PHILIPPE RIGOLLET: [INAUDIBLE] minus xi transpose t. I just pick whatever notation I want from a variable. And let's say it's t.

So that's the least squares estimator. And it turns out that, as I said last time, it's going to be convenient to think of those things as matrices. So here, I already have vectors. I've already gone from one dimension, just real valued random variables through random vectors when I think of each xi, but if I start stacking them together, I'm going to have vectors and matrices that show up.

So the first vector I'm getting is y, which is just a vector where I have y1 to yn. Then I have-- so that's a boldface vector. Then I have x, which is a matrix where I have-- well, the first coordinate is always 1. So I have 1, and then x1 xp minus 1, and that's-- sorry, x1 xp minus 1, and that's for observation 1. And then I have the same thing all the way down for observation n.

OK, everybody understands what this is? So I'm just basically stacking up all the xi's. So this i-th row is xi transpose. I am just stacking them up.

And so if I want to write all these things to be true for each of them, all I need to do is to write a vector epsilon, which is epsilon 1 to epsilon n. And what I'm going to have is that y, the boldface vector, now is equal to the matrix x times the vector beta plus the vector epsilon. And it's really just exactly saying what's there, because for 2-- so this is a vector, right? This is a vector.

And what is the dimension of this vector? n, so this is n observations. And for all these-- for two vectors to be equal, I need to have all the coordinates to be equal, and that's exactly the same thing as saying that this holds for i equal 1 to n.

But now, when I have this, I can actually rewrite the sum for t equals-- sorry, for i equals 1 to n of yi minus xi transpose beta squared, this turns out to be equal to the Euclidean norm of the vector y minus the matrix x times beta squared. And I'm going to put a 2 here so we know we're talking about the Euclidean norm. This just means this is the Euclidean norm.

That's the one we've seen before when we talked about chi squared-- that's the square norm

is the sum of the square of the coefficients, and then I take a square root, but here I have an extra square. So it's really just the sum of the square of the coefficients, which is this. And here are the coefficients.

So then, that I write this thing like that, then minimizing-- so my goal here, now, is going to solve minimum over t in our p of y minus x times t2 squared. And just like we did for one dimension, we can actually write optimality conditions for this. I mean, this is a function.

So this is a function from rp to r. And if I want to minimize it, all I have to do is to take its gradient and set it equal to 0. So minimum, set gradient to 0.

So that's where it becomes a little complicated. Now I'm going to have to take the gradient of this norm. It might be a little annoying to do.

But actually, what's nice about those things-- I mean, I remember that it was a bit annoying to learn. I mean, it's just basically rules of calculus that you don't use that much. But essentially, you can actually expend this norm. And you will see that the rules are basically the same as in one dimension, you just have to be careful about the fact that matrices do not commute.

So let's expand this thing. y minus xt squared-- well, this is equal to the norm of y squared plus the norm of x squared plus 2 times y transpose xt. That's just expanding the square in more dimensions. And this, I'm actually going to write as y squared plus-- so here, the norm squared of this guy, I always have that the norm of x squared is equal to x transpose x. So I'm going to write this as x transpose x, so it's t transpose x transpose xt plus 2 times y transpose xt.

So now, if I'm going to take the gradient with respect to t, I have basically three terms, and each of them has some sort of a different nature. This term is linear in t, and it's going to differentiate the same way that I differentiate a times x. I'm just going to keep the a. This guy is quadratic. t appears twice.

And this guy, I'm going to pick up a 2, and it's going to differentiate just like when I differentiate a times x squared. It's 2 times ax. And this guy is a constant with respect to t, so it's going to differentiate to 0.

So when I compute the gradient-- now, of course, all of these rules that I give you you can check by looking at the partial derivative with respect to each coordinate. But arguably, it's much faster to know the rules of differentiability. It's like if I gave you the function exponential x and I said, what is the derivative, and you started writing, well, I'm going to write exponential x

plus h minus exponential ax divided by h and let h go to 0. That's a bit painful.

AUDIENCE: Why did you transpose your-- why does x have to be [INAUDIBLE]?

PHILIPPE RIGOLLET: I'm sorry?

AUDIENCE: I was wondering why you times t times the [INAUDIBLE]?

PHILIPPE RIGOLLET: The transpose of 2ab is b transpose a transpose. If you're not sure about this, just make a and b have different size, and then you will see that there's some incompatibility. I mean, there's basically only one way to not screw that one up, so that's easy to remember.

So if I take the gradient, then it's going to be equal to what? It's going to be 0 plus-- we said here, this is going to differentiate like-- so think a times x squared. So I'm going to have 2ax. So here, basically, this guy is going to go to x transpose xt.

Now, I could have made this one go away, but that's the same thing as saying that my gradient is-- I can think of my gradient as being either a horizontal vector or a vertical vector. So if I remove this guy, I'm thinking of my gradient as being horizontal. If I remove that guy, I'm thinking of my gradient as being vertical. And that's what I want to think of, typically-- vertical vectors, column vectors.

And then this guy, well, it's like these guys just think a times x. So the derivative is just a, so I'm going to keep only that part here. Sorry, I forgot a minus somewhere-- yeah, here. Minus 2y transpose x.

And what I want is this thing to be equal to 0. So t, the optimal t, is called beta hat and satisfies-- well, I can cancel the 2's and put the minus on the other side, and what I get is that x transpose xt is equal to y transpose x. Yeah, that's not working for me.

