15-251: Great Theoretical Ideas In Computer Science

Notes on Linear Algebra

Venkatesan Guruswami

October 18, 2011

Linear algebra is probably familiar in some form or other to you. It is a branch of mathematics that is very important to a diverse set of areas in computer science. Linear algebra has been steadily increasing in prominence in recent years, fueled by algorithmic applications dealing with massive data sets. In these notes, we will introduce some basic concepts of the subject, and see some nice "extraneous" uses of linear-algebraic arguments.

1 Vector space

We begin by defining one of the principal objects of study in linear algebra, namely vector spaces.

Definition 1.1 (Vector space) *Let $\mathbb{F}$ be a field, whose elements are referred to as scalars. A vector space over $\mathbb{F}$ is a nonempty set V, together with two operations: (i) addition and denoted $+$, which is a binary operation on V, and (ii) scalar multiplication, denoted by $\cdot$ or juxtaposition, that assigns to each pair $(\lambda, v) \in \mathbb{F} \times V$ an element $\lambda v \in V$.The operations must satisfy the following properties:*

1. *$(V, +)$ is a commutative group. (The additive identity will be denoted by 0.)*
2. *For all $\lambda, \mu \in \mathbb{F}$ and $v, w \in V$,*

$$\begin{aligned} \lambda(v + w) &= \lambda v + \lambda w \\ (\lambda + \mu)v &= \lambda v + \mu v \\ (\lambda\mu)v &= \lambda(\mu v) \\ 1v &= v\ . \end{aligned}$$

We will sometimes refer to the elements of the vector space V as vectors. The principal way in which elements of a vector space are combined or manipulated to give other vectors is via linear combinations. This is the "bread and butter" operation for vector spaces.

Definition 1.2 (Linear combinations) *Any (finite) expression of the form*

$$\lambda_1 v_1 + \lambda_2 v_2 + \cdots + \lambda_n v_n$$

where $\lambda_i \in \mathbb{F}$ and $v_i \in V$ for $i = 1, 2, \ldots, n$, is called a linear combination *of the vectors $v_1, v_2, \ldots, v_n$. If at least of the λ_i's is nonzero, we say the linear combination is* non-trivial*, otherwise it is the trivial linear combination (which must equal the 0 vector).*

Let us see some examples of vector spaces:

1. For every field $\mathbb{F}$, $\mathbb{F}$ itself is a vector space over $\mathbb{F}$.
2. The set of m-dimensional real vectors $\mathbb{R}^m$ under addition and scalar multiplication defined component-wise is a vector space over $\mathbb{R}$:

$$\begin{aligned} (a_1, a_2, \ldots, a_m) + (b_1, b_2, \ldots, b_m) &= (a_1 + b_1, a_2 + b_2, \ldots, a_m + b_m) \\ \lambda(a_1, a_2, \ldots, a_m) &= (\lambda a_1, \lambda a_2, \ldots, \lambda a_m) \end{aligned}$$

3. The set $M_{p\times q}(\mathbb{Q})$ of $p \times q$ matrices with rational entries is a vector space over the field $\mathbb{Q}$ of rationals, under the operations of matrix addition and scalar multiplication.

4. The set of real-valued functions on $\{0,1\}^n$, $\{f : f : \{0,1\}^n \to \mathbb{R}\}$ is a vector space over $\mathbb{R}$, under the operation of addition and scalar multiplication of functions:

$$(f+g)(x) = f(x) + g(x)$$
$$(\lambda f)(x) = \lambda(f(x))$$

   This example is really the same as the second example $\mathbb{R}^m$ with $m = 2^n$, dressed up differently. (Why?)

5. The set of polynomials in $\mathbb{Z}_p[X]$ of degree at most d, under the operation of polynomial addition (i.e, $(P+Q)(X) = P(X)+Q(X)$) and scalar multiplication defined as multiplication by the corresponding constant polynomial (i.e., $(\lambda P)(X) = \lambda(P(X))$).

6. The above are examples of finite-dimensional vectors spaces[1], which are all the spaces that will concern us here. But here is an infinite dimensional one just to give you an example: the space ℓ^1 of real-valued sequences whose series is *absolutely convergent*. That is, infinite sequences $(a_1, a_2, \ldots, a_n, a_{n+1}, \ldots)$ of real numbers such that $\sum_{i=1}^{\infty} |a_i|$ converges. The vector sum is defined componentwise

$$(a_1, a_2, \ldots, a_n, \ldots) + (b_1, b_2, \ldots, b_n, \ldots) = (a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n, \ldots)$$

   (why is ℓ^1 closed under the above operation?) and scalar action is also defined componentwise

$$\lambda(a_1, a_2, \ldots, a_n, \ldots) = (\lambda a_1, \lambda a_2, \ldots, \lambda a_n, \ldots)\ .$$

As with most interesting algebraic structures, vector spaces also have a natural notion of substructures.

