PHILIPPE RIGOLLET: --bunch of x's and a bunch of y's. The y's were univariate, just one real valued random variable. And the x's were vectors that described a bunch of attributes for each of our individuals or each of our observations. Let's assume now that we're given essentially only the x's. This is sometimes referred to as unsupervised learning. There is just the x's. Usually, supervision is done by the y's.

And so what you're trying to do is to make sense of this data. You're going to try to understand this data, represent this data, visualize this data, try to understand something, right? So, if I give you a d-dimensional random vectors, and you're going to have n independent copies of this individual-- of this random vector, OK?

So you will see that I'm going to have-- I'm going to very quickly run into some limitations about what I can actually draw on the board because I'm using [? boldface ?] here. I'm also going to use the blackboard [? boldface. ?] So it's going to be a bit difficult. So tell me if you're actually a little confused by what is a vector, what is a number, and what is a matrix. But we'll get there.

So I have X in Rd, and that's a random vector. And I have X1 to Xn that are IID. They're independent copies of X. OK, so you can think of those as being-- the realization of these guys are going to be a cloud of n points in R to the d. And we're going to think of d as being fairly large. And for this to start to make sense, we're going to think of d as being at least 4, OK?

And meaning that you're going to have a hard time visualizing those things. If it was 3 or 2, you would be able to draw these points. And that's pretty much as much sense you're going to be making about those guys, just looking at the [INAUDIBLE]

All right, so I'm going to write each of those X's, right? So this vector, X, has d coordinate. And I'm going to write them as X1, to Xd. And I'm going to stack them into a matrix, OK? So once I have those guys, I'm going to have a matrix. But here, I'm going to use the double bar. And it's X1 transpose, Xn transpose.

So what it means is that the coordinates of this guy, of course, are X1,1. Here, I have-- I'm of size d, so I have X1d. And here, I have Xn1. Xnd. And so the i-th, j-th-- i-th row and j-th column is the matrix, Xij, right-- is the entry, Xi to-- sorry. OK, so each-- so the rows here are the observations. And the columns are the covariance over attributes. OK? So this is an n by d matrix.

All right, this is really just some bookkeeping. How do we store this data somehow? And the fact that we use a matrix just like for regression is going to be convenient because we're going to able to talk about projections-- going to be able to talk about things like this.

All right, so everything I'm going to say now is about variances or covariances of those things, which means that I need two moments, OK? If the variance does not exist, there's nothing I can say about this problem. So I'm going to assume that the variance exists. And one way to just put it to say that the two norm of those guys is finite, which is another way to say that each of them is finite. I mean, you can think of it the way you want.

All right, so now, the mean of X, right? So I have a random vector. So I can talk about the expectation of X. That's a vector that's in Rd. And that's just taking the expectation entrywise. Sorry. X1, Xd.

OK, so I should say it out loud. For this, the purpose of this class, I will denote by subscripts the indices that corresponds to observations. And superscripts, the indices that correspond to coordinates of a variable. And I think that's the same convention that we took for the regression case. Of course, you could use whatever you want. If you want to put commas, et cetera, it becomes just a bit more complicated.

All right, and so now, once I have this, so this tells me where my cloud of point is centered, right? So if I have a bunch of points-- OK, so now I have a distribution on Rd, so maybe I should talk about this-- I'll talk about this when we talk about the empirical version. But if you think that you have, say, a two-dimensional Gaussian random variable, then you have a center in two dimension, which is where it peaks, basically. And that's what we're talking about here.

But the other thing we want to know is how much does it spread in every direction, right? So in every direction of the two dimensional thing, I can then try to understand how much spread I'm getting. And the way you measure this is by using covariance, right? So the covariance matrix, sigma-- that's a matrix which is d by d. And it records-- in the j, k-th entry, it records the covariance between the j-th coordinate of X and the k-th coordinate of X, OK?

So with entries-- OK, so I have sigma, which is sigma 1,1, sigma dd, sigma 1d, sigma d1. OK, and here I have sigma jk And sigma jk is just the covariance between Xj, the j-th coordinate and the k-th coordinate. OK? So in particular, it's symmetric because the covariance between Xj and Xk is the same as the covariance between Xk and Xj. I should not put those parentheses here. I do not use them in this, OK?

Just the covariance matrix. So that's just something that records everything. And so what's nice about the covariance matrix is that if I actually give you X as a vector, you actually can build the matrix just by looking at vectors times vectors transpose, rather than actually thinking about building it coordinate by coordinate. So for example, if you're used to using MATLAB, that's the way you want to build a covariance matrix because MATLAB is good at manipulating vectors and matrices rather than just entering it entry by entry.

OK, so, right? So, what is the covariance between Xj and Xk? Well by definition, it's the expectation of Xj and Xk minus the expectation of Xj times the expectation of Xk, right? That's the definition of the covariance. I hope everybody's seeing that.

And so, in particular, I can actually see that this thing can be written as-- sigma can now be written as the expectation of XX transpose minus the expectation of X times the expectation of X transpose.

Why? Well, let's look at the jk-th coefficient of this guy, right? So here, if I look at the jk-th coefficient, I see what? Well, I see that it's the expectation of XX transpose jk, which is equal to the expectation of XX transpose jk. And what are the entries of XX transpose? Well, they're of the form, Xj times Xk exactly. So this is actually equal to the expectation of Xj times Xk.

