Phys 412-1 Quantum Mechanics

(Dated: 2022 Fall)

Abstract

This document contains the TA notes for the course Phys 412-1 Quantum Mechanics in the fall quarter of the academic year 2022-2023.

CONTENTS

I. DISCUSSION 1 (SEP 29)

What is an observable? It is an *abstract operator* which can be *represented* in a matrix in terms of some entries using a suitable *basis*. Suppose we have an observable A. The matrix representation of A is hermitian. We need to know what we are talking about when we say a matrix is hermitian.

Hermitian matrices have the three important properties:

- They have *real eigenvalues*.
- Their *eigenvectors* are *orthogonal* [1].
- Their eigenvectors form a *complete basis*.

The last item is purely mathematical but it is physically useful.

Now suppose I want to write down the eigenvalue equation for my operator A:

$$A|n\rangle = a_n|n\rangle \tag{1}$$

I'll be very simplistic in my notation and just put the quantum number into my kets and bras. A more proper notation is for the eigenkets is $|A;n\rangle$; namely, you'd indicate the operator, as well. Now, here we know that $a_n \in \mathbb{R}$ for all n [2]. As for the eigenkets, we know that they are orthogonal. But let's not worry about the magnitudes, so suppose everything is normalized. Then, we have the orthonormality relation

$$\langle n|m\rangle = \delta_{nm} \tag{2}$$

Let's talk about the completeness now. But to do that, I'll introduce the *projection operator*, Λ_n. Suppose I write down an operator like $|n\rangle\langle m|$. What does this guy do for a living?

It is clearly not a ket vector due to the bra on the right. It is a tensor, which can be represented by some matrix in a proper basis. Consider the operator $|n\rangle\langle n|$. What does this do? If I give you a generic state ket, say $|\psi\rangle$ and tell you to act this operator on this state, you'd get $|n\rangle\langle n|\psi\rangle$. The second factor, i.e. $\langle n|\psi\rangle$, is certainly a scalar. We have a ket, $|n\rangle$, next to it, so the operator $|n\rangle\langle n|$ most certainly produces this ket, times some complex number. We call this number, the *probability amplitude* (or the *transition amplitude*). It measures *how much* $|n\rangle$ *is contained in* $|\psi\rangle$, for lack of a better explanation. This is no different than the usual projection of vectors. Suppose I have this vector $\boldsymbol{v}$ and I want to project it onto an axis in the direction of some other vector $\boldsymbol{n}$. Then, we express this new, projected vector as $(\boldsymbol{v}\cdot\hat{\boldsymbol{n}})\hat{\boldsymbol{n}}$, where I normalized my direction vector. This object, $\langle n|\psi\rangle$ has 1:1 correspondence to the object $\hat{\boldsymbol{n}}\cdot\boldsymbol{v}$.

The moral of the story is, $|n\rangle\langle n|$ projects any state onto the state $|n\rangle$. We define our projection operator then as

$$\Lambda_n := |n\rangle\langle n| \tag{3}$$

Let's study some properties of this projection operator. What happens if I act $\Lambda_n\Lambda_m$ ($m \neq n$) on some general state $|\psi\rangle$? Without doing explicit mathematics, we immediately see that we get zero. The reason is, we project our initial state onto $|m\rangle$ first, but then we try to project this ket onto $|n\rangle$. We cannot do that since our eigenkets are orthogonal. Let's prove this mathematically:

$$\Lambda_n\Lambda_n|\psi\rangle = |n\rangle\langle n|m\rangle\langle m|\psi\rangle \tag{4}$$

This is equal to zero because $\langle n|m\rangle = 0$. Suppose now I act Λ_n on $|\psi\rangle$ twice. What do I get? I get $|n\rangle$ again. Once I'm in this state, I don't go anywhere. So we may write

$$\Lambda_n\Lambda_m = \Lambda_n\delta_{nm} \tag{5}$$

Now let's go back to the completeness. Suppose I have this vector $\boldsymbol{v}$ in $3D$. How do I get the components along the Cartesian axes? I take the dot product of the vector and tack in the unit vector in that direction. So I'm actually projecting my vector onto all these axes. I also know that the three-dimensional real space is spanned by $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$ *completely*. There is no vector left out. You can express any vector using these unit vectors. An identical story takes place here. If I have a general state $|\psi\rangle$, then, in order to be able express it

using the eigenkets of this operator A, I need to project it onto each and every one of these eigenkets:

$$|\psi\rangle = |1\rangle\langle 1|\psi\rangle + |2\rangle\langle 2|\psi\rangle + \cdots = \sum_n |n\rangle\langle n|\psi\rangle = \sum_n \Lambda_n |\psi\rangle \tag{6}$$

