**PHILIPPE RIGOLLET:** We keep on talking about principal component analysis, which we essentially introduced as a way to work with a bunch of data. So the data that's given to us when we want to do PCA is a bunch of vectors X1 to Xn. So they are random vectors. in Rd.

And what we mentioned is that we're going to be using linear algebra-- in particular, the spectral theorem-- that guarantees to us that if I look at the convenience matrix of this guy, or its empirical covariance matrix, since they're symmetric real matrices and they are positive semidefinite, there exists a diagonalization into non-negative eigenvalues. And so here, those things live in Rd, so it's a really large space. And what we want to do is to map it down into a space that we can visualize, hopefully a space of size 2 or 3. Or if not, then we're just going to take more and start looking at subspaces altogether.

So think of the case where d is large but not larger than n. So let's say, you have a large number of points. The question is, is it possible to project those things onto a lower dimensional space, d prime, which is much less than d-- so think of d prime equals, say, 2 or 3-- and so that you keep as much information about the cloud of points that you had originally.

So again, the example that we could have is that X1 to Xn for, say, Xi for patient i's recording a bunch of body measurements and maybe blood pressure, some symptoms, et cetera. And then we have a cloud of n patients. And we're trying to visualize maybe to see if-- If I could see, for example, that there's two groups of patients, maybe I would know that I have two groups of different disease or maybe two groups of different patients that respond differently to a particular disease or drug et cetera. So visualizing is going to give us quite a bit of insight about what the spatial arrangement of those vectors are.

And so PCA says, well, here, of course, in this question, one thing that's not defined is what is information. And we said that one thing we might want to do when we project is that points do not collide with each other. And so that means we're trying to find directions, where after I project, the points are still pretty spread out. And so I can see what's going on.

And PCA says-- OK, so there's many ways to answer this question. And PCA says, let's just find a subspace of dimension d prime that keeps as much covariance structure as possible. And the reason is that those directions are the ones that maximize the variance, which is a

proxy for the spread. There's many, many ways to do this.

There's actually a Google video that was released maybe last week about the data visualization team of Google that shows you something called t-SNE, which is essentially something that tries to do that. It takes points in very high dimensions and tries to map them in lower dimensions, so that you can actually visualize them. And t-SNE is some alternative to PCA that gives an other definition for the word information. I'll talk about this towards the end, how you can actually somewhat automatically extend everything we've said for PCA to an infinite family of procedures.

So how do we do this? Well, the way we do this is as follows. So remember, given those guys, we can form something which is called S, which is the sample, or the empirical covariance matrix. And since from couple of slides ago, we know that S has a eigenvalue decomposition, S is equal to PDP transpose, where P is orthogonal. So that's where we use our linear algebra results. So that means that P transpose P is equal to PP transpose, which is the identity.

So remember, S is a d by d matrix. And so P is also d by d. And d is diagonal. And I'm actually going to take, without loss of generality, I'm going to assume that d-- so it's going to be diagonal-- and I'm going to have something that looks like lambda 1 to lambda d. Those are called the eigenvalues of S.

What we know is that lambda j's are non-negative. And actually, what I'm going to assume without loss of generalities is lambda 1 is larger than lambda 2, which is larger than lambda d. Because in particular, this decomposition-- the spectrum decomposition-- is not entirely unique. I could permute the columns of P, and I would still have an orthogonal matrix. And to balance that, I would also have to permute the entries of d.

So there's as many decompositions as there are permutations. So there's actually quite a bit. But the bag of eigenvalues is unique. The set of eigenvalues is unique. The ordering is certainly not unique.

So here, I'm just going to pick-- I'm going to nail down one particular permutation-- actually, maybe two in case I have equalities. But let's say, I pick one that satisfies this. And the reason why I do this is really not very important. It's just to say, I'm going to want to talk about the largest of those eigenvalues.

So this is just going to be easier for me to say that this one is lambda 1, rather than say it's

lambda 7. So this is just to say that the largest eigenvalue of S is lambda 1. If I didn't do that, I would just call it maybe lambda max, and you would just know which one I'm talking about.

So what's happening now is that if I look at d, then it turns out that if I start-- so if I do P transpose Xi, I am actually projecting my Xi's-- I'm basically changing the basis for my Xi's. And now, D is the empirical covariance matrix of those guys. So let's check that.

So what it means is that if I look at-- so what I claim is that P transpose Xi-- that's a new vector, let's call it Yi, it's also in Rd-- and what I claim is that the covariance matrix of this guy is actually now this diagonal matrix, which means in particular that if they were Gaussian, then they would be independent. But I also know now that there's no correlation across coordinates of Yi.

So to prove this, let me assume that X bar is equal to 0. And the reason why I do this is because it's just annoying to carry out all this censuring constantly and I talk about S. So when X bar is equal to 0, that implies that S has a very simple form. It's of the form sum from i equal 1 to n of Xi Xi transpose. So that's my S.

But what I want is the S of Y-- So OK, that implies also that P times X bar, which is equal to P times X bar is also equal to 0. So that means that Y bar-- Y has mean 0, if this is 0. So if I look at the sample covariance matrix of Y, it's just going to be something that looks like the sum of the outer products or the Yi Yi transpose.

And again, the reason why I make this assumption is so that I don't have to write minus X bar X bar transpose. But you can do it. And it's going to work exactly the same.

So now, I look at this S prime. And so what is this S prime? Well, I'm just going to replace Yi with PXi. So it's the sum from i equal 1 to n of PXi PXi transpose, which is equal to the sum from-- sorry there's a 1/n.

So it's equal to 1/n sum from i equal 1 to n of PXi Xi transpose P transpose. Agree? I just said that the transpose of AB is the transpose of B times the transpose of A.

And so now, I can push the sum in. P does not depend on i. So this thing here is equal to PS P transpose, because the sum of the Xi Xi transpose divided by n is S.

But what is PS P transpose? Well, we know that S is equal to-- sorry that's P transpose. So this was with a P transpose.

I'm sorry, I made an important mistake here. So Yi is P transpose Xi. So this is P transpose and P transpose here, which means that this is P transpose and this is double transpose, which is just nothing and that transpose and nothing.

