[SQUEAKING]

[RUSTLING]

[CLICKING]

ALAN EDELMAN: So welcome, everybody, to this IAP class on matrix calculus. I'm Professor Alan Edelman. And the other professor is Steven Johnson, who's going to have-- he had to go out of town. And so we're going to take advantage of Zoom technology so that he could give lectures remotely. We did some testing of the audio and the video. So hopefully it'll work out just fine.

So maybe just a quick introduction. So-- oh, good, good. Yes, Steven-- recording?

STEVEN JOHNSON: I already started it.

ALAN EDELMAN: All right. Thank you for the reminder. Yeah. So I even put the two reminders on the blackboard. Yeah, so that was reminder number one. Good.

OK. So let's see. So both of us are in the math department. I'm also in CSAIL. And I run the Julia lab over in CSAIL. And my own research involves both mathematics as well as computing software. Steven, do you want to say a few words quickly about yourself?

STEVEN JOHNSON: Hello, everyone. So some of you may remember me if you took 18.06 in the spring--

ALAN EDELMAN: How many of you took 18.06 in the spring?

STEVEN JOHNSON: --or in the fall with me.

ALAN EDELMAN: Or in the fall. So there's two, or three, or four hands. OK.

STEVEN JOHNSON: Yep. Yeah. So those people who took 18.06 with me will have gotten one hour of these 16 hours of lectures that we're going to do this IAP on. And that'll be familiar.

But yeah. So I'm also in the math department. And I'm also in physics. And I come into matrix calculus through a lot of PDE-constrained optimization mainly. But I've also worked with Julia, which will also be used on the problem sets. And so I'm sorry I can't be there with you in person. I'll be missing-- I'll be doing it remotely for the first week and a half. And then I'll be back in person.

ALAN EDELMAN: OK. So let me quickly show everybody the GitHub site that we're going to be using. Where-- did it disappear on me? Where did it go? Oh, there it is. OK. So-- oh, no. That's not the right one. This was another course.

All right. I think we should just dive right in. So let me go ahead and do that. So yes, here are the lectures, the problem sets-- three units. And we assume people are familiar with linear algebra and not much more. OK? And there's no assumption that you've already used Julia before.

But of course, if you've taken 18.06 with Professor Johnson or some other courses, you may already be somewhat familiar. But you could easily learn. The kind of Julia we're using here is sort of basic calculator Julia. And so you won't need to know much. OK?

So let me delve right into where does matrix calculus fit in. So if you look at MIT's course catalog, which I've copied over here-- this is replicated probably in universities all over the planet, right? There's single-variable calculus. This is, like, the first semester of calculus. And it's required at MIT for all undergraduates to take 18.01 and learn how to take the derivative or an integral of a function of one variable. OK?

And then there's that second semester of the sequence, 18.02, where you learn vector calculus or multivariate calculus, right? And so you learn the basic definitions of a gradient or a Jacobian. And that's kind of this whole 18.02 thing. And like I said, this is in every university around the world. OK? And I bet everybody in this room has gone through these two classes one way or another, or learned it on the streets, or something.

So it seems to me that there's actually a sequence that's sort of been cut off completely arbitrarily, right? The sequence should go scalar, vector, right? So that's one, two dimensions-- matrix. Right? Scalar is zero-dimensional; vector is one-dimensional; and matrices are two-dimensional; and then, of course, higher dimensional arrays, right? It just makes sense that these objects are-- you should be able to do calculus on any of these objects.

When you talk about programming languages, for example, some but not-- every programming language has two-dimensional structures and more, right-- arrays. So if you're familiar with Matlab, it does not have a one-dimensional array. People talk about n-by-1 column vectors and 1-by-n row vectors. But they're really just matrices, right?

Other languages have one-dimensional arrays. Some even have zero-dimensional arrays, right? So these are the things that you find in programming languages. And this is Julia. The size of a matrix, for example, has two components. The size of a vector has one component, et cetera.

So it seems to me-- I've been teaching at MIT now-- I don't even want to count the years. We're pushing on 30 years now for me. And when I started here, linear algebra was not what it is today. I mean, maybe you have-- I don't know if you folks know, but linear algebra was this thing to avoid. It was this, like, required course maybe. And a few people needed it perhaps.

But I think because of machine learning, and statistics, and lots of other reasons, linear algebra has gradually taken over a much bigger part of today's tools for lots and lots of areas compared to when I started 30 years ago. And so machine learning, statistics, engineering-- everybody needs linear algebra.

And so it stands to reason that you'd want to be able to do calculus on matrices and higher-dimensional objects, especially because everybody these days-- they're doing machine learning. I don't have to tell you because everybody in MIT is already doing it, right? Machine learning-- you need to take gradients. And you need to take gradients of complicated objects, right? So you want to do gradient descent? You need to be able to do this sort of thing.

But I've always been interested in matrix calculus. I always just thought it was-- I always liked calculus to begin with. And I thought the idea of doing calculus with matrices just seemed like a-- I always liked linear algebra. I always thought it would be sort of fun to marry calculus and linear algebra.