We have discovered something now. Notice that we have $|\psi\rangle$ on both sides, namely a *vector* equal to itself with some operator in front. This can mean only one thing:

$$\sum_n \Lambda_n = \mathbb{1} \tag{7}$$

This is called the *completeness* relation. Inserting a sum of eigenkets and bras between two operators, which we'll do every now and then, is called the *resolution of identity*.

* * *

I like my projection operators because that's how I (used to) compute my transitions (and hopefully we'll do it together in the third quarter). If I want to measure the probability of an event taking place between some initial and final states, I immediately write down the transition amplitude as

$$T_{f\leftarrow i} = \langle f|\Lambda|i\rangle \tag{8}$$

where Λ is effectively the projection operator that causes this transition. By squaring this element, we'll get the probability. This may seem a bit vague but it will become much clearer when we start talking about time-dependent phenomena.

Let's do a little exercise now in which I'll stress the importance of these projections operators again. Suppose I have this function f and, whatever it is, I want to evaluate it at my observable A represented in a suitable basis. What can I do?

Whenever in doubt, expand in a power series. That's one of my top quotes by anyone on this planet. Fermi said that, and that's what we are going to do here. Suppose I pretend to expand my function f in a Taylor series but I'll do it in a vague manner, i.e. I'll absorb all the factors into a constant, c_j, and just focus on the expansion parameter, namely our operator, A:

$$f(A) = \sum_{j\geq 0} c_j A^j \tag{9}$$

Now, I want to express my operator in terms of *something*. That something will turn out to be its eigenvalues and eigenkets. Suppose I take my operator A and multiply it by 1:

$$A = A1 = A\left[\sum_n \Lambda_n\right] = A\sum_n |n\rangle\langle n| \tag{10}$$

Then I know how to act A on the ket:

$$A = \sum_n a_n |n\rangle\langle n| \tag{11}$$

This is called the *spectral decomposition* of an operator in terms of its eigenvalues and eigenkets. Let's take the powers of this operator:

$$A = \sum_n a_n |n\rangle\langle n| = \sum_n a_n \Lambda_n \tag{12}$$

$$A^2 = \left[\sum_n a_n |n\rangle\langle n|\right]\left[\sum_m a_m |m\rangle\langle m|\right] = \sum_n\sum_m a_n a_m |n\rangle \underbrace{\langle n|m\rangle}_{\delta_{nm}}\langle m| = \sum_n a_n^2 |n\rangle\langle n| = \sum_n a_n^2 \Lambda_n \tag{13}$$

or, let's use the properties of the projection operator:

$$A^3 = \left[\sum_n a_n \Lambda_n\right]\left[\sum_m a_m \Lambda_m\right]\left[\sum_k a_k \Lambda_k\right] = \sum_n\sum_m\sum_k a_n a_m a_k \underbrace{\underbrace{\Lambda_n \Lambda_m}_{\Lambda_n \delta_{nm}} \Lambda_k}_{\Lambda_n \delta_{nm}\delta_{nk}} = \sum_n a_n^3 \Lambda_n = \sum_n a_n^3 \Lambda_n \tag{14}$$

So we get the idea:

$$A^j = \sum_n a_n^j \Lambda_n \tag{15}$$

With this, our series expansion becomes

$$f(A) = \sum_{j\geq 0} c_j \left[\sum_n a_n^j \Lambda_n\right] = \sum_n \left[\sum_{j\geq 0} c_j a_n^j\right] \Lambda_n = \sum_n f(a_n)\Lambda_n \tag{16}$$

This way, we generalize the spectral decomposition into any functional form. Now, we don't need to worry about the operator itself. We just deal with the projections, which are easier to work with.

