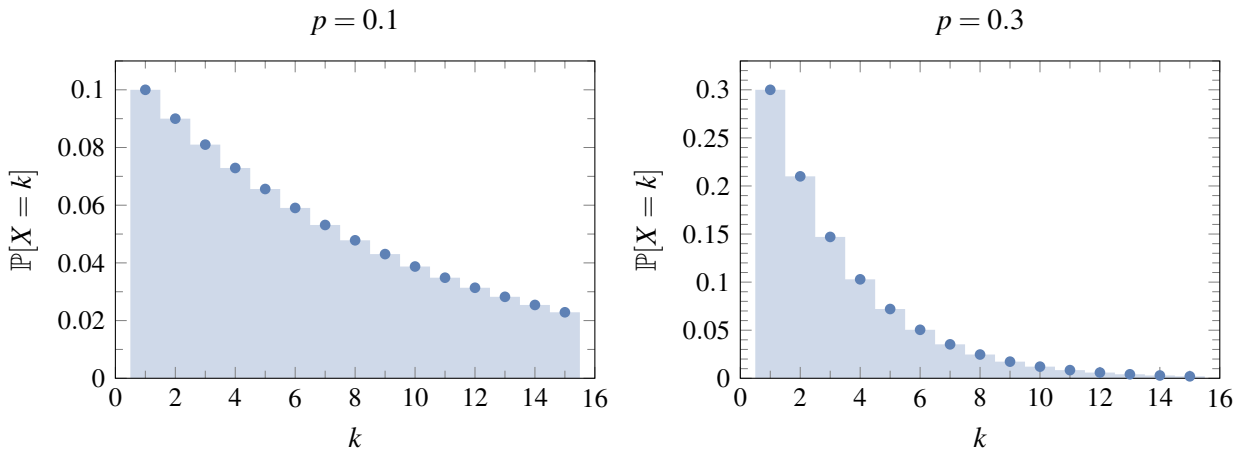


Figure 1: Illustration of the Geometric(p) distribution for $p = 0.1$ and $p = 0.3$.

1.1 Mean and Variance of a Geometric Random Variable

Let us now compute the expectation $\mathbb{E}[X]$. Applying the definition of expected value directly gives us:

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

However, the final summation is a little tricky to evaluate (though it can be done). Instead, we will use the following alternative formula for expectation that simplifies the calculation and is useful in its own right.

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

$$\mathbb{E}[X] = \sum_{i=1}^{\infty} \mathbb{P}[X \geq i].$$

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

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

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:

$$\mathbb{E}[X] = \sum_{j=1}^{\infty} j \times \mathbb{P}[X = j] = \sum_{j=1}^{\infty} \sum_{i=1}^{j} \mathbb{P}[X = j] = \sum_{i=1}^{\infty} \sum_{j=i}^{\infty} \mathbb{P}[X = j] = \sum_{i=1}^{\infty} \mathbb{P}[X \geq i],$$

where the third equality follows from interchanging the order of summations. □

We can now use Theorem 19.1 to compute $\mathbb{E}[X]$ more easily.

Theorem 19.2. *For $X \sim \text{Geometric}(p)$, we have $\mathbb{E}[X] = \frac{1}{p}$.*

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

$$\mathbb{P}[X \geq i] = (1-p)^{i-1} \quad \text{for } i = 1, 2, \ldots. \tag{1}$$

We can obtain this simply by summing $\mathbb{P}[X = j]$ 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

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

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 $\frac{1}{p}$. Thus for a fair coin, the expected number of tosses until the first Head is 2 (but of course the actual number of tosses can be any positive integer).

Let us now compute the variance of X.

Theorem 19.3. *For $X \sim \text{Geometric}(p)$, we have $\text{Var}(X) = \frac{1-p}{p^2}$.*

Proof. We use the formula

$$\text{Var}(X) = \mathbb{E}\left[X^2\right] - (\mathbb{E}[X])^2 = \mathbb{E}\left[X^2\right] - \frac{1}{p^2}. \tag{2}$$

Thus we just have to compute $\mathbb{E}\left[X^2\right]$, which we can do using a calculus trick as follows. Starting from

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

and differentiating with respect to p, we get

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

Here we dropped the first term, for $i = 0$, since its derivative is zero, and multiplied both sides by -1. (Note incidentally that (3) gives an alternative proof of the fact that $\mathbb{E}[X] = \frac{1}{p}$; why?)

Now we multiply both sides of (3) by $(1-p)$ and differentiate again to get

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

Since the left-hand side here is exactly $\frac{1}{p}\mathbb{E}\left[X^2\right]$, we conclude that

$$\mathbb{E}\left[X^2\right] = \frac{2-p}{p^2}.$$

Finally, plugging this into (2) gives us

$$\text{Var}(X) = \frac{2-p}{p^2} - \frac{1}{p^2} = \frac{1-p}{p^2},$$

as claimed. □

1.2 Memoryless Property

The geometric distribution has a special property called *memorylessness*[1]. To get some intuition for this property, suppose we revisit our earlier motivational example of repeatedly tossing a biased coin with heads probability p, and let X denote the number of tosses until the first head appears. Thus $X \sim \text{Geometric}(p)$.

As we've seen, the probability that we need *more than* n tosses before getting the first head is $\mathbb{P}[X > n] = (1-p)^n$, since the first n tosses must all have been tails.

What if we've already tossed the coin m times without getting a head? What is the probability that we need more than n *additional* tosses before getting our first head? We can calculate this probability as:

$$\begin{aligned}\mathbb{P}[X > n+m \mid X > m] &= \frac{\mathbb{P}[X > n+m]}{\mathbb{P}[X > m]} \\ &= \frac{(1-p)^{n+m}}{(1-p)^m} \\ &= (1-p)^n \\ &= \mathbb{P}[X > n]\end{aligned}$$

This gives us the formal statement of the memoryless property of geometric distributions:

$$\mathbb{P}[X > n+m \mid X > m] = \mathbb{P}[X > n]..$$

The geometric distribution is memoryless in the sense that the length of time we have already been waiting does not affect the additional time we have to wait until we get our first head. (This, of course, defies the "common wisdom" that if we've tossed a coin ten times and it's come up tails every time, then surely it's more likely to come up heads on the 11th toss!)

1.3 Application: Coupon Collecting

Recall the coupon collecting problem from a previous note: we are trying to collect a set of n different baseball cards by buying boxes of cereal, each of which contains exactly one random card (uniformly distributed among the n cards). How many boxes do we need to buy until we have collected at least one copy of every card?

Let S_n denote the number of boxes we need to buy in order to collect all n cards. The distribution of S_n is difficult to compute directly (try it for $n = 3$!), and we did some calculations in a previous note to estimate the likely values of S_n. But if we are only interested in the expected value $\mathbb{E}[S_n]$, then we can evaluate it easily using linearity of expectation and what we have just learned about the geometric distribution.

We start by writing

$$S_n = X_1 + X_2 + \cdots + X_n \tag{4}$$

for simpler random variables X_i, where X_i is the number of boxes we buy while trying to get the i-th new card (starting immediately after we have gotten the $(i-1)$st new card). With this definition, make sure you believe equation (4) 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 $\mathbb{P}[X_1 = 1] = 1$, and thus $\mathbb{E}[X_1] = 1$.

[1] This property is shared by the *exponential distribution*, the continuous analog of the geometric distribution, which we'll cover in a future note. Indeed, these are the *only* two distributions that are memoryless.

How about X_2? Each time we buy a box, we will 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_2 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

$$\mathbb{E}[X_2] = \frac{n}{n-1}.$$

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

$$\mathbb{E}[X_3] = \frac{n}{n-2}.$$

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

$$\mathbb{E}[X_i] = \frac{n}{n-i+1}.$$

Finally, applying linearity of expectation to equation (4), we get

$$\mathbb{E}[S_n] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \frac{n}{n} + \frac{n}{n-1} + \cdots + \frac{n}{2} + \frac{n}{1} = n \sum_{i=1}^{n} \frac{1}{i}. \tag{5}$$

This is an exact expression for $\mathbb{E}[S_n]$. We can obtain a tidier form by noting that the sum in (5) actually has a very good approximation,[2] namely:

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

where $\gamma_E = 0.5772\ldots$ 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 (5) even for quite small values of n. So for example, for $n = 100$, we expect to buy about 518 boxes.

2 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*

$$\mathbb{P}[X = i] = \frac{\lambda^i}{i!} e^{-\lambda}, \qquad \text{for } i = 0, 1, 2, \ldots \tag{6}$$

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

To make sure this is a valid definition, let us check that (6) is in fact a distribution, i.e., that the probabilities sum to 1. We have

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

[2] This is another of the little tricks you might like to carry around in your toolbox.

In the second-to-last step, we used the Taylor series expansion $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$.

The Poisson distribution is a very widely accepted model for so-called "rare events," such as disconnected 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 λ 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

$$\mathbb{P}[X = 5] = \frac{1^5}{5!}e^{-1} = \frac{1}{120e} \approx \frac{1}{326}.$$

Now suppose the article has 200 pages. If we assume the numbers of typos on each page are independent, then the probability that there is at least one page with exactly 5 typos is

$$\begin{aligned}
\mathbb{P}[\exists \text{ a page with exactly 5 typos}] &= 1 - \mathbb{P}[\text{every page has} \neq 5 \text{ typos}] \\
&= 1 - \prod_{k=1}^{200} \mathbb{P}[\text{page } k \text{ has} \neq 5 \text{ typos}] \\
&= 1 - \prod_{k=1}^{200} (1 - \mathbb{P}[\text{page } k \text{ has exactly 5 typos}]) \\
&= 1 - \left(1 - \frac{1}{120e}\right)^{200} \approx 0.46,
\end{aligned}$$

where in the last step we have used our earlier calculation for $\mathbb{P}[X = 5]$. □

2.1 Mean and Variance of a Poisson Random Variable

Let us now calculate the expectation and variance of a Poisson random variable.

Theorem 19.4. *For a Poisson random variable $X \sim \text{Poisson}(\lambda)$, we have $\mathbb{E}[X] = \lambda$ and $\text{Var}(X) = \lambda$.*

Proof. We can calculate $\mathbb{E}[X]$ directly from the definition of expectation:

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{i=0}^{\infty} i \times \mathbb{P}[X = i] \\
&= \sum_{i=1}^{\infty} i \times \frac{\lambda^i}{i!} e^{-\lambda} && \text{(the } i = 0 \text{ term is equal to 0 so we omit it)} \\
&= \lambda e^{-\lambda} \sum_{i=1}^{\infty} \frac{\lambda^{i-1}}{(i-1)!} \\
&= \lambda e^{-\lambda} e^{\lambda} && \text{(since } e^{\lambda} = \sum_{j=0}^{\infty} \frac{\lambda^j}{j!} \text{ with } j = i-1) \\
&= \lambda.
\end{aligned}$$

Similarly, we can calculate $\mathbb{E}\left[X^2\right]$ as follows:

$$\begin{aligned}
\mathbb{E}\left[X^2\right] &= \sum_{i=0}^{\infty} i^2 \times \mathbb{P}[X=i] \\
&= \sum_{i=1}^{\infty} i^2 \frac{\lambda^i}{i!} e^{-\lambda} && \text{(the } i=0 \text{ term is equal to 0 so we omit it)} \\
&= \lambda e^{-\lambda} \sum_{i=1}^{\infty} i \frac{\lambda^{i-1}}{(i-1)!} \\
&= \lambda e^{-\lambda} \left[\sum_{i=2}^{\infty} \frac{\lambda^{i-1}}{(i-2)!} + \sum_{i=1}^{\infty} \frac{\lambda^{i-1}}{(i-1)!} \right] && \text{(replacing } i \text{ by } (i-1)+1) \\
&= \lambda e^{-\lambda} \left[\lambda e^{\lambda} + e^{\lambda} \right] && \text{(using } e^{\lambda} = \sum_{j=0}^{\infty} \frac{\lambda^j}{j!} \text{ twice, with } j=i-1 \text{ and } j=i-2) \\
&= \lambda^2 + \lambda.
\end{aligned}$$

Therefore,

$$\mathrm{Var}(X) = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 = \lambda^2 + \lambda - \lambda^2 = \lambda,$$

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 illustrates the Poisson(λ) distribution for $\lambda = 1, 2, 5, 20$.

2.2 Sum of Independent Poisson Random Variables

As illustrated in Figure 2, the Poisson(λ) distribution becomes more symmetric and resembles a "bell curve" as λ increases. As we will learn later, this phenomenon is closely related to the following useful fact regarding a sum of independent Poisson random variables.