And I used to go to the library. And then when Google came along, you'd Google "matrix calculus." And up until fairly recently, I was rather disappointed with what I found, right? Matrix calculus was-- I found, like, three books I think with the title of "matrix calculus."

And the moment I opened the book, I realized, no, no, no, this is not what I wanted at all, right? I wanted to be able to do the kind of calculus we did in college-- 18.01, 18.02 calculus-- but on higher-dimensional objects. OK?

So here's a little bit of a quick question for all of you to think about if you don't already know the answer. You might ask yourself, for example, suppose you have the function that squares a matrix, right? What should the derivative be? In fact, if you've never done this before, you might not even know what it should look like, right?

Should it be 2x, like the scalar case? Well, I'll tell you right now that that's not correct when x is not a scalar. And then, of course, you could ask about what is the derivative of the matrix inverse, or what is the derivative of the matrix inverse squared, and so forth. Or yeah, is the derivative of x inverse the same as negative xn minus 2? And the answer is no.

So the first point I want to make is, matrix calculus is not-- it's more complicated than scalar and vector calculus. It's not a lot more complicated, but it's unfamiliar to everybody who hasn't studied it. That's what I want to say-- that you can't just say, oh, I know vector calculus, I know scalar calculus, it's probably just some simple generalization.

So my first message to you is, no, it's not. I mean, you can master it. You'll learn it during this IEP course. But it's not just a simple, obvious generalization.

OK. And so as far as applications go, this is a bit of a collage that Steven put together from a year or two ago. So you could just look at this yourself. But I think you all probably-- you're all here for some reason. So you probably already all know the buzzwords.

But "machine learning," "parameter optimization," "stochastic gradient descent," "automatic differentiation," "backpropagation," right-- so these are all the reasons for doing matrix calculus. And this little collage here-- you could just see the places where things are happening in various parts of the web. OK? So you can take a quick look yourselves.

The part that interests some of us are the applications to physical problems as well as machine learning and statistics. And so for example, in the upper left here, you have the so-called topology-optimized aircraft wing, where in the old days, when people used to engineer an airplane wing, what they would do is they would just pick a design. And then they would just somehow figure out the aerodynamics. And if they wanted to try again, they would take another design and figure out the aerodynamics.

But now, with faster computers and machine learning techniques, you could have the computer decide what is the-- you can optimize for whatever you want-- minimize fuel, or minimize metal that you use, or whatever it is that you want to optimize for. And you can let the computer figure out the shape. And so that becomes a big optimization problem. And that gets called topology optimization. It happens for an airplane wing. It happens for fluid dynamics.

So it doesn't really matter what you do. The point is that, in the old days, it was hard enough to just simulate the physics around anything, like an airplane wing. But now we can actually put that physics inside an inner loop. And we can put an optimization problem around it, right? And that's happening everywhere.

So data science and multivariate statistics, of course, is another big area, especially the computer scientists. They're doing this sort of stuff all over the place nowadays. OK. So there is-- you can take a look at this video. Or here's a book.

And let me say some roles-- let me say a few words about the role of autodiff, or automatic differentiation. So automatic differentiation is a very exciting technology. And one of the things that I like to say-- I mean, you can look at the slide. But when I first learned about automatic differentiation, I was rather surprised as to what it is and what it isn't.

So how many of you are familiar with autodiff already a little bit? OK. So about maybe five or six people have raised their hands-- a minority of the course, for sure. So everybody at MIT is good at differentiating, right? You didn't get into MIT if you can't differentiate functions.

You're all good at it. I have no doubt, right, that-- certainly if you came to this class. But everybody at MIT can differentiate everything. I don't care if I write x-squared, square root, sine, cosine-- arctan maybe you have to remind yourself. But yeah, you can differentiate everything.

And nowadays, automatic differentiation has become almost more of a compiler technology than a math course, right? It's kind of become this new thing. It's not numerical differentiation, like if you take a numerical analysis course. Or maybe you see a little bit at the end of your calculus course, where you learn to take a delta y over a delta x. You remember? You take a small difference over a small change. So it's not that.

And it's not-- like, some of you may have used Wolfram Alpha or Mathematica, where you let the computer differentiate symbolically. And it's not that either. And I think that's what makes it interesting-- that it's neither of those two things, right? It's something different. And we'll show you a little bit about how it's done.

And what you learn is that you should never-- well, we teach in 18.01 to do things-- to me, it's almost like what's happened with long division. Like, maybe-- how many of you actually learned to do long division? How many of you think you could still do it today if forced to, right? So to tell Steven-- the whole class raised their hand for the first one and the minority for the second question.

But of course, one of the good things is you kind of understand division maybe by learning long division. I'm not even sure. But calculus is, I think, becoming the same way, where we teach it the old-fashioned way. And I guess that's good because you can learn it. You understand it.

But as to what should-- we don't really think humans should do long division. And it's getting to the point where probably-- this may be heresy here in Building 2 in the math-- this is home of mathematics at MIT. But I would say that taking derivatives may or may not be as important-- certainly complicated derivatives are beyond human ability anyway, no matter how good you are at it. The things we want to differentiate these days are just too complicated.