Definition 1.3 (Subspace) *A nonempty subset $S \subseteq V$ is said to be a* subspace *of V if it is a vector space under the restriction of the operations of V to S. Equivalently, $S \subseteq V$ is a subspace if S is closed under linear combinations, i.e.,*

$$\lambda, \mu \in \mathbb{F},\ \ v, w \in S \ \Rightarrow \lambda v + \mu w \in S\ .$$

When a subspace S doesn't equal the whole vector space V, we say that S is a proper subspace *of V.*

For example, the subset of diagonal $n \times n$ matrices is a subspace of $M_{n\times n}(\mathbb{Q})$, as is the subset of upper-triangular matrices. Note that every subspace contains 0, and just the set $\{0\}$ is a trivial subspace. Here is a simple example of a more interesting subspace.

Exercise 1.4 *Let $H_n = \mathbb{Z}_2^n$ be the vector space of all binary n-tuples over the field $\mathbb{Z}_2 = \{0,1\}$ of two elements. Let E_n be the subset of H_n consisting of strings with an even number of 1's. Prove that E_n is a subspace of H_n.*

[1] We will define dimension of a vector space shortly.

2 Linear independence, span, and basis

As mentioned above, the central operation one performs in a vector space is taking linear combinations. Some subsets $S \subset V$ are "non-redundant" in the sense that for any vector $v \in V$, there is at most one way to generate v as a linear combination of elements of S. The extremely important definition which captures this property is the following.

Definition 2.1 (Linear independence) *A non-empty subset $S \subset V$ is said to be* linearly independent *if for any $v_1, v_2, \ldots, v_n \in S$,*

$$\lambda_1 v_1 + \lambda_2 v_2 + \cdots + \lambda_n v_n = 0 \implies \lambda_1 = \lambda_2 = \cdots = \lambda_n = 0 \,.$$

If S is not linearly independent, then it is said to be linearly dependent.

The set of all linear combinations of vectors from S is an important concept, called span, defined next.

Definition 2.2 (Span) *The subspace* spanned *(or* generated*) by a subset S of a vector space V, denoted $\mathrm{span}(S)$, is the set of all linear combinations of vectors from S:*

$$\mathrm{span}(S) = \{\lambda_1 v_1 + \lambda_2 v_2 + \cdots + \lambda_n v_n \mid \lambda_i \in \mathbb{F}, v_i \in S\} \,.$$

If $\mathrm{span}(S) = V$, we say that S spans V or generates V.

It is easy to show the following from the definitions.

Exercise 2.3 *Let S be a subset of a vector space V. The following statements are equivalent.*

1. *S is linearly independent.*
2. *No element of S can be expressed as a linear combination of the other elements in S.*
3. *Each element in $\mathrm{span}(S)$ has a* **unique** *expression as a linear combination of elements from S.*

The most common way to represent and reason about a vector space is by choosing a basis for it.

Definition 2.4 (Basis) *A subset $S \subset V$ that spans V and is linearly independent is called a* basis *for V.*

By Exercise 2.3, if S is a basis for V, *every* element of V has a *unique* expression as a linear combination of elements from S.

But why must a basis exist? It turns out that a set S is a basis precisely when either of the following two conditions hold (the two conditions can be shown to be equivalent, so if one holds, both of them hold):

- S is a maximal linearly independent set, i.e., S is linearly independent but every proper superset of S is linearly dependent.
- S is a minimal spanning set, i.e., $\mathrm{span}(S) = V$, but no proper subset of S spans V.

It can be shown, using Zorn's lemma, that maximal linearly independent sets exist in any vector space. In fact, the following important statement can be shown.

Theorem 2.5 (Existence of basis) *Let V be a non-trivial vector space (i.e., $V \neq \{0\}$). Then,*

1. *V has a basis.*
2. *Any linearly independent subset $S \subset V$ can be extended to a basis.*
3. *Every subset $S \subset V$ that spans V contains a basis.*

Here is an example of a basis. For $1 \leq i \leq n$, define the i'th coordinate vector $e_i \in \mathbb{R}^n$ to be the vector whose i'th coordinate is 1 and the rest are 0. For example, $e_1 = (1, 0, 0, 0, \ldots, 0)$. The set of vectors

$$\{e_1, e_2, \ldots, e_n\}$$

is a basis for $\mathbb{R}^n$. It is often referred to as the standard basis.