And this is actually not the way I want to write it. I want to write it-- OK? Is that clear? That when I have a rank 1 matrix of this form, XX transpose, the entries are of this form, right? Because if I take-- for example, think about x, y, z, and then I multiply by x, y, z. What I'm getting here is x-- maybe I should actually use indices here.

x1, x2, x3. x1, x2, x3. The entries are x1x1, x1x2, x1x3; x2x1, x2x2, x2x3; x3x1, x3x2, x3x3, OK? So indeed, this is exactly of the form if you look at jk, you get exactly Xj times Xk, OK? So that's the beauty of those matrices.

So now, once I have this, I can do exactly the same thing, except that here, if I take the jk-th entry, I will get exactly the same thing, except that it's not going to be the expectation of the product, but the product of the expectation, right? So I get that the jk-th entry of E of X, E of X transpose, is just the j-th entry of E of X times the k-th entry of E of X.

So if I put those two together, it's actually telling me that if I look at the j, k-th entry of sigma, which I called little sigma jk, then this is actually equal to what? It's equal to the first term minus the second term. The first term is the expectation of Xj, Xk minus the expectation of Xj, expectation of Xk, which-- oh, by the way, I forgot to say this is actually equal to the expectation of Xj times the expectation of Xk because that's just the definition of the expectation of random vectors. So my j and my k are now inside. And that's by definition the covariance between Xj and Xk, OK?

So just if you've seen those manipulations between vectors, hopefully you're bored out of your mind. And if you have not, then that's something you just need to get comfortable with, right? So one thing that's going to be useful is to know very quickly what's called the outer product of a vector with itself, which is the vector of times the vector transpose, what the entries of these things are. And that's what we've been using on this second set of boards.

OK, so everybody agrees now that we've sort of showed that the covariance matrix can be written in this vector form. So expectation of XX transpose minus expectation of X, expectation of X transpose.

OK, just like the covariance can be written in two ways, right we know that the covariance can also be written as the expectation of Xj minus expectation of Xj times Xk minus expectation of Xk, right?

That's the-- sometimes, this is the original definition of covariance. This is the second definition of covariance. Just like you have the variance which is the expectation of the square of X minus c of X, or the expectation X squared minus the expectation of X squared. It's the same thing for covariance.

And you can actually see this in terms of vectors, right? So this actually implies that you can also rewrite sigma as the expectation of X minus expectation of X times the same thing transpose.

Right? And the reason is because if you just distribute those guys, this is just the expectation of XX transpose minus X, expectation of X transpose minus expectation of XX transpose. And then I have plus expectation of X, expectation of X transpose.

Now, things could go wrong because the main difference between matrices slash vectors and numbers is that multiplication does not commute, right? So in particular, those two things are not the same thing. And so that's the main difference that we have before, but it actually does not matter for our problem. It's because what's happening is that if when I take the expectation of this guy, then it's actually the same as the expectation of this guy, OK?

And so just because the expectation is linear-- so what we have is that sigma now becomes equal to the expectation of XX transpose minus the expectation of X, expectation of X transpose minus expectation of X, expectation of X transpose. And then I have-- well, really, what I have is this guy. And then I have plus the expectation of X, expectation of X transpose.

And now, those three things are actually equal to each other just because the expectation of X transpose is the same as the expectation of X transpose. And so what I'm left with is just the expectation of XX transpose minus the expectation of X, expectation of X transpose, OK?

So same thing that's happening when you want to prove that you can write the covariance either this way or that way. The same thing happens for matrices, or for vectors, right, or a covariance matrix. They go together. Is there any questions so far? And if you have some, please tell me, because I want to-- I don't know to which extent you guys are comfortable with this at all or not.

OK, so let's move on. All right, so of course, this is what I'm describing in terms of the distribution right here. I took expectations. Covariances are also expectations. So those depend on some distribution of X, right? If I wanted to compute that, I would basically need to know what the distribution of X is.

Now, we're doing statistics, so I need to [INAUDIBLE] my question is going to be to say, well, how well can I estimate the covariance matrix itself, or some properties of this covariance matrix based on data?

All right, so if I want to understand what my covariance matrix looks like based on data, I'm going to have to basically form its empirical counterparts, which I can do by doing the age-old statistical trick, which is replace your expectation by an average, all right? So let's just-- everything that's on the board, you see expectation, just replace it by an average.

OK, so, now I'm going to be given X1, Xn. So, I'm going to define the empirical mean. OK so, really, the idea is take your expectation and replace it by 1 over n sum, right? And so the empirical mean is just 1 over n. Some of the Xi's-- I'm guessing everybody knows how to average vectors. It's just the average of the coordinates. So I will write this as X bar. And the empirical covariance matrix, often called sample covariance matrix, hence the notation, S.

Well, this is my covariance matrix, right? Let's just replace the expectations by averages. 1 over n, sum from i equal 1 to n, of Xi, Xi transpose, minus-- this is the expectation of X. I will replace it by the average, which I just called X bar, X bar transpose, OK?

And that's when I want to use the-- that's when I want to use the notation-- the second definition, but I could actually do exactly the same thing using this definition here. Sorry, using this definition right here. So this is actually 1 over n, sum from i equal 1 to n, of Xi minus X bar, Xi minus X bar transpose.

And those are actually-- I mean, in a way, it looks like I could define two different estimators, but you can actually check. And I do encourage you to do this. If you're not comfortable making those manipulations, you can actually check that those two things are actually exactly the same, OK?

So now, I'm going to want to talk about matrices, OK? And remember, we defined this big matrix, X, with the double bar. And the question is, can I express both X bar and the sample covariance matrix in terms of this big matrix, X? Because right now, it's still expressed in terms of the vectors. I'm summing those vectors, vectors transpose. The question is, can I just do that in a very compact way, in a way that I can actually remove this sum term, all right?