So now, I write S as PD P transpose. That's the spectral decomposition that I had before. That's my eigenvalue decomposition, which means that now, if I look at S prime, it's P transpose times PD P transpose P. But now, P transpose P is the identity, P transpose P is the identity. So this is actually just equal to D.

And again, you can check that this also works if you have to center all those guys as you go. But if you think about it, this is the same thing as saying that I just replaced Xi by Xi minus X bar. And then it's true that Y bar is also P times Xi minus X bar.

So now, we have that D is the empirical covariance matrix of those guys-- the Yi's, which are P transpose Xi's. And so in particular, what it means is that if I look at the covariance of Yj Yk-- So that's the covariance of the j-th coordinate of Y and the k-th coordinate of Y. I'm just not putting an index. But maybe, let's say the first one or something like this-- any of them, their IID.

Then what is this covariance? It's actually 0 if j is different from k. And the covariance between Yj and Yj, which is just the variance of Yj, is equal to lambda j-- the j-th largest eigenvalue. So the eigenvalues capture the variance of my observations in this new coordinate system. And they're completely orthogonal.

So what does that mean? Well, again, remember, if I chop off the head of my Gaussian in multi dimensions, we said that what we started from was something that looked like this. And we said, well, there's one direction that's important, that's this guy, and one important that's this guy.

When I applied a transformation P transpose, what I'm doing is that I'm realigning this thing with the new axes. Or in a way, rather to be fair, I'm not actually realigning the ellipses with the axes. I'm really realigning the axes with the ellipses. So really, what I'm doing is I'm saying, after I apply P, I'm just rotating this coordinate system. So now, it becomes this guy.

And now, my ellipses actually completely align. And what happens here is that this coordinate is independent of that coordinate. And that's what we write here, if they are Gaussian. I didn't really tell this-- I'm only making statements about covariances. If they are Gaussians, those

implied statements about independence.

So as I said, the variance now, lambda 1, is actually the variance of P transpose Xi. But if I look now at the-- so this is a vector, so I need to look at the first coordinate of this guy. So it turns out that doing this is actually the same thing as looking at the variance of what? Well, the first column of P times Xi.

So that's the variance of-- I'm going to call it v1 transpose Xi, where P-- So the v1 vd in Rd are eigenvectors. And each vi is associated to lambda i. So that's what we saw when we talked about this eigen decomposition a couple of slides back. That's the one here.

So if I call the columns of P v1 to vd, this is what's happening. So when I look at lambda 1, it's just the variance of Xi inner product with v1. And we made this picture when we said, well, let's say v1 is here and then x1 is here. And if vi has a unique norm, then the inner product between Xi and v1 is just the length of this guy here.

So that's the variance of the Xi says the length of Xi-- so this is 0-- that's the length of Xi when I project it onto the direction that span by v1. If v1 has length 2, this is really just twice this length. If vi has length 3, it's three times this.

But it turns out that since P satisfies P transpose P is equal to the identity-- that's an orthogonal matrix, that's right here-- then this is actually saying the same thing as vj transpose vj, which is really the norm squared of vj, is equal to 1. And vj transpose vk is equal to 0, if j is different from k. The eigenvectors are orthogonal to each other. And they're actually all of norm 1.

So now, I know that this is indeed a direction. And so when I look at v1 transpose Xi, I'm really measuring exactly this length. And what is this length? It's the length of the projection of Xi onto this line. That's the line that's spanned by v1.

So if I had a very high dimensional problem and I started to look at the direction v1-- let's say v1 now is not a eigenvector, it's any direction-- then if I want to do this lower dimensional projection, then I have to understand how those Xi's project onto the line that's spanned by v1, because this is all that I'm going to be keeping at the end of the day about Xi's.

So what we want is to find the direction where those Xi's, those projections, have a lot of variance. And we know that the variance of Xi on this direction is actually exactly given by

lambda 1. Sorry, that's the empirical var-- yeah, I should call variance hat. That's the empirical variance.

Everything is in empirical here. We're talking about the empirical covariance matrix. And so I also have that lambda 2 is the empirical variance of when I project Xi onto v2, which is the second one, just for exactly this reason. Any question?

So lambda j's are going to be important for us. Lambda j measure the spread of the points when I project them onto a line which is a one dimensional space. And so I'm going to have-- let's say I want to pick only one, I'm going to have to find the one dimensional space that carries the most variance. And I claim that v1 is the one that actually maximizes the spread.

So the claim-- so for any direction, u in Rd-- and by direction, I really just mean that the norm of u is equal to 1. I need to play fair-- I'm going to compare myself to other things of lengths one, so I need to play fair and look at directions of length 1.

Now, if I'm interested in the empirical variance of X1 transpose-- sorry, u transpose X1 u transpose Xn, then this thing is maximized for u equals v1, where v1 is the eigenvector associated to lambda 1 and lambda 1 is not any eigenvalues, it's the largest of all those. So it's the largest eigenvalue.

So why is that true? Well, there's also a claim that for any direction u-- so that's 1 and 2-- the variance of u transpose X-- now, this is just a random variable, and I'm looking about the true variance-- this is maximized for u equals, let's call it w1, where w1 is the eigenvector of sigma-- Now, I'm talking about the true variance. Whereas, here, I was talking about the empirical variance. So the true variance is the eigenvectors of the true sigma associated to the largest eigenvalue of sigma.

So I did not give it a name. Here, that was lambda 1 for the empirical one. For the true one, you can give it another name, mu 1 if you want. But that's just the same thing. All it's saying is like, wherever I see empirical, I can remove it.

So why is this claim true? Well, let's look at the second one, for example. So what is the variance of u transpose X? So that's what I want to know. So that's the expectation-- so let's assume that X is 0, again, for same reasons as before.

So what is the variance? It's just the expectation of the square? I don't need to remove the expectation. And the expedition of the square is just the expectation of u transpose X. And

then I'm going to write the other one X transpose u.

And we know that this is deterministic. So I'm just going to take that this is just u transpose expectation of X X transpose u. And what is this guy? That's covariance sigma. That's just what sigma is.