Theorem 19.5. *Let $X \sim$ Poisson(λ) and $Y \sim$ Poisson(μ) be independent Poisson random variables. Then, $X+Y \sim$ Poisson($\lambda+\mu$).*

Proof. For all $k = 0, 1, 2, \ldots$, we have

$$\begin{aligned}
\mathbb{P}[X+Y=k] &= \sum_{j=0}^{k} \mathbb{P}[X=j, Y=k-j] \\
&= \sum_{j=0}^{k} \mathbb{P}[X=j]\mathbb{P}[Y=k-j] \\
&= \sum_{j=0}^{k} \frac{\lambda^j}{j!} e^{-\lambda} \frac{\mu^{k-j}}{(k-j)!} e^{-\mu} \\
&= e^{-(\lambda+\mu)} \frac{1}{k!} \sum_{j=0}^{k} \frac{k!}{j!(k-j)!} \lambda^j \mu^{k-j} \\
&= e^{-(\lambda+\mu)} \frac{(\lambda+\mu)^k}{k!},
\end{aligned}$$

where the second equality follows from independence, and the last equality from the binomial theorem. □

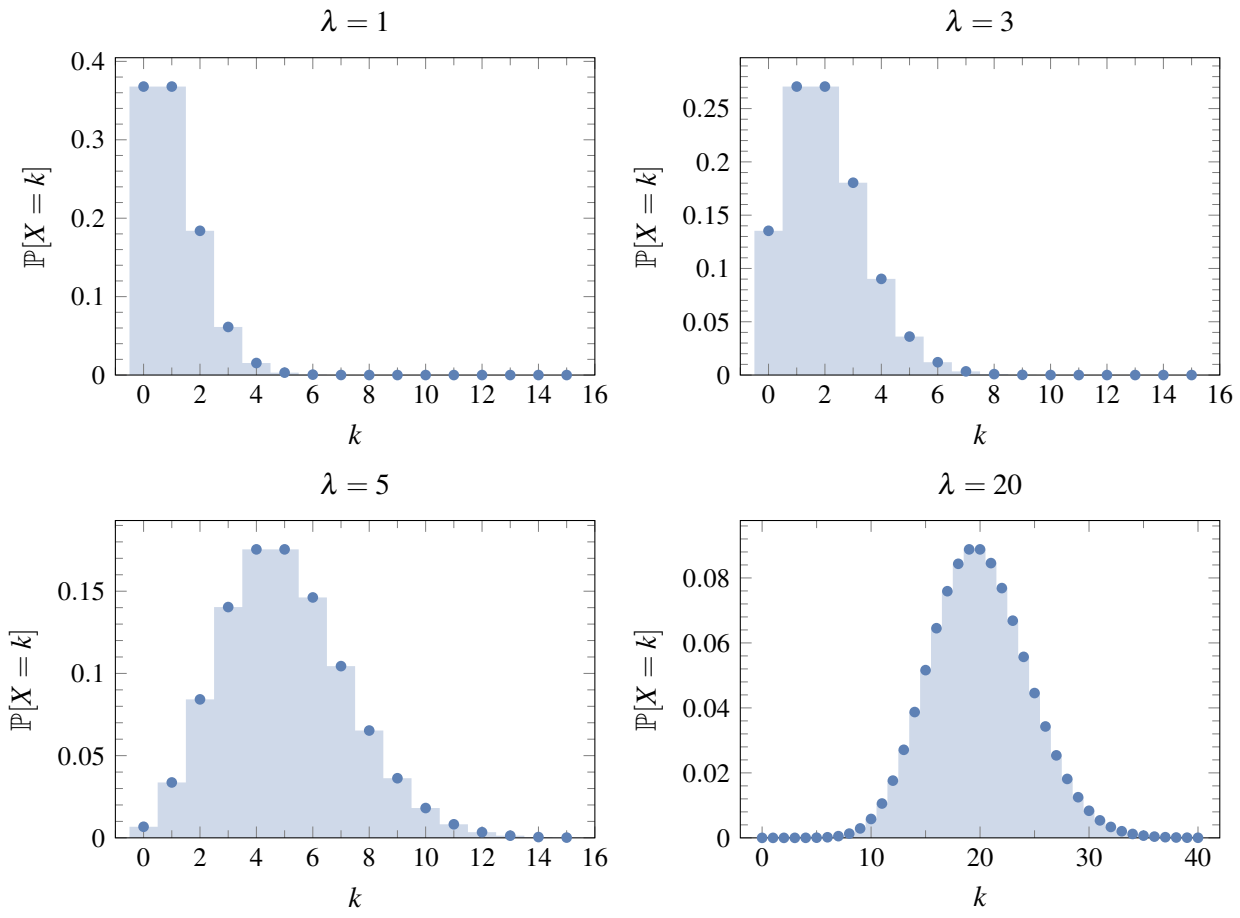


Figure 2: Illustration of the Poisson(λ) distribution for $\lambda = 1, 2, 5, 20$.

By induction, we conclude that if $X_1, X_2, \ldots, X_n$ are independent Poisson random variables with parameters $\lambda_1, \lambda_2, \ldots, \lambda_n$, respectively, then

$$X_1 + X_2 + \cdots + X_n \sim \text{Poisson}(\lambda_1 + \lambda_2 + \cdots + \lambda_n).$$

2.3 Poisson as Limit of Binomial

To see a concrete example of how the 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 initiations of calls in disjoint time intervals are independent events.

Then, if we divide the one-minute time period into n disjoint intervals, the number of calls X in that time period can be modeled as a Binomial(n, p) random variable, where 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 $\mathbb{E}[X] = \lambda$ and so $np = \lambda$, i.e.,

$p = \lambda/n$. So $X \sim \text{Binomial}(n, \frac{\lambda}{n})$. As we shall see below, as we let n tend to infinity, this distribution tends to the Poisson distribution with parameter λ. This explains why the Poisson distribution is a model for "rare events": it counts the number of heads in a very long sequence ($n \to \infty$) of coin flips, where the expected number of heads is a small finite number.

Now we will prove the claim above that Poisson(λ) is the limit of Binomial($n, \frac{\lambda}{n}$), as n tends to infinity.

Theorem 19.6. *Let $X \sim \text{Binomial}(n, \frac{\lambda}{n})$ where $\lambda > 0$ is a fixed constant. Then for every $i = 0, 1, 2, \ldots$,*

$$\mathbb{P}[X = i] \longrightarrow \frac{\lambda^i}{i!} e^{-\lambda} \quad \text{as } n \to \infty.$$

That is, the probability distribution of X converges to the Poisson distribution with parameter λ.

Proof. Fix $i \in \{0, 1, 2, \ldots\}$, and assume $n \geq i$ (because we will let $n \to \infty$). Then, because X has binomial distribution with parameter n and $p = \frac{\lambda}{n}$,

$$\mathbb{P}[X = i] = \binom{n}{i} p^i (1-p)^{n-i} = \frac{n!}{i!(n-i)!} \left(\frac{\lambda}{n}\right)^i \left(1 - \frac{\lambda}{n}\right)^{n-i}.$$

Let us collect the factors into

$$\mathbb{P}[X = i] = \frac{\lambda^i}{i!} \left(\frac{n!}{(n-i)!} \cdot \frac{1}{n^i}\right) \cdot \left(1 - \frac{\lambda}{n}\right)^n \cdot \left(1 - \frac{\lambda}{n}\right)^{-i}. \tag{7}$$

For any fixed i, the first parenthesis above becomes, as $n \to \infty$,

$$\frac{n!}{(n-i)!} \cdot \frac{1}{n^i} = \frac{n \cdot (n-1) \cdots (n-i+1) \cdot (n-i)!}{(n-i)!} \cdot \frac{1}{n^i} = \frac{n}{n} \cdot \frac{(n-1)}{n} \cdots \frac{(n-i+1)}{n} \to 1.$$

Since λ is a fixed constant, the second parenthesis in (7) becomes, as $n \to \infty$,

$$\left(1 - \frac{\lambda}{n}\right)^n \to e^{-\lambda}.$$

And since i is fixed, the third parenthesis in (7) becomes, as $n \to \infty$,

$$\left(1 - \frac{\lambda}{n}\right)^{-i} \to (1-0)^{-i} = 1.$$

Substituting these results back into (7) gives us

$$\mathbb{P}[X = i] \to \frac{\lambda^i}{i!} \cdot 1 \cdot e^{-\lambda} \cdot 1 = \frac{\lambda^i}{i!} e^{-\lambda},$$

as desired. □

Prediction!

Prediction is one of the most compelling applications of probability, particularly in computer science.

In our setting, prediction is the problem of making an estimate for a random variable from available information; today that information is a distribution and joint distribution. The goal of prediction is to minimize the error (sometimes called loss) in the prediction in a formal sense.

Today, we consider the *mean squared error*. That is, for a random variable X with a known mean, we wish to give an estimate for x which minimizes the following expression:

$$\mathbb{E}\left[(X-x)^2\right].$$

The solution to this prediction problem is $\mathbb{E}[X]$!

1 The parabola

Mathematically, we revisit those childhood days of understanding the parabola. Recall that for a parabola defined by $f(x) = Ax^2 + Bx + C$, the vertex of the parabola is at $x = -\frac{B}{2A}$. Long ago, you derived this by "completing the square".

Exercise: Try this at home!

One can also use calculus and set the derivative, $f'(x) = 2Ax + B$, to zero, and solving for x to find a critical point. It is a minimum when A is positive (open upwards graphically) and a maximum when A is negative.

2 Expected value is optimal!

Recall, we wish to find the value for x which minimizes $\mathbb{E}\left[(X-x)^2\right]$. Let's do this!

$$\begin{aligned}\mathbb{E}\left[(X-x)^2\right] &= \mathbb{E}\left[X^2\right] - 2\,\mathbb{E}[X]\,\mathbb{E}[x] + \mathbb{E}[x]^2 \\ &= \mathbb{E}\left[X^2\right] - 2\,\mathbb{E}[X]x + x^2\end{aligned}$$

The second line follows from the fact that the expectation of x which is a constant with respect the expectation. We see that this is a parabola of the form $Ax^2 + Bx + C$ where $A = 1, B = -2\,\mathbb{E}[X]$ and $C = \mathbb{E}\left[X^2\right]$. Thus, the minimum is at $x = -\frac{-2\,\mathbb{E}[X]}{2} = \mathbb{E}[X]$. Middle school is cool.[1]

An important observation is the minimum value of the mean squared error of this estimator is exactly the variance of X, i.e., $\mathbb{E}\left[(X - \mathbb{E}[X])^2\right]$.

In sum, the best mean squared error of estimator for X is $\mathbb{E}[X]$ and this estimate results in a mean squared error of $\mathrm{Var}(X)$.

[1] The joke is that one learns about the parabola in middle school. Unfortunately, middle school is anything but fun. Also, unfortunately, having to explain a joke perhaps means it is a poor one.

3 Joint Distributions: conditional expectation

Recall the following definition that we presented previously.

Definition 20.1. *The joint distribution for two discrete random variables X and Y is the collection of values $\{((a,b),\mathbb{P}[X=a,Y=b]) : a \in \mathscr{A}, b \in \mathscr{B}\}$, where $\mathscr{A}$ is the set of all possible values taken by X and $\mathscr{B}$ is the set of all possible values taken by Y.*

When given a joint distribution for X and Y, the distribution $\mathbb{P}[X=a]$ for X is called the *marginal distribution* for X, and can be found by "summing" over the values of Y. That is,

$$\mathbb{P}[X=a] = \sum_{b\in\mathscr{B}} \mathbb{P}[X=a,Y=b].$$

The marginal distribution for Y is analogous, as is the notion of a joint distribution for any number of random variables.

From a joint distribution, we can compute the *conditional probability* of $X=x$ given that $Y=y$ from the definition of conditional probability as follows:

$$\mathbb{P}[X=x \mid Y=y] = \frac{\mathbb{P}[X=x,Y=y]}{\mathbb{P}[Y=y]}.$$

The *conditional expectation* of X given $Y=y$ is defined naturally as follows:

$$\mathbb{E}[X \mid Y=y] = \sum_{x\in\mathscr{A}} x \cdot \mathbb{P}[X=x \mid Y=y].$$

That is, $\mathbb{E}[X \mid Y=y]$ is simply the expectation of X given that $Y=y$. This is useful for prediction in the same sense that expectation is useful as we will return to later in this note.

3.1 Iterated Expectation and Wald's identity

Before returning to prediction, we discuss *the law of iterated expectations* which is

$$\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X \mid Y]] = \sum_{y\in\mathscr{B}} \mathbb{E}[X \mid Y=y]\mathbb{P}[Y=y]. \tag{1}$$

This simple concept can be quite useful. For example, consider choosing an integer N at random and forming a random variable $Y = X_1 + \cdots + X_N$ where the X_i are identical and independently distributed. Note the *number* of terms is a random variable here! We wish to compute $\mathbb{E}[Y]$ and assuming that the random variables X_i is independent of the value of N.

$$\begin{aligned}
\mathbb{E}[Y] &= \mathbb{E}[\mathbb{E}[Y \mid N]] \\
&= \sum_n \mathbb{E}[Y \mid N=n]\mathbb{P}[N=n] \\
&= \sum_n n\,\mathbb{E}[X_1]\mathbb{P}[N=n] \\
&= \mathbb{E}[X_1]\sum_n n\mathbb{P}[N=n] \\
&= \mathbb{E}[X_1]\,\mathbb{E}[N]
\end{aligned}$$

The third line follows from $Y = X_1 + \cdots + X_n$ and the fact that X_1 are identically distributed and are independent of the value of N. Thus, we have $\mathbb{E}[Y] = \mathbb{E}[X_1]\,\mathbb{E}[N]$ which is the basic form of *Wald's identity.*

This can be useful for modelling the total time to serve customers in a time interval, where we have "Poisson" arrivals, and each customer's service time is from the same distribution. That is, we have a Poisson random variable, $N \sim \text{Poisson}(\lambda)$, that determines the number of customers and each X_i is a random variable that corresponds to the time needed to serve customer i.

To conclude, we note that the law of iterated expectations is sometimes called the tower rule as one can extend the concept to more than two random variables, e.g., $\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[\mathbb{E}[X \mid Y,Z]]]$, where the outer expectations are over the values of Y and Z analogously to (1).

4 Minimum Mean Square Estimate (MMSE): an example

Returning to prediction: one predicts given more information just than one's prior expectations or (in our words) distributions to predict a value. We do this constantly in our behavior, e.g., when will a traffic light turn red after we see it turn yellow?

A simpler example is to predict the weight of someone from their height. Here the information is two fold, the first being the joint distribution of the weights, W, and heights, H. That is, we have $\mathbb{P}[W = w, H = h]$. The second being the value of height itself, h.

Exercise: What estimate should one use for W to minimize mean squared error if one does not have a height?