That's going to be the goal. I mean, that's not a notational goal. That's really something that we want-- that's going to be convenient for us just like it was convenient to talk about matrices when we did linear regression.

OK, X bar. We just said it's 1 over n, sum from I equal 1 to n of Xi, right? Now remember, what does this matrix look like? We said that X bar-- X is this guy. So if I look at X transpose, the columns of this guy becomes X1, my first observation, X2, my second observation, all the way to Xn, my last observation, right? Agreed? That's what X transpose is.

So if I want to sum those guys, I can multiply by the all-ones vector. All right, so that's what the definition of the all-ones 1 vector is. Well, it's just a bunch of 1's in Rn, in this case. And so when I do X transpose 1, what I get is just the sum from i equal 1 to n of the Xi's. So if I divide by n, I get my average, OK? So here, I definitely removed the sum term.

Let's see if with the covariance matrix, we can do the same. Well, and that's actually a little more difficult to see, I guess. But let's use this definition for S, OK? And one thing that's actually going to be-- so, let's see for one second, what-- so it's going to be something that involves X, multiplying X with itself, OK? And the question is, is it going to be multiplying X with X transpose, or X tranpose with X?

To answer this question, you can go the easy route, which says, well, my covariance matrix is of size, what? What is the size of S?

AUDIENCE: d by d.

PHILIPPE RIGOLLET: d by d, OK? X is of size n by d. So if I do X times X transpose, I'm going to have something which is of size n by n. If I do X transpose X, I'm going to have something which is d by d. That's the easy route. And there's basically one of the two guys.

You can actually open the box a little bit and see what's going on in there. If you do X transpose X, which we know gives you a d by d, you'll see that X is going to have vectors that are of the form, Xi, and X transpose is going to have vectors that are of the form, Xi transpose, right? And so, this is actually probably the right way to go. So let's look at what's X transpose X is giving us.

So I claim that it's actually going to give us what we want, but rather than actually going there, let's-- to actually-- I mean, we could check it entry by entry, but there's actually a nice thing we can do. Before we go there, let's write X transpose as the following sum of variables, X1 and then just a bunch of 0's everywhere else. So it's still d by n. So n minus 1 of the columns are equal to 0 here.

Then I'm going to put a 0 and then put X2. And then just a bunch of 0's, right? So that's just 0, 0 plus 0, 0, all the way to Xn, OK? Everybody agrees with it? See what I'm doing here? I'm just splitting it into a sum of matrices that only have one nonzero columns. But clearly, that's true.

Now let's look at the product of this guy with itself. So, let's call these matrices M1, M2, Mn. So when I do X transpose X, what I do is the sum of the Mi's for i equal 1 to n, times the sum of the Mi transpose, right? Now, the sum of the Mi's transpose is just the sum of each of the Mi's transpose, OK? So now I just have this product of two sums, so I'm just going to re-index the second one by j. So this is sum for i equal 1 to n, j equal 1 to n of Mi Mj transpose. OK?

And now what we want to notice is that if i is different from j, what's happening? Well if i is different from j, let's look at say, M1 times XM2 transpose. So what is the product between those two matrices?

AUDIENCE: It's a new entry and [INAUDIBLE]

PHILIPPE RIGOLLET: There's an entry?

AUDIENCE: Well, it's an entry. It's like a dot product in that form next to [? transpose. ?]

PHILIPPE RIGOLLET: You mean a dot product is just getting [INAUDIBLE] number, right? So I want-- this is going to be a matrix. It's the product of two matrices, right? This is a matrix times a matrix. So this should be a matrix, right, of size d by d. Yeah, I should see a lot of hands that look like this, right?

Because look at this. So let's multiply the first-- let's look at what's going on in the first column here. I'm multiplying this column with each of those rows. The only nonzero coefficient is here, and it only hits this column of 0's. So every time, this is going to give you 0, 0, 0, 0. And it's going to be the same for every single one of them. So this matrix is just full of 0's, right? They never hit each other when I do the matrix-matrix multiplication. There's no-- every non-zero hits a 0.

So what it means is-- and this, of course, you can check for every i different from j. So this means that Mi times Mj transpose is actually equal to 0 when i is different from j, Right? Everybody is OK with this? So what that means is that when I do this double sum, really, it's a simple sum. There's only just the sum from i equal 1 to n of Mi Mi transpose. Because this is the only terms in this double sum that are not going to be 0 when [INAUDIBLE] [? M1 ?] with M1 itself.

Now, let's see what's going on when I do M1 times M1 transpose. Well, now, if I do Mi times and Mi transpose, now this guy becomes [? X1 ?] [INAUDIBLE] it's here. And so now, I really have X1 times X1 transpose. So this is really just the sum from i equal 1 to n of Xi Xi transpose, just because Mi Mi transpose is Xi Xi transpose. There's nothing else there.

So that's the good news, right? This term here is really just X transpose X divided by n. OK, I can use that guy again, I guess. Well, no. Let's just-- OK, so let me rewrite S. All right, that's the definition we have.

And we know that this guy already is equal to 1 over n X transpose X. x bar x bar transpose-- we know that x bar-- we just proved that x bar-- sorry, little x bar was equal to 1 over n X bar transpose times the all-ones vector. So I'm just going to do that.

So that's just going to be minus. I'm going to pull my two 1 over n's-- one from this guy, one from this guy. So I'm going to get 1 over n squared. And then I'm going to get X bar-- sorry, there's no X bar here. It's just X. Yeah. X transpose all ones times X transpose all ones transpose, right?

