The following content is provided under a Creative Commons license. Your support will help MIT OpenCourseWare continue to offer high quality educational resources for free. To make a donation, or to view additional materials from hundreds of MIT courses, visit MIT OpenCourseWare at ocw.mit.edu.

GILBERT STRANG: OK. Just as we're getting started, I thought I'd add a few words about a question that came up after class. Suppose, in that discussion last time, where you were given three-- you were given a distance matrix-- you were given the distance between x1 and x2, between x2 and x3, and between x1 and x3, and you wanted to find points that satisfied that.

Well, we're going to fail on this example, because if the distance here is 1, the distance here is 1, then by the triangle inequality, the distance from x1 to x3 could not be more than 2. And when we square it, it could not be more than 4. And here it's 6. So what's going to happen? What goes wrong in that case? Yeah. I hadn't commented on that, and I'm not sure that the paper that I referenced does so.

So I had to do a little search back in the literature, because people couldn't overlook this problem. So this is the triangle inequality fails. And it's not going to help to go into 10 dimensions, because the triangle inequalities doesn't change. And it's still there in 10 dimensions. And we're still failing. So what happens? Well, what could happen? Do you remember?

And you'll have to remind me, the key equation. You remember, we had an equation connecting the-- so what is the matrix D for this problem? So D is-- this is a 3 by 3 matrix with these distances squared. And it was convenient to use distances squared, because that's what comes into the next steps. So of course, the distance from each x to itself is zero.

The distance from x distance squared was that. This one was that. But this one is 6. OK. So that's the distance matrix. And we would like to find-- the job was to find-- and I'm just going to write down, we cannot find x1, x2, and x3 to match those distances. So what goes wrong? Well, there's only one thing that could go wrong. When you connect this distance matrix D to the matrix X transpose X-- you remember the position matrix-- maybe I called it G? This is giving-- so Gij is the dot product of xi with xj.

Make that into a j. Thank you. So Gij is the matrix of dot product. And the great thing was that we can discover what that matrix-- that matrix G comes directly from D-- comes directly from D. And of course, what do we know about this matrix of cross products? We know that is positive semidefinite. So what goes wrong? Well, just in a word, when we write out that equation and discover what G is, if the triangle inequality fails, we learn that G doesn't come out positive definite.

That's really all I want to say. And I could push through the example. G will not come out positive definite if D-- if that's D because it can't. If it came out positive definite, then we could find an X. So if we had the G, then the final step, you remember, is to find an X. Well we know that if G is positive semidefinite, there are multiple ways to find an X.

This is positive semidefinite matrices is what you get out of X transpose X's. And we can find an x given a G. We can find G given an x. So it has to be that this won't be true-- that the matrix G that comes out of that equation will turn out not to be positive definite. So it's really quite nice. It's a beautiful little bit of mathematics, that if, and only if, the triangle inequality is satisfied by these numbers-- if and only if-- then the matrix in the D matrix-- then the G matrix that comes out of this equation-- which I haven't written-- is positive semidefinite.

If the triangle inequality is OK, we can find the points. If the triangle inequality is violated-- like here-- then the matrix G is not positive semidefinite, has negative eigenvalues, and we cannot find the point. Yeah. I could recall the G matrix but-- the G equation, but it's coming to you in the two page section that does distance matrices. OK. That's just-- I should have made a point. It's nice to have specific numbers.

And I could get the specific numbers for G, and we would see, no way. It's not positive definite. OK. So that's just tidying up last time. I have another small problem to talk about, and then a big question of whether deep learning actually works. I had an email from an expert last night, which changed my view of the world about that question, as you can imagine. The change in my world was, I had thought the answer was yes, and I now think the answer is no. So that's like rather a big issue for 18.065.

But we'll-- let's see about that later. OK. Now Procrustes' problem. So Procrustes-- and it's included in the notes-- that name comes from a Greek myth. Are you guys into Greek myths? So what was the story of Procrustes? Was it Procrustes who adjusted the length of his-- so he had a special bed. Procrustes' bed-- certain length. And then, he had visitors coming. And instead of adjusting the length of the bed to fit the visitor, Procrustes adjusted the length of the visitor to fit the bed. So either stretched the visitor or chopped off part of the visitor.

So anyway-- the Greeks like this sort of thing. OK. So anyway, that's a Greek myth for 18.065. OK. So the whole idea, the Procrustes problem, is to make something fit something else. So the two things are-- so suppose I'm just in three dimensions and I have two vectors here. So I have a basis for a two dimensional space. And over here I have-- people-- space scientists might have one computation of the positions of satellites.