OK. So what did I put over here? Let's see. So yeah. So today's courses are mostly symbolic. I think that's fair to say-- that 18.01 is probably a 90% symbolic course. And I'm saying, yeah, today's differentiation-- neither of these two things. The math is fun. We'll learn about it in this class. OK? So--

STEVEN JOHNSON: And it's not just fun.

ALAN EDELMAN: Wait, wait, wait. Let me put the microphone on you.

STEVEN JOHNSON: Even if you're using the computer to take derivatives, right-- so you write a program. You let it take the derivative for you. To use that effectively, you really have to have some idea of what's going on under the hood because there are-- and especially there are cases where it doesn't work well. You need to know about those.

And you need to know something about the technology and about what its capabilities are-- to know when to use forward mode or reverse mode differentiation, and what a vector Jacobian product is and why it matters-- in order to really be an effective user of these kinds of things.

ALAN EDELMAN: Yes, yes Steven's getting fancy already a little bit faster than I would have. But he's quite right that-- I mean, I would actually tell people that it's not such a bad idea to know how an engine works if you want to drive a car, but I guess you don't really have to. And these days, cars don't even have engines anymore, many of them.

But so what Steven's saying is that automatic differentiation is probably not quite as easy as driving a car-- that actually understanding the technology underneath it will really help you to use it. And sometimes it's necessary.

OK. So let me start with something you all know just to establish some notation. OK? And then I think Steven in the second part of today's lecture will kind of reiterate and talk a little bit more about the notation that we'll be using.

But I want to emphasize the concept of linearization. I just want to get that word out. And you'll see more of that pretty soon. But instead, the idea that a derivative is taking a nonlinear function and pretending at least locally that it's a linear function-- I want to emphasize that point of view. And of course, you all know that.

So looking over here at this expression, you all know that, in some sense, if you're sitting at a point-- x0, y0-- then the linear function that's tangent to the curve can be written as y minus y0 is-- well, the linear function is equal to f prime of x0 x minus x0, right? So if I put this down in the denominator, it's just saying the change in y over the change of x is the derivative, right?

So in some sense, why we take derivatives-- this is all very trivial in one-dimension. But it's good to kind of keep track of this point of view so that, when we go into higher dimensions, you kind of remember that calculus is really about pretending some complicated curved surface is just locally linear. And for one variable, that's just a tangent line. For multivariables, you have planes, hyperplanes, and so forth. But they're flat. That's what a linearization is.

So here are some notations. We'll use delta to indicate finite perturbations. And so I have this approximate symbol over here. OK? So delta y is approximately f prime of x delta x. Or you can take the infinitesimal limit. And many of you are familiar with dy over dx is f prime of x.

But how many of you have actually seen it written this way, where-- or we're told, you're not allowed to do that, where you go-- right? So everybody knows this notation. And maybe they told you that this is really not a division. Or maybe they told you it is kind of a division.

I don't know what your calculus teacher told you. But has anyone ever-- have you seen it like this? How many of you have seen it like this, where you're allowed to-- OK. So almost everybody. Good. OK. So we'll talk a little bit about what that really means.

Here's the linearization that's-- instead of just using y's, I'm using x over here. And this one is my favorite, where you write, if y is equal to f of x, right, so then-- in fact, there's no real y at all. If you just have a function of x, that df is equal to f prime of x dx.

And so we have this-- f prime is a linear function in the end. I mean, in one variable, it's just a constant, right? But a constant times the change of x is the change of f. OK?

So you all know this. It's one-dimensional calculus-- but just to set the stage. OK. And--

STEVEN JOHNSON: Hey, can I say something? So the reason we don't want to put dx on the other side is we don't want to divide by dx. One way of thinking about it is that pretty soon this-- right now, it's a scalar. We can divide by numbers. But pretty soon, x could be a vector.

Or it could be some other thing. You can't divide by a vector. So it's harder to-- it's easier to generalize this kind of notation to other kinds of objects to where you can multiply, you can operate, but you can't divide.

ALAN EDELMAN: Yep. Good. Good. Yeah. Try dividing a vector by a vector. It doesn't really mean much.

All right. So again, more trivialities just to set the stage, just to go baby steps. So here, what I'm doing is I'm looking at the square function. OK? And I'm looking at it at a particular point-- the point 3, 9. And you all know that the derivative is 2x. And so if x equals 3, 2x is equal to 6, right? So that's the derivative.

The square of 3, of course, is 9. And even if you do this on a computer, this is easy to do. You don't even need a computer, really. It's actually easy to check this even on paper and pencil. But if you get closer and closer to 3, as I'm doing in these four lines over here, of course you get closer and closer to 9. And what we're interested in is the difference.

So you see that if I add a little delta x to 3, I get a 9 plus delta y, which is 9 plus 6 delta x plus some higher-order term, which we don't care about, right? And so we see this as delta y is f prime of x0 delta x, right? delta y is 6 delta x.

So the thing I want you to walk away with is really that I want you to look at these numbers, like this one here-- I think of this as the delta x and think of this red part here as the sixth delta x, right? So again, this idea that Steven also just said-- that this is a linear functional. This is the function that multiplies by 6.