And X transpose all ones transpose-- right, the rule-- if I have A times B transpose, it's B transpose times A transpose, right? That's just the rule of transposition. So this is 1 transpose X transpose. And so when I put all these guys together, this is actually equal to 1 over n X transpose X minus one over n squared X transpose 1, 1 transpose X. Because X transpose transposes X, OK?

So now, I can actually-- I have something which is of the form, X transpose X-- [INAUDIBLE] to the left, X transpose; to the right, X. Here, I have X transpose to the left, X to the right. So it can factor out whatever's in there. So I can write S as 1 over n-- sorry, X transpose times 1 over n times the identity of Rd. And then I have minus 1 over n, 1, 1 transpose X.

OK, because if you-- I mean, you can distribute it back, right? So here, I'm going to get what? X transpose identity times X, the whole thing divided by n. That's this term. And then the second one is going to be-- sorry, 1 over n squared. And then I'm going to get 1 over n squared times X transpose 1, 1 transpose which is this guy, times X, and that's the [? right ?] [? thing, ?] OK?

So, the way it's written, I factored out one of the 1 over n's. So I'm just going to do the same thing as on this slide. So I'm just factoring out this 1 over n here. So it's 1 over n times X transpose identity of our d divided by n divided by 1 this time, minus 1 over n 1, 1 transpose times X, OK? So that's just what's on the slides.

What does the matrix, 1, 1 transpose, look like?

AUDIENCE: All 1's.

PHILIPPE RIGOLLET: It's just all 1's, right? Because the entries are the products of the all-ones-- of the coordinates of the all-ones vectors with the coordinates of the all-ones vectors, so I only get 1's. So it's a d by d matrix with only 1's. So this matrix, I can actually write exactly, right? H, this matrix that I called H which is what's sandwiched in-between this X transpose and X. By definition, I said this is the definition of H. Then this thing, I can write its coordinates exactly.

We know it's identity divided by n minus-- sorry, I don't know why I keep [INAUDIBLE]. Minus 1 over n 1, 1 transpose-- so it's this matrix with the only 1's on the diagonals and 0's and elsewhere-- minus a matrix that only has 1 over n everywhere. OK, so the whole thing is 1 minus 1 over n on the diagonals and then minus 1 over n here, OK?

And now I claim that this matrix is an orthogonal projector. Now, I'm writing this, but it's completely useless. This is just a way for you to see that it's actually very convenient now to think about this problem as being a matrix problem, because things are much nicer when you think about the actual form of your matrices, right? They could tell you, here is the matrix.

I mean, imagine you're sitting at a midterm, and I say, here's the matrix that has 1 minus 1 over n on the diagonals and minus 1 over n on the [INAUDIBLE] diagonal. Prove to me that it's a projector matrix. You're going to have to basically take this guy times itself. It's going to be really complicated, right? So we know it's symmetric. That's for sure.

But the fact that it has this particular way of writing it is going to make my life super easy to check this. That's the definition of a projector. It has to be symmetric and it has to square to itself because we just said in the chapter on linear regression that once you project, if you apply the projection again, you're not moving because you're already there.

OK, so why is H squared equal to H? Well let's just write H square. It's the identity minus 1 over n 1, 1 transpose times the identity minus 1 over n 1, 1 transpose, right? Let's just expand this now. This is equal to the identity minus-- well, the identity times 1, 1 transpose is just the identity. So it's 1, 1 transpose, sorry. So 1 over n 1, 1 transpose minus 1 over n 1, 1 transpose.

And then there's going to be what makes the deal is that I get this 1 over n squared this time. And then I get the product of 1 over n trans-- oh, let's write it completely. I get 1, 1 transpose times 1, 1 transpose, OK?

But this thing here-- what is this? n, right, is the end product of the all-ones vector with the all-ones vector. So I'm just summing n times 1 squared, which is n. So this is equal to n. So I pull it out, cancel one of the ends, and I'm back to what I had before. So I had identity minus 2 over n 1, 1 transpose plus 1 over n 1, 1 transpose which is equal to H. Because one of the 1 over n's cancel, OK?

So it's a projection matrix. It's projecting onto some linear space, right? It's taking a matrix. Sorry, it's taking a vector and it's projecting onto a certain space of vectors. What is this space? Right, so, how do you-- so I'm only asking the answer to this question in words, right? So how would you describe the vectors onto which this matrix is projecting?

Well, if you want to answer this question, the way you would tackle it is first by saying, OK, what does a vector which is of the form, H times something, look like, right? What can I say about this vector that's going to be definitely giving me something about the space on which it projects? I need to know a little more to know that it projects exactly onto this.

But one way we can do this is just see how it acts on a vector. What does it do to a vector to apply H, right? So I take v. And let's see what taking v and applying H to it looks like.

Well, it's the identity minus something. So it takes v and it removes something from v. What does it remove? Well, it's 1 over n times v transpose 1 times the all-ones vector, right? Agreed? I just wrote v transpose 1 instead of 1 transpose v, which are the same thing.

What is this thing? What should I call it in mathematical notation? v bar, right? I should all it v bar because this is exactly the average of the entries of v, agreed? This is summing the entries of v's, and this is dividing by the number of those v's. Sorry, now v is in our-- sorry, why do I divide by-- I'm just-- OK, I need to check what my dimensions are now. No, it's in Rd, right? So why do I divide by n?

So it's not really v bar. It's the sum of the v's divided by-- right, so it's v bar.

AUDIENCE: [INAUDIBLE]

[INTERPOSING VOICES]

AUDIENCE: Yeah, v has to be [INAUDIBLE]

PHILIPPE RIGOLLET: Oh, yeah. OK, thank you. So everywhere I wrote Hd, that was actually Hn. Oh, man. I wish I had a computer now. All right. So-- yeah, because the-- yeah, right? So why it's not-- well, why I thought it was this is because I was thinking about the outer dimension of X, really of X transpose, which is really the inner dimension, didn't matter to me, right?