Exercise 2.6 *Can you give a basis for the vector space of polynomials in $\mathbb{Z}_p[X]$ of degree at most d?*

A vector space can (and typically does) have many different bases. For example $e_1, e_2, \ldots, e_{n-1}, \mathbf{1}$ is also a basis for $\mathbb{R}^n$ where $\mathbf{1}$ is the all-ones vector. Turns out that all bases, however, will have the same size. Let us prove this very important fact for the case when there is a finite size basis.

Theorem 2.7 *If $A = \{a_1, a_2, \ldots, a_n\}$ is a basis for vector space V, and $B = \{b_1, b_2, \ldots, b_m\}$ is a spanning set, then $m \geq n$. If further B is a basis, then $m = n$.*

Proof: Since B spans V, a_1 can be expressed as a linear combination of the $\{b_1, b_2, \ldots, b_m\}$. As $a_1 \neq 0$ (because A is a basis), there must be an i, $1 \leq i \leq m$, such that b_i can be expressed as a linear combination of $a_1, b_1, \ldots, b_{i-1}$. This implies that $\mathrm{span}(\{a_1, b_1, b_2, \ldots, b_m\} \setminus \{b_i\}) = V$. Define $B_1 = \{a_1, b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_m)$. We have that B_1 is also a spanning set of size m. Now we will repeat this process. As A is linearly independent, for $j = 2, 3, \ldots, n$, we can always swap in a_j for an element from the list B_j (which is not $a_1, a_2, \ldots, a_{j-1}$) to create B_{j+1}. After n iterations, we will be left with a spanning set of size m that contains $a_1, a_2, \ldots, a_n$. Thus, $m \geq n$.

If B is also a basis, then we can apply the argument with the roles of A and B switched to conclude $n \geq m$, implying that in fact $m = n$. ■

Definition 2.8 *A vector space V is said to be* finite dimensional *if it is the zero space $\{0\}$ or has a finite basis. The dimension of such a V, denoted $\dim(V)$, is the size of any basis for V (if $V = \{0\}$, $\dim(V) = 0$).*

We state the following important corollary of Theorems 2.5 and 2.7.

Corollary 2.9 *If V is finite dimensional, then*

1. *Any two bases of V have the same size.*
2. *If $S \subset V$ is linearly independent, then $|S| \leq \dim(V)$, with equality if and only if S is a basis for V.*

Note that if $S \subset V$ is a linearly independent set, then $\mathrm{span}(S)$ is a subspace of V of dimension $|S|$. By virtue of the above, every proper subspace of a finite-dimensional vector space V has a strictly lower dimension than V.

Corollary 2.10 *Let V be a finite-dimensional vector space, and let W be a proper subspace of V. Then $\dim(W) < \dim(V)$.*

Some comments on basis representations and basis change. Once a basis $B = \{b_1, b_2, \ldots, b_n\}$ of an n-dimensional vector space V over a field $\mathbb{F}$ is picked, elements of V can be represented as n-tuples from $\mathbb{F}$, with the n-tuple $(\lambda_1, \lambda_2, \ldots, \lambda_n) \in \mathbb{F}^n$ corresponding to $\lambda_1 b_1 + \lambda_2 b_2 + \cdots + \lambda_n b_n \in V$. For example, the usual coordinate representation of $\mathbb{R}^n$ is the representation of vectors with respect to the standard basis. Thus, all n-dimensional vector spaces V over $\mathbb{F}$ are "essentially" the same as $\mathbb{F}^n$ — the formal statement is that V is *isomorphic* to $\mathbb{F}^n$, though we don't make the concept of isomorphism precise here.

In particular, when $\mathbb{F}$ is finite, an n-dimensional subspace over $\mathbb{F}$ has exactly $|\mathbb{F}|^n$ elements. The following is a useful way to state Corollary 2.10 for the finite field case, which is the analog of the fact that a proper subgroup of a finite group G has at most $|G|/2$ elements.