Then, of course, they wouldn't be off by as much as this figure shows. But then they have another computation using different coordinates. So it partly rotated from this picture, but also it's partly got round off errors and error in it between the two. So the question is, what's the best orthogonal transformation? So this is a bunch of vectors, x1, x2, to xn, let's say. And I want to modify them by an orthogonal matrix-- maybe I'd do it on the other side. I think I do. Yeah.

Q, to be as close as possible to this other set, y1, y2 up to yn. So let me just say it again. I have two sets of vectors. And I'm looking, and they're different-- like those two sets. And I'm looking for the orthogonality matrix that, as well as possible, takes this set into this one. Of course, if this was an orthogonal basis, and this was an orthogonal basis, then we would be home free. Q-- we could get equality. We could take an orthogonal basis directly into an orthogonal basis with a orthogonal matrix Q.

In other words, if x was an orthogonal matrix, and y was an orthogonal matrix, we would get the exact correct Q. But that's not the case. So we're looking for the best possible. So that's the problem there-- minimize over orthogonal matrix-- matrices Q. And I just want to get my notation to be consistent here. OK. So I've-- I see that starting with the y's and mapping them to x's-- so let me ask the question. What orthogonal matrix Q multiplies the y's to come as close as possible to the x's?

So over all orthogonal Q's I want to minimize YQ minus X in the Frobenius norm. And I might as well square it. So Frobenius-- we're into the Frobenius norm. Remember the-- of a matrix? This is a very convenient norm in data science, to measure the size of a matrix. And we have several possible formulas for it. So let me call the matrix A. And the Frobenius norm squared-- so what's one expression, in terms of the entries of the matrix-- the numbers Aij in the matrix?

The Frobenius norm just treats it like a long vector. So it's a11 squared, plus a12 squared, of all the way along the first plus second row, just-- I'll say nn squared. OK. Sum of all the squares-- just treating it like a long vector. OK. This-- but that's a awkward expression to write down. So what other ways do we have to find the Frobenius norm of a matrix?

Let's see. I could look at this as A transpose A. Is that right? A transpose A. So what what's happening there? Remind me what-- yeah. I would get all that. I would get all these by taking the matrix A transpose times A. But what-- sorry. I'm not-- I haven't-- I've lost my thread of talk here. So here's-- oh, and then I take the trace, of course. So that first row-- first column of A times that one will give me the-- one set of squares. And then, that one times the other, and the next one, will give me the next set of squares, right?

So this is going to-- if I look at the trace-- so now, let me. So I just want to look at the diagonal here. So it's the trace. You remember, the trace of a matrix-- of a matrix M is the sum down the diagonal M11, M22, down to Mnn. It's the diagonal sum. And-- everybody with me here now? So that term on the diagonal-- A transpose A-- gives me all of that. Then-- or maybe I should be doing AA transpose. The point is, it doesn't matter.

Or the trace of AA transpose. That would be-- those would both give the correct Frobenius norm squared. So traces are going to come into this little problem. Now there's another formula for the Frobenius norm-- even shorter-- well, certainly shorter than this one-- involving a sum of squares. And what's that one? What's the other way to get the same answer? If I look at the SVD-- look at singular values. I think that this is also equal to the sum square of all the singular values.

So it's three nice expressions for the Frobenius norm. The nice ones involve A transpose A, or AA transpose. And of course, that connects to the singular values, because what are-- what's the connection between singular values and those-- and these guys-- A transpose A, or AA transpose? The singular values are the-- or the singular values squared are the--

AUDIENCE: Eigenvalues.

GILBERT STRANG: Eigenvalues of A transpose A. And then when I add up the trace, I'm adding up the eigenvalues and that's the-- that gives me the Frobenius norm squared. So this is a-- that tells us something important, which we can see in different ways, that the-- so to solve this problem, we're going to need various facts, like the QA in the Frobenius norm is the same as A in the Frobenius norm. Why is that? Why? So here I'm multiplying every column by the matrix Q. What happens to the length of the column when I multiply it by q?

AUDIENCE: It doesn't change.

GILBERT STRANG: Doesn't change. So I could add up the length of the columns all squared. Here I wrote it in terms of rows. But I could have reordered that, and got it in terms of columns. That's because the length of Q times any vector squared is the same as the vector squared. And these-- take these to be the columns of A. So for column by column, the multiplication by Q doesn't change the length. And then when I add up all the columns squared, I get the Frobenius norm squared.

And another way to say it-- let's make that connection between this fact-- that Q didn't change the Frobenius norm-- and this fact, that the Frobenius norm is expressed in terms of the sigmas. So what does Q do to the sigmas? I want to see in another way the answer to why. So if I have a matrix A with singular values, I multiply by Q-- what happens to the singular values?