So the thing that I can sandwich between X transpose and X has to be n by n. So this was actually n by n. And so that's actually n by n. Everything is n by n. Sorry about that. So this is n. This is n. This is-- well, I didn't really tell you what the all-ones vector was, but it's also in our n. Yeah, OK. Thank you. And n-- actually, I used the fact that this was of size n here already.

OK, and so that's indeed v bar. So what is this projection doing to a vector? It's removing its average on each coordinate, right? And the effect of this is that v is a vector. What is the average of Hv?

AUDIENCE: 0.

PHILIPPE RIGOLLET: Right, so it's 0. It's the average of v, which is v bar, minus the average of something that only has v bar's entry, which is v bar. So this thing is actually 0. So let me repeat my question. Onto what subspace does H project? Onto the subspace of vectors that have mean 0.

A vector that has mean 0 is a vector. So if you want to talk more linear algebra, v bar-- for a vector you have mean 0, it means that v is orthogonal to the span of the all-ones vector. That's it. It projects to this space. So in words, it projects onto the space of vectors that have 0 mean. In linear algebra, it says it projects onto the hyperplane which is orthogonal to the all-ones vector, OK? So that's all.

Can you guys still see the screen? Are you good over there? OK.

All right, so now, what it means is that, well, I'm doing this weird thing, right? I'm taking the inner product-- so S is taking X. And then it's removing its mean of each of the columns of X, right? When I take H times X, I'm basically applying this projection which consists in removing the mean of all the X's. And then I multiply by H transpose.

But what's actually nice is that, remember, H is a projector. Sorry, I don't want to keep that. Which means that when I look at X transpose HX, it's the same as looking at X transpose H squared X. But since H is equal to its transpose, this is actually the same as looking at X transpose H transpose HX, which is the same as looking at HX transpose HX, OK?

So what it's doing, it's first applying this projection matrix, H, which removes the mean of each of your columns, and then looks at the inner products between those guys, right? Each entry of this guy is just the covariance between those centered things. That's all it's doing.

All right, so those are actually going to be the key statements. So everything we've done so far is really mainly linear algebra, right? I mean, looking at expectations and covariances was just-- we just used the fact that the expectation was linear. We didn't do much. But now there's a nice thing that's happening.

And that's why we're going to switch from the language of linear algebra to more statistical, because what's happening is that if I look at this quadratic form, right? So I take sigma. So I take a vector, u. And I'm going to look at u-- so let's say, in Rd. And I'm going to look at u transpose sigma u. OK?

What is this doing? Well, we know that u transpose sigma u is equal to what? Well, sigma is the expectation of XX transpose minus the expectation of X expectation of X transpose, right? So I just substitute in there.

Now, u is deterministic. So in particular, I can push it inside the expectation here, agreed? And I can do the same from the right. So here, when I push u transpose here, and u here, what I'm left with is the expectation of u transpose X times X transpose u. OK? And now, I can do the same thing for this guy. And this tells me that this is the expectation of u transpose X times the expectation of X transpose u.

Of course, u transpose X is equal to X transpose u. And u-- yeah. So what it means is that this is actually equal to the expectation of u transpose X squared minus the expectation of u transpose X, the whole thing squared.

But this is something that should look familiar. This is really just the variance of this particular random variable which is of the form, u transpose X, right? u transpose X is a number. It involves a random vector, so it's a random variable. And so it has a variance. And this variance is exactly given by this formula. So this is just the variance of u transpose X. So what we've proved is that if I look at this guy, this is really just the variance of u transpose X, OK?

I can do the same thing for the sample variance. So let's do this. And as you can see, spoiler alert, this is going to be the sample variance. OK, so remember, S is 1 over n, sum of Xi Xi transpose minus X bar X bar transpose. So when I do u transpose, Su, what it gives me is 1 over n sum from i equal 1 to n of u transpose Xi times Xi transpose u, all right?

So those are two numbers that multiply each other and that happen to be equal to each other, minus u transpose X bar X bar transpose u, which is also the product of two numbers that happen to be equal to each other. So I can rewrite this with squares.

So we're almost there. All I need to know to check is that this thing is actually the average of those guys, right? So u transpose X bar. What is it? It's 1 over n sum from i equal 1 to n of u transpose Xi. So it's really something that I can write as u transpose X bar, right? That's the average of those random variables of the form, u transpose Xi.

So what it means is that u transpose Su, I can write as 1 over n sum from i equal 1 to n of u transpose Xi squared minus u transpose X bar squared, which is the empirical variance that we need noted by small s squared, right? So that's the empirical variance of u transpose X1 all the way to u transpose Xn.

OK, and here, same thing. I use exactly the same thing. I just use the fact that here, the only thing I use is really the linearity of this guy, of 1 over n sum or the linearity of expectation, that I can push things in there, OK?

AUDIENCE: So what you have written at the end of that sum for uT Su?

PHILIPPE RIGOLLET: This one?

AUDIENCE: Yeah.

PHILIPPE RIGOLLET: Yeah, I said it's equal to small s, and I want to make a difference between the big S that I'm using here. So this is equal to small-- I don't know, I'm trying to make it look like a calligraphic s squared.

OK, so this is nice, right? This covariance matrix-- so let's look at capital sigma itself right now. This covariance matrix, we know that if we read its entries, what we get is the covariance between the coordinates of the X's, right, of the random vector, X. And the coordinates, well, by definition, are attached to a coordinate system. So I only know what the covariance of X in of those two things are, or the covariance of those two things are.