Yeah, that's because when I took the derivative, I still need to make sure-- so it's the same question of whether I want things to be columns or rows. So this is not a column. If I remove that guy, y transpose t is a row. So I'm just going to take the transpose of this guy to make things work, and this is just going to be x transpose y. And this guy is x transpose y so that I have columns.

So this is just the linear equation in t. And I have to solve it, so it's of the form some matrix

times t is equal to another vector. And so that's basically in your system.

And the way to solve it, at least formally, is to just take the inverse of the matrix on the left. So if x transpose x is invertible, then-- sorry, that's beta hat is the t I want. I get that beta hat is equal to x transpose x inverse x transpose y. And that's the least squares estimator.

So here, I use this condition. I want it to be invertible so I can actually write its inverse. Here, I wrote, rank of x is equal to p. What is the difference?

Well, there's basically no difference. Basically, here, I have to assume-- what is the size of the matrix x transpose x?

[INTERPOSING VOICES]

PHILIPPE RIGOLLET: Yeah, so what is the size?

AUDIENCE: p by p.

PHILIPPE RIGOLLET: p by p. So this matrix is invertible if it's a rank p, if you know what rank means. If you don't, that just rank p means that it's invertible. So it's full rank and it's invertible.

And the rank of x transpose x is actually just the rank of x because this is the same matrix that you apply twice. And that's all it's saying. So if you're not comfortable with the notion of rank that you see here, just think of this condition just being the condition that x transpose x is invertible. And that's all it says.

What it means for it to be invertible-- this was true. We made no assumption up to this point. If x is not invertible, it means that there might be multiple solutions to this equation.

In particular, for a matrix to not be invertible, it means that there's some vector v. So if x transpose x is not invertible, then this is equivalent to there exists a vector v, which is not 0, and such that x transpose xv is equal to 0. That's what it means to not be invertible.

So in particular, if beta hat is a solution-- so this equation is sometimes called score equations, because the gradient is called the score, and so you're just checking if the gradient is equal to 0. So if beta hat satisfies star, then so does beta hat plus lambda v for all lambda in the real line. And the reason is because, well, if I start looking at-- what is x transpose x times beta hat plus lambda v? Well, by linearity, this is just x transpose x beta hat plus lambda x transpose x

times v.

But this guy is what? It's 0, just because that's what we assumed. We assumed that x transpose xv was equal to 0, so we're left only with this part, which, by star, is just x transpose y.

So that means that x transpose x beta hat plus lambda v is actually equal to x transpose y, which means that there's another solution, which is not just beta hat, but any move of beta hat along this direction v by any size. So that's going to be an issue, because you're looking for one estimator. And there's not just one, in this case, there's many. And so this is not going to be well-defined and you're going to have some issues.

So if you want to talk about the least squares estimator, you have to make this assumption. What does it imply in terms of, can I think of p being too n, for example, in this case? What happens if p is equal to 2n?

AUDIENCE: Well, then the rank of your matrix is only p/2.

PHILIPPE RIGOLLET: So the rank of your matrix is only p/2, so that means that this is actually not going to happen. I mean, it's not only p/2, it's at most p/2. It's at most the smallest of the two dimensions of your matrix. So if your matrix is n times 2n, it's at most n, which means that it's not going to be full rank, so it's not going to be invertible.

So every time the dimension p is larger than the sample size, your matrix is not invertible, and you cannot talk about the least squares estimator. So that's something to keep in mind. And it's actually a very simple thing. It's essentially saying, well, if p is lower than n, it means that you have more parameters to estimate than you have equations to estimate it.

So you have this linear system. There's one equation per observation. Each row, which was each observation, was giving me one equation. But then the number of unknowns in this linear system is p, and so I cannot solve linear systems that have more unknowns than they have equations. And so that's basically what's happening.

Now, in practice, if you think about what data sets look like these days, for example, people are trying to express some phenotype. So phenotype is something you can measure on people-- maybe the color of your eyes, or your height, or whether you have diabetes or not, things like this, so things that are macroscopic. And then they want to use the genotype to do that. They want to measure your-- they want to sequence your genome and try to use this to

predict whether you're going to be responsive to a drug or whether your r's are going to be blue, or something like this.

Now, the data sets that you can have-- people, maybe, for a given study about some sort of disease. Maybe you will sequence the genome of maybe 100 people. n is equal to 100. p is basically the number of genes they're sequencing. This is of the order of 100,000.

So you can imagine that this is a case where n is much, much smaller than p, and you cannot talk about the least squares estimator. There's plenty of them. There's not just one line like that, lambda times v that you can move away. There's basically an entire space in which you can move, and so it's not well-defined.

So at the end of this class, I will give you a short introduction on how you do this. This actually represents more and more. It becomes a more and more preponderant part of the data sets you have to deal with, because people just collect data.

When I do the sequencing, the machine allows me to sequence 100,000 genes. I'm not going to stop at 100 because doctors are never going to have cohorts of more than 100 patients. So you just collect everything you can collect.

And this is true for everything. Cars have sensors all over the place, much more than they actually gather data. There's data, there's-- we're creating, we're recording everything we can. And so we need some new techniques for that, and that's what high-dimensional statistics is trying to answer. So this is way beyond the scope of this class, but towards the end, I will give you some hints about what can be done in this framework because, well, this is the new reality we have to deal with.

So here, we're in a case where p's less than n and typically much smaller than n. So the kind of orders of magnitude you want to have is maybe p's of order 10 and n's of order 100, something like this. So you can scale that, but maybe 10 times larger.