Corollary 2.11 *Let V be a finite-dimensional vector space over a* **finite** *field $\mathbb{F}$, and let W be a proper subspace of V. Then $|W| \leq \frac{|V|}{|\mathbb{F}|}$.*

A different basis $B' = \{b'_1, b'_2, \ldots, b'_n\}$ for V can be expressed by n vectors in $\mathbb{F}^n$, giving the coefficients in the representation of each b'_i in terms of the b_j's. The $n \times n$ matrix $\mathbf{B'}$ consisting of these vectors as columns gives a representation of *basis B' in terms of B*. A vector represented as $\mathbf{y} \in \mathbb{F}^n$ in basis B' has representation $\mathbf{x} = \mathbf{B'y}$ with respect to basis B. An important choice of convention choice we made here is that coefficient vectors in $\mathbb{F}^n$ for a vector (such as $\mathbf{x}$ and $\mathbf{y}$) are considered *column vectors*.

In other words, multiplying by $\mathbf{B'}$ allows us to change from basis B' to B. Likewise, multiplying by $\mathbf{B'^{-1}}$ gives us a way to change representations from basis B to basis B'. This is how matrices naturally enter linear algebra — they are an important tool in working with vector spaces when vectors are represented in a basis dependent way.

A careful choice of basis drives several neat applications of linear algebra — a great example is the use of the Fourier basis for $\mathbb{R}^n$ instead of the standard basis when n is a power of two. (The details of this are, however, beyond our scope.)

We could take a detour here and talk about linear transformations and the related matrix algebra, eigenvalues, eigenvectors, and what not. But let us change gears a bit now, so we can present an elegant application of linear algebra in a seemingly alien context.

3 Oddtown clubs

Oddtown has a population of n people. People started forming numerous clubs in this town, and in order to limit the number of clubs, the following rules were strictly imposed:

1. Each club must have an odd number of members.
2. Every pair of clubs must have an even number of members in common.

Turns out these rules do the intended job, as we will show in this section.

Theorem 3.1 *Under these rules, there can be at most n clubs in Oddtown.*

While this combinatorial problem does not seem to have much to do with linear algebra, we will give a linear-algebraic proof, illustrating a simple but powerful method to use dimension arguments in combinatorics. But first, we need to endow vector spaces with more structure, namely an inner product, which is the abstraction of the familiar dot product of vectors with real coefficients.

Definition 3.2 (Inner product) *Given a real vector space V, an inner product on V is a map from $V \times V$ to $\mathbb{R}$, with $\langle v, w \rangle$ denoting the inner product of v and w, satisfying the following properties:*

- *For all $v \in V$, $\langle v, v \rangle \geq 0$ with equality if and only if $v = 0$.*
- *For all $\lambda, \mu \in \mathbb{F}$, and $u, v, w \in V$, $\langle \lambda u + \mu v, w \rangle = \lambda \langle u, w \rangle + \mu \langle v, w \rangle$.*
- *For $v, w \in V$, $\langle v, w \rangle = \langle w, v \rangle$.*

We will refer to a vector space with an inner product defined on it as an inner product space.

We are now ready to prove the "Oddtown theorem."

Proof: (Of Theorem 3.1) Suppose $C_1, C_2, \ldots, C_m \subseteq \{1, 2, \ldots, n\}$ are m clubs such that:

- $|C_i|$ is odd for $1 \leq i \leq m$, and
- $|C_i \cap C_j|$ is even for $1 \leq i < j \leq m$.

We need to prove that $m \leq n$. Define $v_i = (v_{i,1}, \ldots, v_{i,n}) \in \{0, 1\}^n$ to be the characteristic vector of C_i, i.e.,

$$v_{i,j} = \begin{cases} 1 & \text{if } j \in C_i \\ 0 & \text{if } j \notin C_i \end{cases}$$

Define $\langle x, y \rangle = \sum_{j=1}^{n} x_j y_j$ to be the usual inner product for vectors in $\mathbb{R}^n$. Note that $\langle v_i, v_i \rangle = |C_i|$ is odd for every $i \in \{1, 2, \ldots, m\}$, and $\langle v_i, v_j \rangle = |C_i \cap C_j|$ is even for $1 \leq i < j \leq m$.

Consider the vector space $V = \mathbb{Q}^n$ over $\mathbb{Q}$. We will prove that $v_1, v_2, \ldots, v_m$ are linearly independent vectors in V. Together with Corollary 2.9 this will immediately imply that $m \leq n$, since $\mathbb{Q}^n$ has dimension n.