If I want to know the change in y, I have that change in x. And I multiply by 6. That's my-- you see, eventually this is going to be not just multiplied by 6. But it's going to be, you have a vector change, and you're going to multiply by a matrix. OK? But just going slowly, we make a little change. And you multiply by 6. And that's what's going on here.

So I like to think of dx and dy as really small numbers on a computer. The mathematicians call them infinitesimals. From-- there's a lot of rigorous math that makes infinitesimals work, which I think is sort of a great human achievement but of little value to practical computations.

I think for practical computations, to think of-- you could think of it any way you like, but I like to think of dx and dy-- when I'm actually working and I'm not thinking theoretically, I like to think of dx and dy as, like, the limit of very small numbers. And when I play on the computer, I type 0.0001 or 0.00001. And I say, that's small enough. It depends on context, of course. But that's usually what I do. OK?

So all right. Now we get to go to where the big boys and big girls go, right? So we're leaving the world of scalar calculus and entering this new world of matrix calculus. So to get started, let me just mention a little bit of notation. So it's handy to have the element-wise-- so yeah, just a little bit of vector and matrix notation.

So I like to use the Julia notation for element-wise product of vectors. So this dot times this point-- I like to call it point-wise times, right? So 2, 3 point-wise times 10, 11 is 20, 33.

If you happen to use languages-- I think Python does it, too-- and certainly Julia. They pronounce it as "broadcasting." I always hated that word because I don't feel it really indicates what's going on. I think point-wise multiply-- to me, a dot is like a decimal point. It says it better. But if you're used to the term "broadcasting," that's fine as well.

So you see we're doing element-wise multiplication. And in this little demo here, you'll also see some people use this dot with a circle around it to indicate point-wise multiply. I remind you, in case linear algebra was too long ago, the notion of a trace of a matrix is, if you have a big square matrix, then you look at the diagonal, and you add up all the numbers. And that's the trace.

And if you go to this matrix.calculus.org, there was-- oh, yes. There's a story here. Let me step back and tell you the story. So the reason why we started-- one of the reasons we started teaching this class during IEP was because there was a question on the math Piazza-- all the undergraduates have access to a Piazza page, not for an individual class, but for the majors. And there was this question that arose, which is basically this question. If y is an n-by-m matrix-- here, I'll use the mouse so it'll get recorded properly.

So if y is an n-by-m matrix, x is an n-by-k matrix, and theta is also a matrix-- a k-by-m matrix-- then this scalar, the trace, makes sense, right? The trace of this multiply makes sense. And one could try to understand what does it mean to take the derivative. OK? And this student on Piazza asked, how do you do it? And how do you learn how to do things like this?

And that really is the origin of this IEP class-- that, here's the answer. The answer is itself a matrix. It's minus 2x transpose times y minus x theta. And you can actually get the answer through this matrixcalculus.org, right?

So those of you who want to, you could look at it. It's kind of a nice web page. You can type in some matrices. It's got its limitations, but you can look at some of these and see what's going on. OK.

But let me go back. How do I get back to my slides? I'll never find them again. Where are my slides? They're underneath here, aren't they?

STEVEN JOHNSON: Ctrl-left arrow or right arrow?

ALAN EDELMAN: No, I think they're underneath. Yeah, they're underneath. Oh, you think I could have gotten it that way?

STEVEN JOHNSON: Full screen mode, I think, yeah.

ALAN EDELMAN: Oh, OK. Let's see. Can I really get to that? Ctrl-- never mind. I don't know how I can get back. Anyway, yeah. So you might play with this a little bit and see some of the things you can do. It has limitations. For example, it doesn't do a good job when the answer is higher than two-dimensional, right? But you can take a look and see what it will do.

One of the things-- you might see some matrix calculus in some classes. For example-- I think this is on the next slide, but I'll just put it on the blackboard. For example, you might ask for, how do you take the gradient of x transpose x, and things like that-- or x transpose ax. I think it's coming up on the next slide, anyway.

So how do you take the gradient with respect to x of objects like these? And I've seen many classes at MIT do that. And what I would say is they would do it the old-fashioned way. They do it variable by variable by variable as opposed to the holistic way.

So one of the things that I'd like to kind of invite all of you to think about as we go through this course is to think of a matrix holistically or think of a vector holistically. Stop thinking of a vector as a bunch of elements or a matrix as an m-by-n table, as you would see in an early linear algebra class.

Think of a matrix as sort of having-- like, we're more than our hands, and our feet, and our noses, and our mouths, right? We're people, right? I want you to think of matrices that way-- as a holistic object. And there are ways of doing matrix calculus without going down to the element level.

To kind of get that point across a little bit more, I still see this today. Though, it happened more and more. I, as a professor, would walk into a classroom to start to give a lecture. And in the old days, professors used chalk and blackboards. I guess they still do that sometimes.

And I would walk in. And I would see what's on the blackboard from the previous class. And as I'm erasing it, I would actually look and see what would be written. And I would see people taking-- essentially, I would stand back. And I would look at the entire blackboard.

And I realized that, this is just a couple of matrix multiplies written out element-wise. But it fills up the whole blackboard because, for whatever reason, the professor didn't use matrix notation. They just used scalar notation, with indices all over the place.