So maybe you cannot solve this guy b for b hat, but actually, you can talk about x times b hat, even if p is larger than n. And the reason is that x times b hat is actually something that's very well-defined. So what is x times b hat? Remember, I started with the model. So if I look at this definition, essentially, what I had as the original thing was that the vector y was equal to x times beta plus the vector epsilon. That was my model.

So beta is actually giving me something. Beta is actually some parameter, some coefficients that are interesting. But a good estimator for-- so here, it means that the observations that I have are of the form x times beta plus some noise. So if I want to adjust the noise, remove the noise, a good candidate to do noise is x times beta hat. x times beta hat is something that should actually be useful to me, which should be close to x times beta.

So in the one-dimensional case, what it means is that if I have-- let's say this is the true line, and these are my x's, so I have-- these are the true points on the real line, and then I have my little epsilon that just give me my observations that move around this line. So this is one of epsilons, say epsilon i. Then I can actually either talk-- to say that I recovered the line, I can actually talk about recovering the right intercept or recovering the right slope for this line. Those are the two parameters that I need to recover. But I can also say that I've actually found a set of points that's closer to being on the line that are closer to this set of points right here than the original crosses that I observed.

So if we go back to the picture here, for example, what I could do is say, well, for this point here-- there was an x here-- rather than looking at this dot, which was my observation, I can say, well, now that I've estimated the red line, I can actually just say, well, this point should really be here. And actually, I can move all these dots so that they're actually on the red line. And this should be a better value, something that has less noise than the original y value that I should see. It should be close to the true value that I should be seeing without the extra noise.

So that's definitely something that could be of interest. For example, in imaging, you're not trying to understand-- so when you do imaging, y is basically an image. So think of a pixel image, and you just stack it into one long vector. And what you see is something that should look like some linear combination of some feature vectors, maybe.

So there's people created a bunch of features. They're called, for example, Gabor frames or wavelet transforms-- so just well-known libraries of variables x such that when you take linear combinations of those guys, this should looks like a bunch of images. And what you want for your image-- you don't care what the coefficients of the image are in these bases that you came up with. What you care about is the noise in the image. And so you really want to get x beta.

So if you want to estimate x beta, well, you can use x beta hat. What is x beta hat? Well, since beta hat is x transpose x inverse x transpose y, this is x transpose. That's my estimator for x

beta.

Now, this thing, actually, I can define even if I'm not low rank. So why is this thing interesting? Well, there's a formula for this estimator, but actually, I can visualize what this thing is.

So let's assume, for the sake of illustration, that n is equal to 3. So that means that y lives in a three-dimensional space. And so let's say it's here. And so I have my, let's say, y's here.

And I also have a plane that's given by the vectors x1 transpose x2 transpose, which is, by the way, 1-- sorry, that's not what I want to do. I'm going to say that n is equal to 3 and that p is equal to 2. So I basically have two vectors, 1, 1 and another one, let's assume that it's, for example, abc.

So those are my two vectors. This is x1, and this is x2. And those are my three observations for this guy.

So what I want when I minimize this, I'm looking at the point which can be formed as the linear combination of the columns of x, and I'm trying to find the guy that's the closest to y.

So what does it look like? Well, the two points, 1, 1, 1 is going to be, say, here. That's the point 1, 1, 1. And let's say that abc is this point.

So now I have a line that goes through those two guys. That's not really-- let's say it's going through those two guys. And this is the line which can be formed by looking only at linear combination.

So this is the line of x times t for t in r2. That's this entire line that you can get. Why is it-- yeah, sorry, it's not just a line, I also have to have t, all the 0's thing. So that actually creates an entire plane, which is going to be really hard for me to represent.

I don't know. I mean, maybe I shouldn't do it in these dimensions. So I'm going to do it like that. So this plane here is the set of xt for t and r2.

So that's a two-dimensional plane, definitely goes to 0, and those are all these things. So think of a sheet of paper in three dimensions. Those are the things I can get.

So now, what I'm going to have as y is not necessarily in this plane. y is actually something in this plane, x beta plus some epsilon. y is x beta plus epsilon. So I start from this plane, and then I have this epsilon that pushes me, maybe, outside of this plane.

And what least squares is doing is saying, well, I know that epsilon should be fairly small, so the only thing I'm going to be doing that actually makes sense is to take y and find the point that's on this plane that's the closest to it. And that corresponds to doing an orthogonal projection of y onto this thing, and that's actually exactly x beta hat. So in one dimension, just because this is actually a little hard-- in one dimension, so that's if p is equal to 1.

So let's say this is my point. And then I have y, which is in two dimensions, so this is all on the plane. What it does, this is my-- the point that's right here is actually x beta hat.

That's how you find x beta hat. You take your point y and you project it on the linear span of the columns of x. And that's x beta hat. This does not tell you exactly what beta should be.

And if you know a little bit of linear algebra, it's pretty clear, because if you want to find beta hat, that means that you should be able to find the coordinates of a point in the system of columns of x. And if those guys are redundant, there's not going to be unique coordinates for these guys, so that's why it's actually not easy to find. But x beta hat is uniquely defined. It's a projection. Yeah?

AUDIENCE: And epsilon is the distance between the y and the--

PHILIPPE RIGOLLET: No, epsilon is the vector that goes from-- so there's a true x beta. That's the true one. It's not clear. I mean, x beta hat is unlikely to be exactly equal to x beta.