But what if I want to find coordinates between linear combination of the X's? Sorry, if I want to find covariances between linear combination of those X's. And that's exactly what this allows me to do. It says, well, if I pre- and post-multiply by u, this is actually telling me what the variance of X along direction u is, OK? So there's a lot of information in there, and it's just really exploiting the fact that there is some linearity going on in the covariance.

So, why variance? Why is variance interesting for us, right? Why? I started by saying, here, we're going to be interested in having something to do dimension reduction. We have-- think of your points as [? being in a ?] dimension larger than 4, and we're going to try to reduce the dimension. So let's just think for one second, what do we want about a dimension reduction procedure?

If I have all my points that live in, say, three dimensions, and I have one point here and one point here and one point here and one point here and one point here, and I decide to project them onto some plane-- that I take a plane that's just like this, what's going to happen is that those points are all going to project to the same point, right? I'm just going to not see anything.

However, if I take a plane which is like this, they're all going to project into some nice line. Maybe I can even project them onto a line and they will still be far apart from each other. So that's what you want. You want to be able to say, when I take my points and I say I project them onto lower dimensions, I do not want them to collapse into one single point. I want them to be spread as possible in the direction on which I project.

And this is what we're going to try to do. And of course, measuring spread between points can be done in many ways, right? I mean, you could look at, I don't know, sum of pairwise distances between those guys. You could look at some sort of energy. You can look at many ways to measure of spread in a direction.

But variance is a good way to measure of spread between points. If you have a lot of variance between your points, then chances are they're going to be spread. Now, this is not always the case, right? If I have a direction in which all my points are clumped onto one big point and one other big point, it's going to choose this because that's the direction that has a lot of variance. But hopefully, the variance is going to spread things out nicely.

So the idea of principal component analysis is going to try to identify those variances-- those directions along which we have a lot of variance. Reciprocally, we're going to try to eliminate the directions along which we do not have a lot of variance, OK? And let's see why.

Well, if-- so here's the first claim. If you transpose Su is equal to 0, what's happening? Well, I know that an empirical variance is equal to 0. What does it mean for an empirical variance to be equal to 0? So I give you a bunch of points, right? So those points are those points-- u transpose X1, u transpose-- those are a bunch of numbers. What does it mean to have the empirical variance of those points being equal to 0?

AUDIENCE: They're all the same.

PHILIPPE RIGOLLET: They're all the same. So what it means is that when I have my points, right? So, can you find a direction for those points in which they project to all the same point? No, right? There's no such thing. For this to happen, you have to have your points which are perfectly aligned. And then when you're going to project onto the orthogonal of this guy, they're going to all project to the same point here, which means that the empirical variance is going to be 0.

Now, this is an extreme case. This will never happen in practice, because if that happens, well, I mean, you can basically figure that out very quickly. So in the same way, it's very unlikely that you're going to have u transpose sigma u, which is equal to 0, which means that, essentially, all your points are [INAUDIBLE] or let's say all of them are orthogonal to u, right? So it's exactly the same thing. It just says that in the population case, there's no probability that your points deviate from this guy here. This happens with zero probability, OK?

And that's just because if you look at the variance of this guy, it's going to be 0. And then that means that there's no deviation. By the way, I'm using the name projection when I talk about u transpose X, right? So let's just be clear about this. If you-- so let's say I have a bunch of points, and u is a vector in this direction. And let's say that u has the-- so this is 0. This is u. And let's say that u has norm, 1, OK?

When I look, what is the coordinate of the projection? So what is the length of this guy here? Let's call this guy X1. What is the length of this guy? In terms of inner products? This is exactly u transpose X1. This length here, if this is X2, this is exactly u transpose X2, OK? So those-- u transpose X measure exactly the distance to the origin of those-- I mean, it's really-- think of it as being just an x-axis thing. You just have a bunch of points. You have an origin. And it's really just telling you what the coordinate on this axis is going to be, right?

So in particular, if the empirical variance is 0, it means that all these points project to the same point, which means that they have to be orthogonal to this guy. And you can think of it as being also maybe an entire plane that's orthogonal to this line, OK? So that's why I talk about projection, because the inner products, u transpose X, is really measuring the coordinates of X when u becomes the x-axis.

Now, if u does not have norm 1, then you just have a change of scale here. You just have a change of unit, right? So this is really u times X1. The coordinates should really be divided by the norm of u.

OK, so now, just in the same way-- so we're never going to have exactly 0. But if we [INAUDIBLE] the other end, if u transpose Su is large, what does it mean? It means that when I look at my points as projected onto the axis generated by u, they're going to have a lot of variance. They're going to be far away from each other in average, right? That's what large variance means, or at least large empirical variance means. And same thing for u.

So what we're going to try to find is a u that maximizes this. If I can find a u that maximizes this so I can look in every direction, and suddenly I find a direction in which the spread is massive, then that's a point on which I'm basically the less likely to have my points project onto each other and collide, right? At least I know they're going to project at least onto two points.

So the idea now is to say, OK, let's try to maximize this spread, right? So we're going to try to find the maximum over all u's of u transpose Su. And that's going to be the direction that maximizes the empirical variance. Now of course, if I read it like that for all u's in Rd, what is the value of this maximum?

It's infinity, right? Because I can always multiply u by 10, and this entire thing is going to multiplied by 100. So I'm just going to take u as large as I want, and this thing is going to be as large as I want, and so I need to constrain u. And as I said, I need to have u of size 1 to talk about coordinates in the system generated by u like this. So I'm just going to constrain u to have Euclidean norm equal to 1, OK?