* * *

Now let's do an example on representations and measurements. Suppose I give you an observable A with the following properties:

$$\langle 1|A|1\rangle = a \tag{17}$$

$$\langle 1|A|2\rangle = b \tag{18}$$

$$\langle 2|A|1\rangle = b \tag{19}$$

$$\langle 2|A|2\rangle = a \tag{20}$$

Then, I give you a general state:

$$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|1\rangle + i|2\rangle\right) \tag{21}$$

The question is, what are the probabilities that if I measure A on this state (say, the energy or the spin or any other observable if you want to leave this abstract realm), I get a, b, $a+b$, $a-b$ (in separate measurements, not within the same measurement)?

$$* * *$$

Measurement is nothing but taking projections. Using the projection operator and just the very basic assumptions about a measurement, Schwinger created a *measurement formalism* of quantum mechanics, which turned out to be identical to Schrödinger's wavefunction approach and Heisenberg's matrix mechanics.

We start with the *spectrum* of this operator A. Here, spectrum is just a fancy way of saying its eigenvalues and eigenstates. At this point, we are going to resort to matrices because it is highly convenient.

Now, we are given the A *sandwiches* above. What do we do with them? They appear to be the matrix elements of the operator A in the $\{|1\rangle, |2\rangle\}$ basis. So we may write

$$A \doteq \begin{pmatrix} a & b \\ b & a \end{pmatrix} \tag{22}$$

where $\doteq$ is read *can be represented by*. I just put a in the $(1,1)$ and $(2,2)$ entries and b in the $(1,2)$ and $(2,1)$ entries. How did I do that? This is a step that's usually taken without saying but it's crucial to be aware of that. We have tacitly assumed the representations

$$|1\rangle \doteq \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |2\rangle \doteq \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{23}$$

Why? Because it's convenient. Now, I want to obtain the eigenvalues and the eigenvectors of this matrix. Let's use Mathematica [3].

```
A = {
    {a, b},
    {b, a}
};
```

Mathematica is row-major, so we enter the components row by row as we write them on paper. The newlines and the spacing that I'm using here are totally optional. Note the semicolon at the end. It just suppresses the output if you are working with a Mathematica notebook. Let's now play with this.

```
EigenValues[A]
EigenVectors[A]
```

When you evaluate this, the first output should be like

```
{a - b, a + b}
```

and the second output is

```
{{-1, 1}, {1, 1}}
```

So the eigenvalues are

$$\lambda_{\pm} = a \pm b \tag{24}$$

The unnormalized eigenvector corresponding to the *plus* eigenvalue is

$$|\widetilde{+}\rangle \doteq \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{25}$$

and the unnormalized eigenvector corresponding to the *minus* eigenvalue is

$$|\widetilde{-}\rangle \doteq \begin{pmatrix} -1 \\ 1 \end{pmatrix} \tag{26}$$

I'm going to normalize the *plus* eigenket by $1/\sqrt{2}$ and the *minus* eigenket by $-1/\sqrt{2}$. There is nothing that prevents me from choosing some other factor—even a complex one—but this is what I like. So, the normalized eigenkets are

$$|\pm\rangle \doteq \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ \pm 1 \end{pmatrix} \tag{27}$$

But remember these are just representations. We don't want to work with representations because they are just tools. Let's express the eigenkets in the basis $\{|1\rangle, |2\rangle\}$—even though they look abstract, it's these states that are our reality:

$$|\pm\rangle \doteq \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \pm \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}}|1\rangle \pm \frac{1}{\sqrt{2}}|2\rangle = \frac{|1\rangle \pm |2\rangle}{\sqrt{2}} \tag{28}$$

With this, let's perform a spectral decomposition of the observable A:

$$\begin{aligned} A &= \sum_n a_n |n\rangle\langle n| \\ &= (a+b)|+\rangle\langle +| + (a-b)|-\rangle\langle -| \end{aligned} \tag{29}$$

where the projection operator to the *plus* eigenstate is given by

$$\Lambda_+ = |+\rangle\langle +| = \frac{|1\rangle + |2\rangle}{\sqrt{2}} \frac{\langle 1| + \langle 2|}{\sqrt{2}} \tag{30}$$

and the projection operator to the *minus* eigenstate is given by

$$\Lambda_- = |-\rangle\langle -| = \frac{|1\rangle - |2\rangle}{\sqrt{2}} \frac{\langle 1| - \langle 2|}{\sqrt{2}} \tag{31}$$

Finally, let's do some measurement. The spectral decomposition tells you what you can get if you try to measure the observable A on any state. You can either get $a+b$ or $a-b$. There is no other possibility. In a way, you can only measure what is in your spectrum [4]. Therefore, without doing any other calculation, we can directly conclude that the probability of measuring A to be a or b on any state is 0.