And then the epsilon is the one that starts from this line. It's the vector that pushes you away. So really, this is this vector. That's epsilon. So it's not a length. The lengths of epsilon is the distance, but epsilon is just the actual vector that takes you from one to the other. So this is all in two dimensions, and it's probably much clearer than what's here.

And so here, I claim that this x beta hat-- so from this picture, I implicitly claim that forming this operator that ticks y and maps it into this vector x times x transpose y, blah, blah, blah, this should actually be equal to the projection of y onto the linear span of the columns of x. That's what I just drew for you. And what it means is that this matrix must be the projection matrix.

So of course, anybody-- who knows linear algebra here? OK, wow. So what are the conditions that a projection matrix should be satisfying?

AUDIENCE: Squares through itself.

PHILIPPE RIGOLLET: Squares through itself, right. If I project twice, I'm not moving. If I keep on iterating projection, once I'm in the space I'm projecting onto, I'm not moving. What else? Do they have to be symmetric, maybe?

AUDIENCE: If it's an orthogonal projection.

PHILIPPE RIGOLLET: Yeah, so this is an orthogonal projection. It has to be symmetric. And that's pretty much it. So from those things, you can actually get quite a bit of things.

But what's interesting is that if you actually look at the eigenvalues of this matrix, they should be either 0 or 1, essentially. And they are 1 if the eigenvector associated is within this space, and 0 otherwise. And so that's basically what you can check. This is not an exercise in linear algebra, so I'm not going to go too much into those details. But this is essentially what you want to keep in mind.

What's associated to orthogonal projections is Pythagoras theorem. And that's something that's going to be useful for us. What it's essentially telling is that if I look at this norm squared, it's equal to this norm squared-- sorry, this norm squared plus this norm squared is equal to this norm squared. And that's something the norm of y squared.

So Pythagoras tells me that the norm of y squared is equal to the norm of x beta hat squared plus the norm of y minus x beta hat squared. Agreed? It's just because I have a straight angle here. So that's this plus this is equal to this.

So now, to define this, I made no assumption. Epsilon could be as wild. I was just crossing my fingers that epsilon was actually small enough that it would make sense to project onto the linear span, because I implicitly assumed that epsilon did not take me all the way there, so that actually, it makes sense to project back.

And so for that, I need to somehow make assumptions that epsilon is well-behaved and that it's completely wild, that it's moving uniformly in all directions of the space. There's no privileged direction where it's always going, otherwise, I'm going to make a systematic error. And I need that those epsilons are going to average somehow.

So here are the assumptions we're going to be making so that we can actually do some statistical inference. The first one is that the design matrix is deterministic. So I started by saying the x-- I have xi, yi, and maybe they're independent. Here, they are, but the xi's, I want

to think as deterministic.

If they're not deterministic, it can condition on them, but otherwise, it's very difficult to think about this thing if I think of those entries as being random, because then I have the inverse of a random matrix, and things become very, very complicated. So we're to think of those guys as being deterministic. We're going to think of the model as being homoscedastic. And actually, let me come back to this in a second.

Homoscedastic-- well, I mean, if you're trying to find the etymology of this word, "homo" means the same, "scedastic" means scaling. So what I want to say is that the epsilons have the same scaling. And since my third assumption is that epsilon is Gaussian, then essentially, what I'm going to want is that they all share the same sigma squared.

They're independent, so this is definitely in the identity covariance matrix. And I want them to be centered, as well. That means that there's no direction that I'm always privileging when I'm moving away from my plane there.

So these are important conditions. It depends on how much inference you want to do. If you want to write t-tests, you need all these assumptions. But if you only want to write, for example, the fact that your least squares estimator is consistent, you really just need the fact that epsilon has variance sigma squared. The fact that it's Gaussian won't matter, just like Gaussianity doesn't matter for a large number. Yeah?

AUDIENCE: So the first assumption that x has to be deterministic, but I just made up this x1, x2--

PHILIPPE RIGOLLET: x is the matrix.

AUDIENCE: Yeah. So most are random variables, right?

PHILIPPE RIGOLLET: No, that's the assumption.

AUDIENCE: OK. So I mean, once we collect the data and put it in the matrix, it becomes deterministic. So maybe I'm missing something.

PHILIPPE RIGOLLET: Yeah. So this is for the purpose of the analysis. I can actually assume that-- I look at my data, and I think of this. So what is the difference between thinking of data as deterministic or

thinking of it as random? When I talked about random data, the only assumptions that I made were about the distribution. I said, well, if my x is a random variable, I want it to have this variance and I want it to have, maybe, this distribution, things like this.

Here, I'm actually making an assumption on the values that I see. I'm seeing that the value that you give me is-- the matrix is actually invertible. x transpose x will be invertible. So I've never done that before, assuming that some random variable-- assuming that some Gaussian random variable was positive, for example. We don't do that, because there's always some probability that things don't happen if you make things at random.

And so here, I'm just going to say, OK, forget about-- here, it's basically a little stronger. I start my assumption by saying, the data that's given to me will actually satisfy those assumptions. And that means that I don't actually need to make some modeling assumption on this thing, because I'm actually putting directly the assumption I want to see. So here, either I know sigma squared or I don't know sigma squared.

So is that clear? So essentially, I'm assuming that I have this model, where this guy, now, is deterministic, and this is some multivariate Gaussian with mean 0 and covariance matrix identity of rn. That's the model I'm assuming. And I'm observing this, and I'm given this matrix x.