For the known value of h, we wish to find w that minimizes the mean squared error which is:

$$\mathbb{E}\left[(W-w)^2 \mid H = h\right].$$

Recall, that conditioning on $H = h$, in turn yields a distribution on W which we denote as $\mathbb{P}[W = w \mid H = h]$, which in turn has a conditional expectation, denoted by $\mathbb{E}[W \mid H = h]$. This is just the expectation of W in the sample space corresponding to the event $H = h$, and thus by the previous section is the best estimate for W with regards to minimizing the mean squared error.

Note that $\mathbb{E}[W \mid H = h]$ can be viewed as a function of h. And since for each h, the mean squared error is minimized, we can also see that the mean squared error $\mathbb{E}_{W,H}[(W - \mathbb{E}[W \mid H])^2]$ is minimized with respect to the entire sample space.[2] That is, we take the expected squared error over the entire joint distribution. This is subtle, but should be more clear after we discuss linear regression.

4.1 Application and warning

An example of doing this, informally, might be grouping people into buckets according to height, and predicting the weight of a random person in a bucket as the "average" or expectation of the people in the bucket.

The process is a way of approximating the joint distribution of height and weight. Statisticians are careful when doing so to distinguish the sample mean from the real underlying mean (or the conditioned expectation) for example by denoting the sample mean by $\tilde{\mu}$ and the real and unknown mean by μ and being careful about some other terminology. To do this properly is both subtle and interesting but for another time.

In this case, we presume one knows the distribution. This is fine when we are working with a distribution like geometric or binomial, but with the height and weight example, one may never truly know the joint

[2] Here, $\mathbb{E}_{W,H}[(W - \mathbb{E}[W \mid H])^2] = \sum_{w,h} \mathbb{P}[W = w, H = h] \times (w - \mathbb{E}[W \mid H = h])^2$.

distribution. On the other hand, this frame is a solid starting point and is mathematically rigorous given perfect knowledge of the joint distribution and as the number of samples gets large the situation approaches full information.

5 Linear Regression

5.1 Discussion

We introduced covariance and the correlation coefficient earlier in the course, the latter of which is perhaps the most often reported quantity regarding relationships between two quantities. Even in CS70, we often compute this coefficient for midterm scores and final exam scores to "measure" the consistency of our exams. To be sure, this is the primary measure used in science and the first cut in numerous prediction or estimation problems. For example, you may have heard it said that someone's GPA in their freshman year explains for 80% of the variance in their GPA in their senior year or something to that effect.

This is a mathematical statement about the two random variables freshman year GPA and senior year GPA.

The statement is based on a prediction or estimation method called linear regression which we will now discuss.

5.2 The mathematics

We consider two random variables, X and Y on some sample space. Recall that for a given value x for X, the best prediction for Y for minimizing mean squared error is simply $\mathbb{E}[Y \mid X = x]$. Note that this prediction could be any function of x as it is defined by the joint distribution.

Say instead, we predict the weight using a *linear* function of h. That is, what is the function $\mathscr{L}(X) = mX + b$ that minimizes the mean squared error, defined as $\mathbb{E}\left[(Y - \mathscr{L}(X))^2\right]$? In particular, we wish to derive values for m and b to produce the best linear estimator for Y given a value x for the random variable X.

For simplicity, we will shift the variables to have mean zero. That is, we consider $\bar{X} = X - \mathbb{E}[X]$ and $\bar{Y} = Y - \mathbb{E}[Y]$. Note for a value x for the random variable X, we have $\bar{x} = x - \mathbb{E}[X]$ for the random variable $\bar{X}$.

Furthermore, recall that the $\text{Var}(\bar{X}) = \text{Var}(X)$, $\text{Var}(\bar{Y}) = \text{Var}(Y)$, and $\text{Cov}(\bar{X}, \bar{Y}) = \text{Cov}(X, Y)$.

To estimate $\bar{Y}$ for some value x, we will consider a function $\bar{f}(x) = m\bar{x}$ (that is we assume for now that $b = 0$). We wish to minimize the mean squared error over the entire joint distribution over X and Y, which corresponds minimizing $\mathbb{E}_{X,Y}[(\bar{Y} - m\bar{X})^2]$.

We can expand this expression as follows:

$$\mathbb{E}\left[(\bar{Y} - m\bar{X})^2\right] = \mathbb{E}\left[\bar{Y}^2\right] - 2\,\mathbb{E}[\bar{X}\bar{Y}] \times m + \mathbb{E}\left[\bar{X}^2\right] \times m^2.$$

We see a quadratic function here in the unknown m of the form $C + Bm + Am^2$ where $A = \mathbb{E}\left[\bar{X}^2\right]$, $B = -2\,\mathbb{E}[\bar{X}\bar{Y}]$, and $C = \mathbb{E}\left[\bar{Y}^2\right]$, which is minimized at $m = -\frac{B}{2A} = \frac{\mathbb{E}[\bar{X}\bar{Y}]}{\mathbb{E}[\bar{X}^2]}$.

We derived this, again, using just our knowledge of a parabola. Again, we assumed that our linear function, $mx + b$, has $b = 0$ or "goes through" the point $(0,0)$ for $\bar{X}$ and $\bar{Y}$. We justify this by minimizing $\mathbb{E}\left[(\bar{Y} - (m\bar{X} + b))^2\right]$ with respect to the choice of b. Here, we expand to obtain

$$\mathbb{E}[\bar{Y}]^2 - 2(m\,\mathbb{E}[\bar{X}\bar{Y}] + b\,\mathbb{E}[\bar{Y}]) + (m^2\,\mathbb{E}[\bar{X}]^2 + 2mb\,\mathbb{E}[\bar{X}] + b^2).$$

Collecting terms and taking the derivative with respect to b and setting to zero gives the equation:

$$2\,\mathbb{E}[\bar{Y}] + 2m\,\mathbb{E}[\bar{X}] + 2b = 0.$$

Here, we see that $b = 0$ satisfies the equation since $\mathbb{E}[\bar{Y}] = \mathbb{E}[\bar{X}] = 0$ which implies that this is where the minimum is.

Finally, to get our estimate for Y we substitute back and to estimate $Y - \mathbb{E}[Y]$, use the fact that $\text{cov}(\bar{X},\bar{Y}) = \text{cov}(X,Y)$,$\text{Var}(\bar{X}) = \text{Var}(X)$, $\text{Var}(\bar{Y}) = \text{Var}(Y)$, and solving for Y yields:

$$\mathscr{L}(X) = \frac{\text{Cov}(X,Y)}{\text{Var}(X)}(X - \mathbb{E}[X]) + \mathbb{E}[Y].$$

The intuition above is that if X differs from its expectation, then we vary Y from its expectation by an amount proportional to $\text{cov}(X,Y)$. The division by $\text{Var}(X)$ scales the movement in X as well as the dependence of the covariance of X on the typical movement in X.

We can also write the formula as follows to see it as a linear function in the variable X.

$$\mathscr{L}(X) = \frac{\text{Cov}(X,Y)}{\text{Var}(X)}X - \frac{\text{Cov}(X,Y)}{\text{Var}(X)}\,\mathbb{E}[X] + \mathbb{E}[Y].$$

Exercise: Knowing that $\mathscr{L}(X)$ goes through $(0,0)$ for zero mean random variables X and Y, what point does the $\mathscr{L}(X)$ always go through for general random variables?

We sometimes refer to this estimator as the *linear least squares estimate* of Y given X, usually denoted by $\text{LLSE}(Y \mid X)$. Here, we note that this is something that can be calculated from knowing joint distribution for X and Y which is assuming more than having access to sample points from a distribution which is often the only access one has to a distributions in statistics.

5.3 Linear estimation as explanatory of variance

Armed with our derivation, we formally define the phrase that X explains some fraction of the variance of Y as how much the smaller the mean squared error for the regression is than the variance of Y (which recall is the mean squared error one gets using the estimate $\mathbb{E}[Y]$.)

Indeed, recall the mean squared error for our problem was this:

$$\mathbb{E}\left[\bar{Y}^2\right] - 2m\,\mathbb{E}[\bar{X}\bar{Y}] + m^2\,\mathbb{E}\left[\bar{X}^2\right],$$

and plugging in $m = \frac{\text{Cov}(\bar{X},\bar{Y})}{\text{Var}(\bar{X})}$, we obtain the expression:

$$\mathbb{E}\left[\bar{Y}^2\right] - \frac{\text{Cov}(\bar{X},\bar{Y})^2}{\text{Var}(\bar{X})}.$$

Dividing by the $\text{Var}(\bar{Y}) = \mathbb{E}\left[\bar{Y}^2\right]$, we obtain the expression:

$$1 - (\text{Corr}(\bar{X},\bar{Y}))^2.$$

That is the correlation coefficient squared is exactly the fraction by which the mean squared error in the linear regression estimator is less than the error in estimating Y by using the estimate $\mathbb{E}[Y]$. Thus, square of the correlation coefficient tells us how much the variance is explained by a linear estimator given X.

6 Statistics

In statistics, one often has some number of samples from a distribution or joint distribution and one wishes to find the line that "best" fits the data where the error is defined as the average squared distance to the line. One basic approach is to assume a uniform distribution over the points themselves and in the limit of large n the above approach works to converge to the LLSE estimator.

To be sure, a Statistician would carefully understand how this approximates the unknown joint distribution. Indeed, the idea of estimating confidence intervals for the coefficients of the regression line as well, though again in the limit the law of large numbers does apply and these values converge.

Statisticians deal with the issue of samples by thinking through the consequences of the sample mean and sample variances and covariances being different for the true ones. Indeed, we have heard exasperation from our statistics colleagues when one confuses the true expectation (typically denoted by μ) with a sample expectation (typically denoted by $\tilde{\mu}$.)

7 Other types of error

The mean square error developed here is quite common in science and applications of statistics. You might even read the phrase X "explains some percentage of the variability" in Y in your local newspaper. This is just the correlation coefficient squared as noted above.

Other notions of error (for example the absolute value of the error) can be more effective for various applications and one might see them in future EECS, Data Science, Statistics or many other departments' curriculums.

8 Prediction from more variables

These days one predicts from many variables; e.g., predicting whether an image is a cat or dog from various features in the image. The above discussion can be generalized to estimating a variable Y from many random variables, $X_1, \ldots, X_n$. With a bit of linear algebra, the framework above can be adjusted to this situation. Again, this is an oft-used technique that may be discussed in courses that are in your future.

Continuous Probability Distributions

Up to now we have focused exclusively on *discrete* sample 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 a certain incident happens, or the direction of travel of a meteorite. In this note, 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 daunting.

1 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 $[0,\ell]$.[1] 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 $\mathbb{P}[\omega] = \frac{1}{m}$ 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 $\mathbb{P}[\omega]$? By uniformity, this probability should be the same for all ω. But if we assign $\mathbb{P}[\omega]$ to be any positive value, then because there are infinitely many ω in Ω, the sum of all probabilities $\mathbb{P}[\omega]$ will be ∞! Thus, $\mathbb{P}[\omega]$ 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 *interval* $[a,b] \subseteq [0,\ell]$, where $b > a$. 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

[1] As we shall see shortly, it doesn't matter whether we consider the closed interval $[0,\ell]$ or the half-open interval $[0,\ell)$.

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 $\mathbb{P}[E] = \sum_i \mathbb{P}[E_i]$, 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.[2]

2 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 $\mathbb{P}[X = a]$. But for the r.v. X corresponding to the position of the pointer, we have $\mathbb{P}[X = a] = 0$ 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 $\mathbb{P}[X = a]$, we specify $\mathbb{P}[a \leq X \leq b]$ for all intervals $[a, b]$.[3] To do this formally, we need to introduce the concept of a *probability density function* (sometimes referred to just as a "density", or a "p.d.f.").

Definition 21.1 (Probability Density Function)**.** *A* probability density function *(p.d.f.) 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:

$$\mathbb{P}[a \leq X \leq b] = \int_a^b f(x)dx \qquad \text{for all } a < b.$$

Let us 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:

$$\mathbb{P}[X \in \mathbb{R}] = \mathbb{P}[-\infty < X < \infty] = \int_{-\infty}^{\infty} f(x)dx = 1. \tag{1}$$

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

[2] 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.

[3] Note that it does not matter whether or not we include the endpoints a, b; since $\mathbb{P}[X = a] = \mathbb{P}[X = b] = 0$, we have $\mathbb{P}[a < X < b] = \mathbb{P}[a \leq X \leq b]$.

$[0,\ell]$ we want the distribution of X to be uniform, which means we should take f to be a constant $f(x) = c$ for $0 \le x \le \ell$. The value of c is determined by the requirement (1) that the total area under f is 1:

$$1 = \int_{-\infty}^{\infty} f(x)dx = \int_0^{\ell} c\,dx = c\ell,$$

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 $[x, x+dx]$ close to x; then we have

$$\mathbb{P}[x \le X \le x+dx] = \int_x^{x+dx} f(z)dz \approx f(x)dx. \tag{2}$$

Thus, we can interpret $f(x)$ as the "probability per unit length" in the vicinity of x.

2.1 Cumulative Distribution Function

For a continuous random variable X, one often starts the discussion with the *cumulative distribution function (c.d.f.)*, which is the function $F(x) = \mathbb{P}[X \le x]$. It is closely related to the probability density function for X, $f(x)$, as

$$F(x) = \mathbb{P}[X \le x] = \int_{-\infty}^{x} f(z)\,dz. \tag{3}$$

Thus, one can describe a random variable X by its c.d.f., denoted by $F(x)$, which then gives the probability density function, $f(x)$, as

$$f(x) = \frac{dF(x)}{dx}.$$

To connect to discrete probability, one might think of approximating a continuous random variable, X, as the set of probabilities for X being in one of a countably infinite set of intervals of length dx on the real line. That is, the set of probabilities $\mathbb{P}[x_k < X \le x_k + dx]$ where $x_k = k\,dx$ for $k \in \mathbb{Z}$. In this view, $\mathbb{P}[X \le x_i] = \sum_{j \le i} \mathbb{P}[x_j < X \le x_j + dx]$. Connecting this to the probability density function, $f(x)$, we have

$$\mathbb{P}[x_j < X \le x_j + dx] \approx f(x_j)dx$$

for "small" dx, and

$$F(x_i) = \mathbb{P}[X \le x_i] \approx \sum_{j \le i} f(x_j)dx.$$

Calculus comes in when we see the expression above is a Riemann sum. Taking the limit as dx goes to zero yields the integral we see in (3).