We can get $a+b$ and $a-b$, so let compute the probability amplitudes. We have

$$\Lambda_+|\psi\rangle = \frac{|1\rangle + |2\rangle}{\sqrt{2}} \frac{\langle 1| + \langle 2|}{\sqrt{2}} \frac{|1\rangle + i|2\rangle}{\sqrt{2}} \tag{32}$$

Noting that $\langle 1|2\rangle = 0$ and $\langle 1|1\rangle = \langle 2|2\rangle = 1$ (because that's how we chose these basis vectors), we get

$$\Lambda_+|\psi\rangle = |+\rangle \frac{1+i}{2} \tag{33}$$

so the probability of measuring A to be $a+b$ is

$$P(a+b) = \left|\frac{1+i}{2}\right|^2 = \frac{1}{2} \tag{34}$$

It is just a cute exercise of taking the inner product of 1's and 2's to show that measuring A to be $a-b$ has the same probability (so that the total probability is 1).

* * *

At some point in this discussion session, I mentioned that you can do a lot of amazing things on Mathematica other than just symbolic calculation. I claimed that you can even compose music with it. Just for the sake of fun, here is a Mathematica notebook where you can create a random composition, which you then may extract as a .wav or .mp3 file using the `Export` command.

`https://github.com/kagsimsek/random_compose` (35)

If you run it a couple of times, a good melody will come out eventually.

II. DISCUSSION 2 (OCT 6)

Let's do some index manipulation. We start with the well-known mathematical objects that carry some indices:

- Position vector, momentum vector, electric-field vector, etc.: v_i
- The moment of inertia tensor, the quadrupole moment tensor, the field-strength tensor, etc.: T_{ij}
- The unit symmetric tensor, i.e. the Kronecker delta: δ_{ij} (given by `KroneckerDelta[i, j]` on Mathematica)
- The *totally antisymmetric structure constants*: f_{ijk}. Note that the Levi-Civita tensor falls into this class. It's just the structure constant of $SU(2)$, if you want some formalism. Note that *totally symmetric* means here that if you flip any pair of indices, your tensor will receive a minus sign. With this in mind, we can safely say that if any of the two indices are the same, the value of such tensors is zero for that component; i.e. $\epsilon_{112} = -\epsilon_{112}$, where I flipped the 1's, so we have $\epsilon_{112} = 0$. (The Levi-Civita tensor is a built-in tensor on Mathematica, as well. We write ϵ_{ijk} as `LeviCivitaTensor[3][[i, j, k]]`, where 3 here means that we are in the three-dimensional space.)

Now, we say that quantities with only one index are rank-1 tensor, those with two are rank-2, etc. if you want even more formalism. But that's just a language. The important point is the ranges of these indices, which define the size of the matrices that we can form to represent these objects. For an elementary vector, we have i running from 1 to 3 in the good ol' Cartesian space. But why not generalize this? We can have an infinitely large vector space if we let i run from 1 to infinity. That's important to keep in mind but for what follows, we won't really dig into that.

Let's do some index manipulation. We start with the dot product of two vectors. Suppose I have $\boldsymbol{a}$ and $\boldsymbol{b}$ with the same size, i.e. suppose they are both in $\mathbb{R}^n$. Then,

$$\boldsymbol{a} \cdot \boldsymbol{b} = a_1 b_1 + \cdots a_n b_n = \sum_{i=1}^{n} a_i b_i \tag{36}$$

Here, we have expressed the same quantity using an explicit summation over the indices. i here—or any index that's being summed over—is called a *dummy* index. With this, we are ready to drop the summation symbol:

$$\boldsymbol{a} \cdot \boldsymbol{b} = a_i b_i \tag{37}$$

How comfortable is that! Einstein once that his greatest contribution to physics was to drop this summation symbol over repeated indices. By convention, we assume any two repeated indices are meant to be summed over. If you see something like $a_i b_i c_i$, be careful. We need to be aware of the context here, so I won't go into that.

