CSE610 Week 2: Hilbert Spaces, Tensor Products, and Operators

In a calculus or linear algebra course you have likely encountered the spaces $\mathbb{R}^2$ of points (a, b) in the plane and $\mathbb{R}^3$ of points (x, y, z) or (x_1, x_2, x_3) in 3-dimensional space. Then $\mathbb{R}^n$ means n-dimensional real space, whether you called it a vector space or not. Maybe you also covered the complex vector spaces $\mathbb{C}^n$ or specialized to vectors of rational numbers---which make the vector space $\mathbb{Q}^n$. When we care more about the "space" aspect than the particular kind of numbers allowed, we use the umbrella term "Hilbert space" after the mathematician David Hilbert. That term is often employed by physicists not only to avoid having to specify the dimension n but also to allow it to be infinite. We, however, will stay in finite dimensional spaces and care a lot about what the dimension is.

The usual rule for the *product* of two vector spaces is to add the dimensions. Thus a member of $\mathbb{R}^2 \times \mathbb{R}^3$, which formally is an ordered pair like $((a, b), (x, y, z))$, is considered the same as the 5-tuple (a, b, x, y, z), which we could re-label as $(x_1, x_2, x_3, x_4, x_5)$. So

$$\mathbb{R}^2 \times \mathbb{R}^3 = \mathbb{R}^5.$$

The **tensor product**, however, multiplies the dimensions. When defined between our vectors (a, b) and (x, y, z), it doesn't just ram them together. Instead it combines a copy of the second vector into each part of the first vector. In symbols:

$$(a, b) \otimes (x, y, z) = (a \cdot (x, y, z),\ b \cdot (x, y, z)) = (ax, ay, az, bx, by, bz) .$$

The vectors we get have dimension 6 not 5. We will see that not every vector in the target space, here $\mathbb{R}^6$, arises as a tensor product. But if we close out $\mathbb{R}^2 \otimes \mathbb{R}^3$ under linear combinations, then we do get all of $\mathbb{R}^6$.

What does tensor product *do*? We feel again that good intuition comes by thinking about abstract attributes first, numbers later. Let us make the card suits into the abstract "attribute vector"

$$u = (\clubsuit, \diamondsuit, \heartsuit, \spadesuit).$$

And the ranks of cards becomes the attribute vector

$$v = (2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A) .$$

Then the tensor product---in the order $u \otimes v$ (it's not commutative)---is

$$w = (\clubsuit 2, \clubsuit 3, \ldots, \clubsuit K, \clubsuit A, \diamondsuit 2, \diamondsuit 3, \ldots, \diamondsuit A, \heartsuit 2, \ldots, \heartsuit A, \spadesuit 2, \ldots, \spadesuit A) .$$

This sorts the deck by suits. If we tensored the other way around, we'd get

$$v \otimes u = (2\clubsuit, 2\diamondsuit, 2\heartsuit, 2\spadesuit, 3\clubsuit, 3\diamondsuit, 3\heartsuit, 3\spadesuit, 4\clubsuit, \ldots, 4\spadesuit, 5\clubsuit, \ldots, \ldots, A\clubsuit, A\diamondsuit, A\heartsuit, A\spadesuit) ,$$

which sorts the deck by ranks instead. Always the second vector gets "copied inner" while the first vector is "outer." The ordinary product would have just rammed v after u to give a vector of 17 items, four of type `Suit` and thirteen of type `Rank`. This is inhomogeneous mishmash---like "not playing with a full deck" as we say. Whereas, in either order, the tensor product creates a homogeneous length-52 vector of type "`Suit` and `Rank`." (Between `Suit` and `Rank`, the order might or might not matter.)