Where does this make sense? You could say, well, if I think of my rows as being people and I'm collecting genes, it's a little intense to assume that I actually know, ahead of time, what I'm going to be seeing, and that those things are deterministic. That's true, but it still does not prevent the analysis to go through, for one.

And second, a better example might be this imaging example that I described, where those x's are actually libraries. Those are libraries of patterns that people have created, maybe from deep learning nets, or something like this. But they've created patterns, and they say that all images should be representable as a linear combination of those patterns. And those patterns are somewhere in books, so they're certainly deterministic. Everything that's actually written down in a book is as deterministic as it gets.

Any questions about those assumptions? Those are the things we're going to be working with. There's only three of them. One is about x.

Actually, there's really two of them. I mean, this guy already appears here. So there's two--

one on the noise, one on the x's. That's it.

Those things allow us to do quite a bit. They will allow us to-- well, that's actually-- they allow me to write the distribution of beta hat, which is great, because when I know the distribution of my estimator, I know it's fluctuations. If it's centered around the true parameter, I know that it's going to be fluctuating around the true parameter. And it should tell me what kind of distribution the fluctuations are.

I actually know how to build confidence intervals. I know how to build tests. I know how to build everything. It's just like when I told you that asymptotically, the empirical variance was Gaussian with mean theta and standard deviation that depended on n, et cetera, that's basically the only thing I needed. And this is what I'm actually getting here.

So let me start with this statement. So remember, beta hat satisfied this, so I'm going to rewrite it here. So beta hat was equal to x transpose x inverse x transpose y. That was the definition that we found.

And now, I also know that y was equal to x beta plus epsilon. So let me just replace y by x beta plus epsilon here. Yeah?

AUDIENCE: Isn't it x transpose x inverse x transpose y?

PHILIPPE RIGOLLET: Yes, x transpose. Thank you. So I'm going to replace y by x beta plus epsilon. So that's-- and here comes the magic.

I have an inverse of a matrix, and then I have the true matrix, I have the original matrix. So this is actually the identity times beta. And now this guy, well, this is a Gaussian, because this is a Gaussian random vector, and I just multiply it by a deterministic matrix. So we're going to use the rule that if I have, say, epsilon, which is n0 sigma, then b times epsilon is n0-- can somebody tell me what the covariance matrix of b epsilon is?

AUDIENCE: What is capital B in this case?

PHILIPPE RIGOLLET: It's just a matrix. And for any matrix, I mean any matrix that I can premultiply-- that I can postmultiply with epsilon. Yeah?

AUDIENCE: b transpose b.

PHILIPPE b transpose?

RIGOLLET:

AUDIENCE: Times b.

PHILIPPE RIGOLLET: And sigma is gone.

AUDIENCE: Oh, times sigma, sorry.

PHILIPPE RIGOLLET: That's the matrix, right?

AUDIENCE: b transpose sigma b.

PHILIPPE RIGOLLET: Almost. Anybody wants to take a guess at the last one? I think we've removed all other possibilities. It's b sigma b transpose. So if you ever answered to the question, do you know Gaussian random vectors, but you did not know that, there's a gap in your knowledge that you need to fill, because that's probably the most important property of Gaussian vectors. When you multiply them by matrices, you have a simple rule on how to update the covariance matrix.

So here, sigma is the identity. And here, this is the matrix b that I had here. So what this is is, basically, n, some multivariate n, of course. Then I'm going to have 0.

And so what I need to do is b times the identity times b transpose, which is just b b transpose. And what is it going to tell me? It's x transpose x-- sorry, that's inverse-- inverse x transpose, and then the transpose of this guy, which is x x transpose x inverse transpose.

But this matrix is symmetric, so I'm actually not going to make the transpose of this guy. And again, magic shows up. Inverse times the matrix of those two guys cancel, and so this is actually equal to beta plus some n0 x transpose x inverse. Yeah?

AUDIENCE: I'm a little lost on the [INAUDIBLE]. So you define that as the b matrix, and what happens?

PHILIPPE RIGOLLET: So I just apply this rule, right?

AUDIENCE: Yeah.

PHILIPPE RIGOLLET: So if I multiply a matrix by a Gaussian, then let's say this Gaussian had mean 0, which is the case of epsilon here, then the covariance matrix that I get is b times the original covariance

matrix times b transpose. So all I did is write this matrix times the identity times this matrix transpose. And the identity, of course, doesn't play any role, so I can remove it. It's just this matrix, then the matrix transpose.

And what happened? So what is the transpose of this matrix? So I used the fact that if I look at x transpose x inverse x transpose, and now I look at the whole transpose of this thing, that's actually equal 2. And I use the rule that ab transpose is b transpose a transpose-- let me finish-- and it's x x transpose x inverse. Yes?

AUDIENCE: I thought the-- for epsilon, it was sigma squared.

PHILIPPE RIGOLLET: Oh, thank you. There's a sigma squared somewhere. So this was sigma squared times the identity, so I can just pick up a sigma squared anywhere. So here, in our case, so for epsilon, this is sigma. Sigma squared times the identity, that's my covariance matrix.

You seem perplexed.

AUDIENCE: It's just a new idea for me to think of a maximum likelihood estimator as a random variable.

PHILIPPE RIGOLLET: Oh, it should not be. Any estimator is a random variable.

AUDIENCE: Oh, yeah, that's a good point.

PHILIPPE RIGOLLET: [LAUGHS] And I have not told you that this was the maximum likelihood estimator just yet. The estimator is a random variable. There's a word-- some people use estimate just to differentiate the estimator while you're doing the analysis with random variables and the values when you plug in the numbers in there. But then, of course, people use estimate because it's shorter, so then it's confusing.