So the variance I can write as u transpose sigma u. We've made this computation before. And now what I want to claim is that this thing is actually less than the largest eigenvalue, which I actually called lambda 1 here. I should probably not. And the P is-- well, OK. Let's just pretend everything is not empirical.

So now, I'm going to write sigma as P lambda 1 lambda n P transpose. That's just the eigendecomposition, where I admittedly reuse the same notation as I did for S. So I should really put some primes everywhere, so you know those are things that are actually different in practice. So this is just that the decomposition of sigma.

You seem confused, Helen. You have a question? Yeah?

**AUDIENCE:** What is-- when you talked about the empirical data and--

**PHILIPPE RIGOLLET:** So OK-- so I can make everything I'm saying, I can talk about either the variance or the empirical variance. And you can just add the word empirical in front of it whenever you want. The same thing works.

But just for the sake of removing the confusion, let's just do it again with S. So I'm just going to do everything with S. So I'm going to assume that X bar is equal to 0.

And here, I'm going to talk about the empirical variance, which is just 1/n sum from i equal 1 to n of u transpose Xi squared. So it's the same thing. Everywhere you see an expectation, you just put in average.

And then I get 1/n sum from i equal 1 to n of Xi Xi transpose. And now, I'm going to call this guy S, because that's what it is. So this is u transpose Su. But just defined that I could just replace the expectation by averages everywhere, you can tell that the thing is going to work for either one or the other.

So now, this thing was actually-- so now, I don't have any problem with my notation. This is actually the decomposition of S. That's just the spectral decomposition and it's to its

eigenvalues.

And so now, what I have is that when I look at u transpose Su, this is actually equal to P u transpose S Pu.

OK. There's a transpose somewhere. That's this guy. And that's this guy. Now-- sorry, that's not P, that's D. That's D, that's this diagonal matrix.

Let's look at this thing. And let's call P transpose u, let's call it b. So that's also a vector in Rd.

What is it? It's just, I take a unit vector, and then I apply P transpose to it. So that's basically what happens to a unit vector when I apply the same change of basis that I did. So I'm just changing my orthogonal system the same way I did for the other ones.

So what's happening when I write this? Well, now I have that u transpose Su is b transpose Db. But now, doing b transpose Db when D is diagonal and b is a vector is a very simple thing.

I can expand it. This is what? This is just the sum from j equal 1 to d of lambda j bj squared. So that's just like matrix vector multiplication.

And in particular, I know that the largest of those guys is lambda 1 and those guys are all non-negative. So this thing is actually less than lambda 1 times the sum from j equal 1 to d of lambda j squared-- sorry, bj squared. And this is just the norm of b squared.

So if I want to prove what's on the slide, all I need to check is that b has norm, which is--

**AUDIENCE:** 1.

**PHILIPPE RIGOLLET:** At most, 1. It's going to be at most 1. Why? Well, because b is really just a change of basis for u. And so if I take a vector, I'm just changing its basis. I'm certainly not changing its length-- think of a rotation, and I can also flip it, but think of a rotation-- well, actually, for vector, it's just going to be a rotation.

And so now, what I have I just have to check that the norm of b squared is equal to what? Well, it's equal to the norm of P transpose u squared, which is equal to u transpose P P transpose u. But P is orthogonal. So this thing is actually just the identity.

So that's just u transpose u, which is equal to the norm u squared, which is equal to 1, because I took u to have norm 1 in the first place. And so this-- you're right-- was actually of

norm equal to 1. I just needed to have it less, but it's equal. And so what I'm left with is that this thing is actually equal to lambda 1.

So I know that for every u that I pick-- that has norm-- So I'm just reminding you that u here has norm squared equal to 1. For every u that I pick, this u transpose Su is at mostly lambda 1. So that's the u transpose Su is at most lambda 1.

And we know that that's the variance, that's the empirical variance, when I project my points onto direction spanned by u. So now, I have an empirical variance, which is at most lambda 1. But I also know that if I take u to be something very specific-- I mean, it was on the previous board-- if I take u to be equal to v1, then this thing is actually not an inequality, this is an equality.

And the reason is, when I actually take u to be v1, all of these bj's are going to be 0, except for the one that's b1, which is itself equal to 1. So I mean, we can briefly check this. But if I take v-- if u is equal to v1, what I have is that u transpose Su is equal to P transpose v1 D P transpose v1.

But what is P transpose v1? Well, remember P transpose is just the matrix that has vectors v1 transpose here, v2 transpose here, all the way to vd transpose here. And we know that when I take vj transpose vk, I get 0, if j is different from k. And if j is equal to k, I get 1. So P transpose v1 is equal to what?

Take v1 here and multiply it. So the first coordinate is going to be v1 transpose v1, which is 1. The second coordinate is going to be v2 transpose v1, which is 0. And so I get 0's all the way, right?

So that means that this thing here is really just the vector 1, 0, 0. And here, this is just the vector 1, 0, 0. So when I multiply it with this guy, I am only picking up the top left element of D, which is lambda 1. So for every one, it's less lambda 1. And for v1, it's equal to lambda 1, which means that it's maximized for a equals v1.

And that's where I said that this is the fanciest non-convex problem we know how to solve. This was a problem that was definitely non-convex. We were maximizing a convex function over a sphere. But we know that v1, which is something-- I mean, of course, you still have to believe me that you can compute the spectral decomposition efficiently-- but essentially, if you've taken linear algebra, you know that you can diagonalize a matrix.

And so you get that v1 is just the maximum. So you can find your maximum just by looking at the spectral decomposition. You don't have to do any optimization or anything like this.

So let's recap. Where are we? We've established that if I start with my empirical covariance matrix, I can diagonalize it and PD P transpose. And then if I take the eigenvector associated to the largest eigenvalues-- so if I permute the columns of P and of D's in such a way that they are ordered from the largest to the smallest when I look at the diagonal elements of D, then if I pick the first column of P, it's v1. And v1 is the direction on which, if I project my points, they are going to carry the most empirical variance.

Well, that's a good way. If I told you, pick one direction along which if you were to project your points they would be as spread out as possible, that's probably the one you would pick. And so that's exactly what PCA is doing for us. It says, OK, if you ask me to take d prime equal to 1, I will take v1. I will just take the direction that's spanned by v1.