So that's going to be my goal-- trying to find the largest possible u transpose Su, or in other words, empirical variance of the points projected onto the direction u when u is of norm 1, which justifies to use the word, "direction," and because there's no magnitude to this u.

OK, so how am I going to do this? I could just fold and say, let's just optimize this thing, right? Let's just take this problem. It says maximize a function onto some constraints. Immediately, the constraint is sort of nasty. I'm on a sphere, and I'm trying to move points on the sphere. And I'm maximizing this thing which actually happens to be convex. And we know we know how to minimize convex functions, but maximize them is a different question.

And so this problem might be super hard. So I can just say, OK, here's what I want to do, and let me give that to an optimizer and just hope that the optimizer can solve this problem for me. That's one thing we can do. Now as you can imagine, PCA is so well spread, right? Principal component analysis is something that people do constantly. And so that means that we know how to do this fast. So that's one thing.

The other thing that you should probably question about why-- if this thing is actually difficult, why in the world would you even choose the variance as a measure of spread if there's so many measures of spread, right? The variance is one measure of spread. It's not guaranteed that everything is going to project nicely far apart from each other. So we could choose the variance, but we could choose something else. If the variance does not help, why choose it? Turns out the variance helps.

So this is indeed a non-convex problem. I'm maximizing, so it's actually the same. I can make this constraint convex because I'm maximizing a convex function, so it's clear that the maximum is going to be attained at the boundary. So I can actually just fill this ball into some convex ball.

However, I'm still maximizing, so this is a non-convex problem. And this turns out to be the fanciest non-convex problem we know how to solve. And the reason why we know how to solve it is not because of optimization or using gradient-type things or anything of the algorithms that I mentioned during the maximum likelihood. It's because of linear algebra. Linear algebra guarantees that we know how to solve this.

And to understand this, we need to go a little deeper in linear algebra, and we need to understand the concept of diagonalization of a matrix. So who has ever seen the concept of an eigenvalue? Oh, that's beautiful. And if you're not raising your hand, you're just playing "Candy Crush," right? All right, so, OK.

This is great. Everybody's seen it. For my live audience of millions, maybe you have not, so I will still go through it. All right, so one of the basic facts-- and I remember when I learned this in-- I mean, when I was an undergrad, I learned about the spectral decomposition and this diagonalization of matrices. And for me, it was just a structural property of matrices, but it turns out that it's extremely useful, and it's useful for algorithmic purposes.

And so what this theorem tells you is that if you take a symmetric matrix-- well, with real entries, but that really does not matter so much. And here, I'm going to actually-- so I take a symmetric matrix, and actually S and sigma are two such symmetric matrices, right? Then there exists P and D, which are both-- so let's say d by d. Which are both d by d such that P is orthogonal. That means that P transpose P is equal to PP transpose is equal to the identity. And D is diagonal. And sigma, let's say, is equal to PDP transpose, OK?

So it's a diagonalization because it's finding a nice transformation. P has some nice properties. It's really just the change of coordinates in which your matrix is diagonal, right? And the way you want to see this-- and I think it sort of helps to think about this problem as being-- sigma being a covariance matrix. What does a covariance matrix tell you?

Think of a multivariate Gaussian. Can everybody visualize a three-dimensional Gaussian density? Right, so it's going to be some sort of a bell-shaped curve, but it might be more elongated in one direction than another. And then going to chop it like that, all right? So I'm going to chop it off. And I'm going to look at how it bleeds, all right? So I'm just going to look at where the blood is.

And what it's going to look at-- it's going to look like some sort of ellipsoid, right? In high dimension, it's just going to be an olive. And that is just going to be bigger and bigger. And then I chop it off a little lower, and I get something a little bigger like this. And so it turns out that sigma is capturing exactly this, right? The matrix sigma-- so the center of your covariance matrix of your Gaussian is going to be this thing. And sigma is going to tell you which direction it's elongated.

And so in particular, if you look, if you knew an ellipse, you know there's something called principal axis, right? So you could actually define something that looks like this, which is this axis, the one along which it's the most elongated. Then the axis along which is orthogonal to it, along which it's slightly less elongated, and you go again and again along the orthogonal ones.

It turns out that those things here is the new coordinate system in which this transformation, P and P transpose, is putting you into. And D has entries on the diagonal which are exactly this length and this length, right?

So that's just what it's doing. It's just telling you, well, if you think of having this Gaussian or this high-dimensional ellipsoid, it's elongated along certain directions. And these directions are actually maybe not well aligned with your original coordinate system, which might just be the usual one, right-- north, south, and east, west. Maybe I need to turn it. And that's exactly what this orthogonal transformation is doing for you, all right?

So, in a way, this is actually telling you even more. It's telling you that any matrix that's symmetric, you can actually turn it somewhere. And that'll start to dilate things in the directions that you have, and then turn it back to what you originally had. And that's actually exactly the effect of applying a symmetric matrix through a vector, right?

And it's pretty impressive. It says if I take sigma times v. Any sigma that's of this form, what I'm doing is-- that's symmetric. What I'm really doing to v is I'm changing its coordinate system, so I'm rotating it. Then I'm changing-- I'm multiplying its coordinates, and then I'm rotating it back. That's all it's doing, and that's what all symmetric matrices do, which means that this is doing a lot.

All right, so OK. So, what do I know? So I'm not going to prove that this is the so-called spectral theorem. And the diagonal entries of D is of the form, lambda 1, lambda 2, lambda d, 0, 0. And the lambda j's are called eigenvalues of D.