2.2 Expectation and Variance

As in the discrete case, we define the expectation of a continuous r.v. as follows:

Definition 21.2 (Expectation)**.** *The expectation of a continuous r.v. X with probability density function f is*

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x f(x) dx.$$

Note that the integral plays the role of the summation in the discrete formula $\mathbb{E}[X] = \sum_a a\mathbb{P}[X = a]$. Similarly, we can define the variance as follows:

Definition 21.3 (Variance)**.** *The variance of a continuous r.v. X with probability density function f is*

$$\mathrm{Var}(X) = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 = \int_{-\infty}^{\infty} x^2 f(x) dx - \left(\int_{-\infty}^{\infty} x f(x) dx\right)^2.$$

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

$$\mathbb{E}[X] = \int_0^{\ell} x \frac{1}{\ell} dx = \frac{x^2}{2\ell}\bigg|_0^{\ell} = \frac{\ell}{2},$$

as claimed. We can also calculate its variance using the above definition and plugging in the value $\mathbb{E}[X] = \frac{\ell}{2}$ to get:

$$\mathrm{Var}(X) = \int_0^{\ell} x^2 \frac{1}{\ell} dx - \mathbb{E}[X]^2 = \frac{x^3}{3\ell}\bigg|_0^{\ell} - \left(\frac{\ell}{2}\right)^2 = \frac{\ell^2}{3} - \frac{\ell^2}{4} = \frac{\ell^2}{12}.$$

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.

2.3 Joint Distribution

Recall that for discrete random variables X and Y, their joint distribution is specified by the probabilities $\mathbb{P}[X = a, Y = c]$ for all possible values a, c. Similarly, if X and Y are continuous random variables, then their joint distribution is specified by the probabilities $\mathbb{P}[a \le X \le b, c \le Y \le d]$ 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 21.4 (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:

$$\mathbb{P}[a \le X \le b,\ c \le Y \le d] = \int_c^d \int_a^b f(x, y)\, dx\, dy \qquad \text{for all } a \le b \text{ and } c \le d.$$

In analogy with (2), we can connect the joint density $f(x,y)$ with probabilities by looking at a very small square $[x,x+dx] \times [y,y+dy]$ close to (x,y); then we have

$$\mathbb{P}[x \le X \le x+dx,\ y \le Y \le y+dy] = \int_y^{y+dy} \int_x^{x+dx} f(u,v)\,du\,dv \approx f(x,y)\,dx\,dy. \tag{4}$$

Thus we can interpret $f(x,y)$ as the "probability per unit area" in the vicinity of (x,y).

2.4 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 21.5 (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 $c \le d$:*

$$\mathbb{P}[a \le X \le b,\ c \le Y \le d] = \mathbb{P}[a \le X \le b] \cdot \mathbb{P}[c \le Y \le d].$$

What does this definition say about the joint density of independent r.v.'s X and Y? Applying (4) to connect the joint density with probabilities, we get, for small dx and dy:

$$\begin{aligned} f(x,y)\,dx\,dy &\approx \mathbb{P}[x \le X \le x+dx,\ y \le Y \le y+dy] \\ &= \mathbb{P}[x \le X \le x+dx] \cdot \mathbb{P}[y \le Y \le y+dy] \qquad \text{(by independence)} \\ &\approx f_X(x)dx \times f_Y(y)dy \\ &= f_X(x) f_Y(y)\,dx\,dy, \end{aligned}$$

where f_X and f_Y are the (marginal) densities of X and Y respectively. So we get the following result:

Theorem 21.1. *The joint density of independent r.v.'s X and Y is the product of the marginal densities:*

$$f(x,y) = f_X(x)\, f_Y(y) \quad \textit{for all} \quad x,y \in \mathbb{R}.$$

3 Exponential Distribution

We have already seen one important continuous distribution, namely the uniform distribution. In this and the next sections 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.

The exponential distribution is a continuous version of the geometric distribution, which we have already seen. 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.

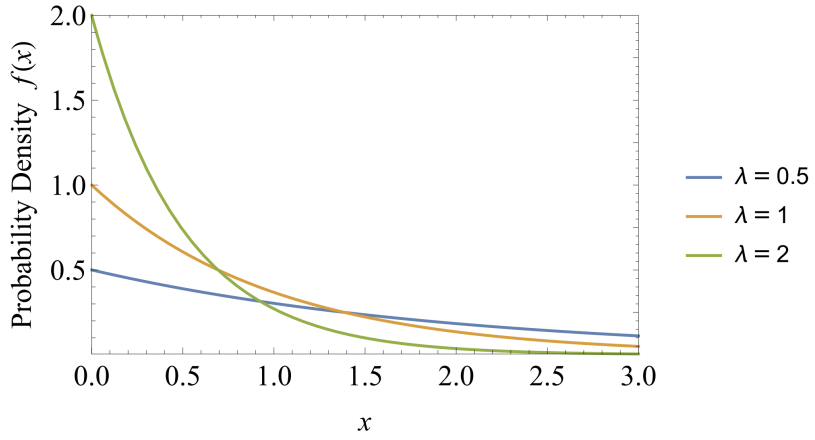


Figure 1: The probability density function $f(x)$ for the exponential distribution with $\lambda = 0.5, 1, 2$.

Definition 21.6 (Exponential Distribution)**.** *For $\lambda > 0$, a continuous random variable X with p.d.f.*

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{if } x \geq 0, \\ 0, & \text{otherwise}, \end{cases}$$

is called an exponential random variable with parameter λ, and we write $X \sim Exp(\lambda)$.

Note that by definition $f(x)$ is non-negative. Moreover, we can check that it satisfies (1):

$$\int_{-\infty}^{\infty} f(x)dx = \int_0^{\infty} \lambda e^{-\lambda x} dx = -e^{-\lambda x}\Big|_0^{\infty} = 1,$$

so $f(x)$ is indeed a valid probability density function. Figure 1 shows the probability density function for the exponential distribution with a few different values of λ.

3.1 Mean and Variance of an Exponential Random Variable

Let us now compute the expectation and variance of $X \sim \text{Exp}(\lambda)$.

Theorem 21.2. *Let X be an exponential random variable with parameter $\lambda > 0$. Then*

$$\mathbb{E}[X] = \frac{1}{\lambda} \qquad \text{and} \qquad \text{Var}(X) = \frac{1}{\lambda^2}.$$

Proof. We can calculate the expected value using integration by parts:

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} xf(x)dx = \int_0^{\infty} \lambda x e^{-\lambda x} dx = -xe^{-\lambda x}\Big|_0^{\infty} + \int_0^{\infty} e^{-\lambda x} dx = 0 + \left(-\frac{e^{-\lambda x}}{\lambda}\right)\Big|_0^{\infty} = \frac{1}{\lambda}.$$

To compute the variance, we first evaluate $\mathbb{E}\left[X^2\right]$, again using integration by parts:

$$ExX^2 = \int_{-\infty}^{\infty} x^2 f(x)dx = \int_0^{\infty} \lambda x^2 e^{-\lambda x} dx = -x^2 e^{-\lambda x}\Big|_0^{\infty} + \int_0^{\infty} 2xe^{-\lambda x} dx = 0 + \frac{2}{\lambda}\mathbb{E}[X] = \frac{2}{\lambda^2}.$$

The variance is therefore

$$\text{Var}(X) = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 = \frac{2}{\lambda^2} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2},$$

as claimed. □

3.2 As 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 $t \geq 0$,

$$\mathbb{P}[X > t] = \int_t^\infty \lambda e^{-\lambda x} dx = -e^{-\lambda x}\Big|_t^\infty = e^{-\lambda t}. \tag{5}$$

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

$$\mathbb{P}[Y > k\delta] = (1-p)^k = (1-\lambda\delta)^k, \qquad \text{for any integer } k \geq 0.$$

Hence, for any $t > 0$, we have

$$\mathbb{P}[Y > t] = \mathbb{P}[Y > (\tfrac{t}{\delta})\delta] = (1-\lambda\delta)^{t/\delta} \approx e^{-\lambda t}, \tag{6}$$

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 (6) with (5), we see that this distribution has the same form as the exponential distribution with parameter λ. Thus we may view the exponential distribution as a continuous time analog of the geometric distribution, where the parameter p (success probability per trial) is replaced by the parameter λ (success probability per unit time). Note, though, that λ is not constrained to be ≤ 1.

4 Normal Distribution

Another continuous distribution we will consider, and by far the most prevalent in applications, is called the *normal* or *Gaussian* distribution. It has two parameters, μ and σ^2, which are the mean and variance of the distribution, respectively.

Definition 21.7 (Normal Distribution)**.** *For any $\mu \in \mathbb{R}$ and $\sigma > 0$, a continuous random variable X with p.d.f.*

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}$$

is called a normal random variable with parameters μ and σ^2, and we write $X \sim N(\mu, \sigma^2)$. 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):

$$\int_{-\infty}^{\infty} f(x)dx = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} e^{-(x-\mu)^2/(2\sigma^2)} dx = 1. \tag{7}$$

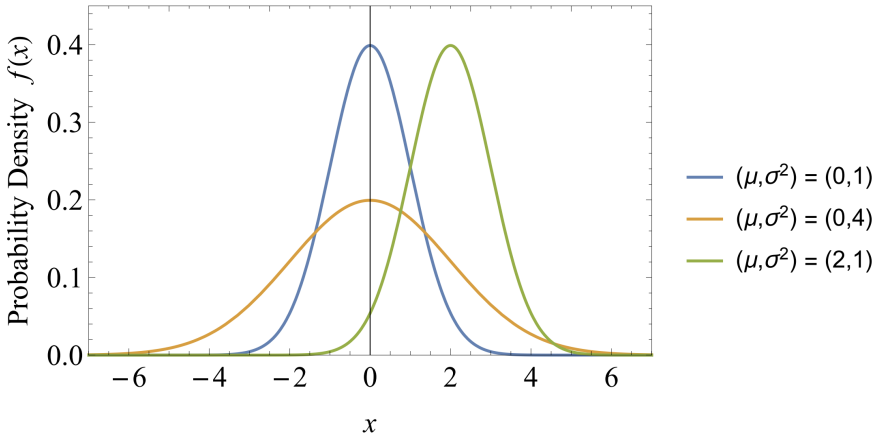


Figure 2: The density function for the normal distribution with several different choices for μ and σ^2.

The fact that this integral evaluates to 1 is a standard exercise in (mutivariable) integral calculus (or feel free to look it up in any standard text on probability or on the internet).

A plot of the p.d.f. f reveals a classical "bell-shaped" curve, centered at (and symmetric around) $x = \mu$, and with "width" determined by σ. Figure 2 shows that the normal densities 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 21.1. *If* $X \sim N(\mu, \sigma^2)$, *then* $Y = \frac{X-\mu}{\sigma} \sim N(0,1)$. *Equivalently, if* $Y \sim N(0,1)$, *then* $X = \sigma Y + \mu \sim N(\mu, \sigma^2)$.

Proof. Given that $X \sim N(\mu, \sigma^2)$, we can calculate the distribution of $Y = \frac{X-\mu}{\sigma}$ as:

$$\mathbb{P}[a \le Y \le b] = \mathbb{P}[\sigma a + \mu \le X \le \sigma b + \mu] = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{\sigma a+\mu}^{\sigma b+\mu} e^{-(x-\mu)^2/(2\sigma^2)} dx = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-y^2/2} dy,$$

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. □

4.1 Mean and Variance of a Normal Random Variable

Let us now calculate the expectation and variance of a normal random variable.

Theorem 21.3. *For* $X \sim N(\mu, \sigma^2)$,

$$\mathbb{E}[X] = \mu \qquad \textit{and} \qquad \mathrm{Var}(X) = \sigma^2.$$

Proof. First consider the case when $X \sim N(0,1)$. By definition, its expectation is

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x f(x) dx = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} x e^{-x^2/2} dx = \frac{1}{\sqrt{2\pi}} \left(\int_{-\infty}^{0} x e^{-x^2/2} dx + \int_0^{\infty} x e^{-x^2/2} dx \right) = 0.$$

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

$$\begin{aligned}\mathrm{Var}(X) = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2 &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} x^2 e^{-x^2/2}dx \\ &= \frac{1}{\sqrt{2\pi}}\left(-xe^{-x^2/2}\right)\Big|_{-\infty}^{\infty} + \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{-x^2/2}dx \\ &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{-x^2/2}dx = 1.\end{aligned}$$

In the first line here we used the fact that $\mathbb{E}[X] = 0$; in the second line we used integration by parts; and in the last line we used (7) in the special case $\mu = 0$, $\sigma = 1$. So the standard normal distribution has expectation $\mathbb{E}[X] = 0 = \mu$ and variance $\mathrm{Var}(X) = 1 = \sigma^2$.

Now consider the general case when $X \sim N(\mu, \sigma^2)$. By Lemma 21.1, we know that $Y = \frac{X-\mu}{\sigma}$ is a standard normal random variable, so $\mathbb{E}[Y] = 0$ 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

$$0 = \mathbb{E}[Y] = \mathbb{E}\left[\frac{X-\mu}{\sigma}\right] = \frac{\mathbb{E}[X]-\mu}{\sigma},$$

and hence $\mathbb{E}[X] = \mu$. 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^2.