Suppose, for contradiction, that the v_i's are linearly dependent over $\mathbb{Q}$, Let $\lambda_1 v_1 + \lambda_2 v_2 + \cdots + \lambda_n v_n$ be a nontrivial linear combination with rational coefficients λ_i that equals 0. By clearing denominators, we can assume that all λ_i's are in fact integers. Further we can divide out by common factors, and assume that at least one λ_j is odd. Assume without loss of generality that λ_1 is odd. We have

$$\begin{aligned} 0 &= \langle v_1, 0 \rangle \\ &= \langle v_1, \lambda_1 v_1 + \lambda_2 v_2 + \cdots + \lambda_k v_k \rangle \\ &= \lambda_1 \langle v_1, v_1 \rangle + \sum_{j=2}^{k} \lambda_j \langle v_1, v_j \rangle \, . \end{aligned}$$

Now, $\sum_{j=2}^{k} \lambda_j \langle v_1, v_j \rangle$ is even as each $\langle v_1, v_j \rangle$, $j > 1$, is even. Also, $\lambda_1 \langle v_1, v_1 \rangle$ is odd as both λ_1 and $\langle v_1, v_1 \rangle$ are odd. Therefore, $\lambda_1 \langle v_1, v_1 \rangle + \sum_{j=2}^{k} \lambda_j \langle v_1, v_j \rangle$ is an odd number and thus can't equal 0. We have the desired contradiction. Therefore, $\{v_1, v_2, \ldots, v_m\}$ is linearly independent over $\mathbb{Q}$, as desired. ■

4 Error correction: A simple bound for codes

Our friends Alice and Bob are back, with Alice trying to transmit messages to Bob across an intervening noisy communication channel. The channel can be used to transmit a sequence of n bits for

n as large as Alice and Bob choose (we will assume they are patient, so Bob can wait to receive 10^{10} bits before he decodes the message Alice sent). The catch is that the channel is noisy, and can flip any subset of less than $n/3$ of the transmitted bits. The question is, how many distinct messages can Alice transmit to Bob reliably (which means Bob can for sure decode the correct message that Alice had in mind even in the wake of the errors)?

First, a warm-up: suppose the channel is more nasty than stated, and can flip up to $n/2$ bits. Now, how many messages can Alice get across?

Well, the channel can always flip all 0's to 1's if there are at most $n/2$ 0's, or all 1's to 0's if there are at most $n/2$ 1's. Thus Bob can be forced to received just the all 0's or all 1's string. So at best Alice can transmit two messages reliably, regardless of how big n is! (And indeed she *can* send two messages, by encoding them as the 0's and all 1's strings.)

Back to our problem of $< n/3$ flips, suppose Alice sends m messages, each encoded as a codeword $c_i \in \{0,1\}^n$ of n bits, for $i = 1, 2, \ldots, m$. Let us begin with a key observation.

Lemma 4.1 *The set of codewords $C = \{c_1, c_2, \ldots, c_m\}$ enables Bob to correctly decode Alice's message if and only if it has the following"distance" property:*

$$\Delta(c_i, c_j) \geq 2n/3 \text{ for } 1 \leq i < j \leq m \ ,$$

where for $x, y \in \{0,1\}^n$, $\Delta(x, y)$ denotes the Hamming distance between x and y, defined to be the number of positions x and y differ on.

Proof: Suppose c_i is the codeword sent on the channel, and Bob receives r such that $\Delta(r, c_i) < n/3$. We note that Bob can confuse r to be the corrupted form of c_j for some $j \neq i$ if and only if we also have $\Delta(r, c_j) < n/3$. In such a case, $\Delta(c_i, c_j) < 2n/3$, contradicting the distance property of C. Thus the distance property ensures reliable communication.

On the other hand, if $\Delta(c_i, c_j) < 2n/3$, then the channel can distort c_i into a point r "mid-way" between c_i and c_j, and Bob upon receiving r can't tell if Alice originally sent c_i or c_j. So the distance property is also a necessary condition for reliable communication. ■

So our question now becomes: suppose there are m bit vectors $c_1, \ldots, c_m$ such that every pair has Hamming distance at least $2n/3$. How large can m be?

We will map the problem to a linear-algebraic/geometric one, and obtain the following strong upper bound. Note that, as with the case of $n/2$ flips, the upper bound is once again independent of n — Alice can't hope to communicate more than 4 distinct messages!

Theorem 4.2 *If $c_1, c_2, \ldots, c_m$ are n-bit vectors such that $\Delta(c_i, c_j) \geq 2n/3$ whenever $i \neq j$, then $m \leq 4$.*

Proof: We will map the codewords $c_1, c_2, \ldots, c_m$ into vectors $v_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, m$, such that the angle between every pair of vectors is more than 90 degrees (i.e., their dot product $\langle v_i, v_j \rangle < 0$). We will then show that one can't "pack" many such vectors in $\mathbb{R}^n$, regardless of how large the dimension n is.