So any questions about this computation? Did I forget any other Greek letter along the way? All right, I think we're good.

So one thing that it says-- and actually, thank you for pointing this out-- I said there's actually a little hidden statement there. By the way, this answers this question. Beta hat is of the form beta plus something that's centered, so it's indeed of the form Gaussian with mean beta and covariance matrix sigma squared x transpose x inverse.

So that's very nice. As long as x transpose x is not huge, I'm going to have something that is

close to what I want. Oh, sorry, x transpose x inverse is not huge.

So there's a hidden claim in there, which is that least squares estimator is equal to the maximum likelihood estimator. Why does the maximum likelihood estimator just enter the picture now? We've been talking about regression for the past 18 slides.

And we've been talking about estimators. And I just dumped on you the least squares estimator, but I never really came back to this thing that we know-- maybe the method of moments, or maybe the maximum likelihood estimator. It turns out that those two things are the same.

But if I want to talk about a maximum likelihood estimator, I need to have a likelihood. In particular, I need to have a density. And so if I want a density, I have to make those assumptions, such as the epsilons have this Gaussian distribution.

So why is this the maximum likelihood estimator? Well, remember, y is x transpose beta plus epsilon. So I actually have a bunch of data.

So what is my model here? Well, its the family of Gaussians on n observations with mean x beta, variance sigma squared identity, and beta lives in rp. Here's my family of distributions. That's the possible distributions for y.

And so in particular, I can write the density of y. Well, what is it? It's something that looks like p of x-- well, p of y, let's say, is equal to 1 over-- so now its going to be a little more complicated, but its sigma squared times 2 pi to the p/2 exponential minus norm of y minus x beta squared divided by 2 sigma squared.

So that's just the multivariate Gaussian density. I just wrote it. That's the density of a multivariate Gaussian with mean x beta and covariance matrix sigma squared times the identity. That's what it is. So you don't have to learn this by heart, but if you are familiar with the case where p is equal to 1, you can check that you recover what you're familiar with, and this makes sense as an extension.

So now, I can actually write my log likelihood. How many observations do I have of this vector y? Do I have n observations of y? I have just one, right?

Oh, sorry, I shouldn't have said p, this is n. Everything is in dimension n. So I can think of either having n independent observations of each coordinate, or I can think of having just one

observation of the vector y.

So when I write my log likelihood, it's just the log of the density at y. And that's the vector y, which I can write as minus n/2 log sigma squared 2 pi minus 1 over 2 sigma squared norm of y minus x beta. And that's, again, my boldface y.

And what is my maximum likelihood estimator? Well, this guy does not depend on beta. And this is just a constant factor in front of this guy. So it's the same thing as just minimizing, because I have a minus sign, over all beta and rp.

y minus x beta squared, and that's my least squares estimator. Is there anything that's unclear on this board? Any question?

So all I used was-- so I wrote my log likelihood, which is just the log of this expression where y is my observation. And that's indeed the observation that I have here. And that was just some constant minus some constant times this quantity that depends on beta.

So maximizing this whole thing is the same thing as minimizing only this thing. The minimizers are the same. And so that tells me that I actually just have to minimize the squared norm to get my maximum likelihood estimator.

But this used, heavily, the fact that I could actually write exactly what my density was, and that when I took the log of this thing, I had exactly the square norm that showed up. If I had a different density, if, for example, I assumed that my coordinates of epsilons were, say, iid double exponential random variables. So it's just half of an exponential. And the plus is half of an exponential on the negatives.

So if I said that, then this would not have the square norm that shows up. This is really idiosyncratic to Gaussians. If I had something else, I would have, maybe, a different norm here, or something different measures the difference between y and x beta. And that's how you come up with other maximum likelihood estimators that leads to other estimators that are not the least squares-- maybe the least absolute deviation, for example, or this fourth movement, for example, that you suggested last time.

So I can come up with a bunch of different things, and they might be tied-- maybe I can come up from them from the same perspective that I came from the least squares estimator. I said, let's just do something smart and check, then, that it's indeed the maximum likelihood

estimator. Or I could just start with the modeling on-- and check, then, what happens-- what was the implicit assumption that I put on my noise. Or I could start with the assumption of the noise, compute the maximum likelihood estimator and see what it turns into.

So that was the first thing. I've just proved to you the first line. And from there, you can get what you want. So all the other lines are going to follow.

So what is beta hat-- so for example, let's look at the second line, the quadratic risk. Beta hat minus beta, from this formula, has a distribution, which is n n0, and then I have x transpose x inverse.

AUDIENCE: Wouldn't the dimension be p on the board?

PHILIPPE RIGOLLET: Sorry, the dimension of what?

AUDIENCE: Oh beta hat minus beta. Isn't beta only a p dimensional?

PHILIPPE RIGOLLET: Oh, yeah, you're right, you're right. That was all p dimensional there. Yeah. So if b here, the matrix that I'm actually applying, has dimension p times n-- so even if epsilon was an n dimensional Gaussian vector, then b times epsilon is a p dimensional Gaussian vector now. So that's how I switch from p to n-- from n to p. Thank you.

So you're right, this is beta hat minus beta is this guy. And so in particular, if I look at the expectation of the norm of beta hat minus beta squared, what is it? It's the expectation of the norm of some Gaussian vector.