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. $X \sim N(\mu, \sigma^2)$ from the formula

$$\mathbb{P}[X \le a] = \mathbb{P}[Y \le \tfrac{a-\mu}{\sigma}],$$

where Y is standard normal.

The normal distribution is ubiquitous throughout the sciences and the social sciences, because it is the standard model for aggregate data that results from averaging a large number of independent observations of the same random variable (such as the weights of mosquitos in Berkeley, or the outcome of a physical experiment). Such averaged 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 discuss in Section 5.

4.2 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 21.1.

Theorem 21.4. *Let $X \sim N(0,1)$ and $Y \sim N(0,1)$ be independent standard normal random variables, and suppose $a, b \in \mathbb{R}$ are constants. Then $Z = aX + bY \sim N(0, a^2 + b^2)$.*

Proof. [4] Since X and Y are independent, by Theorem 21.1 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.

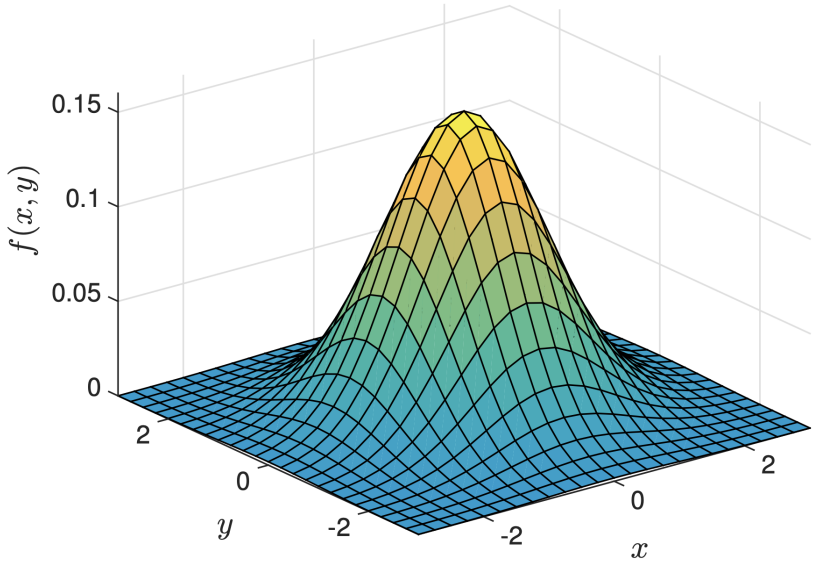


Figure 3: The joint density function $f(x,y) = \frac{1}{2\pi} e^{-(x^2+y^2)/2}$ is rotationally symmetric.

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 $A \subseteq \mathbb{R}^2$,

$$\mathbb{P}[(X,Y) \in A] = \mathbb{P}[(X,Y) \in T(A)] \tag{8}$$

where T is a rotation of $\mathbb{R}^2$. Now given any $t \in \mathbb{R}$, we have

$$\mathbb{P}[Z \le t] \;=\; \mathbb{P}[aX + bY \le t] \;=\; \mathbb{P}[(X,Y) \in A]$$

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

$$T(A) = \left\{ (x,y) \,\middle|\, x \le \frac{t}{\sqrt{a^2+b^2}} \right\}.$$

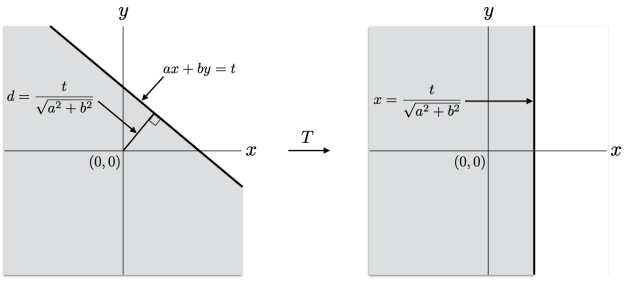


Figure 4: The half plane $ax+by \le t$ is rotated into the half plane $x \le \frac{t}{\sqrt{a^2+b^2}}$.

By (8), this rotation does not change the probability:

$$\mathbb{P}[Z \le t] \;=\; \mathbb{P}[(X,Y) \in A] \;=\; \mathbb{P}[(X,Y) \in T(A)] \;=\; \mathbb{P}\left[X \le \frac{t}{\sqrt{a^2+b^2}}\right] \;=\; \mathbb{P}\left[\sqrt{a^2+b^2}\,X \le t\right].$$

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 21.1 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 21.1 and Theorem 21.4.

Corollary 21.1. *Let $X \sim N(\mu_X, \sigma_X^2)$ and $Y \sim N(\mu_Y, \sigma_Y^2)$ be independent normal random variables. 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 21.1, $Z_1 = (X-\mu_X)/\sigma_X$ 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 21.4, $Z' = a\sigma_X Z_1 + b\sigma_Y Z_2$ 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 21.1 we conclude that $Z = \mu + Z'$ is a normal random variable with mean μ and variance σ^2, as desired. □

5 The Central Limit Theorem

Recall from an earlier note the Law of Large Numbers for i.i.d. random variables $\{X_i\}$: it says that the probability of *any* deviation $\varepsilon > 0$, however small, of the sample average $\frac{S_n}{n}$, where $S_n = \sum_{i=1}^n X_i$, from the mean 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 $\frac{S_n}{n}$, 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 $\frac{S_n}{n}$; 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.

[4]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.

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 standardize $\frac{S_n}{n}$ as

$$\frac{\frac{S_n}{n} - \mu}{\frac{\sigma}{\sqrt{n}}} = \frac{S_n - n\mu}{\sigma\sqrt{n}}.$$

The Central Limit Theorem then says that the distribution of $\frac{S_n - n\mu}{\sigma\sqrt{n}}$ converges to the *standard normal* distribution.

Theorem 21.5 (**Central Limit Theorem**). *Let $X_1, X_2, \ldots$ be a sequence of i.i.d. random variables with common finite expectation $\mathbb{E}[X_i] = \mu$ and finite variance $\mathrm{Var}(X_i) = \sigma^2$. Let $S_n = \sum_{i=1}^n X_i$. Then, the distribution of $\frac{S_n - n\mu}{\sigma\sqrt{n}}$ converges to $N(0,1)$ as $n \to \infty$. In other words, for any constant $c \in \mathbb{R}$,*

$$\mathbb{P}\left[\frac{S_n - n\mu}{\sigma\sqrt{n}} \leq c\right] \to \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{c} e^{-x^2/2} dx \quad \text{as } n \to \infty.$$

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 $\mu = \mathbb{E}[X]$. Thus all trace of the distribution of X disappears as n gets large: all distributions, no matter how complex,[5] 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.

The Central Limit Theorem immediately tells us that *averaged* data tends to look normal, even when the distribution it is drawn from is not itself normal. Since much of the data we work with in real life is the result of averaging (mean wealth, mean height, mean temperature etc.), it's not surprising that we deal with the normal distribution so often. But there's actually a deeper reason why even (say) the height of a population itself tends to follow a normal distribution: this is because a person's height is actually itself the result of averaging many factors (nutrition, environment, various genetic factors, etc.), so intuitively at least we would expect it to follow a normal distribution.

6 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; this is known as *Buffon's needle* problem, after its 18th century inventor Georges-Louis Leclerc, Comte de Buffon.

As illustrated in Figure 5, 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 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

[5] We do need to assume that the mean and variance of X are finite.

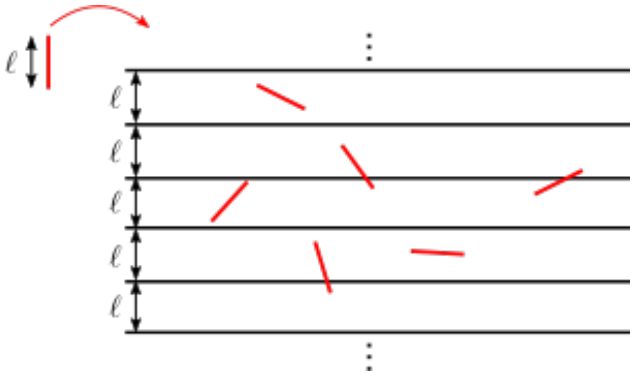


Figure 5: Buffon's Needle.

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 an earlier note to determine how many throws we need in order to achieve specified accuracy and confidence.

6.1 Integrating a Joint Density Function

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

$$f(y,\theta)=\begin{cases}\frac{2}{\pi\ell}, & \text{for } (y,\theta)\in[0,\ell/2]\times[-\pi/2,\pi/2],\\ 0, & \text{otherwise.}\end{cases} \tag{9}$$

Equivalently, Y and Θ are independent random variables, each uniformly distributed in their respective range. To check our answer, let's verify that the integral of this density over all possible values is indeed 1:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(y,\theta)dyd\theta=\int_{-\pi/2}^{\pi/2}\int_{0}^{\ell/2}\frac{2}{\pi\ell}dyd\theta=\int_{-\pi/2}^{\pi/2}\frac{2y}{\pi\ell}\Big|_{0}^{\ell/2}d\theta=\int_{-\pi/2}^{\pi/2}\frac{1}{\pi}d\theta=\frac{\theta}{\pi}\Big|_{-\pi/2}^{\pi/2}=1.$$

This is an analog of (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

$$\mathbb{P}[E]=\mathbb{P}[Y\le\tfrac{\ell}{2}\cos\Theta]=\int_{-\pi/2}^{\pi/2}\int_{0}^{(\ell/2)\cos\theta}f(y,\theta)dyd\theta.$$

Substituting the density $f(y,\theta)$ from (9) and performing the integration we get

$$\mathbb{P}[E]=\int_{-\pi/2}^{\pi/2}\int_{0}^{(\ell/2)\cos\theta}\frac{2}{\pi\ell}dyd\theta=\int_{-\pi/2}^{\pi/2}\frac{2y}{\pi\ell}\Big|_{0}^{(\ell/2)\cos\theta}d\theta=\frac{1}{\pi}\int_{-\pi/2}^{\pi/2}\cos\theta d\theta=\frac{1}{\pi}\sin\theta\Big|_{-\pi/2}^{\pi/2}=\frac{2}{\pi}.$$

This is exactly what we claimed at the beginning of the section!

Finite Markov Chains

This note gives a brief introduction to the theory of finite Markov chains, omitting some proofs. This topic has many applications and will be revisited in several higher-level courses.

1 Introduction

Markov chains are models of random motion in a finite or countable set. These models are powerful because they capture a vast array of systems that we encounter in applications. Yet, the models are simple in that many of their properties can often be determined using elementary matrix algebra. In this course, we limit the discussion to the case of finite Markov chains, i.e., motions in a finite set.

Imagine the following scenario. You flip a fair coin until you get two consecutive 'heads'. How many times do you have to flip the coin, on average? You roll a balanced six-sided die until the sum of the last two rolls is 8. How many times do you have to roll the die, on average?

As another example, say that you play a game of 'heads or tails' using a biased coin that yields 'heads' with probability 0.48. You start with \$10. At each step, if the flip yields 'heads', you earn \$1. Otherwise, you lose \$1. What is the probability that you reach \$100 before \$0? How long does it take until you reach either \$100 or \$0?

You try to go up a ladder that has 20 rungs. At each time step, you succeed in going up by one rung with probability 0.9. Otherwise, you fall back to the ground. How many time steps does it take you to reach the top of the ladder, on average?

You look at a web page, then select randomly one of the links on that page, with equal probabilities. You then repeat on the next page you visit, and so on. As you keep browsing the web in this way, what fraction of the time do you open a given page? How long does it take until you reach a particular page? How likely is it that you visit a given page before another given page?

These questions can all be answered using the methods of Markov chains, as we explain in this note.

2 A First Example

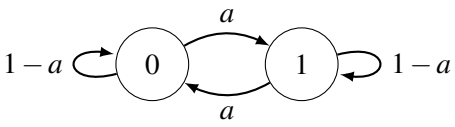


Figure 1: A simple Markov chain

Figure 1 illustrates a simple Markov chain. It describes a random motion in the set $\{0,1\}$. The position at time $n = 0,1,2,\ldots$ is $X_n \in \{0,1\}$. We call X_n the *state* of the Markov chain at step (or time) n. The set

$\{0,1\}$ is the *state space*, i.e., the set of possible values of the state. The motion, i.e., the time evolution, of X_n follows the following rules. One is given a number $a \in [0,1]$ and two nonnegative numbers $\pi_0(0)$ and $\pi_0(1)$ that add up to 1. Then,

$$\mathbb{P}[X_0 = 0] = \pi_0(0) \quad \text{and} \quad \mathbb{P}[X_0 = 1] = \pi_0(1). \tag{1}$$

That is, the *initial* state X_0 is equal to 0 with probability $\pi_0(0)$, otherwise it is 1. Then for $n \geq 0$,

$$\mathbb{P}[X_{n+1} = 0 \mid X_n = 0, X_{n-1}, \ldots, X_0] = 1 - a \tag{2}$$
$$\mathbb{P}[X_{n+1} = 1 \mid X_n = 0, X_{n-1}, \ldots, X_0] = a \tag{3}$$
$$\mathbb{P}[X_{n+1} = 0 \mid X_n = 1, X_{n-1}, \ldots, X_0] = a \tag{4}$$
$$\mathbb{P}[X_{n+1} = 1 \mid X_n = 1, X_{n-1}, \ldots, X_0] = 1 - a \tag{5}$$

Figure 1 summarizes the rules (2)–(5). These rules specify the *transition probabilities* of the Markov chain. Rules (2)–(3) specify that if the Markov chain is in state 0 at step n, then at the next step it stays in state 0 with probability $1-a$ and it moves to state 1 with probability a, regardless of what happened in the previous steps. Rules (4)-(5) are similar. Figure 1 is called the *state transition diagram* of the Markov chain. It captures the transition probabilities in a graphical form.