Now, let's do a cross product. Suppose I have $\boldsymbol{a}$ and $\boldsymbol{b}$ in the familiar three-dimensional space. Then, their cross product is given by the following determinant:

$$\boldsymbol{a} \wedge \boldsymbol{b} = \det \begin{pmatrix} \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{\boldsymbol{z}} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{pmatrix} \tag{38}$$

Note the use of the wedge product here. In most cases, it is used interchangeably by the traditional $\times$ symbol but the latter looks like x, so I prefer the former. Let's expand this cross product:

$$\boldsymbol{a} \wedge \boldsymbol{b} = \hat{\boldsymbol{x}}(a_2 b_3 - a_3 b_2) + \hat{\boldsymbol{y}}(a_3 b_1 - a_1 b_3) + \hat{\boldsymbol{z}}(a_1 b_2 - a_2 b_1) \tag{39}$$

Suppose I want the first component of this cross product.

$$(\boldsymbol{a} \wedge \boldsymbol{b})_1 = a_2 b_3 - a_3 b_2 \tag{40}$$

We have this beautiful cyclic pattern here. Whenever we see something like this, it's best to use resort to the Levi-Civita tensor. Now, I just want to test this out. Let's try the following:

$$(\boldsymbol{a} \wedge \boldsymbol{b})_1 \stackrel{?}{=} \epsilon_{1jk} a_j b_k \quad \text{(sum implied)} \tag{41}$$

For this summation, we say that j and k runs from 1 to 3. Let's expand this:

$$(\boldsymbol{a} \wedge \boldsymbol{b})_1 = \epsilon_{111} a_1 b_1 + \epsilon_{112} a_1 b_2 + \epsilon_{113} a_1 b_3 + \ldots + \epsilon_{133} a_3 b_3 \tag{42}$$

We have nine terms here. But most of them are zero. We want to remember the convention that $\epsilon_{123} = \epsilon_{231} = \epsilon_{312} = 1$ and that if you flip any indices, you'll get -1, and that

if any of the indices are repeated, you get zero. So, we have only two nontrivial terms in this summation, for which all the indices are different:

$$(\boldsymbol{a} \wedge \boldsymbol{b})_1 = \epsilon_{123} a_2 b_3 + \epsilon_{132} a_3 b_2 \tag{43}$$

The first Levi-Civita is just $+1$ and the second one is -1. So we have

$$(\boldsymbol{a} \wedge \boldsymbol{b})_1 = a_2 b_3 - a_3 b_2 \tag{44}$$

It seems to be working. That's what we want, then. Notice how the fact that the cross product is anticommutative is consistent here with the definition of the Levi-Civita tensor:

$$(\boldsymbol{b} \wedge \boldsymbol{a})_1 = b_2 a_3 - b_3 a_2 = -(a_2 b_3 - a_3 b_2) = -\epsilon_{1jk} a_j b_k = -(\boldsymbol{a} \wedge \boldsymbol{b})_1 \tag{45}$$

Next, let's *contract* $\boldsymbol{a} \wedge \boldsymbol{b}$ with another vector. Here—and most of the time—*contraction* means taking the dot product to produce a scalar. Let's investigate the *triple product*. The question is, how would you indexify $\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c}$? Notice that I don't have to put some parentheses to emphasize the multiplication order, it's just what it is; namely, $\boldsymbol{a} \wedge (\boldsymbol{b} \cdot \boldsymbol{c})$ is a meaningless quantity because $\boldsymbol{b} \cdot \boldsymbol{c}$ is a scalar and we can't take the cross of of a scalar with a vector.