These vectors are defined as follows:

$$v_i = \frac{1}{\sqrt{n}}\left((-1)^{c_{i,1}}, (-1)^{c_{i,2}}, \cdots, (-1)^{c_{i,n}}\right) ,$$

where $c_{i,j}$ is the i'th bit of the codeword c_i. It is easy to check that $\langle v_i, v_i \rangle = 1$ for each i, and for $i \neq j$,

$$\langle v_i, v_j \rangle = \frac{1}{n}(n - 2\Delta(c_i, c_j)) \leq \frac{n - 4n/3}{n} = -\frac{1}{3} .$$

By Lemma 4.3 below, we must have $m \leq 4$. ■

Lemma 4.3 *Let $\eta > 0$ and let $v_1, v_2, \ldots, v_m$ be m vectors in $\mathbb{R}^n$ such that*

- *$\langle v_i, v_i \rangle = 1$ for all $1 \leq i \leq m$, and*
- *$\langle v_i, v_j \rangle \leq -\eta$ for all $1 \leq i < j \leq m$.*

Then $m \leq 1 + \frac{1}{\eta}$.

Proof: We have

$$0 \leq \langle \sum_{i=1}^{m} v_i, \sum_{i=1}^{m} v_i \rangle = \sum_{i=1}^{m} \langle v_i, v_i \rangle + 2 \sum_{1 \leq i < j \leq m} \langle v_i, v_j \rangle \leq m - m(m-1)\eta ,$$

which gives $m \leq 1 + 1/\eta$ after rearranging. ■

5 Correcting a single bit flip

Since Alice would like to transmit more than one of four possible messages to Bob, they now invest in a better communication medium. The new, much improved, communication channel, while still noisy, never flips more than 1 out of the n bits that are transmitted. In this section, we will use linear algebra to help Alice and Bob comunicate in the most efficient manner on such a channel. More precisely, Alice will make a judicious choice of a *code* $C \subset \{0,1\}^n$ and transmit only codewords from C. The structure of C should ensure that Bob can decipher the original codeword unambiguously even when one of its bits gets flipped by the channel. We would like to construct as large a C as possible that enables this.

Well, before we go about *correcting* errors, let's think about a coding scheme that will ensure that Bob will at least be able *detect* the presence of an error. A simple argument (similar to Lemma 4.1) shows that this is possible precisely when every $c \neq c' \in C$ satisfy $\Delta(c, c') \geq 2$. That is, no two codewords in C differ in exactly one bit. Here is a little exercise for you (it's just simple combinatorics, no linear algebra!)

Exercise 5.1 *Suppose $C \subset \{0,1\}^n$ is such that every $c \neq c' \in C$ satisfy $\Delta(c, c') \geq 2$. Then $|C| \leq 2^{n-1}$.*

Here is a construction that actually achieves this bound:

$$C_0 = \{(c_1, c_2, \ldots, c_n) \in \{0,1\}^n \mid c_1 + c_2 + \cdots + c_n \equiv 0 \pmod 2\} . \tag{1}$$

This is the *parity check code* — called thus because the parity of all the bits in the codeword must be even. Or equivalently, if Alice wants to send a message $m \in \{0,1\}^{n-1}$, she appends the parity of all the bits as the n'th bit, and sends m together with the parity bit.

Exercise 5.2 *Prove that $|C_0| = 2^{n-1}$ and for every $c \neq c' \in C_0$, $\Delta(c, c') \geq 2$. Show also that there exist $c_1, c_2 \in C_0$ such that $\Delta(c_1, c_2) = 2$.*

By the way, does the subset C_0 sound familiar from earlier in the notes? (Hint: Exercise 1.4.) As there exist $c_1, c_2 \in C_0$ such that $\Delta(c_1, c_2) = 2$, the code C_0, however, can't be used if Bob actually wants to rectify the bit flip (i.e., identify the position where the bit flip occurred) rather than just detect the presence of an error. (Why?)

To enable correcting the bit flip, we will pick a smaller subset $C^* \subset \{0,1\}^n$ as the code, where the codeword bits must obey more than one parity check condition. We will assume from now on that n is of the form $n = 2^t - 1$.