The key property of a Markov chain is that it is *amnesic*: at step n, it forgets what it did before getting to the current state and its future steps depend only on that current state. Here is one way to think of the rules of motion. When the Markov chain gets to state 0, it flips a coin with heads probability a. If the outcome is H then it goes to state 1; otherwise, it stays in state 0 and flips the coin again. The situation is similar when the Markov chain gets to state 1.

An equivalent, more compact way of describing a Markov chain is its *transition probability matrix* P, which for the above chain is given by

$$P(0,0) = 1-a; \qquad P(0,1) = a;$$
$$P(1,0) = a; \qquad P(1,1) = 1-a.$$

That is,

$$P = \begin{bmatrix} 1-a & a \\ a & 1-a \end{bmatrix}.$$

Hence,

$$\mathbb{P}[X_{n+1} = j \mid X_n = i,\ X_{n-1}, \ldots,\ X_0] = P(i,j), \quad \text{for } n \geq 0 \text{ and } i, j \in \{0,1\}.$$

Figure 2 shows some simulations of the Markov chain with different values of a. When $a = 0.1$, it is unlikely that the state of the Markov chain changes in one step. As the figure shows, the Markov chain spends many steps in one state before switching. For larger values of a, the state of the Markov chain changes more frequently. Note that, by symmetry, over the long term the Markov chain spends half of the time in each state. (More about this phenomenon later.)

3 A Second Example

Figure 3 shows the state transition diagram of a small web browsing experiment. Each state in the figure represents a web page. The arrows out of a state correspond to links on the page that point to other pages. The transition probabilities are not indicated on the figure, but the model is that each outgoing link from a

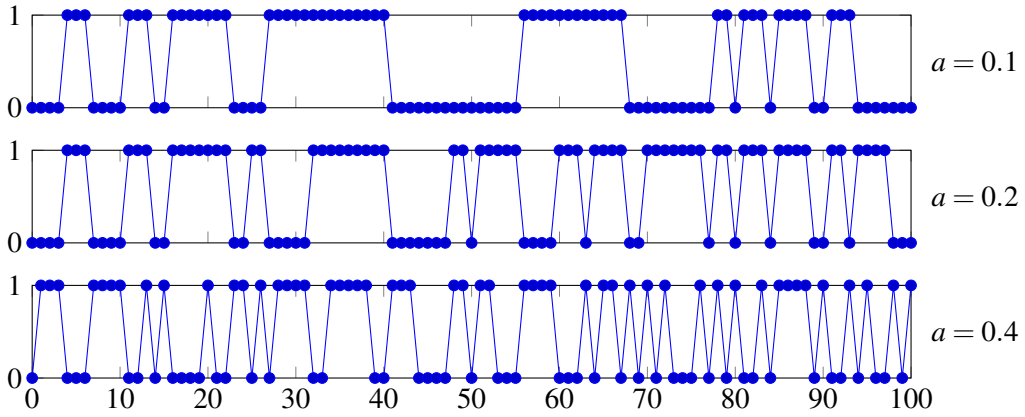


Figure 2: Simulations of the two-state Markov chain

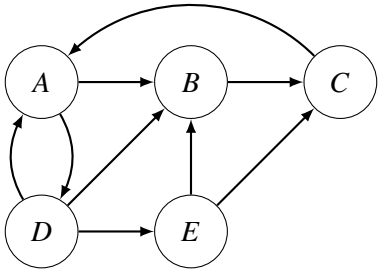


Figure 3: A five-state Markov chain. The outgoing arrows are equally likely.

particular page is equally likely. The figure corresponds to the following probability transition matrix, with rows/columns indexed by states A, B, C, D, E, respectively:

$$P = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \end{bmatrix}.$$

Figure 4 shows a simulation of the five-state Markov chain.

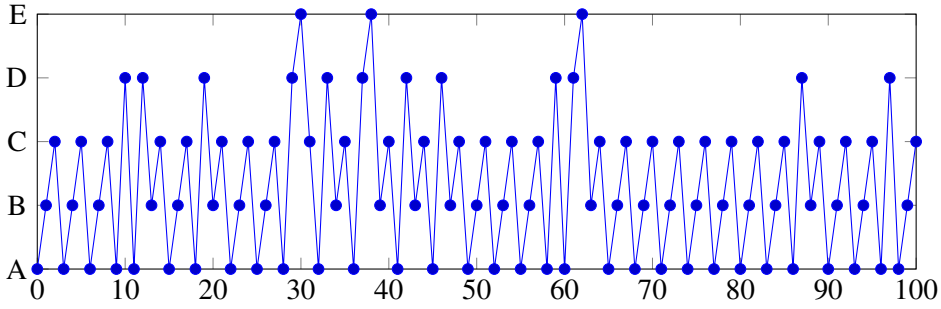


Figure 4: Simulation of the five-state Markov chain.

4 Finite Markov Chains

We define a general finite Markov chain as follows. The *state space* is $\mathscr{X} = \{1, 2, \ldots, K\}$ for some finite K. The *transition probability matrix* P is a $K \times K$ matrix such that

$$P(i,j) \geq 0, \quad \forall i, j \in \mathscr{X}$$

and

$$\sum_{j=1}^{K} P(i,j) = 1, \quad \forall i \in \mathscr{X}.$$

The *initial distribution* is a vector $\pi_0 = \{\pi_0(i) \mid i \in \mathscr{X}\}$ where $\pi_0(i) \geq 0$ for all $i \in \mathscr{X}$ and $\sum_{i \in \mathscr{X}} \pi_0(i) = 1$. (In many applications, π_0 is concentrated on a single state, i.e., $\pi_0(i) = 1$ for some i, which corresponds to starting deterministically in state i.)

One then defines the random sequence $\{X_n \mid n = 0, 1, 2, \ldots\}$ by

$$\mathbb{P}[X_0 = i] = \pi_0(i), \quad \forall i \in \mathscr{X};$$
$$\mathbb{P}[X_{n+1} = j \mid X_n = i, X_{n-1}, \ldots, X_0] = P(i,j), \quad \forall n \geq 0, \forall i, j \in \mathscr{X}.$$

Note that

$$\begin{aligned}
&\mathbb{P}[X_0 = i_0, X_1 = i_1, \ldots, X_n = i_n] \\
&= \mathbb{P}[X_0 = i_0]\mathbb{P}[X_1 = i_1 \mid X_0 = i_0]\mathbb{P}[X_2 = i_2 \mid X_0 = i_0, X_1 = i_1] \cdots \mathbb{P}[X_n = i_n \mid X_0 = i_0, \ldots, X_{n-1} = i_{n-1}] \\
&= \pi_0(i_0) P(i_0, i_1) \cdots P(i_{n-1}, i_n).
\end{aligned}$$

Consequently,

$$\begin{aligned}
\mathbb{P}[X_n = i_n] &= \sum_{i_0, \ldots i_{n-1}} \mathbb{P}[X_0 = i_0, X_1 = i_1, \ldots, X_n = i_n] \\
&= \sum_{i_0, \ldots i_{n-1}} \pi_0(i_0) P(i_0, i_1) \cdots P(i_{n-1}, i_n) \\
&= [\pi_0 P^n](i_n),
\end{aligned}$$

where the last expression denotes the i_n component of the product of the row vector π_0 times the nth power of the matrix P.

Thus, if we designate by π_n the distribution of X_n, so that $\mathbb{P}[X_n = i] = \pi_n(i)$, then the last derivation proves the following result.

Theorem 22.1. *For all $n \geq 0$, one has*

$$\pi_n = \pi_0 P^n.$$

In particular, if $\pi_0(i) = 1$ for some i, then $\pi_n(j) = P^n(i,j) = \mathbb{P}[X_n = j \mid X_0 = i]$.

For the two-state Markov chain in Figure 1, one can verify that (see Appendix for details)

$$P^n = \begin{bmatrix} 1-a & a \\ a & 1-a \end{bmatrix}^n = \begin{bmatrix} \frac{1}{2} + \frac{1}{2}(1-2a)^n & \frac{1}{2} - \frac{1}{2}(1-2a)^n \\ \frac{1}{2} - \frac{1}{2}(1-2a)^n & \frac{1}{2} + \frac{1}{2}(1-2a)^n \end{bmatrix}. \tag{6}$$

Note that if $0 < a < 1$,

$$P^n \to \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} \quad \text{as } n \to \infty.$$

Consequently, for $0 < a < 1$, one has $\pi_n = \pi_0 P^n \to \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$ as $n \to \infty$, which means that after a long time the chain is equally likely to be in either of the two states.

5 Invariant Distribution

The following definition introduces the important notion of invariant distribution.

Definition 22.1. *A distribution π is* invariant *for the transition probability matrix P if it satisfies the following* balance equations*:*

$$\pi = \pi P. \tag{7}$$

The relevance of this definition is stated in the next result.

Theorem 22.2. *One has $\pi_n = \pi_0$ for all $n \geq 0$ if and only if π_0 is invariant.*

Proof. If $\pi_n = \pi_0$ for all $n \geq 0$, then $\pi_0 = \pi_1 = \pi_0 P$, so that π_0 satisfies (7) and is thus invariant.

If $\pi_0 P = \pi_0$, then $\pi_1 = \pi_0 P = \pi_0$. And similarly, by induction, we get that $\pi_n = \pi_{n-1} P = \pi_0 P = \pi_0$. □

For instance, in the case of the two-state Markov chain, the balance equations are

$$\begin{aligned}\pi(0) &= \pi(0)(1-a) + \pi(1)a;\\ \pi(1) &= \pi(0)a + \pi(1)(1-a).\end{aligned}$$

Each of these two equations is equivalent to

$$\pi(0) = \pi(1).$$

Thus, the two equations are redundant. If we add the condition that the components of π add up to one, we find that the only solution is $\begin{bmatrix}\pi(0) & \pi(1)\end{bmatrix} = \begin{bmatrix}\frac{1}{2} & \frac{1}{2}\end{bmatrix}$, which is not surprising in view of symmetry.

For the five-state Markov chain, the balance equations are

$$\begin{bmatrix}\pi(A) & \pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} = \begin{bmatrix}\pi(A) & \pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \end{bmatrix}.$$

Once again, these five equations in the five unknowns are redundant: they do not determine π uniquely. However, if we add the condition that the components of π add up to one, then we find that the solution is unique and given by (see the Appendix for the calculations):

$$\begin{bmatrix}\pi(A) & \pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} = \frac{1}{39}\begin{bmatrix}12 & 9 & 10 & 6 & 2\end{bmatrix}. \tag{8}$$

Thus, in this web-browsing example, in the long term page A is visited most often, then page C, then page B. A Google search would return the pages in order of most frequent visits, i.e., in the order A, C, B, D, E. This ranking of the pages is called *PageRank* and can be determined by solving the balance equations. (In fact, the actual ranking by Google combines the estimate of π with other factors.)

How many invariant distributions does a Markov chain have? We have seen that for the two examples, the answer was one. However, that is not always the case. For instance, consider the two-state Markov chain with $a = 0$ instead of $0 < a < 1$ as we assumed previously. This Markov chain does not change state. Its transition probability matrix is $P = I$ where I denotes the identity matrix. Since $\pi I = \pi$ for any vector π, we see that *any* distribution is invariant for this Markov chain.

However, we will see in the next section that two simple conditions guarantee the uniqueness of the invariant distribution.

6 Convergence to the Invariant Distribution

In this section, we will state (without proof) the so-called Fundamental Theorem of Markov Chains, which says that, under mild conditions, a finite Markov chain will always converge, as time tends to infinity, to a unique invariant distribution, regardless of its initial state. This implies that, in the long term, the probability of finding the chain in any given state has a certain fixed value that doesn't change over time.

To guarantee this nice behavior, we need two conditions. The first is known as *irreducibility*.

Definition 22.2 (Irreducible)**.** *A Markov chain is* irreducible *if it can go from every state i to every other state j in a finite number of steps.*

We can picture this condition graphically as follows. For a Markov chain to be irreducible, its state transition diagram must be "strongly connected", i.e., there must exist a directed path of transitions (each with non-zero probability) from every state i to every state j. For example, the two-state Markov chain in Figure 1 is irreducible for all $0 < a \leq 1$, but not for $a = 0$. And the Markov chain in Figure 3 is irreducible. Note that irreducibility is a necessary condition for convergence, since otherwise there will be states that are unreachable from some initial states, which will therefore never be visited. It turns out that irreducibility guarantees that a stationary distribution π with $\pi(i) > 0$ for all i not only exists but is also unique.

For the remainder of this section, since we are focusing on convergence, we will restrict our attention to irreducible Markov chains. In later sections, when we shift our focus to hitting times, we will naturally also consider non-irreducible chains that have absorbing states.

The second condition we need for convergence requires a short detour into the notion of *periodicity*.

Definition 22.3 (Period)**.** *For any state i in an irreducible finite Markov chain, the* period *of i is defined as*

$$d(i) := \gcd\{n > 0 : P^n(i,i) > 0\}.$$

Let's parse this definition. The set of values in the gcd expression are exactly the lengths of all possible paths in the Markov chain from state i back to itself: that's because the entry $P^n(i,j)$ is the probability that the chain goes from state i to state j in exactly n steps, so $P^n(i,i) > 0$ if and only if there is at least one length-n path from i back to itself. (Note that we're considering *all* paths here, not just simple paths.) The value $d(i)$ determines whether there is any kind of periodic behavior in these path lengths: if $d(i) > 1$ then the return visits to i, starting from i, can only occur at intervals of at least $d(i)$ steps; on the other hand, if $d(i) = 1$ there is no such periodic behavior. A very simple example of a situation where $d(i) > 1$ is the 2-state example in Figure 1 when $a = 1$. In this case the chain just bounces back and forth between the two states (the self-loops have probability 0 so should be removed from the graph), so all paths from state 0 back to itself have even length and $d(i) = 2$.