And so it turns out-- so maybe we don't have-- well, that's just also a property of a Gaussian vector. So if epsilon is n0 sigma, then the expectation of the norm of epsilon squared is just the trace of sigma. Actually, we can probably check this by saying that this is the sum from j equal 1 to p of the expectation of beta hat j minus beta j squared. Since beta j squared is the expectation-- beta j is the expectation of beta hat. This is actually equal to the sum from j equal 1 to p of the variance of beta hat j, just because this is the expectation of beta hat.

And how do I read the variances in a covariance matrix? There are just the diagonal elements. So that's really just sigma jj. And so that's really equal to-- so that's the sum of the diagonal elements of this matrix. Let's call it sigma.

So that's equal to the trace of x transpose x inverse. The trace is the sum of the diagonal elements of a matrix. And I still had something else. I'm sorry, this was sigma squared. I forget it all the time.

So the sigma squared comes out. It's there. And so the sigma squared comes out because the trace is a linear operator. If I multiply all the entries of my matrix by the same number, then all the diagonal elements are multiplied by the same number, so when I sum them, the sum is multiplied by the same number. So that's for the quadratic risk of beta hat.

And now I need to tell you about x beta hat. x beta hat was something that was actually telling me that that was the point that I reported on the red line that I estimated. That was my x beta hat. That was my y minus the noise.

Now, this thing here-- so remember, we had this line, and I had my observation. And here, I'm really trying to measure this distance squared. This distance is actually quite important for me because it actually shows up in the Pythagoras theorem. And so you could actually try to estimate this thing.

So what is the prediction error? So we said we have y minus x beta hat, so that's the norm of this thing we're trying to compute. But let's write this for what it is for one second.

So we said that beta hat was x transpose x inverse extra transpose y, and we know that y is x transpose beta plus epsilon. So let's write this-- x beta plus epsilon plus x. And actually, maybe I should not write it. Let me keep the y for what it is now.

So that means that I have, essentially, the identity of rn times y minus this matrix times y. So I can factor y out, and that's the identity of rn minus x x transpose x inverse x transpose, the whole thing times y. We call this matrix p because this was the projection matrix onto the linear span of the x's. So that means that if I take a point x and I apply p times x, I'm projecting onto the linear span of the columns of x.

What happens if I do i minus p times x? It's x minus px. So if I look at the point on which-- so this is the point on which I project. This is x. I project orthogonally to get p times x. And so what it means is that this operator i minus px is actually giving me this guy, this vector here-- x minus p times x.

Let's say this is 0. This means that this vector, I can put it here. It's this vector here. And that's actually the orthogonal projection of x onto the orthogonal complement of the span of the

columns of x.

So if I project x, or if I look of x minus its projection, I'm basically projecting onto two orthogonal spaces. What I'm trying to say here is that this here is another projection matrix p prime. That is just the projection matrix onto the orthogonal-- projection onto orthogonal of column span of x. Orthogonal means the set of vectors that's orthogonal to everyone in this linear space.

So now, when I'm doing this, this is exactly what-- I mean, in a way, this is illustrating this Pythagoras theorem. And so when I want to compute the norm of this guy, the norm squared of this guy, I'm really computing-- if this is my y now, this is px of y, I'm really controlling the norm squared of this thing. So if I want to compute the norm squared-- so I'm almost there.

So what am I projecting here onto the orthogonal projector? So here, y, now, I know that y is equal to x beta plus epsilon. So when I look at this matrix p prime times y, It's actually p prime times x beta plus p prime times epsilon.

What's happening to p prime times x beta? Let's look at this picture. So we know that p prime takes any point here and projects it orthogonally on this guy. But x beta is actually a point that lives here. It's something that's on the linear span. So where do all the points that are on this line get projected to?

AUDIENCE: The origin.

PHILIPPE RIGOLLET: The origin, to 0. They all get projected to 0. And that's because I'm basically projecting something that's on the column span of x onto its orthogonal. So that's always 0 that I'm getting here.

So when I apply p prime to y, I'm really just applying p prime to epsilon. So I know that now, this, actually, is equal to the norm of some multivariate Gaussian. What is the size of this Gaussian? What is the size of this matrix?

Well, I actually had it there. It's i n, so it's n dimensional. So it's some n dimensional with mean 0. And what is the covariance matrix of p prime times epsilon?

AUDIENCE: p p transpose.

PHILIPPE RIGOLLET: Yeah, p prime p prime transpose, which we just said p prime transpose is p, so that's p squared. And we see that when we project twice, it's as if we projected only once. So here, this

is n0 p prime p prime transpose. That's the formula for the covariance matrix. But this guy is actually equal to p prime times p prime, which is equal to p prime.

So now, what I'm looking for is the norm squared of the trace. So that means that this whole thing here is actually equal to the trace. Oh, did I forget again a sigma squared? Yeah, I forgot it only here, which is good news.

So I should assume that sigma squared is equal to 1. So sigma squared's here. And then what I'm left with is sigma squared times the trace of p prime.

At some point, I mentioned that the eigenvalues of a projection matrix were actually 0 or 1. The trace is the sum of the eigenvalues. So that means that the trace is going to be an integer number as the number of non-0 eigenvalues. And the non-0 eigenvalues are just the dimension of the space onto which I'm projecting.

Now, I'm projecting from something of dimension n onto the orthogonal of a space of dimension p. What is the dimension of the orthogonal of a space of dimension p when thought of space in dimension n?

AUDIENCE: [? 1. ?]