The set C^* will be specified by imposing parity checks on certain subsets of the n bits of the codeword. For $j = 1, 2, \ldots, t$, define S_j to be the subset of positive integers m, $1 \leq m \leq 2^t - 1$, such that the j'th least significant bit in the binary representation of m equals 1. For instance, S_1 is the set of odd integers in the range $[1, 2^t - 1]$. (Question: What's the size of each S_j?). We define the code C^* — which, by the way, is called the *Hamming code* — as follows:

$$C^* = \{(c_1, c_2, \ldots, c_n) \in \{0,1\}^n \mid \bigoplus_{i \in S_j} c_i = 0 \text{ for } j = 1, 2, \ldots, t\} \,. \tag{2}$$

For instance, for $n = 7$, the code is the following subset of $\{0,1\}^7$

$$C^*_7 = \{(c_1, c_2, \ldots, c_7) \mid c_1 \oplus c_3 \oplus c_5 \oplus c_7 = 0;\;\; c_2 \oplus c_3 \oplus c_6 \oplus c_7 = 0;\;\; c_4 \oplus c_5 \oplus c_6 \oplus c_7 = 0\} \,.$$

The construction has a particularly nice view in matrix form – C^*_7 is the subset $(c_1, c_2, \ldots, c_7) \in \mathbb{Z}_2^7$ satisfying

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_7 \end{pmatrix} = 0 \tag{3}$$

where the matrix multiplication is performed in the field $\mathbb{Z}_2$. Note that the columns of the 3×7 matrix are all nonzero 3-bit vectors, or the binary representations of the integers in the range $[1, 7]$.

Exercise 5.3 *Prove that C^*_7 is a 4-dimensional subspace of $\mathbb{Z}_2^7$.*

Similarly to (3), the code C^* for length n, defined in (2), is the set of $(c_1, c_2, \ldots, c_n) \in \mathbb{Z}_2^n$ such that

$$M_t \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = 0$$

where M_n is the $t \times (2^t - 1)$ matrix whose i'th column is the binary representation of i, for $i = 1, 2, \ldots, 2^t - 1$.

Our code C^* has the adequate distance property to be able to correct one bit flip.

Exercise 5.4 *Prove that for every $c \neq c' \in C^*$, $\Delta(c, c') \geq 3$. Show also that there exist $c_1, c_2 \in C^*$ such that $\Delta(c_1, c_2) = 3$.*

We finally turn to how Bob can locate (and therefore correct) the bit flip caused by the channel. Suppose Alice sent $c \in C^*$, and the channel flips the ℓ'th bit. This means that Bob receives the string

$$r = c + e_\ell$$

where e_ℓ is the ℓ'th coordinate vector (which has a 1 in the ℓ'th coordinate, and 0's elsewhere). Bob's goal is to find ℓ, so he can flip back the ℓ'th bit to recover the codeword c Alice sent.

Here's what Bob can do. Compute $M_t r \in \mathbb{Z}_2^t$. Note that

$$M_t r = M_t(c + e_\ell) = M_t c + M_t e_\ell = M_t e_\ell$$

where the last step uses the fact that by definition of the code C^*, $M_t c = 0$ when $c \in C^*$. What is $M_t e_\ell$? A moment's recollection of matrix-vector product reveals that $M_t e_\ell$ is the ℓ'th column of M_t.

To conclude, $M_t r$ is simply equal to the ℓ'th column of M_t. But what is the ℓ'th column of M_t? Yes, that's right, it is the binary representation of ℓ. So, Bob can compute $M_t r$, and the answer tells him the binary representation of the location of the bit flip. Reliable transmission against single bit flips achieved!

Similar to Exercise 5.3, it can be shown that $|C^*|$ has dimension $n - t$ as a subspace of $\mathbb{Z}_2^n$. Thus $|C^*| = 2^{n-t} = \frac{2^n}{n+1}$. That's a whole lot of possible messages Alice can send Bob — just a factor $(n + 1)$ smaller than the total number of n-bit strings. It can also be shown that this is optimal. In summary, for n of the form $2^t - 1$, Hamming codes give the optimal way to correct a single bit flip.