An important fact about periods is that, in an irreducible Markov chain, they are the same for all states!

Proposition 22.1. *In an irreducible finite Markov chain, $d(i) = d(j)$ for all pairs of states i, j.*

Proof. Since the chain is irreducible, there is a path $\mathscr{P}_{ij}$ from i to j and a path $\mathscr{P}_{ji}$ from j to i. Call the lengths of these paths ℓ_{ij} and ℓ_{ji}, respectively.

Now consider the following path from i back to itself: take path $\mathscr{P}_{ij}$ to j, then *any* path (of length k, say) from j back to itself, and finally path $\mathscr{P}_{ji}$ back to i. (Note that paths from j back to itself certainly exist, since e.g. $\mathscr{P}_{ji}$ followed by $\mathscr{P}_{ij}$ is one such path.) The total length of this path is $\ell_{ij} + \ell_{ji} + k$. The following two observations follow from Definition 22.3:

- $d(i) \mid (\ell_{ij} + \ell_{ji} + k)$ (since this is the length of the above path from i back to itself);
- $d(i) \mid (\ell_{ij} + \ell_{ji})$ (since $\mathscr{P}_{ij}$ followed by $\mathscr{P}_{ji}$ is also a path from i back to itself.

Putting these together immediately implies that $d(i) \mid k$. But since k was the length of an *arbitrary* path from j back to itself, $d(i)$ must in fact divide *all* such path lengths, and therefore $d(i) \mid d(j)$ (since $d(j)$ is the gcd of all such path lengths).

By a symmetrical argument, we also see that $d(j) \mid d(i)$. But if $d(i) \mid d(j)$ and $d(j) \mid d(i)$, then we must have $d(i) = d(j)$, as claimed. □

We are now ready to define the second property required for convergence.

Definition 22.4 (Aperiodic)**.** *An finite irreducible Markov chain is* aperiodic *if the period $d(i) = 1$ for all states i.*

For an example of an aperiodic Markov chain, consider Figure 3. Note that there are paths of length 2 and 3 from A back to itself, so $d(A) = 1$. Since the chain is irreducible, this implies by Proposition 22.1 that all states have period 1, so the chain is aperiodic. Another useful fact to bear in mind is that if an irreducible Markov chain contains a self-loop on *any* state, then it is aperiodic. (Why?) This applies, for example, to the chain in Figure 1 for any $a \in (0,1)$.

When a chain is periodic, it can't converge to an invariant distribution from a given starting state, since the probability of being at that state will oscillate between zero and non-zero values according to the period. You can see this in the very simple example of Figure 1 when $a = 1$. However, it turns out that this is the *only* obstacle to convergence for an irreducible chain, as we now state.

Theorem 22.3 (Fundamental Theorem of Markov Chains)**.** *For any finite, irreducible, aperiodic Markov chain, the probability distribution at time n for any initial state X_0 converges as $n \to \infty$ to π, where π is the unique invariant distribution and $\pi(i) > 0$ for all states i. In other words, for any X_0 and any state i, $\mathbb{P}[X_n = i] \to \pi(i)$ as $n \to \infty$.*

There are several alternative proofs of this theorem, all of which are beyond the scope of the course. However, we will briefly discuss some sample applications.

First, recall that the chain in Figure 1 with $0 < a < 1$ is aperiodic and irreducible with invariant distribution $\begin{bmatrix}\frac{1}{2} & \frac{1}{2}\end{bmatrix}$. Thus $\mathbb{P}[X_n = 0] \to \frac{1}{2}$ as $n \to \infty$, and similarly for $\mathbb{P}[X_n = 1]$, regardless of where we start. Second, we have seen that the chain in Figure 3 is aperiodic and irreducible with invariant distribution $\frac{1}{39}\begin{bmatrix}12 & 9 & 10 & 6 & 2\end{bmatrix}$. Thus, e.g., as $n \to \infty$, $\mathbb{P}[X_n = D] \to \frac{6}{39}$.

For a more interesting example, consider the problem of shuffling a deck of 52 cards. Here our goal is to construct a *uniform* distribution over all 52! permutations of the deck (a huge number, around 8×10^{67}). How do we achieve this feat? We can model the shuffling process as a Markov chain whose states are these 52! permutations, and whose transitions correspond to the kind of "riffle shuffles" that card players perform. To model real-life shuffling, we would need a mathematical model of these riffles, which does exist but is a little tricky to describe. Instead, we'll consider a "slow" shuffle in which, at each step, we pick two random cards (with replacement) in the current deck and switch their positions. We claim that this shuffle is aperiodic and irreducible and that its invariant distribution is uniform.

The fact that it's irreducible follows from the standard fact that *any* permutation can be written as a sequence of transpositions (i.e., switches of two cards); thus we can get from any permutation to any other one via a sequence of our simple card switching operations. To see that it's aperiodic, note that $P(i,i) > 0$ for any

state i (because we could pick the same card, meaning that nothing happens); this means that $\gcd\{n > 0 : P^n(i,i) > 0\} = 1$ for all i, so the chain is aperiodic. Finally, to see that the invariant distribution is uniform, we note that the transition matrix P is *symmetric*, i.e., $P(i,j) = P(j,i)$ for all i, j; this follows because either the permutations i, j differ by the transposition of just two cards, c, c' (say), in which case $P(i,j) = P(j,i) = \frac{2}{52^2}$ because both transitions are effected by picking cards c and c' (in either order); or they don't, in which case $P(i,j) = P(j,i) = 0$. Now *any* irreducible, aperiodic Markov chain that is symmetric has uniform invariant distribution, as we can easily check from the balance equations, as follows. We need to verify that the uniform distribution $\pi(i) = \frac{1}{N}$ satisfies the balance equations $\pi(j) = [\pi P](j)$, where in this case $N = 52!$. But

$$[\pi P](j) = \sum_i \pi(i)P(i,j) = \sum_i \frac{1}{N}P(j,i) = \frac{1}{N} = \pi(j),$$

where in the second equality we used the symmetry of P and in the third equality we used the fact that $\sum_i P(j,i) = 1$.

Putting all this together gives us the following perhaps surprising conclusion: if you start from any ordering of the deck, and perform enough random switches of pairs of cards, then you will eventually reach an (almost) perfectly shuffled deck! The same holds for mathematical models of the shuffles used in casinos (namely, the "riffle" or "dovetail" shuffle), which reach the uniform distribution after fewer steps. (For professional dealers, it is generally accepted that 7 riffle shuffles achieve a deck that is shuffled well enough that even professional card players cannot exploit any remaining structure.)

7 Random Walk on an Undirected Graph

A class of Markov chains that arises very often in practice are known as random walks on graphs. Let $G = (V,E)$ be an undirected graph, as we have seen earlier in the class. Define a Markov chain as follows: the state space is V (the vertices of the graph), and at each step, if the process is at v, it moves to a neighbor u of v chosen uniformly at random.

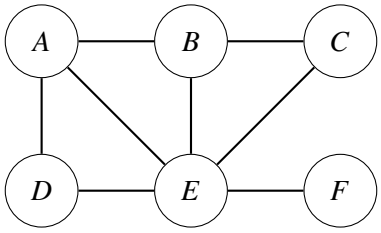


Figure 5: An undirected graph on six vertices. To interpret this as the transition diagram of a Markov chain, view each undirected edge as a pair of directed edges, one in each direction. Transition probabilities for all edges out of a given vertex are equal.

Figure 5 shows a simple example. For this graph, you should check that the transition matrix of the random walk is

$$P = \begin{bmatrix} 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{1}{5} \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

What can we say about random walk on a graph? First, it should be clear that the process is irreducible if and only if G is connected (since that corresponds precisely to the condition that there is a path from every vertex to every other vertex). Second, the random walk is aperiodic if and only if the graph is *not* bipartite. (You may like to prove this: the "only if" direction is easy, while the "if" direction follows from the fact that a non-bipartite graph must contain a cycle of odd length.) So, assuming G is connected and not bipartite, the random walk on G converges to a unique invariant distribution.

What is that distribution? We claim that it has a very nice form, namely

$$\pi(v) = \frac{\deg(v)}{D}, \tag{9}$$

where $\deg(v)$ denotes the degree of vertex v and $D = \sum_{v \in V} \deg(v)$. ($D$ here is just a normalizing factor to make the probabilities sum to 1. Note that in fact $D = 2|E|$.) To prove this we just have to verify that π as defined in (9) satisfies the balance equations $[\pi P](v) = \pi(v)$ for all v. This follows from

$$[\pi P](v) = \sum_{u:\{u,v\} \in E} \pi(u) P(u,v) = \sum_{u:\{u,v\} \in E} \frac{\deg(u)}{D} \times \frac{1}{\deg(u)} = \sum_{u:\{u,v\} \in E} \frac{1}{D} = \frac{\deg(v)}{D} = \pi(v).$$

In the second equality here, we used the fact that $P(u,v) = \frac{1}{\deg(u)}$ for all neighbors v of u.

Thus we see that, for random walk on a (connected, non-bipartite) graph, the probability of finding the walk in a given vertex v after many steps is proportional to $\deg(v)$. For the example graph in Figure 5, these proportions are

$$\pi(A) = \frac{3}{16}; \quad \pi(B) = \frac{3}{16}; \quad \pi(C) = \frac{2}{16}; \quad \pi(D) = \frac{2}{16}; \quad \pi(E) = \frac{5}{16}; \quad \pi(F) = \frac{1}{16}.$$

8 Hitting Time

Consider the Markov chain in Figure 3. Assume it starts in state A. What is the average number of steps until it reaches state E for the first time? To calculate this average time, for $i \in \{A,B,C,D,E\}$ define $\beta(i)$ to be the average time until the Markov chain reaches state E given that it starts from state i.

Thus, $\beta(E) = 0$ since it takes 0 steps to reach E when starting in state E. We want to calculate $\beta(A)$. However, it turns out that to calculate $\beta(A)$, one also has to calculate $\beta(B), \ldots, \beta(D)$. We do this by finding equations that these quantities satisfy and then solving these equations.

We claim that

$$\beta(A) = 1 + \frac{1}{2}\beta(B) + \frac{1}{2}\beta(D). \tag{10}$$

To see this, note that when the Markov chain starts in state A, it stays there for one step. Then, with probability $\frac{1}{2}$ it moves to state B. In that case, the average time until it reaches E is $\beta(B)$. With probability $1/2$, the Markov chain moves to state D and then takes $\beta(D)$ steps (on average) to reach E. Thus, the time to reach E starting from state A is 1 step plus an average of $\beta(B)$ steps with probability $\frac{1}{2}$ and an average of $\beta(D)$ steps with probability $\frac{1}{2}$. Equation (10) captures this decomposition.

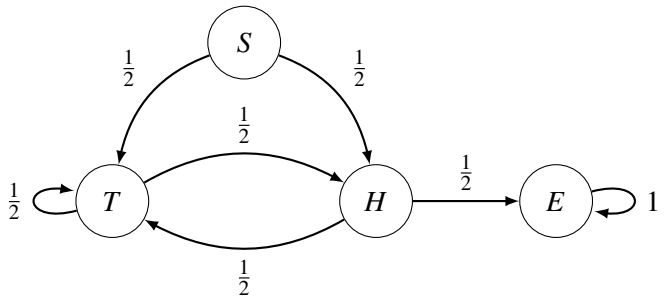


Figure 6: Flipping a fair coin until two heads in row.

An identity similar to (10) can be written for every starting state. We find

$$\begin{aligned}
\beta(A) &= 1 + \frac{1}{2}\beta(B) + \frac{1}{2}\beta(D) \\
\beta(B) &= 1 + \beta(C) \\
\beta(C) &= 1 + \beta(A) \\
\beta(D) &= 1 + \frac{1}{3}\beta(A) + \frac{1}{3}\beta(B) + \frac{1}{3}\beta(E) \\
\beta(E) &= 0
\end{aligned}$$

These equations are called the *first step equations*.

Solving these equations, we find (see the Appendix for the calculations):

$$\beta(A) = 17 \qquad \beta(B) = 19 \qquad \beta(C) = 18 \qquad \beta(D) = 13 \qquad \beta(E) = 0 \tag{11}$$

Let us now consider a general finite Markov chain with transition probability matrix P on the state space $\mathscr{X}$. Let $A \subset \mathscr{X}$ be a set of states. For each $i \in \mathscr{X}$, let $\beta(i)$ be the average number of steps until the Markov chain enters one of the states in A, given that it starts in state i.

Then one has the first step equations

$$\begin{aligned}
\beta(i) &= 0 \quad \text{if } i \in A \\
\beta(i) &= 1 + \sum_{j \in \mathscr{X}} P(i,j)\beta(j) \quad \text{otherwise.}
\end{aligned}$$

As another example, consider the Markov chain in Figure 1. Let $\beta(i)$ be the average number of steps until the Markov chain enters state 1, starting in state i. The first step equations are

$$\begin{aligned}
\beta(0) &= 1 + (1-a)\beta(0) + a\beta(1) \\
\beta(1) &= 0
\end{aligned}$$

Solving, we find $\beta(0) = \frac{1}{a}$. Note that the time to enter state 1 starting from state 0 is the number of times one has to flip a biased coin with $\mathbb{P}[H] = a$ until the first heads. This number of steps has a geometric distribution with parameter a. Thus, we have rediscovered the fact that the mean value of a Geometric(a) random variable is $\frac{1}{a}$.