And one of the things you'll see is how to kind of grow up from that point of view to actually-- sometimes it's useful to work with indices. And sometimes it's comforting. You feel like you're getting the right answer. But in some sense, it's more elegant when you don't have to do that. And we'll show you how to do that as well.

OK. So let's take a look at sort of the various types of cases of what it even means to take a derivative, right? And so 18.01 is very much in the upper-left corner, right? So you have a scalar in, and a scalar comes out, right? That's everybody's first semester of calculus. OK?

If you go along the first row over here, for example, probably at least in physics-- but very simple physics-- you learn the idea that you could have, say, your position in three dimensions as a function of time, right? So your input is time. It's a scalar. And your output is a position in space, like a-- for those of you who can't see what I'm doing online, I'm moving my hand to indicate a trajectory in space. OK?

And of course, the derivative is now the velocity vector, right? That's what the derivative is of the function of time. It's tangent to the curve. And its magnitude tells you how fast you're going, right? And that's the derivative of a vector with a scalar.

Now, this is not the sort of thing you would see much in previous classes, I wouldn't think. But it's perfectly reasonable to also talk about a trajectory in matrix space, right? And so you could have an m-by-n matrix that's a function of time. I could say it as every element as a function of time.

And of course, you can take the derivative of that thing, right, which will be a fixed matrix, which, if you could imagine an m-by-n matrix space-- mathematicians love to imagine high-dimensional matrix spaces. high-dimensional spaces of all kinds. So if you can imagine an m-by-n matrix space, then the derivative would be a tangent in that big high-dimensional space.

OK. So that's the first row. Maybe it's a good time to now go down the first column. So this blue-- it looks-- yeah, I guess it looks more purple on the screen, but it's blue-- is sort of the-- this is what you do in multivariate calculus-- 18.02-- and machine learning everywhere, which is, we take gradients.

So it's very typical where you have a function with many variables going in. I'll just talk about vectors. But in fact, you could have lots more variables than just a vector. But let's just say, in machine learning, you have a vector going in and a scalar going out. The scalar is often called the loss function in machine learning.

And how many of you have heard the word "loss function"? Everybody. OK. I think students are learning it in kindergarten these days. So yeah, everybody-- right. So you're all familiar with many, many variables coming in. That's the vector, the input, and the loss function, the scalar coming out. And you want to take a gradient of that thing, right?

And so if you actually literally have a vector going in, then the gradient is-- in fact, for every scalar function, the gradient always has the same shape as the inputs. So if your input is a vector, then the gradient is a vector, right?

So that's often denoted with this nabla notation. In LaTeX, it's backslash nabla-- so usually pronounced grad f, or gradient of f. And in Julia, it would be a vector. We don't have to talk about column vectors and row vectors. It's simple to just say "vectors." It gets too confused when you have column vectors, but people are used to that notation. It's just a vector. But you could call it a column vector.

But the derivative is the transpose. It's a row vector, right? And Steven and I-- we actually went back and forth on this a couple of years ago. And we were absolutely convinced that this is the best way to do it-- that the gradient is a vector, or a column vector, and the f prime is a row vector.

So in particular, I'll just give you the answer for x transpose x. So here's f of x, right? And we're talking about x being a vector, right? And so the gradient of f is equal to 2x. OK? But df will be 2x transpose dx. Or f prime of x will be 2x transpose.

And I hope you can start to appreciate why this is a good idea. Because we want this to be a linear operator-- that we put in a little vector change. And we want the output to be a scalar, right?

So it makes no sense to go-- you see, it makes no sense to go to 2x dx, where everything-- this is a little vector change. And this is a little vector. There's no such thing as multiplying vector by vector.

But if we write this-- 2x transpose-- then when we multiply by dx, as we're doing here, out comes a scalar, right? So I make a little small change to x-- infinitesimal or a tiny little change, however you want to think of it. And out comes a scalar change to f. And so this actually makes perfect sense.

And I don't think you would learn it that way in 18.02. And so this is-- but this seems to lead to consistent answers. And so we're going to encourage you to take that point of view.

All right. So I've covered four of these boxes. Let me point out that there's a color-coding that you might have noticed to these boxes, if you haven't noticed it already. The green is when the derivative is zero-dimensional. It goes along the diagonals, right? The blue is one-dimensional answers, right-- the vector and the gradient, those two boxes.

And then the next diagonal-- it's a funny diagonal, right? It's going from northeast to southwest. But this funny diagonal-- the next step is when the derivative itself is a matrix, right? And so one example was in the top right, when you just have a trajectory in matrix space.

Another one is in the bottom left-- is, if your parameters are coming into a machine learning algorithm as an array and you have a loss function, then the gradient-- as I said, the gradient of a scalar function always matches the shape of the input, right? And so the gradient of a matrix is a matrix. OK?

And then in the middle of these 3-by-3 arrays, I think you would have all seen in 18.02-- I don't know if you remember it anymore-- most of you probably do. Some of you may have forgotten.

STEVEN JOHNSON: 18.02 doesn't always cover it, unfortunately.

ALAN EDELMAN: Doesn't cover Jacobians?

STEVEN JOHNSON: And certainly not as linearization. They sometimes do Jacobian matrices determinants for integrals. But they don't even always cover that, apparently.