And that's just when I come back to this picture that was here before, this is v1. Of course, here, I only have two of them. So v2 has to be this guy, or this guy, or I mean or this thing. I mean, I don't know them up to sine.

But then if I have three-- think of like an olive in three dimensions-- then maybe I have one direction that's slightly more elongated than the other one. And so I'm going to pick the second one. And so the procedure is to say, well, first, I'm going to pick v1 the same way I pick v1 in the first place. So the first direction I am taking is the leading eigenvector.

And then I'm looking for a direction. Well, if I found one-- the one I'm going to want to find-- if you say you can take d equal 2, you're going to need the basis for this guy. So the second one has to be orthogonal to the first one you've already picked. And so the second one you pick is the one that's just, among all those that are orthogonal to v1, maximized the empirical variance when you project onto it.

And it turns out that this is actually exactly v2. You don't have to redo anything again. You're eigendecomposition, this is just the second column of P. Clearly, v2 is orthogonal to v1. We just used it here. This 0 here just says this v2 is orthogonal to v1.

So they're like this. And now, what I said-- what this slide tells you extra-- is that v2 among all those directions that are orthogonal-- I mean, there's still d minus 1 of them-- this is the one that maximizes the, say, residual empirical variance-- the one that was not explained by the

first direction that you picked. And you can check that. I mean, it's becoming a bit more cumbersome to write down, but you can check that. If you're not convinced, please raise your concern.

I mean, basically, one way you view this to-- I mean, you're not really dropping a coordinate, because v1 is not a coordinate. But let's assume actually for simplicity that v1 was actually equal to e1, that the direction that carries the most variance is the one that just says, just look at the first coordinate of X. So if that was the case, then clearly the orthogonal directions are the ones that comprise only of the coordinates 2 to d.

So you could actually just drop the first coordinate and do the same thing on a slightly shorter vector of length d minus 1. And then you would just look at the largest eigenvector of these guys, et cetera, et cetera. So in a way, that's what's happening, except that you rotate it before you actually do this. And that's exactly what's happening.

So what we put together here is essentially three things. One was statistics. Statistics says, if you won't spread, if you want information, you should be looking at variance.

The second one was optimization. Optimization said, well, if you want to maximize spread, well, you have to maximize variance in a certain direction. And that means maximizing over the sphere of vectors that have unique norm. And that's an optimization problem, which actually turned out to be difficult.

But then the third thing that we use to solve this problem was linear algebra. Linear algebra said, well, it looks like it's a difficult optimization problem. But it turns out that the answer comes in almost-- I mean, it's not a closed form, but those things are so used, that it's almost a closed form-- says, just pick the eigenvectors in order of their associated eigenvalues from largest to smallest.

And that's why principal component analysis has been so popular and has gained huge amount of traction since we had computers that were allowed to compute eigenvalues and eigenvectors for matrices of gigantic sizes. You can actually do that. If I give you-- I don't know, this Google video, for example, is talking about words.

They want to do just the, say, principal component analysis of words. So I give you all the words in the dictionary. And-- sorry, well, you would have to have a representation for words, so it's a little more difficult. But how do I do this?

Let's say, for example, pages of a book. I want to understand the pages of a book. And I need to turn it into a number. And a page of a book is basically the word count. So I just count the number of times "the" shows up, the number of times "and" shows up, number of times "dog" shows up. And so that gives me a vector.

It's in pretty high dimensions. It's as many dimensions as there are words in the dictionary. And now, I want to visualize how those pages get together-- are two pages very similar or not. And so what you would do is essentially just compute the largest eigenvector of this matrix-- maybe the two largest-- and then project this into a plane. Yeah.

**AUDIENCE:** Can we assume the number of points was far larger than the dimension?

**PHILIPPE RIGOLLET:** Yeah, but there's many pages in the world. There's probably more pages in the world than there's words in the dictionary. Yeah, so of course, if you are in high dimensions and you don't have enough points, it's going to be clearly an issue. If you have two points, then the leading eigenvector is going to be just the line that goes through those two points, regardless of what the dimension is. And clearly, you're not learning anything.

So you have to pick, say, the k largest one. If you go all the way, you're just reordering your thing, and you're not actually gaining anything. You start from d and you go too d. So at some point, this procedure has to stop. And let's say it stops at k.

Now, of course, you should ask me a question, which is, how do you choose k? So that's, of course, a natural question. Probably the basic answer is just pick k equals 3, because you can actually visualize it.

But what happens if I take k is equal to 4? If I take is equal to 4, I'm not going to be able to plot points in four dimensions. Well, I could, I could add color, or I could try to be a little smart about it. But it's actually quite difficult.

And so what people tend to do, if you have four dimensions, they actually do a bunch of two dimensional plots. And that's what a computer-- a computer is not very good-- I mean, by default, they don't spit out three dimensional plots. So let's say they want to plot only two dimensional things. So they're going to take the first directions of, say, v1, v2.

Let's say you have three, but you want to have only two dimensional plots. And then it's going to do v1, v3; and then v2, v3. So really, you take all three of them, but it's really just showing

you all choices of pairs of those guys. So if you were to keep k is equal to 5, you would have five, choose two different plots.

So this is the actual principal component algorithm, how it's implemented. And it's actually fairly simple. I mean, it looks like there's lots of steps. But really, there's only one that's important.

So the first one is the input. I give you a bunch of points, x1 to xn in d dimensions. And step two is, well, compute their empirical covariance matrix S. The points themselves, we don't really care. We care about their empirical covariance matrix. So it's a d by d matrix.

Now, I'm going to feed that. And that's where the actual computation starts happening. I'm going to feed that to something that knows how to diagonalize this matrix. And you have to trust me, if I want to compute the k largest eigenvalues and my matrix is d by d, it's going to take me about k times d squared operations.

So if I want only three, it's 3 times d squared, which is about-- d squared is the time for me it takes to just even read the matrix sigma. So that's not too bad. So what it's going to spit out, of course, is the diagonal matrix D. And those are nice, because they allow me to tell me what is the order in which I should be taking the columns of P. But what's really important to me is v1 to vd, because those are going to be the ones I'm going to be using to draw those plots.