Now suppose you flip a fair coin repeatedly until you get two heads in a row. How many times do you have to flip the coin, on average? Figure 6 shows a state transition diagram that corresponds to this situation. The

Markov chain starts in state S. The state is H or T if the last coin flip was H or T, respectively, except that the state is E if the last two flips were heads. This state E is *absorbing*, i.e., $P(E,E) = 1$ (the chain never leaves it). For $i \in \{S,T,H,E\}$, let $\beta(i)$ be the average number of steps until the Markov chain enters state E. The first step equations are

$$\begin{aligned}
\beta(S) &= 1 + \frac{1}{2}\beta(T) + \frac{1}{2}\beta(H) \\
\beta(T) &= 1 + \frac{1}{2}\beta(T) + \frac{1}{2}\beta(H) \\
\beta(H) &= 1 + \frac{1}{2}\beta(T) + \frac{1}{2}\beta(E) \\
\beta(E) &= 0
\end{aligned}$$

Solving, we find

$$\beta(S) = 6 \tag{12}$$

(See the Appendix for the calculations.)

Consider now the 20-rung ladder. A man starts on the ground. At each step, he moves up one rung with probability p and falls back to the ground otherwise. Let $\beta(i)$ be the average number of steps needed to reach the top rung, starting from rung $i \in \{0,1,\ldots,20\}$ where rung 0 refers to the ground. The first step equations are

$$\begin{aligned}
\beta(i) &= 1 + (1-p)\beta(0) + p\beta(i+1), \quad i = 0,\ldots,19 \\
\beta(20) &= 0
\end{aligned}$$

Solving, we find

$$\beta(0) = \frac{p^{-20} - 1}{1-p} \tag{13}$$

(See the Appendix for the calculations.) For instance, if $p = 0.9$, then $\beta(0) \approx 72$. Also, if $p = 0.8$, then $\beta(0) \approx 429$. The moral of the story is that you should be careful on a ladder!

9 Probability of A before B

Let X_n be a finite Markov chain with state space $\mathscr{X}$ and transition probability matrix P. Let also A and B be two disjoint subsets of $\mathscr{X}$. We want to determine the probability $\alpha(i)$ that, starting in state i, the Markov chain enters one of the states in A before one of the states in B.

The first step equations for $\alpha(i)$ are

$$\begin{aligned}
\alpha(i) &= \sum_j P(i,j)\alpha(j), \quad \forall i \notin A \cup B \\
\alpha(i) &= 1, \quad \forall i \in A \\
\alpha(i) &= 0, \quad \forall i \in B
\end{aligned}$$

To see why the first set of equations hold, we observe that the event that the Markov chain enters A before B starting from i is partitioned into the events that it does so by first moving to state j, for all possible value of j. Now, the probability that it enters A before B starting from i after moving first to j is the probability that it enters A before B starting from j, because the Markov chain is amnesic. The second and third sets of equations are obvious.

As an illustration, suppose we play a game of heads-or-tails with a coin such that $\mathbb{P}[H] = p$. For every heads, your fortune increases by 1 and for every tails, it decreases by 1. Your initial fortune is n. You stop playing when either you go bankrupt (your fortune reaches zero), or your fortune reaches some target value $M > n$. We want to calculate the probability that your fortune reaches M before you go bankrupt. Call this probability $\alpha(n)$. As usual, we will set up a system of equations to compute $\alpha(n)$ for all n.

The first step equations are

$$\begin{aligned} \alpha(n) &= (1-p)\alpha(n-1) + p\alpha(n+1), \quad 0 < n < M \\ \alpha(M) &= 1 \\ \alpha(0) &= 0 \end{aligned}$$

Solving these equations, we find

$$\alpha(n) = \frac{1-\rho^n}{1-\rho^M} \tag{14}$$

where $\rho := (1-p)p^{-1}$. (See the Appendix for the calculations.) Note that $\rho < 1$ under the reasonable assumption that $p < 0.5$ (so that the casino has an advantage). For instance, with $p = 0.48$ and $M = 100$, we find that $\alpha(10) \approx 4 \times 10^{-4}$, which is sobering when contemplating a trip to Las Vegas. Note that for each gambler who plays this game, the Casino makes \$10.00 with probability $1 - 4 \times 10^{-4}$ and loses \$990.00 with probability 4×10^{-4}, so that the expected gain of the Casino per gambler is approximately $(1 - 4 \times 10^{-4}) \times \$10.00 - 4 \times 10^{-4} \times \$990.00 \approx \$9.60$. Observe that the probability of winning in one step is 48%, so that if the gambler did bet everything on a single game and stopped after one step, the Casino would only make $0.52 \times \$10.00 - 0.48 \times \$10.00 = \$0.40$ on average per gambler, instead of \$9.60. (Of course, if $p > 0.5$ then $\rho > 1$ and you as the gambler have the advantage; similar conclusions then hold with the roles of you and the casino reversed.)

Appendix: Calculations

This section presents the details of the calculations of this note. The actual calculations are not very important and are included here for completeness.

Equation (6)

By symmetry, we can write

$$P^n = \begin{bmatrix} 1-\alpha_n & \alpha_n \\ \alpha_n & 1-\alpha_n \end{bmatrix}$$

for some α_n that we determine below. Note that $\alpha_1 = a$. Also,

$$P^{n+1} = \begin{bmatrix} 1-\alpha_{n+1} & \alpha_{n+1} \\ \alpha_{n+1} & 1-\alpha_{n+1} \end{bmatrix} = PP^n = \begin{bmatrix} 1-a & a \\ a & 1-a \end{bmatrix} \begin{bmatrix} 1-\alpha_n & \alpha_n \\ \alpha_n & 1-\alpha_n \end{bmatrix}.$$

Consequently, by looking at component $(0,1)$ of this product,

$$\alpha_{n+1} = (1-a)\alpha_n + a(1-\alpha_n) = a + (1-2a)\alpha_n.$$

Let us try a solution of the form $\alpha_n = b + c\lambda^n$. We need

$$\begin{aligned} \alpha_{n+1} &= b + c\lambda^{n+1} \\ &= a + (1-2a)\alpha_n \\ &= a + (1-2a)(b + c\lambda^n) \\ &= a + (1-2a)b + (1-2a)c\lambda^n. \end{aligned}$$

Matching the terms, we see that this identity holds if

$$b = a + (1-2a)b \quad \text{and} \quad \lambda = 1-2a.$$

The first equation gives $b = \frac{1}{2}$. Hence, $\alpha_n = \frac{1}{2} + c(1-2a)^n$. To find c, we use the fact that $\alpha_1 = a$, so that $\frac{1}{2} + c(1-2a) = a$, which yields $c = -\frac{1}{2}$.

Hence,

$$\alpha_n = \frac{1}{2} - \frac{1}{2}(1-2a)^n.$$

Equation (8)

The balance equations are $\pi = \pi P$.

We know that the equations do not determine π uniquely. Let us choose arbitrarily $\pi(A) = 1$. We then solve for the other components of π and we renormalize later. We can ignore any equation we choose. Let us ignore the first one. The new equations are

$$\begin{bmatrix} \pi(B) & \pi(C) & \pi(D) & \pi(E) \end{bmatrix} = \begin{bmatrix} 1 & \pi(B) & \pi(C) & \pi(D) & \pi(E) \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \end{bmatrix}.$$

Equivalently,

$$\begin{bmatrix}\pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} = \begin{bmatrix}\frac{1}{2} & 0 & \frac{1}{2} & 0\end{bmatrix} + \begin{bmatrix}\pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0\\ \frac{1}{3} & 0 & 0 & \frac{1}{3}\\ \frac{1}{2} & \frac{1}{2} & 0 & 0\end{bmatrix}.$$

By inspection, we see that $\pi(D) = \frac{1}{2}$, then $\pi(E) = \frac{1}{3}\pi(D) = \frac{1}{6}$, then

$$\pi(B) = \frac{1}{2} + \frac{1}{3}\pi(D) + \frac{1}{2}\pi(E) = \frac{1}{2} + \frac{1}{6} + \frac{1}{12} = \frac{3}{4}.$$

Finally,

$$\pi(C) = \pi(B) + \frac{1}{2}\pi(E) = \frac{3}{4} + \frac{1}{12} = \frac{5}{6}.$$

The components $\pi(A) + \cdots + \pi(E)$ add up to $1 + \frac{3}{4} + \frac{5}{6} + \frac{1}{2} + \frac{1}{6} = \frac{39}{12}$. To normalize, we multiply each component by $\frac{12}{39}$ and we get

$$\pi = \begin{bmatrix}\frac{12}{39} & \frac{9}{39} & \frac{10}{39} & \frac{6}{39} & \frac{2}{39}\end{bmatrix}.$$

We could have proceeded differently and observed that our identity implies that

$$\begin{bmatrix}\pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} \left(I - \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0\\ \frac{1}{3} & 0 & 0 & \frac{1}{3}\\ \frac{1}{2} & \frac{1}{2} & 0 & 0\end{bmatrix} \right) = \begin{bmatrix}\frac{1}{2} & 0 & \frac{1}{2} & 0\end{bmatrix}$$

$$\begin{bmatrix}\pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} \begin{bmatrix}1 & -1 & 0 & 0\\ 0 & 1 & 0 & 0\\ -1/3 & 0 & 1 & -1/3\\ -1/2 & -1/2 & 0 & 1\end{bmatrix} = \begin{bmatrix}\frac{1}{2} & 0 & \frac{1}{2} & 0\end{bmatrix}$$

Hence,

$$\begin{bmatrix}\pi(B) & \pi(C) & \pi(D) & \pi(E)\end{bmatrix} = \begin{bmatrix}\frac{1}{2} & 0 & \frac{1}{2} & 0\end{bmatrix} \begin{bmatrix}1 & -1 & 0 & 0\\ 0 & 1 & 0 & 0\\ -\frac{1}{2} & 0 & 1 & -\frac{1}{3}\\ -\frac{1}{2} & -\frac{1}{2} & 0 & 1\end{bmatrix}^{-1}.$$

This procedure is a systematic way to solve the balance equations by computer.

Equation (11)

Using the third equation in the second, we find $\beta(B) = 2 + \beta(A)$. The fourth equation then gives

$$\beta(D) = 1 + \frac{1}{3}\beta(A) + \frac{1}{3}(2 + \beta(A)) = \frac{5}{3} + \frac{2}{3}\beta(A).$$

The first equation then gives

$$\beta(A) = 1 + \frac{1}{2}(2 + \beta(A)) + \frac{1}{2}\left(\frac{5}{3} + \frac{2}{3}\beta(A)\right) = \frac{17}{6} + \frac{5}{6}\beta(A).$$

Hence, $\frac{1}{6}\beta(A) = \frac{17}{6}$, so that $\beta(A) = 17$. Consequently, $\beta(B) = 19$ and $\beta(D) = \frac{5}{3} + \frac{34}{3} = 13$. Finally, $\beta(C) = 18$.

Equation (12)

The last two equations give $\beta(H) = 1 + \frac{1}{2}\beta(T)$. If we substitute this expression in the second equation, we get

$$\beta(T) = 1 + \frac{1}{2}\beta(T) + \frac{1}{2}\left(1 + \frac{1}{2}\beta(T)\right) = \frac{3}{2} + \frac{3}{4}\beta(T).$$

Hence, $\beta(T) = 6$. Consequently, $\beta(H) = 1 + \frac{1}{2}\cdot 6 = 4$. Finally, $\beta(S) = 1 + \frac{1}{2}\cdot 6 + \frac{1}{2}\cdot 4 = 6$.

Equation (13)

Let us look for a solution of the form $\beta(i) = a + b\lambda^i$. Then

$$a + b\lambda^i = 1 + (1-p)(a+b) + p[a + b\lambda^{i+1}] = 1 + (1-p)(a+b) + pa + bp\lambda^{i+1}.$$

This identity holds if

$$a = 1 + (1-p)(a+b) + pa \quad \text{and} \quad \lambda = p^{-1},$$

i.e.,

$$b = -(1-p)^{-1} \quad \text{and} \quad \lambda = p^{-1}.$$

Then,

$$\beta(i) = a - (1-p)^{-1}p^{-i}.$$

Since $\beta(20) = 0$, we need

$$0 = a - (1-p)^{-1}p^{-20},$$

so that $a = (1-p)^{-1}p^{-20}$ and

$$\beta(i) = (1-p)^{-1}p^{-20} - (1-p)^{-1}p^{-i} = \frac{p^{-20} - p^{-i}}{1-p}.$$

Equation (14)

We again look for a solution of the form $\alpha(n) = a\lambda^n + b$ for suitable constants a, b, λ. Plugging this into the first step equations gives

$$a\lambda^n + b = (1-p)(a\lambda^{n-1} + b) + p(a\lambda^{n+1} + b),$$

which simplifies to

$$\lambda^n = (1-p)\lambda^{n-1} + p\lambda^{n+1}.$$

Canceling a factor of λ^{n-1} and rearranging gives the quadratic equation

$$p\lambda^2 - \lambda + (1-p) = 0,$$

whose solutions are

$$\lambda = \frac{1 \pm \sqrt{1 - 4p(1-p)}}{2p} = \frac{1 \pm (1-2p)}{2p}.$$

One solution is $\lambda = 1$, which is not interesting, so we take $\lambda = \frac{1-p}{p} =: \rho$. To compute a and b, we use the boundary conditions $\alpha(M) = 1$ and $\alpha(0) = 0$, which become

$$\begin{aligned} a\rho^M + b &= 1 \\ a + b &= 0. \end{aligned}$$

Solving gives us $a = -\frac{1}{1-\rho^M}$ and $b = \frac{1}{1-\rho^M}$. This yields our final answer

$$\alpha(n) = \frac{1-\rho^n}{1-\rho^M}.$$