ALAN EDELMAN: But you mean students in 18.02 will not see a Jacobian matrix in some form or another?

STEVEN JOHNSON: Apparently.

ALAN EDELMAN: What has MIT come to? How many of you have actually taken 18.02 at MIT? OK. And among those, how many of you think you once learned about a Jacobian matrix? You don't have to have remembered it. OK. Well, we've got a good set of students because apparently they learned Jacobian matrices. I would have thought that that would just go without saying, but--

STEVEN JOHNSON: But it's very-- often, it's only for multi-dimensional integration. So it's only determinants of Jacobians that they ever see. So the Jacobian and its determinant-- they kind of get mixed together.

ALAN EDELMAN: All right. That sounds unfortunate. [CHUCKLES] But OK. Thanks for the update on 18.02. A few years back, I actually looked at the 18.02 ASE notes. And I know that it was there. But I don't know what changes have been made.

OK. But in any event, just to kind of set the stage, this is where you have a vector input and a vector output, right? So you have a function in, say, three dimensions. And the output is also a function, say, in three-- or it could be lower or higher dimensions, right?

And if you have n dimensions going in and m-- as in "Mary"-- dimensions going out, then the derivative is most naturally done if you do it as a matrix, not as a linear operator. But if you do it as a matrix, it will be an m-by-n matrix. OK?

And then, of course, we get into the interesting situations as you move down this table-- the situations where that web page doesn't do a very good job at it. But we're going to show you how to do this-- how to think about just even what's a good notation when you start moving into higher-order arrays, right, where the derivative is no longer best expressed with two-dimensional matrices anymore, right? And so that's down on the bottom right. OK?

So here are some answers to set the stage. And again, we're going to show you how to do this. But I think it's kind of nice to foreshadow a little bit of what you're going to see.

So here, I'm going to use this sort of operator notation, starting with one you're familiar with. The derivative of x-cubed is 3x-squared dx, right? So everybody kind of knows that.

And just to encourage you, just to remember what I've said before-- that I want you to think of this as, if I make a little small change to x, my change to my function will be 3x squared times that small change, right? If I'm at x equals 7, right, and I make a 0.01 change, then the result is going to be 3 7 squared times 0.01 approximately, right? Right? And of course, that gets better as 0.01 gets closer and closer to 0. OK?

So here's one that-- in fact, let's use the mouse again just to emphasize. But this one here-- whoops. I didn't want to do that. This one here is-- let's see. We have a-- I want this to be a vector input and a scalar output. So it's this box again.

And it's the same thing I wrote over here. I'm just using prime instead of transpose. But again, I want to encourage you-- that if x is a vector and dx is a small change to that vector, then you take the dot product, right?

So I hope everybody realizes that, if you ever go x transpose y, that's the same thing as a dot product, right? That means multiply all the elements and then add everything up, right? So this is a dot product between 2x and-- this is, I'm at the point x. And my change is a little dx. This dot product will be the change to x transpose x. OK?

And here's your first matrix answer. I told you that, if you square a matrix, the answer is not, like, 2x, right? What is it? Well, in fact, because matrices don't commute-- the x and the dx don't commute-- in a way, you could think about it this way. Should we go x dx times 2 or dx times x times 2? Well, in fact, there's no reason to prefer 1 over the other. And in fact, the correct answer is to add the multiplications both ways.

So if you make a small change to a matrix and you ask, what is the change to the square, then this is the correct answer. This is-- and you see now maybe for the first time why we need to think of it as a linear operator. We can write this out as a big matrix. But it's not a good idea, right? It might be comfortable, but we don't recommend it.

So you see, you could think of x as n-squared variables. If x is an n-by-n matrix, right-- here's x. That's n-by-n. We have n-squared variables. And so we could think of the function that squares our matrix. If you want, you could, like, flatten everything and think of it as going from n-squared variables to n-squared variables.

And then the derivative would have n to the fourth things in it. You could write it as an n-squared by n-squared matrix, but we're not recommending that. We're saying, just think of it as this linear operator. OK? And that's going to be the answer for the derivative of x-squared, OK, for a matrix square. Obviously, it reduces to 2x dx when it scalars, but it's much better than that. OK?

So sometimes I open up a Julia to do this, but I think these slides kind of say it all. So just to kind of hit home on both the matrix square and the x transpose x, I'll just give you a numerical example just to make sure this is completely clear. So let's consider the function x transpose x, right? So I hope you all realize that's just the sum of the squares of the entries of x, right?

And so if I'm at the point 3, 4, right-- so I'm at a point in space, a two-dimensional space-- 3, 4, right? So my function value is 9 plus 16 is 25, right?

And let's make a little change. Let's say my dx is-- I'm going to go 0.001 in my first direction and 0.002 in the next direction. Well, you could do the math. And basically, the answer is about 25.022. OK?

And so here, we're explicitly making those changes, right? But the whole point of calculus is to have, like, a formula for doing that, right? The reason why we don't just do the calculation that's on this line is because the magic of calculus is there's an exact way to do it.

And this is that exact way. If you go 2x0 transpose dx, you see we're going to get that right answer. OK? And that's what we love about calculus. OK? And so the delta f is going to be that. Any questions about that so far? OK.