6 Checking matrix multiplication

Multiplying two $n \times n$ matrices is a basic computational problem. It is of course inhertently a problem from the domain of linear algebra. Recall that if A, B are two $n \times n$ matrices, their product $C = AB$ is given by

$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj} \ .$$

How long does it take to compute C? Well, it has n^2 entries, so just writing them down would take n^2 steps. The "obvious" way of computing each entry takes $O(n)$ steps, for a total of $O(n^3)$ steps. As you may know, there are surprising, non-trivial algorithms that can multiply $n \times n$ matrices in time faster than n^3. The first such algorithm was published by Strassen in 1969, and could multiply matrices in time roughly $n^{\log_2 7} \approx n^{2.81}$. Subsequently there were several improvements to the 2.81 exponent, and the current world record is due to (Coppsermith and Winograd, 1987) which achieves time roughly $n^{2.376}$ (though it is only of theoretical interest and not currently practical).

I think the most common opinion is that matrix multiplication can be done in $O(n^{2+\epsilon})$ for any positive ϵ, though nobody knows how to do it, and this is an outstanding open problem in algorithm design.

Here, we won't discuss algorithms for matrix multiplication. Rather we will discuss *verification* of such algorithms.

Suppose a new startup springs up in Pittsburgh, and sells software FastMM that supposedly multiplies matrices really fast. Since we obviously don't want wrong answers, we would like to have a simple *checking program* appended to FastMM that would check whether the output C is indeed the product of the input matrices A and B.

How can we do this? Of course we could simply multiply A and B and compare the result with C, but this makes little sense, as we do not know how to multiply matrices as fast as FastMM. Turns out, if we allow a small probability of error, there is a very simple and efficient checker for

matrix multiplication. (More broadly, the theme of probabilistic checking of computations with significantly fewer resources than required to actually performing them has been a very influential one in computer science, though we won't have occasion to delve into this in this course.)

Just to be concrete, we will consider matrices with rational entries, though what we say holds verbatim for matrices over any field. Without further ado, here is the checking algorithm. Upon receiving three $n \times n$ matrices A, B, C with C supposedly equal to AB, the checker is a *randomized algorithm* that does the following:

1. Toss a fair coin n times to pick a random n-bit string $x = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$, i.e., for each i, x_i is independently 0 or 1 with probability $1/2$.
2. Compute $y = Cx$.
3. Compute $z = ABx$ (which of course means $A(Bx)$).
4. If $y = z$, say YES, else say NO.

If $C = AB$, the algorithm obviously always says YES. However, when $C \neq AB$, the algorithm may give either answer, depending on its choice of x. Below is the crucial result, which shows that the algorithm errs with probability at most $1/2$. (While a failure probability of $1/2$ might seem high, one can simply repeat the check a few times to drive down the failure probability; for instance repeating it 50 times would reduce the probability of failure to 2^{-50}, an extremely small quantity.)

Lemma 6.1 *Suppose A, B, C are $n \times n$ matrices over $\mathbb{Q}$ and $C \neq AB$. Then for a random $x \in \{0,1\}^n$, the probability that $Cx = ABx$ is at most $1/2$.*

Proof: Define $M = C - AB$. Note that $M \neq 0$. Let p, q be indices such that $M_{pq} \neq 0$. Let $w = Mx$. We will prove that the probability that w_p, the p'th entry of w, is nonzero is at least $1/2$, which would prove the statement claimed in the lemma.

We have $w_p = M_{p1}x_1 + M_{p2}x_2 + \cdots + M_{pn}x_n$. Let us express w_p as

$$w_p = M_{pq}x_q + s$$

where

$$s = \sum_{\substack{1 \leq j \leq n \\ j \neq q}} M_{pj}x_j \ .$$

Imagine we pick the bits of x according to successive tosses of a fair coin, and the toss deciding the value of x_q is made last. (Why can we make this assumption?) Just before x_q is picked, the value of s is already determined. Once x_q is picked, we either leave s unchanged (if $x_q = 0$) or add M_{pq} to s (if $x_q = 1$). Since $M_{pq} \neq 0$, we can't have both $s = 0$ and $s + M_{pq} = 0$. Thus in at least one of the two cases $w_p \neq 0$. So $w_p \neq 0$ with probability at least $1/2$, and we are done. ■

Pretty neat, huh? Here's another exercise, which you can easily solve by the above reasoning, but which I'd like you to solve by appealing to Corollary 2.11.

Exercise 6.2 *Let $\mathbb{F}$ be a finite field, and M be an $n \times n$* nonzero *matrix with entries from $\mathbb{F}$.*

1. *Prove that there exists a column vector $x \in \mathbb{F}^n$ such that $Mx \neq 0$.*
2. *Prove the stronger fact that at most a $1/|\mathbb{F}|$ fraction of column vectors $x \in \mathbb{F}^n$ satisfy $Mx = 0$.*