Now let's see how this works numerically. The vector $\mathbf{u} = [0, 1, 0, 0]^T$ is the standard basis vector corresponding to "diamonds" in our scheme for suits. For ranks, the seven is indexed by

$$\mathbf{v} \ = \ [0,0,0,0,0,1,0,0,0,0,0,0,0,0] \ .$$

Then

$$\begin{aligned} \mathbf{u} \otimes \mathbf{v} \ &= \ [0 \cdot [0,0,0,0,0,1,0,0,0,0,0,0,0],\ 1 \cdot [0,0,0,0,0,1,0,0,0,0,0,0,0],\ \dots, 0]^T \\ &= \ [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,\ \dots .0]^T. \end{aligned}$$

The single 1 corresponds to the position of the $\Diamond 7$ in the abstract indexing scheme. So we write:

$$\mathbf{u} \otimes \mathbf{v} = |\Diamond 7\rangle \ = \ |\Diamond\rangle|7\rangle.$$

Thus the tensor product of two standard basis vectors gives us a standard basis vector in the larger space. Indeed, we can get the entire standard basis of 52 vectors this way. We've also started writing the "invisible dot" product of kets---which are in quantum coordinates---to stand for the tensor product in the underlying coordinates.

Suppose we next take the tensor product of w with itself. Doing this with the attribute vectors, we get

$$w \otimes w \ = \ (\clubsuit 2 \clubsuit 2, \clubsuit 2 \clubsuit 3, \clubsuit 2 \clubsuit 4,\ \dots, \clubsuit 2 \clubsuit A, \clubsuit 2 \Diamond 2,\ \dots, \clubsuit 2 \spadesuit A,\ \clubsuit 3 \clubsuit 2, \clubsuit 3 \clubsuit 3,\ \dots,\ \dots, \spadesuit A \spadesuit A) \ .$$

What does this represent? It conveys the idea of playing two cards in sequence. This allows them to be the same card---casinos usually play blackjack with eight decks shuffled together, for instance---but we will encounter algorithms whose point is either *amplifying* or eliminating such particular possibilities. The point is that tensor product is the underlying way of Nature simply doing a *sequence*, which means *concatenating* the symbolic representations.

Tensor Product = Simple Concatenation = Flow of Events.

This kind of representation is not just useful in quantum. It underlies the idea of the "TensorFlow" API and library in machine learning.

The common example that matters most is when both spaces have dimension 2. This case is innately confusing because $2+2 \ = \ 2 \cdot 2 \ = \ 2^2$. But hopefully the above will help us avoid confusion.

Two Qubits

In 4-space, the standard basis is given by the vectors:

$$e_0 = (1,0,0,0),\ e_1 = (0,1,0,0),\ e_2 = (0,0,1,0),\ e_3 = (0,0,0,1)\ .$$

The indexing scheme for **quantum coordinates** changes the labels to come from $\{0,1\}^2$ instead of from $\{1,2,3,4\}$, using the canonical binary order $00, 01, 10, 11$. Then we have:

$$e_{00} = (1,0,0,0),\ e_{01} = (0,1,0,0),\ e_{10} = (0,0,1,0),\ e_{11} = (0,0,0,1)\ .$$

The big advantage is that these basis elements are all separable and the labels respect the tensor products involved:

$$\begin{array}{lllllll} |00\rangle = e_{00} = (1,0,0,0) = (1,0)\otimes(1,0) = e_0 \otimes e_0 = |0\rangle\otimes|0\rangle = |0\rangle|0\rangle \\ |01\rangle = e_{01} = (0,1,0,0) = (1,0)\otimes(0,1) = e_0 \otimes e_1 = |0\rangle\otimes|1\rangle = |0\rangle|1\rangle \\ |10\rangle = e_{10} = (0,0,1,0) = (0,1)\otimes(1,0) = e_1 \otimes e_0 = |1\rangle\otimes|0\rangle = |1\rangle|0\rangle \\ |11\rangle = e_{11} = (0,0,0,1) = (0,1)\otimes(0,1) = e_1 \otimes e_1 = |1\rangle\otimes|1\rangle = |1\rangle|1\rangle \end{array}$$

It is OK to picture the tensoring with row vectors, but because humanity chose to write matrix-vector products as Mv rather than vM, they need to be treated as column vectors. This will lead to cognitive dissonance when we read quantum circuits left-to-right but have to compose matrices right-to-left.