PHILIPPE RIGOLLET: N minus p-- that's the so-called rank theorem, I guess, as a name. And so that's how I get this n minus p here. This is really just equal to n minus p. Yeah?

AUDIENCE: Here, we're taking the expectation of the whole thing.

PHILIPPE RIGOLLET: Yes, you're right. So that's actually the expectation of this thing that's equal to that. Absolutely.

But I actually have much better. I know, even, that the norm that I'm looking at, I know it's going to be this thing. What is going to be the distribution of this guy? Norm squared of a Gaussian, chi squared. So there's going to be some chi squared that shows up.

And the number of degrees of freedom is actually going to be also n minus p. And maybe it's actually somewhere-- yeah, right here-- n minus p times sigma hat squared over sigma squared. This is my sigma hat squared. If I multiply n minus p, I'm left only with this thing, and so that means that I get sigma squared times-- because they always forget my sigma squared-- I get sigma squared times this thing. And it turns out that the square norm of this

guy is actually exactly chi squared with n minus b degrees of freedom.

So in particular, so we know that the expectation of this thing is equal to sigma squared times n minus p. So if I divide both sides by n minus p, I'm going to have that something whose expectation is sigma squared. And this something, I can actually compute.

It depends on y, and x that I know, and beta hat that I've just estimated. I know what n is. And pr's are the dimensions of my matrix x. So I'm actually given an estimator whose expectation is sigma squared.

And so now, I actually have an unbiased estimator of sigma squared. That's this guy right here. And it's actually super useful.

So those are called the-- this is the normalized sum of square residuals. These are called the residuals. Those are whatever is residual when I project my points onto the line that I've estimated. And so in a way, those guys-- if you go back to this picture, this was yi and this was xi transpose beta hat.

So if beta hat is close to beta, the difference between yi and xi transpose beta should be close to my epsilon i. It's some sort of epsilon i hat. Agreed? And so that means that if I think of those as being epsilon i hat, they should be close to epsilon i, and so their norm should be giving me something that looks like sigma squared.

And so that's why it actually makes sense. It's just magical that everything works out together, because I'm not projecting on the right line, I'm actually projecting on the wrong line. But in the end, things actually work out pretty well.

There's one thing-- so here, the theorem is that this thing not only has the right expectation, but also has a chi squared distribution. That's what we just discussed. So here, I'm just telling you this. But it's not too hard to believe, because it's actually the norm of some vector. You could make this obvious, but again, I didn't want to bring in too much linear algebra.

So to prove this, you actually have to diagonalize the matrix p. So you have to invoke the eigenvalue decomposition and the fact that the norm is invariant by rotation. So for those who are familiar with, what I can do is just look at the decomposition of p prime into ud u transpose where this is an orthogonal matrix, and this is a diagonal matrix of eigenvalues.

And when I look at the norm squared of this thing, I mean, I have, basically, the norm squared

of p prime times some epsilon. It's the norm of ud u transpose epsilon squared. The norm of a rotation of a vector is the same as the norm of the vector, so this guy goes away.

This is not actually-- I mean, you don't have to care about this if you don't understand what I'm saying, so don't freak out. This is really for those who follow. What is the distribution of u transpose epsilon? I take a Gaussian vector that has covariance matrix sigma squared times the [? identity, ?] and I basically rotate it. What is its distribution? Yeah?

AUDIENCE: The same.

PHILIPPE RIGOLLET: It's the same. It's completely invariant, because the Gaussian think of all directions as being the same. So it doesn't really matter if I take a Gaussian or a rotated Gaussian. So this is also a Gaussian, so I'm going to call it epsilon prime. And I am left with just the norm of epsilon primes.

So this is the sum of the dj's squared times epsilon j squared. And we just said that the eigenvalues of p are either 0 or 1, because it's a projector. And so here, I'm going to get only 0's and 1's. So I'm really just summing a certain number of epsilon i squared.

So square root of standard Gaussians-- sorry, with a sigma squared somewhere. And basically, how many am I summing? Well, the n minus p, the number of non-0 eigenvalues of p prime. So that's how it shows up.

When you see this, what theorem am I using here? Cochran's theorem. This is this magic book. I'm actually going to dump everything that I'm not going to prove to you and say, oh, this is actually Cochran's. No, Cochran's theorem is really just telling me something about orthogonality of things, and therefore, independence of things.

And Cochran's theorem was something that I used when I wanted to use what? That's something I used just one slide before. Student t-test, right? I used Cochran's theorem to see that the numerator and the denominator of the student statistic were independent of each other. And this is exactly what I'm going to do here.

I'm going to actually write a test to test, maybe, if the beta j's are equal to 0. I'm going to form a numerator, which is beta hat minus beta. This is normal. And we know that beta hat has a Gaussian distribution.

I'm going to standardized by something that makes sense to me. And I'm not going to go into

details, because we're out of time. But there's the sigma hat that shows up. And then there's a gamma j, which takes into account the fact that my x's-- if I look at the distribution of beta, which is gone, I think-- yeah, beta is gone. Oh, yeah, that's where it is.

The covariance matrix depends on this matrix x transpose x. So this will show up in the variance. In particular, diagonal elements are going to play a role here. And so that's what my gammas are. The gammas is the j's diagonal element of this matrix.

So we'll resume that on Tuesday, so don't worry too much if this is going too fast. I'm not supposed to cover it, but just so you get a hint of why Cochran's theorem actually was useful. So I don't know if we actually ended up recording. I have your homework. And as usual, I will give it to you outside.