And now, I'm going to say, OK, I need to actually choose some set k. And I'm going to basically truncate and look only at the first k columns of P. Once I have those columns, what I want to do is to project onto the linear span of those columns.

And there's actually a simple way to do this, which is just take this matrix P, which is really the matrix of projection onto the linear span of those k columns. And you just take Pk transpose. And then you apply this to every single one of your points. Now Pk transpose, what is the size of the matrix Pk? Yeah, [INAUDIBLE]?

**AUDIENCE:** [INAUDIBLE]

**PHILIPPE RIGOLLET:** So Pk is just this matrix. I take the v1 and I stop at vk-- well--

**AUDIENCE:** [INAUDIBLE]

**PHILIPPE RIGOLLET:** d by k, right? Each of the column is an eigenvector. It's of dimension d. I mean, that's a vector in the original space. So I have this d by k matrix.

So all it is is if I had my-- well, I'm going to talk in a second about Pk transpose. Pk transpose is just this guy, where I stop at the k-th vector. So Pk transpose is k by d.

So now, when I take Yi, which is Pk transpose Xi, I end up with a point which is in k dimensions. I have only k coordinates. So I took every single one of my original points Xi, which had d coordinates, and I turned it into a point that has only k coordinates. Particularly, I could have k is equal to 2.

This matrix is exactly the one that projects. If you think about it for one second, this is just the matrix that says-- well, we actually did that several times. The matrix, so that was this P transpose u that showed up somewhere. And so that's just the matrix that take your point X in, say, three dimensions, and then just project it down to two dimensions.

And that's just-- it goes to the closest point in the subspace. Now, here, the floor is flat. But we can pick any subspace we want, depending on what the lambdas are.

So the lambdas were important for us to be able to identify which columns to pick. The fact that we assumed that they were ordered tells us that we can pick the first ones. If they were not ordered, it would be just a subset of the columns, depending on what the size of the eigenvalue is. So each column is labeled.

And so then, of course, we still have this question of, how do I pick k? So there's definitely the matter of convenience. Maybe 2 is convenient. If it works for 2, you don't have to go any farther.

But you might want to say, well-- originally, I did that to actually keep as much information as possible. I know that the ultimate thing is to keep as much information, which would be to k is equal d-- that's as much information as you want. But it's essentially the same question about, well, if I want to compress a JPEG image, how much information should I keep so it's still visible?

And so there's some rules for that. But none of them is actually really a science. So it's really a matter of what you think is actually tolerable. And we're just going to start replacing this choice by maybe another parameter. So here, we're going to basically replace k by alpha, and so we just do stuff.

So the first one that people do that is probably the most popular one-- OK, the most popular one is definitely take k is equal to 2 or 3, because it's just convenient to visualize. The second most popular one is the scree plot. So the scree plot-- remember, I have my values, lambda j's. And I've chosen the lambda j's to decrease. So the indices are chosen in such a way that lambda is a decreasing function.

So I have lambda 1, and let's say it's this guy here. And then I have lambda 2, and let's say it's this guy here. And then I have lambda 3, and let's say it's this guy here, lambda 4, lambda 5, lambda 6. And all I care about is that this thing decreases.

The scree plot says something like this-- if there's an inflection point, meaning that you can sort of do something like this and then something like this, you should stop at 3. That's what the scree plot tells you. What it's saying in a way is that the percentage of the marginal increment of explained variance that you get starts to decrease after you pass this inflection point.

So let's see why I way this. Well, here, what I have-- so this ratio that you see there is actually the percentage of explained variance. So what it means is that, if I look at lambda 1 plus lambda k, and then I divide by lambda 1 plus lambda d, well, what is this?

Well, this lambda 1 plus lambda d is the total amount of variance that I get in my points. That's the trace of sigma. So that's the variance in the first direction plus the variance in the second direction plus the variance in the third direction. That's basically all the variance that I have possible.

Now, this is the variance that I kept in the first direction. This is the variance that I kept in the second direction, all the way to the variance that I kept in the k-th direction. So I know that this number is always less than or equal to 1. And it's larger than 1. And this is just the proportion, say, of variance explained by v1 to vk, or simply, the proportion of explained variance by my PCA, say.

So now, what this thing is telling me, its says, well, if I look at this thing and I start seeing this inflection point, it's saying, oh, here, you're gaining a lot and lot of variance. And then at some point, you stop gaining a lot in your proportion of explained variance. So this will translate in something where when I look at this ratio, lambda 1 plus lambda k divided by lambda 1 plus lambda d, this would translate into a function that would look like this.

And what it's telling you, it says, well, maybe you should stop here, because here every time you add one, you don't get as much as you did before. You actually get like smaller marginal returns. So explained variance is the numerator of this ratio. And the total variance is the denominator. Those are pretty straightforward terms that you would want to use for this.

So if your goal is to do data visualization-- so why would you take k larger than 2? Let's say, if you take k larger than 6, you can start to imagine that you're going to have six, choose two, which starts to be annoying. And if you have k is equal to 10-- because you could start in dimension 50,000-- and then k equal to 10 would be the place where you have this thing that's a lot of plots that you would have to show. So it's not always for data visualization.

Once I've actually done this, I've actually effectively reduced the dimension of my problem. And what I could do with what I have is do a regression on those guys. The v1-- so I forgot to tell you-- why is that called principal component analysis? Well, the vj's that I keep, v1 to vk are called principal components.

And they effectively act as the summary of my Xi's. When I mentioned image compression, I started with a point Xi that was d numbers-- let's say 50,000 numbers. And now, I'm saying, actually, you can throw out those 50,000 numbers. If you actually know only the k numbers that you need-- the 6 numbers that you need-- you're going to have something that was pretty close to getting what information you had. So in a way, there is some form of compression that's going on here.

And what you can do is that those principal components, you can actually use now for regression. If I want to regress Y onto X that's very high dimensional, before I do this, if I don't have enough points, maybe what I can actually do is to do principal component analysis throughout my exercise, replace them by those compressed versions, and do linear aggression on those guys. And that's called principal component regression, not surprisingly. And that's something that's pretty popular. And you can do with k is equal to 10, for example.