And I could-- I don't know. Well, let me just see. Do you want to see me do this with a matrix? I don't think I put it in the slides. Should I do a matrix quickly in Julia? We could do it. Do you want-- oh, people are saying, yes, I should do it.

All right. So let's open up a Julia quickly. I think this is the sort of thing that's just easy enough I could actually just do it in the repl. Should I do it with VS Code or just do it in the repl? I'll just do it in the repl.

OK. So let's get a Julia. This is a bit impromptu, but why not? OK. So here, let's just grab a Julia. OK. And what I'd like to do is demonstrate-- so the plan here is to demonstrate that dx-- how do I do this? Like that. Dx-squared is going to equal dx times x plus x dx. OK. So that's going to be my goal.

Any particular size you want to see? 3-by-3 good enough? I guess so. Let's just do 3-by-3. Yeah, we'll make this a little bit bigger. So I don't even think I need linear algebra. Let's see. I don't even think I need any packages.

So should I take-- all right. Let me hear your nine favorite integers-- small. Don't make them too big. Come on. Just shout out nine integers.

AUDIENCE: 5.

ALAN EDELMAN: 5.

AUDIENCE: 7.

ALAN EDELMAN: 7.

AUDIENCE: Negative 3.

ALAN EDELMAN: Negative 3.

AUDIENCE: 17.

AUDIENCE: 1, 2, 3.

ALAN EDELMAN: 1, 2, 3. Thank you. OK. Hurrying along here. Three more?

AUDIENCE: 6, 7, 8.

AUDIENCE: 0.

ALAN EDELMAN: 6, 7, 8. All right. Just to show you that there's nothing up my sleeve. OK. And let's go, y is equal to x-squared maybe. So here's the matrix square. OK? So you could check that yourselves.

And let's take dx to be-- oh, let's just go 0.001 times rand of 3,3. Or is that too ugly? Oh, it's fine. Let's just do it. OK? So here's a bunch of ugly numbers. OK?

So everybody understands that I'm sitting at this matrix X. And I'm moving over to X plus dx, right? And I want to know the difference. So let's do that, right?

So dy-- we'll do it first numerically-- is going to be X plus dx-squared minus X-squared. OK? So there's the dy. OK? And now let's compare that with this magic matrix calculus formula. The matrix calculus formula is X times dx plus dx times X.

And this is where I always have that fear that I made a mistake or a typo. But let's hit Enter and see what happens. And look at that. You see? So this is the numerical way of calculating a derivative. Just see where I am over there minus where I started. Or there's this magic formula.

And just to show you that these other ways you might think of doing it won't work, like 2x dx-- that's not right. Or 2 times dx times X-- I don't need that times. That's not right. But the one that is right-- here, let's put it back up-- is clearly that one. OK?

OK. So I haven't showed you how to derive it yet, but that's coming soon. But I just wanted you to sort of-- really, for me-- I don't know if your minds work this way. But for me, seeing this is pretty convincing, right? I feel like I know what's going on, right? I really like looking at it this way. OK?

So after all, if I were doing calculus on a computer for scalars, I would just take X plus dx-squared for scalars. And this is what I would do. And I would get 2X. And in fact, just to remind you that this really is the same thing, we could take X to be 3 and dx to be 0.001. OK? And then we can go like this.

And I can compare that with what everybody knows from ordinary calculus, 2X times dx. Right? And to the right number of digits, they match. OK? Any questions about this? All right. So hopefully you now kind of believe at least s not theoretically, that matrix calculus kind of works.

All right. So getting back to the slides here-- so it's helpful to know certain rules that you all learned in freshman calculus extend to matrices. So you might remember that you learned the product rule, right? So what's the product rule say? That-- people learn it with different notations. Many people learn it as duv equals udv plus vdu, right? Other people learn the product rule with other notations.

But the thing that I want you to know is that the product rule just works for matrices and vectors as well as for scalars. So any time you have a compatible vector product or-- like a vector dot product or a matrix times vector, where the sizes all work out, then it's perfectly OK to take the matrix rule dab to be dab plus adb. OK?

So this would-- the only thing that you have to remember is matrices don't commute. So if a is on the left and b is on the right, a must always-- so in all the formats, a must be on the left. And b must always be on the right. Like, if-- right?

So for example, this was a little bit giving you the wrong idea. I mean, it might have been better to say udv plus duv, you see, because this will work for matrices, right? I must always keep the u on the left and the v on the right. And that's the only rule that makes matrices just a little bit different from scalars. OK?

So let's apply that to x transpose x, right? And so if we apply it to x transpose x, it says that d x transpose x will be dx transpose x plus x transpose dx. OK? And here's where students get confused. I'm going to go ahead, and combine them, and violate every rule that I just said. I'm going to say that I can combine them and get 2x transpose dx.

Wait a minute. What are-- how is it that I can combine this here when I just told you that you can't put things in other orders? Anybody want to tell me why, in this case, I can get away with it? I just said to you the a must be on the left and the b must be on the right.

And here I am. I'm telling you that dx transpose x and x transpose dx are actually the same thing. How come I'm allowed to do that in this one case and combine these two? Can anybody tell me? Yes, please, in the back.