We start with the dot product because it's easier to expand, compared to the cross product. Suppose I define $\boldsymbol{u} := \boldsymbol{a} \wedge \boldsymbol{b}$ for a moment to write

$$\boldsymbol{u} \cdot \boldsymbol{c} = u_i c_i \tag{46}$$

I'm going to insert a Kronecker delta here:

$$\boldsymbol{u} \cdot \boldsymbol{c} = \delta_{ij} u_i c_j \tag{47}$$

Why does it work? Well, let's expand it to confirm:

$$\boldsymbol{u} \cdot \boldsymbol{c} = \delta 11 u_1 c_1 + \delta_{12} u_1 c_2 + \delta_{13} u_1 c_3 + \cdots + \delta_{33} u_3 c_3 \tag{48}$$

But all the terms expect for the ones that go like δ_{11}, δ_{22}, and δ_{33} drop out because of the properties of this tensor:

$$\boldsymbol{u} \cdot \boldsymbol{c} = \delta_{11} u_1 c_1 + \delta_{22} u_2 c_2 + \delta_{33} u_3 c_3 \tag{49}$$

But now all these delta terms are just 1:

$$\boldsymbol{u} \cdot \boldsymbol{c} = u_1 c_1 + \boldsymbol{u}_2 c_2 + \boldsymbol{u}_3 c_3 \tag{50}$$

This is like inserting an identity matrix in between when we express these vectors in arrays:

$$\boldsymbol{u} \cdot \boldsymbol{c} = \begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} \tag{51}$$

So, we are cool on this. Let's get back to the triple product:

$$\boldsymbol{u} \cdot \boldsymbol{c} = \delta_{ij} u_i c_j \tag{52}$$

Now let's switch back to $\boldsymbol{a}$ and $\boldsymbol{b}$:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \delta_{ij} (\boldsymbol{a} \wedge \boldsymbol{b})_i c_j \tag{53}$$

We will now express the i^{th} component of this cross product using the Levi-Civita but we have to be extra careful with all the other indices lurking around. We can't use j anywhere else. Let's use the next couple of letters:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \delta_{ij} (\epsilon_{ik\ell} a_k b_\ell) c_j = \delta_{ij} \epsilon_{ik\ell} a_k b_\ell c_j \tag{54}$$

How many summations do we have here? Well, there is no *free* index, meaning there is no index on the left-hand side of this equality, so all the indices are *dummy*. We have these indices i, j, k, and ℓ, so all four of them are meant to be summed over from 1 to 3:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \sum_{i=1}^{3} \sum_{j=1}^{3} \sum_{k=1}^{3} \sum_{\ell=1}^{3} \delta_{ij} \epsilon_{ik\ell} a_k b_\ell c_j \tag{55}$$

Now, what does this δ_{ij} do for a living? It just replaces indices. You have the option to let $i \to j$ or $j \to i$ in all the terms that are multiplied by this Kronecker delta. Then you drop the delta term. I think I'll just let $j \to i$:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{ik\ell} a_k b_\ell c_i \tag{56}$$

Now, this looks hideous. We have all these indices but they are successive letters. For some unknown reason, I find this disturbing. We note that, since these are all dummy

indices, meaning they will not be used anymore once we carry out the summation, I can relabel them as I wish. N.B. this relabeling is important only when you have some free indices around. Otherwise, you can just go crazy on them however you want. Here, I want to have i under a, j under b, and k under c. That is, I want to let $k \to i$, $\ell \to j$, and $i \to k$ (or if you want, you can always define some intermediate dummy variables by putting some primes on them before doing these replacements). Let's see what we get:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{kij} a_i b_j c_k \tag{57}$$

Well, now it looks better. But we have this epsilon messed up. I want to bring it to the alphabetical order, as well—just for the sake of visual pleasure, nothing more. I want to have ϵ_{ijk}, so first, I flip k and i to have $-\epsilon_{ikj}$, where the minus sign comes from this flipping, and then I flip k and j to get $-(-\epsilon_{ijk})$. Here, the second minus sign comes from the second flipping. Then we get

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{ijk} a_i b_j c_k \tag{58}$$

Notice how everything is in order here.