So for data visualization, I did not find a Thanksgiving themed picture. But I found one that has turkey in it. Get it? So this is actually a gene data set that was-- so when you see something like this, you can imagine that someone has been preprocessing the hell out of this thing. This is not like, oh, I collect data on 23andMe and I'm just going to run PCA on this. It just doesn't happen like that.

And so what happened is that-- so let's assume that this was a bunch of preprocessed data, which are gene expression levels-- so 500,000 genes among 1,400 Europeans. So here, I actually have less observations than I have samples. And that's when you use principal component regression most of the time, so it doesn't stop you.

And then what you do is you say, OK, have those 500,000 genes among-- so here, that means that there's 1,400 points here. And I actually take those 500,000 directions. So each person has a vector of, say, 500,000 genes that are attached to them. And I project them onto two dimensions, which should be extremely lossy.

I lose a lot of information. And indeed, I do, because I'm one of these guys. And I'm pretty sure I'm very different from this guy, even though probably from an American perspective, we're all the same. But I think we have like slightly different genomes.

And so the thing is now we have this-- so you see there's lots of Swiss that participate in this. But actually, those two principal components recover sort of the map of Europe. I mean, OK, again, this is actually maybe fine-grained for you guys. But right here, there's Portugal and Spain, which are those colors. So here is color-coded.

And here is Turkey, of course, which we know has very different genomes. So Turks are very at the boundary. So you can see all the greens. They stay very far apart from everything else. And then the rest here is pretty mixed.

But it sort of recovers-- if you look at the colors, it sort of recovers that. So in a way, those two principal components are just the geographic feature. So if you insist to compress all the genomic information of these people into two numbers, what you're actually going to get is longitude and latitude, which is somewhat surprising, but not so much if you think that's it's been preprocessed.

So what do you do beyond practice? Well, you could try to actually study those things. If you think about it for a second, we did not do any statistics. I talked to you about IID observations, but we never used the fact that they were independent.

The way we typically use independence is to have central limit theorem, maybe. I mentioned the fact that the covariances of the word Gaussian would actually give me something which is independent. We didn't care.

This was a data analysis, data mining process that we did. I give you points, and you just put

them through the crank. There was an algorithm in six steps. And you just put it through and that's what you got.

Now, of course, there's some work which studies says, OK, if my data is actually generated from some process-- maybe, my points are multivariate Gaussian with some structure on the covariance-- how well am I recovering the covariance structure? And that's where statistics kicks in. And that's where we stop. So this is actually a bit more difficult to study.

But in a way, it's not entirely satisfactory, because we could work for a couple of boards and I would just basically sort of reverse engineer this and find some models under which it's a good idea to do that. And what are those models? Well, those are the models that sort of give you sort of prominent directions that you want to find. And it will say, yes, if you have enough observations, you will find those directions along which your data is elongated. So that's essentially what you want to do.

So that's exactly what this thing is telling you. So where does the statistics lie from? Well, everything, remember-- so actually that's where Alana was confused-- the idea was to say, well, if I have a true covariance matrix sigma and I never really have access to it, I'm just running PCA on the empirical covariance matrix, how do those results relate?

And this is something that you can study. So for example, if n goes to infinity and the number of points, your dimension, is fixed, then S goes to sigma in any sense you want. Maybe each entry is going to each entry of sigma, for example. So S is a good estimator.

We know that the empirical covariance is a consistent as the mater. And if d is fixed, this is actually not an issue. So in particular, if you run PCA on the sample covariance matrix, you look at, say, v1, then v1 is going to converge to the largest eigenvector of sigma as n goes to infinity, but for d fixed. And that's a story that we know since the '60s.

More recently, people have started challenging this. Because what's happening when you fix the dimension and let the sample size go to infinity, you're certainly not allowing for this. It's certainly not explaining to you anything about the fact when d is equal to 500,000 and n is equal to 1,400. Because when d is fixed and n goes to infinity, in particular, n is much larger than d, which is not the case here.

And so when n is much larger than d, things go well. But if d is less than n, it's not clear what happens. And particularly, if d is of the order of n, what's happening?

So there's an entire theory in mathematics that's called random matrix theory that studies the behavior of exactly this question-- what is the behavior of the spectrum-- the eigenvalues and eigenvectors-- of a matrix in which I put random numbers and I let-- so the matrix I'm interested in here is the matrix of X's. When I stack all my X's next to each other, so that's a matrix of size, say, d by n, so each column is of size d, it's one person. And so I put them.

And when I let the matrix go to infinity, I let both d and n to infinity. But I want the aspect ratio, d/n, to go to some constant. That's what they do.

And what's nice is that in the end, you have this constant-- let's call it gamma-- that shows up in all the asymptotics. And then you can replace it by d/n. And you know that you still have a handle of both the dimension and the sample size. Whereas, usually the dimension goes away, as you let n go to infinity without having dimension going to infinity.

And so now, when this happens, as soon as d/n goes to a constant, you can show that essentially there's an angle between the largest eigenvector of sigma and the largest eigenvector of S, as n and d go to infinity. There is always an angle-- you can actually write it explicitly. And it's an angle that depends on this ratio, gamma-- the asymptotic ratio of d/n.

And so there's been a lot of understanding how to correct, how to pay attention to this. This creates some biases that were sort of overlooked before. In particular, when I do this, this is not the proportion of explained variance, when n and d are similar. This is an estimated number computed from S.

This is computed from S. All these guys are computed from S. So those are actually not exactly where you want them to be. And there's some nice work that allows you to recalibrate what this ratio should be, how this ratio should be computed, so it's a better representative of what the proportion of explained variance actually is.

So then, of course, there's the question of-- so that's when d/n goes to some constant. So the best case-- so that was '60s-- d is fixed and it's much larger than d. And then random matrix theory tells you, well, d and n are sort of the same order of magnitude. When they go to infinity, the ratio goes to some constant. Think of it as being order 1.

To be fair, if d is 100 times larger than n, it still works. And it depends on what you think what the infinity is at this point. But I think the random matrix theory results are very useful.