AUDIENCE: A dot product is a scalar.

ALAN EDELMAN: Yeah, the dot product is just a scalar. And it doesn't matter what order you do it in. Exactly right. Just to kind of show you in Julia-- oh. Oh, you know what? [CHUCKLES] There's a tiny little hole where the wire comes out. And if the chair leg falls into that hole, I go down with it. [LAUGHS]

All right. So here we go. So let's say X is "1 2 3." Oh, well, actually, let's put it with commas. OK? And if dx is 0.001 times rand of three-- see, I think if I did it this way, you'd all believe it.

If I go-- can I do dot in Base? Let's see. Maybe, maybe not. "not defined." OK. But anyway, I don't even want to do it that way.

Let's go x transpose dx, OK, and dx transpose x. You see they're exactly the same thing. After all, the dot product of the-- right? So my point is the dot product of x and dx is, of course, the same as dx and x. OK? So just to remind you, a times b is not b times a. But x transpose y equals y transpose x if x and y are vectors.

Understood? You understand the difference between the general case, where things don't commute, and the dot product, where things do commute? OK? So students seem to get confused by that sometimes. And so I just wanted to say that as clearly as I know how. OK?

So one more time, using the product rule on dx transpose x-- you can use this approach to write it as-- you use the product rule. OK? And then you recognize that those two dot products are the same. And so you can combine them into 2X transpose dx. OK?

And so in a way-- now, of course, you could do it with-- to take the gradient of x transpose x is not hard to do the old-fashioned way. And there's no-- you could decide for yourselves. But on the blackboard, let me just quickly remind you that-- here's the old-fashioned way to do it. The old fashioned way is to say f of x equals x transpose x is the sum of the xi-squared.

And so this is like the baby way to do it, but this is the sort of thing that you'll see in a lot of classes, right? And you'll learn that the gradient of f is the vector that has df dx1 all the way down to df dxn. And so you'll look f of xi. And you'll say, oh, df of dxi is equal to 2xi. And so you'll put them all in a vector. And you will say, ha, the gradient of f is equal to 2x, right?

And there's nothing wrong with that. I mean, you get good at this sort of thing. But I hope you could sort of see the advantage of doing it holistically. I don't have to use indices. I don't have to think about what this thing is made of. I just do this little trick here. And boom, I get the answer right away, right?

It's kind of like a magic trick for vectors and matrices, that-- like, if you take courses where people are taking gradients, all the other students in the class are going to do it this way. And you're going to know the secret trick of being able to get the gradient this way, right? So you'll have super powers that they don't have. OK. So of course, both ways work. But OK.

OK. So here, let's see. We have du transpose v is-- this is sort of the point that I was making already. So I think we have that. OK? Any questions? All right.

OK. So let's-- just, again, a little warm up. And so sometimes it's good to count how many parameters are needed to express an answer. "Parameters" is sort of a vague term. But think about it as, how many numbers do I need if I'm going to put something in a matrix-- how many elements.

So let me ask you quickly, if I have a function that takes n inputs to m outputs and I do want to express the derivative as a matrix, how many numbers do I need? So n inputs, m outputs-- how many numbers do I need to express it as a matrix?

AUDIENCE: m times n.

ALAN EDELMAN: m times n. All right. Everybody knows. OK. And I think this is my last slide. So I'll just say a few words about second derivatives quickly. And then maybe we take, like, a five-minute break or something. And then Steven will speak.

So I don't know how much we're going to talk about second derivatives in this class. But there's one example that comes up so often that it's probably just worth mentioning. And in a way, it-- I think when I first learned about differentiating vectors, I always got confused between the Hessian, which is a matrix-- it's a second derivative matrix-- and the Jacobian, which is also a matrix, but it's a first derivative matrix, right?

So let me be clear. If you have a function from rm to rn, you can express the answer as an m-by-n matrix. And that's what we call the Jacobian, right? So the big point I'm just saying is it's a function from vectors to vectors.

The other case that comes up very often is if you have a function from vectors to scalars. OK? And so examples like x transpose ax would be vector transpose matrix times vector as a function of the vector. OK?

And so now the-- so if it's going from vector to scalar, remember that the gradient is a vector. It's one-dimensional. But the second derivative is often expressed as a matrix. And that's what we call the Hessian.

So you might ask yourself, if we're trying to get away from element-wise matrix representations, what's sort of the abstraction that we will need to talk about second derivatives? And the answer in advanced linear algebra classes is called a quadratic form. OK?

And we'll see what that's-- So we can get past the clunkiness of matrices, like 18.06, and get into the more abstraction. And those are the quadratic forms.

OK. So I think that's the end of what I'm going to say today. Last chance for questions. Otherwise, what should we take? Should we take a five-minute break, Steven? What do you think-- or a little less?

STEVEN JOHNSON: Sure, five minutes is good.

ALAN EDELMAN: Five minutes is good? All right. So I've got 12:03 on the clock back here. So let's just take a five-minute break-- until 12:08. And people can get some water, or ask me some questions, or read their email, or whatever you do in five minutes. All right? And then, Steven, you can grab the screen and set yourself up.