Now, I want to play around with these indices to prove a useful identity. Suppose I shift my indices in a cyclic manner, i.e. I let $i \to j$ and $j \to k$ and $k \to i$. Again, I can do that because these are all dummy indices:

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{jki} a_j b_k c_i \tag{59}$$

Let's reorder these vector components here so that the indices are ordered alphabetically, namely

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{jki} c_i a_j b_k \tag{60}$$

Good one. Now I want to bring the indices of the epsilon to the alphabetical order, as well: $\epsilon_{jki} = -\epsilon_{jik} = +\epsilon_{ijk}$. Now we have

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \epsilon_{ijk} c_i a_j b_k \tag{61}$$

If we compare this to $\epsilon_{ijk} a_i b_j c_k = \boldsymbol{a} \wedge \boldsymbol{b} \cdot c$, we see that the right-hand side can be written compactly as

$$\boldsymbol{a} \wedge \boldsymbol{b} \cdot \boldsymbol{c} = \boldsymbol{c} \wedge \boldsymbol{a} \cdot b \tag{62}$$

If you want, try showing that this is also equal to $\boldsymbol{b} \wedge \boldsymbol{c} \cdot \boldsymbol{a}$.

These types of manipulations are essential when you have some objects with indices in an expanded form (as opposed to a compact one) because you'll want to recognize the patterns of contractions.

Let's do some exercises before we connect these to the Pauli matrices and all the other good stuff. I'll just to some basic examples and see where we get with them.

- $x_i x_i = \boldsymbol{x} \cdot \boldsymbol{x} = |\boldsymbol{x}|^2$
- $x_i y_j \delta_{ij} = \boldsymbol{x} \cdot \boldsymbol{y}$
- $(a_i)_r (a_j)_s \epsilon_{rst} = (\boldsymbol{a}_i \wedge \boldsymbol{a}_j)_t$: This will turn out to be important if you want to solve the last problem of the first homework with index manipulation. Note that I pulled a trick on you by not saying that what a_i is. It's not the i^{th} component of some vector $\boldsymbol{a}$. In reality, we have three $\boldsymbol{a}_i$ vectors and it's just one of them. Otherwise, the cross product would be meaningless.
- Let's make it spicier by including the derivative. There is this Laplace identity in electromagnetic theory. Suppose you have some magnetic field, $\boldsymbol{B}$, and you are interested in its curl, i.e. $\boldsymbol{\nabla} \wedge \boldsymbol{B}$. We want to express the magnetic field in terms of the magnetic potential, $\boldsymbol{A}$, which satisfies $\boldsymbol{B} = \boldsymbol{\nabla} \wedge \boldsymbol{A}$. Thus, we want to actually simplify $\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})$. Notice the importance of parentheses here. I can't just omit them because the order is important here.

  Now, the question is, what is this object? In terms its rank maybe? It's a certainly a vector. So, we can investigate its i^{th} component:

  $$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i \tag{63}$$

  We start with the outermost objects and expand our way inward:

  $$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \epsilon_{ijk} \nabla_j (\boldsymbol{\nabla} \wedge \boldsymbol{A})_k \tag{64}$$

  (N.B. some people write $\nabla_i = \partial_i$ and that's totally fine.) I've taken my Levi-Civita with i as the first index, and then my first vector here, which is the derivative, with an index that matches the second index of the Levi-Civita, and then I have my second vector—whatever it is, it's a vector—with an index that matches the

third index of the Levi-Civita. So far, so good. Let's now expand this second cross product. I have to be careful to choose indices different than i (which is a free index here because it appears on both sides of the equality) and j and k. But we are interested in the k^{th} component of this vector, so our Levi-Civita starts with a k. And I'm going to pick r and s as my new indices, why not:

$$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \epsilon_{ijk} \nabla_j (\epsilon_{krs} \nabla_r A_s) \tag{65}$$

Remember that ϵ_{ijk} or ϵ_{krs} are just 1's and 0's—they are just numbers. They come out of derivative:

$$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \epsilon_{ijk} \epsilon_{krs} \nabla_j \nabla_r A_s \tag{66}$$

We have something interesting here: two Levi-Civitas but with one common index. Notice that I can write $\epsilon_{krs} = -\epsilon_{rks} = +\epsilon_{rsk}$, so I have

$$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \epsilon_{ijk} \epsilon_{rsk} \nabla_j \nabla_r A_s \tag{67}$$