But then even in this case, I told you that the leading eigenvector of S is actually an angle of the leading eigenvector of-- So what's happening is that-- so let's say that d/n goes to some gamma. And what I claim is that, if you look at-- so that's v1, that's the v1 of S. And then there's the v1 of-- so this should be of size 1.

So that's the v1 of sigma. Then those things are going to have an angle, which is some function of gamma. It's complicated, but there's a function of gamma that you can see there. And there's some models. When gamma goes to infinity, which means that d is now much larger than n, this angle is 90 degrees, which means that you're getting nothing. Yeah.

**AUDIENCE:** If d is not on your lower plane, so like gamma is 0, is there still angle?

**PHILIPPE RIGOLLET:** No, but that's consistent-- the fact that it's consistent when-- so the angle is a function--

**AUDIENCE:** d is not a constant [INAUDIBLE]?

**PHILIPPE RIGOLLET:** d is not a constant? So if d is little of n? Then gamma goes to 0 and f of gamma goes to 0. So f of gamma is a function that-- so for example, if f of gamma-- this is the sine of the angle, for example-- then it's a function that starts at 0, and that goes like this. But as soon as gamma is positive, it goes away from 0.

So now when gamma goes to infinity, then this thing goes to a right angle, which means I'm getting just junk. So this is not my leading eigenvector. So how do you do this?

Well, just like everywhere in statistics, you have to just make more assumptions. You have to assume that you're not looking for the leading eigenvector or the direction that carries the most variance. But you're looking, maybe, for a special direction.

And that's what sparse PCA is doing. Sparse PCA is saying, I'm not looking for any direction new that carries the most variance. I'm only looking for a direction new that is sparse. Think of it, for example, as having 10 non-zero coordinates. So that's a lot of directions still to look for.

But once you do this, then you actually have not only-- there's a few things that actually you get from doing this. The first one is you actually essentially replace d by k, which means that n now just-- I'm sorry, let's say S non-zero coefficients. You replace d by S, which means that n only has to be much larger than S for this thing to actually work.

Now, of course, you've set your goal weaker. Your goal is not to find any direction, only a sparse direction. But there's something very valuable about sparse directions, is that they actually are interpretable. When I found the v-- let's say that the v that I found before was 0.2, and then 0.9, and then 1.1 minus 3, et cetera. So that was the coordinates of my leading eigenvector in the original coordinate system.

What does it mean? Well, it means that if I see a large number, that means that this v is very close-- so that's my original coordinate system. Let's call it e1 and e2. So that's just 1, 0; and then 0, 1. Then clearly, from the coordinates of v, I can tell if my v is like this, or it's like this, or it's like this.

Well, I mean, they should all be of the same size. So I can tell if it's here or here or here, depending on-- like here, that means I'm going to see something where the Y-coordinate it much larger than the X-coordinate. Here, I'm going to see something where the X-coordinate is much larger than the Y-coordinate. And here, I'm going to see something where the X-coordinate is about the same size of the Y-coordinate.

So when things starts to be bigger, you're going to have to make choices. What does it mean to be bigger-- when d is 100,000, I mean, the sum of the squares of those guys have to be equal to 1. So they're all very small numbers. And so it's hard for you to tell which one is a big number and which ones is a small number.

Why would you want to know this? Because it's actually telling you that if v is very close to e1, then that means that e1-- in the case of the gene example, that would mean that e1 is the gene that's very important. Maybe there's actually just two genes that explain those two things. And those are the genes that have been picked up. There's two genes that I encode geographic location, and that's it.

And so it's very important for you to be able to interpret what v means. Where it has large values, it means that maybe it has large values for e1, e2, and e3. And it means that it's a combination of e1, e2, and e3. And now, you can interpret, because you have only three variables to find.

And so sparse PCA builds that in. Sparse PCA says, listen, I'm going to want to have at most 10 non-zero coefficients. And the rest, I want to be 0. I want to be able to be a combination of at most 10 of my original variables. And now, I can do interpretation.

So the problem with sparse PCA is that it becomes very difficult numerically to solve this problem. I can write it. So the problem is simply maximize the variance u transpose, say, Su subject to-- well, I wanted to have u2 equal to 1. So that's the original PCA. But now, I also want that the sum of the indicators of the uj that are not equal to 0 is at most, say, 10.

This constraint is very non-convex. So I can relax it to a convex one like we did for linear aggression. But now, I've totally messed up with the fact that I could use linear algebra to solve this problem.

And so now, you have to go through much more complicated optimization techniques, which are called semidefinite programs, which do not scale well in high dimensions. And so you have to do a bunch of tricks-- numerical tricks. But there are some packages that implements some heuristics or some other things-- iterative thresholding, all sorts of various numerical tricks that you can do.

But the problem they are trying to solve is exactly this. Among all directions that I have norm 1, of course, because it's the direction that have at most, say, 10 non-zero coordinates, I want to find the one that maximizes the empirical variance.

Actually, let me let me just so you this. I wanted to show you an output of PCA where people are actually trying to do directly-- maybe-- there you go.

So right here, you see this is SPSS. That's a statistical software. And this is an output that was preprocessed by a professional-- not preprocessed, post-processed. So that's something where they read PCA.

So what is the data? This is raw data about you ask doctors what they think of the behavior of a particular sales representative for pharmaceutical companies. So pharmaceutical companies are trying to improve their sales force. And they're asking doctors how would they rate-- what do they value about their interaction with a sales representative.

So basically, there's a bunch of questions. One offers credible point of view on something trends, provides valuable networking opportunities. This is one question. Rate this on a scale from 1 to 5. That was the question. And they had a bunch of questions like this.

And then they asked 1,000 doctors to make those ratings. And what they want-- so each doctor now is a vector of ratings. And they want to know if there's different groups of doctors, what do doctors respond to. If there's different groups, then maybe they know that they can

actually address them separately, et cetera.

And so to do that, of course, there's lots of questions. And so what you want is to just first project into lower dimensions, so you can actually visualize what's going on. And this is what was done for this. So these are the three first principal component that came out.