AUDIENCE: Don't change.

GILBERT STRANG: Don't change. Don't change. That's the key point about singular values. If I multiply-- so A has a SVD, U sigma V transpose. And QA will have the SVD QU sigma V transpose. So all I've changed when I multiply by Q-- all I changed was the first factor-- the first orthogonal factor in the SVD. I didn't change the sigmas. They're still sitting there. So-- and of course, I could do also A on the other side-- different Q. Same Q or a different Q on the other side would show up here, and would not change the sigmas, and therefore would not change the Frobenius norm.

So these are important properties of this Frobenius norm. It's a-- it looks messy to write down in that form, but it's much nicer in these forms and in that form. OK. So now, if I can just-- then we saw that it involves traces. So let me make a few observations about traces. So I'll just-- we want to be able to play with traces, and that's something we really haven't done. Here's a fact-- that the trace of A transpose B is equal to the trace of B transpose A.

Of course, if B is A, it's clear, and it's equal to the trace of BA transpose. So even do little changes in your matrix without changing the trace. Let's see why one of these is true. Why is that first statement true? How is that matrix related to this matrix?

AUDIENCE: [INAUDIBLE] transpose.

GILBERT STRANG: It's just a transpose. If I take the transpose of that matrix, I get that. So what happens to the trace? I'm adding down the diagonal. The transpose has no effect. Clearly, this is just a fact that the trace doesn't change-- is not changed when you transpose a matrix, because the diagonal is not changed. Now what about this guy? I guess we're getting back to old fashioned 18.065, remembering facts about linear algebra, because this is a pure linear algebra.

So what's this one about? This says that I can reverse the order of two matrices. So I'm now looking at the connection between those two. And so let me just-- to use different letters-- CD equals the trace of DC. I can flip the order. That's all I've done here is. I've reversed B with A transpose. I reversed C with D. So why is that true? Why is that true? Well, how shall we see the truth of that fact?

So these are really convenient facts, that make a lot of people use the trace more often than we have in 18.065. I'm not a big user of arguments based on trace, but these are identities that go a long way with many problems. So let's see why that's true. Any time you think about trace, you've got two languages to use. You can use the eigenvalues. It's the sum of the eigenvalues. Or you can use the diagonal entries, because it's the sum of the diagonal entries.

Let's use eigenvalues. How are the eigenvalues of CD related to the eigenvalues of DC? They're the same. If these matrices are rectangular, then there might be some extra zero eigenvalues, because they would have different shapes. But zeros are not going to affect the trace. So this is the same nonzero eigenvalues. OK. And so on. Yeah. OK. Let me just-- let me try to tell you the steps now to get the correct Q. And let me tell you the answer first.

And I'm realizing that all important question four-- does deep learning actually work? We're going to run out of time today, because we only have a few minutes left. I suggest we bring that question back up, because it's pretty important to a lot of people. There's-- I had lunch with Professor Edelman, and he said, you know, deep learning and neural nets have had a record amount of publicity and hype for sort of computational algorithm.

And-- but I had-- I've had people now tell me that typical first-- if you create a network-- using Alex's design, for example-- the chances are that it won't be successful-- that the successful networks have been worked on, and experimented with. And a good structure has emerged, but didn't-- wasn't there at the start. So I think that's a topic for Monday. And I'm really just realizing, from talking to people in the field, that it's by no means automatic.

That structure-- even if you put in a whole bunch of layers-- it may not be what you want. OK. So I'm-- let me finish this argument today. Let me give you the answer. So what's the good Q?

I have matrices Y and X. And the idea is that I take it-- I look at Y transpose X. So that'll be all the dot products of one set of vectors or the other set of vectors. That's a matrix. And I do its SVD-- U sigma V transpose. So multiply this. Multiply Y-- the two bases that you're given.

Of course, if Y was the same as X-- if it was an orthogonal basis-- you'd have the identity, no questions. But generally, we have-- it has an SVD. And we're looking for a orthogonal matrix of the best Q is-- Da dun da duh. I mean, it's the right time for expressions of amazement. It is UV transpose. OK. So that gives us the answer. We're given X and Y. We're looking for the best Q. And the answer comes in the simplest possible way. Compute Y transpose X. Compute its SVD, and use the orthogonal matrices from the SVD. Yeah.

And I'm out of time so proof-- it's [INAUDIBLE] line later-- either to just send you the section online, or to discuss it in class Monday. But I'm really planning Monday to start with question 4. And meanwhile to ask a whole lot of people-- everybody I can find-- about that important question, is-- does deep learning usually work? How-- what can you do to make sure it works, or give yourself a better chance to have it work? So let's-- that's up for Monday then. Good.