We can also take tensor products of non-basis vectors of length 2. Let's us try

$$\mathbf{u} = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \sqrt{0.5}\,[1,1]^T.$$

When we do $\mathbf{u}\otimes\mathbf{u}$, the first thing that happens is that the scalars in front multiply to get $\sqrt{0.5}\cdot\sqrt{0.5} = \frac{1}{2}$ as the multiplier on the whole thing. The vector bodies combine as

$$[1\cdot[1,1],\ 1\cdot[1,1]] = [1,1,1,1]\,.$$

(Strictly speaking, we should do this as column vectors---maybe we'll show on the whiteboard---but it's always fine to do as row vectors and remember to transpose when needed at the end.) So

$$\mathbf{u}\otimes\mathbf{u} = \left[\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2}\right]^T.$$

This is a unit vector. We can do the same with

$$\mathbf{v} = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \sqrt{0.5}[1,-1]^T$$

We get $\mathbf{v} \otimes \mathbf{v} = \frac{1}{2}$ times $[1,-1] \otimes [1,-1] = \frac{1}{2}[1,-1,-1,1]$. OK, $\frac{1}{2}[1,-1,-1,1]^T$ to be strict. We also get:

$$\mathbf{u} \otimes \mathbf{v} = \frac{1}{2}[1,-1,1,-1]^T.$$

$$\mathbf{v} \otimes \mathbf{u} = \frac{1}{2}[1,1,-1,-1]^T.$$

Let's ignore the normalizing multipliers out front for the moment, since they do not matter to the ability to combine the vector bodies. How about the simpler vector

$$\mathbf{w} = [1,0,0,1]?$$

This equals $e_{00} + e_{11}$, i.e., $|00\rangle + |11\rangle$, ignoring the normalizing constant $\sqrt{0.5}$. Can we get this as a tensor product of two vectors of length 2?

The answer is no. We can prove it by representing the general 2-by-2 tensor product as

$$[a,b] \otimes [c,d] = [ac, ad, bc, bd].$$

To get $[1,0,0,1]$ as the result, we need to solve the equations

$$ac = 1,\ ad = 0,\ bc = 0, \text{ and } bd = 1.$$

But $ad = 0$ entails that either a is 0 or d is 0. If $a = 0$, then $ac = 1$ is impossible. But if $d = 0$, then $bd = 1$ is impossible. So there is no solution.

Definition. A vector is **separable** if it can be written as the tensor product of two smaller vectors. Otherwise---and especially when the vector represents a quantum state---we call it **entangled**.

To introduce some more quantum terminology, when a unit vector is not a basis vector, it is necessarily a linear combination of two or more basis vectors. Then it is a **superposition**. One of the amazing verities of physics is that we really can put particle-level sustems into superpositions and interact with them. The math of how those interactions behave involves the kind of vectors we are already seeing. The vectors $\mathbf{u}$ and $\mathbf{v}$ above are superpositions. When we re-interpret the attributes $|0\rangle$ and $|1\rangle$ as $|dead\rangle$ and $|alive\rangle$, then $\mathbf{u}$ becomes the superposition

$$\frac{|dead\rangle + |alive\rangle}{\sqrt{2}}$$

This is said to be the state of **Schrödinger's Cat**. The philosophical issue is whether a macro-level being, not a particle, can be put into superposition. (We will later argue the answer is "yes...but...") For now we prefer to take the simple realist view that a particle can have the state $\mathbf{u}$. It is not a cat, but it has a pet-name, indeed a ket-name: $|+\rangle$. The vector $\mathbf{v}$, with its prominent minus sign, is called $|-\rangle$.

[Lecture moved onto the whiteboard for showing examples of matrices and tensor products of matrices. Here are the main definitions and several examples that were covered.]

Definition. The **conjugate transpose** of an $m \times n$ matrix A is obtained by first transposing A to make the $n \times m$ matrix A^T, and then taking the complex conjugate of every element. We write A^* for the resulting $n \times m$ matrix. (Many other sources write $A^\dagger$ instead.)