And even though we ordered the values of the lambdas, there's no reason why the entries of v should be ordered. And if you look at the values of v here, they look like they're pretty much ordered. It starts at 0.784, and then you're at 0.3 around here. There's something that goes up again, and then you go down. Actually, it's marked in red every time it goes up again.

And so now, what they did is they said, OK, I need to interpret those guys. I need to tell you what this is. If you tell me, we found the principal component that really discriminates the doctors in two groups, the drug company is going to come back to you and say, OK, what is this characteristic? And you say, oh, it's actually a linear combination of 40 characteristics. And they say, well, we don't need you to do that. I mean, it cannot be a linear combination of anything you didn't ask.

And so for that, first of all, there's a post-processing of PCA, which says, OK, once I actually, say, found three principal components, that means that I found the dimension three space on which I want to project my points. In this base, I can pick any direction I want. So the first thing is that you do some sort of local arrangements, so that those things look like they are increasing and then decreasing. So you just change, you rotate your coordinate system in this three dimensional space that you've actually isolated.

And so once you do this, the reason to do that is that it sort of makes them big, sharp differences between large and small values of the coordinates of the thing you had. And why do you want this? Because now, you can say, well, I'm going to start looking at the ones that have large values. And what do they say? They say in-depth knowledge, in-depth knowledge, in-depth knowledge, knowledge about.

This thing is clearly something that actually characterizes the knowledge of my sales representative. And so that's something that doctors are sensitive to. That's something that really discriminates the doctors in a way. There's lots of variance along those things, or at least a lot of variance-- I mean, doctors are separate in terms of their experience with respect to this.

And so what they did is said, OK, all these guys, some of those they have large values, but I don't know how to interpret them. And so I'm just going to put the first block, and I'm going to call it medical knowledge, because all those things are knowledge about medical stuff.

Then here, I didn't know how to interpret those guys. But those guys, there's a big clump of large coordinates, and they're about respectful of my time, listens, friendly but courteous. This is all about the quality of interaction. So this block was actually called quality of interaction.

And then there was a third block, which you can tell starts to be spreading a little thin. There's just much less of them. But this thing was actually called fair and critical opinion.

And so now, you have three discriminating directions. And you can actually give them a name. Wouldn't it be beautiful if all the numbers in the gray box came non-zero and all the other numbers came zero-- there was no ad hoc choice. I mean, this is probably an afternoon of work to like scratch out all these numbers and put all these color codes, et cetera.

Whereas, you could just have something that tells you, OK, here are the non-zeros. If you can actually make a story around why this group of thing actually makes sense, such as it is medical knowledge, then good for you. Otherwise, you could just say, I can't. And that's what sparse PCA does for you.

Sparse PCA outputs something where all those numbers would be zero. And there would be exactly, say, 10 non-zero coordinates. And you can turn this knob off 10. You can make it 9.

Depending on what your major is, maybe you can actually go on with 20 of them and have the ability to tell the story about 20 different variables and how they fit in the same group. And depending on how you feel, it's easy to rerun the PCA depending on the value that you want here. And so you could actually just come up with the one you prefer.

And so that's the sparse PCA thing which I'm trying to promote. I mean, this is not super well-spread. It's a fairly new idea, maybe at most 10 years old. And it's not completely well-spread in statistical packages. But that's clearly what people are trying to emulate currently. Yes?

**AUDIENCE:** So what exactly does it mean that the doctors have a lot of variance in medical knowledge, quality of interaction, and fair and critical opinion? Like, it was saying that these are like the main things that doctors vary on, some doctors care. Like we could sort of characterize a doctor by, oh, he cares this much about medical knowledge, this much about the quality of

interaction, and this much about critical opinion. And that says most of the story about what this doctor wants from a drug representative?

**PHILIPPE RIGOLLET:** Not really. I mean, OK, let's say you pick only one. So that means that you would take all your doctors, and you would have one direction, which is quality of interaction. And there would be just spread out points here.

So there are two things that can happen. The first one is that there's a clump here, and then there's a clump here. That still represents a lot of variance. And if this happens, you probably want to go back in your data and see were these people visited by a different group than these people, or maybe these people have a different specialty.

I mean, you have to look back at your data and try to understand why you would have different groups of people. And if it's like completely evenly spread out, then all it's saying is that, if you want to have a uniform quality of interaction, you need to take measures on this. You need to have this to not be discrimination.

But I think really when it's becoming interesting it's not when it's complete spread out. It's when there's a big group here. And then there's almost no one here, and then there's a big group here. And then maybe there's something you can do. And so those two things actually give you a lot of variance.

So actually, maybe I'll talk about this. Here, this is sort of a mixture. You have a mixture of two different populations of doctors. And it turns out that principal component analysis-- so a mixture is when you have different populations-- think of like two Gaussians that are just centered at two different points, and maybe they're in high dimensions. And those are clusters of people, and you want to be able to differentiate those guys.

If you're in very high dimensions, it's going to be very difficult. But one of the first processing tools that people do is to do PCA. Because if you have one big group here and one big group here, it means that there's a lot of variance along the direction that goes through the centers of those groups.

And that's essentially what happened here. You could think of this as being two blobs in high dimensions. But you're really just projecting them into one dimension. And this dimension, hopefully, goes through the center.

And so as preprocessing-- so I'm going to stop here. But PCA is not just made for dimension

reduction. It's used for mixtures, for example. It's also used when you have graphical data.

What is the idea of PCA? It just says, if you have a matrix that seems to have low rank-- meaning that there's a lot of those lambda i's that are very small-- and then I see that plus noise, then it's a good idea to do PCA on this thing. And in particular, people use that in networks a lot.

So you take the adjacency matrix of a graph-- well, you sort of preprocess it a little bit, so it looks nice. And then if you have, for example, two communities in there, it should look like something that is low rank plus some noise. And low rank means that there's just very few non-zero-- well, low rank means this.

Low rank means that if you do the scree plot, you will see something like this, which means that if you throw out all the smaller ones, it should not really matter in the overall structure. And so you can use all-- these techniques are used everywhere these days, not just in PCA. So we call it PCA as statisticians. But people call it the spectral methods or SVD. So everyone--