Now in general, those numbers can be positive, negative, or equal to 0. But here, I know that sigma and S are-- well, they're symmetric for sure, but they are positive semidefinite. What does it mean? It means that when I take u transpose sigma u for example, this number is always non-negative. Why is this true? What is this number?

It's the variance of-- and actually, I don't even need to finish this sentence. As soon as I say that this is a variance, well, it has to be non-negative. We know that a variance is not negative. And so, that's also a nice way you can use that. So it's just to say, well, OK, this thing is positive semidefinite because it's a covariance matrix. So I know it's a variance, OK? So I get this.

Now, if I had some negative numbers-- so the effect of that is that when I draw this picture, those axes are always positive, which is kind of a weird thing to say. But what it means is that when I take a vector, v, I rotate it, and then I stretch it in the directions of the coordinate, I cannot flip it. I can only stretch or shrink, but I cannot flip its sign, all right? But in general, for any symmetric matrices, I could do this.

But when it's positive symmetric definite, actually what turns out is that all the lambda j's are non-negative. I cannot flip it, OK? So all the eigenvalues are non-negative. That's a property of positive semidef. So when it's symmetric, you have the eigenvalues. They can be any number. And when it's positive semidefinite, in particular that's the case of the covariance matrix and the empirical covariance matrix, right? Because the empirical covariance matrix is an empirical variance, which itself is non-negative. And so I get that the eigenvalues are non-negative.

All right, so principal component analysis is saying, OK, I want to find the direction, u, that maximizes u transpose Su, all right? I've just introduced in one slide something about eigenvalues. So hopefully, they should help. So what is it that I'm going to be getting? Well, let's just see what happens.

Oh, I forgot to mention that-- and I will use this. So the lambda j's are called eigenvectors. And then the matrix, P, has columns v1 to vd, OK? The fact that it's orthogonal-- that P transpose P is equal to the identity-- means that those guys satisfied that vi transpose vj is equal to 0 if i is different from j. And vi transpose vi is actually equal to 1, right, because the entries of PP transpose are exactly going to be of the form, vi transpose vj, OK?

So those v's are called eigenvectors. And v1 is attached to lambda 1, and v2 is attached to lambda 2, OK? So let's see what's happening with those things. What happens if I take sigma-- so if you know eigenvalues, you know exactly what's going to happen. If I look at, say, sigma times v1, well, what is sigma? We know that sigma is PDP transpose v1.

What is P transpose times v1? Well, P transpose has rows v1 transpose, v2 transpose, all the way to vd transpose. So when I multiply this by v1, what I'm left with is the first coordinate is going to be equal to 1 and the second coordinate is going to be equal to 0, right? Because they're orthogonal to each other-- 0 all the way to the end. So that's when I do P transpose v1.

Now I multiply by D. Well, I'm just multiplying this guy by lambda 1, this guy by lambda 2, and this guy by lambda d, so this is really just lambda 1. And now I need to post-multiply by P. So what is P times this guy? Well, P is v1 all the way to vd. And now I multiply by a vector that only has 0's except lambda 1 on the first guy. So this is just lambda 1 times v1.

So what we've proved is that sigma times v1 is lambda 1 v1, and that's probably the notion of eigenvalue you're most comfortable with, right? So just when I multiply by v1, I get v1 back multiplied by something, which is the eigenvalue. So in particular, if I look at v1, transpose sigma v1, what do I get? Well, I get lambda 1 v1 transpose v1, which is 1, right? So this is actually lambda 1 v1 transpose v1, which is lambda 1, OK?

And if I do the same with v2, clearly I'm going to get v2 transpose sigma. v2 is equal to lambda 2. So for each of the vj's, I know that if I look at the variance along the vj, it's actually exactly given by those eigenvalues, all right? Which proves this, because the variance along the eigenvectors is actually equal to the eigenvalues. So since they're variances, they have to be non-negative.

So now, I'm looking for the one direction that has the most variance, right? But that's not only among the eigenvectors. That's also among the other directions that are in-between the eigenvectors. If I were to look only at the eigenvectors, it would just tell me, well, just pick the eigenvector, vj, that's associated to the largest of the lambda j's.

But it turns out that that's also true for any vector-- that the maximum direction is actually one direction which is among the eigenvectors. And among the eigenvectors, we know that the one that's the largest-- that carries the largest variance is the one that's associated to the largest eigenvalue, all right?

And so this is what PCA is going to try to do for me. So in practice, that's what I mentioned already, right? We're trying to project the point cloud onto a low-dimensional space, D prime, by keeping as much information as possible. And by "as much information," I mean we do not want points to collide.

And so what PCA is going to do is just going to try to project [? on two ?] directions. So there's going to be a u, and then there's going to be something orthogonal to u, and then the third one, et cetera, so that once we project on those, we're keeping as much of the covariance as possible, OK? And in particular, those directions that we're going to pick are actually a subset of the vj's that are associated to the largest eigenvalues.

So I'm going to stop here for today. We'll finish this on Tuesday. But basically, the idea is it's just the following. You're just going to-- well, let me skip one more.

Yeah, this is the idea. You're first going to pick the eigenvector associated to the largest eigenvalue. Then you're going to pick the direction that orthogonal to the vector that you've picked, and that's carrying the most variance. And that's actually the second largest-- the eigenvector associated to the second largest eigenvalue.

And you're going to go all the way to the number of them that you actually want to pick, which is in this case, d, OK? And wherever you choose to chop this process, not going all the way to d, is going to actually give you a lower-dimensional representation in the coordinate system that's given by v1, v2, v3, et cetera, OK? So we'll see that in more details on Tuesday. But I don't want to get into it now. We don't have enough time. Are there any questions?