(Confirm that you can write $\epsilon_{ijk}\epsilon_{rsk} = \epsilon_{kij}\epsilon_{krs}$ because some people like to see the common index as the first index. I don't know why.) We have this beautiful theorem for this contraction (N.B. the contraction is over k only. It's still a dot product but for only some parts of these tensors or matrices.):

$$\epsilon_{ijk}\epsilon_{imn} = \det \begin{pmatrix} \delta_{jm} & \delta_{km} \\ \delta_{jn} & \delta_{kn} \end{pmatrix} = \delta_{jm}\delta_{kn} - \delta_{km}\delta_{jn} \tag{68}$$

Notice how we've just managed to get rid of these antisymmetric tensors carrying three indices and now we just have the Kronecker deltas [5]. The Levi-Civita takes the cross product but the Kronecker just replaces indices. The latter has the easiest job in the world. Let's finish our calculation now:

$$\begin{aligned} [\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i &= (\delta_{ir}\delta_{js} - \delta_{jr}\delta_{is}) \nabla_j \nabla_r A_s \\ &= \delta_{ir}\delta_{js} \nabla_j \nabla_r A_s - \delta_{jr}\delta_{is} \nabla_j \nabla_r A_s \end{aligned} \tag{69}$$

In the first term, using our Kroneckers, we'll replace r by i and s by j, and in the second term, we'll do the other way around:

$$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \nabla_j \nabla_i A_j - \nabla_j \nabla_j A_i$$

$$= \nabla_i \nabla_j A_j - \nabla_j \nabla_j A_i \tag{70}$$

where, in the second line, I've just flipped the order of the derivatives as they commute. Now, what's the contraction pattern here, meaning who talks to whom? In the first term, the second derivative talks to the A vector because they have the same index, so we have a dot product in between. In the second term, the derivatives will talk to each other, i.e. we'll get the Laplacian, $\nabla^2 = \boldsymbol{\nabla} \cdot \boldsymbol{\nabla}$:

$$[\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A})]_i = \nabla_i (\boldsymbol{\nabla} \cdot \boldsymbol{A}) - \nabla^2 \boldsymbol{A}_i \tag{71}$$

so, in the full vector notation now, without just the i^{th} index, we have

$$\boldsymbol{\nabla} \wedge (\boldsymbol{\nabla} \wedge \boldsymbol{A}) = \boldsymbol{\nabla}(\boldsymbol{\nabla} \cdot \boldsymbol{A}) - \nabla^2 \boldsymbol{A} \tag{72}$$

- Here are some little exercises:
  - Show that $v_i v_j$ is a tensor symmetric with respect to i and j.
  - Show that the contraction of a symmetric tensor with an antisymmetric one over their symmetric and antisymmetric indices is zero; to be more precise, i.e. show that $\epsilon_{ijk} v_i v_j = 0$. (This explains why $\boldsymbol{v} \wedge \boldsymbol{v} = 0$.)
  - Prove that divergence of curl of a vector is zero.
  - Prove that curl of a gradient of a scalar is zero.

  Just to visualize, I mean, show that $\boldsymbol{\nabla} \cdot \boldsymbol{\nabla} \wedge V = 0$ and $\boldsymbol{\nabla} \wedge \boldsymbol{\nabla} f = 0$ using the index manipulation.

---

[1] This is true in the absence of degeneracies. When we have degeneracies, we say that the eigenvectors corresponding to the nondegenerate subset are orthogonal.

[2] Suppose we are working in an infinite-dimensional basis.

[3] To get a copy of Mathematica, go to https://www.it.northwestern.edu/software/mathematica-fac/index.html and click Students Wolfram Activation Key Request Form site under Activation and Download.

[4] But when you perform lots of measurement, your *average* may be some other value, depending on your state.

[5] The extension to the most general case is trivial, i.e. when we have no common indices:

$$\epsilon_{ijk}\epsilon_{rst} = \det \begin{pmatrix} \delta_{ir} & \delta_{jr} & \delta_{kr} \\ \delta_{is} & \delta_{js} & \delta_{ks} \\ \delta_{it} & \delta_{jt} & \delta_{kt} \end{pmatrix} \tag{73}$$

Yes, you will end up more terms than you could imagine, but all of them are deltas. Just expand your terms and start replacing indices. That's what this tells you.