Examples: $A = \begin{bmatrix} 1+i & 1-i \\ 2 & 0 \end{bmatrix}, A^T = \begin{bmatrix} 1+i & 2 \\ 1-i & 0 \end{bmatrix}, A^* = \begin{bmatrix} 1-i & 2 \\ 1+i & 0 \end{bmatrix}$.

$$\mathbf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \mathbf{Y}^T = \begin{bmatrix} 0 & i \\ -i & 0 \end{bmatrix}, \mathbf{Y}^* = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = \mathbf{Y} \text{ back again.}$$

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \mathbf{H}^T = \mathbf{H}^* = \mathbf{H} \text{ because } \mathbf{H} \text{ is symmetric and has all-real entries.}$$

This is our first look at the **Hadamard matrix** $\mathbf{H}$. Note that

$$\mathbf{H}|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle$$

$$\mathbf{H}|1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix} = |-\rangle .$$

Thus $\mathbf{H}$ carries the standard basis onto the $|+\rangle, |-\rangle$ basis. It also maps that basis back to the standard one, because $\mathbf{H}^2 = \mathbf{I}$, the 2×2 identity matrix. A vector that looks superposed in the standard basis can be simple in the changed basis. Thus superposition is relative---"in the eye of the beholder" one might say---but in many concrete cases the observer is Nature.

The matrix $\mathbf{Y}$ is one of four named after the quantum phsyicist Wolfgang Pauli. The others are

$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$, and the identity $\mathbf{I}$. Note that $\mathbf{X}|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$ and similarly, $\mathbf{X}|1\rangle = |0\rangle$. Thus applying $\mathbf{X}$ negates the bit label of a standard basis state, and this functions just like the Boolean NOT operation. Moreover, $\mathbf{X}$ is a **permutation matrix**. In upcoming lectures we will show how permutation matrices used in quantum circuits confer exactly the power of classical Boolean circuit gates. The extra quantum power starts coming in with the Hadamard gate. Now for two key definitions (which apply to any size matrices, not just 2×2):

Definition: A matrix A is **unitary** if $A^*A = \mathbf{I}$.

[Added: Note, incidentally, that A must be invertible, and furthermore

$$AA^* = AA^*(AA^{-1}) = A(A^*A)A^{-1} = A\mathbf{I}A^{-1} = AA^{-1} = \mathbf{I}.$$

This also works vice-versa: if $AA^* = \mathbf{I}$, then $A^*A = \mathbf{I}$. So an equivalent definition of unitary is that $AA^* = \mathbf{I}$.]

Definition: A matrix A is **Hermitian** if $A = A^*$.

The Pauli matrices are all both Hermitian and unitary. So is the Hadamard matrix. If we took away the factor $\frac{1}{\sqrt{2}}$, the resulting matrix $\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ (which is often referred to as "the Hadamard natrix" in non-quantum contexts) is Hermitian but not unitary. The matrix $\mathbf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$ is unitary but not Hermitian. In Part I of the text we toe the line of identifying unitary matrices with "legal quantum operations." In Chapter 14 we tend toward the view that Hermitian operators are the real ones. They restore a kind of left-right symmetry that we are about to abandon further here, and even when a Hermitian matrix has complex entries, it is "as good as real" in a sense we will cover later.

Here I finished by defining the tensor product of matrices.

Definition. The **tensor product** of an $m \times n$ matrix A and a $p \times q$ matrix B, written $A \otimes B$, is the $(mp) \times (nq)$ matrix defined schematically in block form as

$$\begin{bmatrix} a_{1,1}B & a_{1,2}B & \dots & a_{1,n}B \\ a_{2,1}B & a_{2,2}B & \dots & a_{2,n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1}B & a_{m,2}B & \dots & a_{m,n}B \end{bmatrix}.$$

The definition of tensor product of vectors is just a special case of this: $n = q = 1$ for column vectors, or $m = p = 1$ for row vectors. Note that unlike the ordinary matrix product AB, which requires $n = p$ for the dimensions to match up and give an $m \times q$ result, the tensor product is defined regardless of what the dimensions are.