JAMES SWAN: OK, shall we begin? So on the syllabus, today's class is supposed to be a review for the next quiz. And I'm happy to answer any questions you have about the material or about the quiz to the best of my ability.

And when we've exhausted those questions, I have some notes on another method for solving partial differential equations. We didn't really cover it in class, but I think it's quite interesting and applicable for chemical engineers in particular because of the classes of problems that we study. And then notes on how to construct a simulation of a PDE that follow from what Professor Green discussed during your last PDE lecture. And we'll do an example where we actually develop a code to solve the PDE and maybe refine it so that it runs more quickly by noticing certain things about the structure of the problem. OK?

I have your gift cards. I remembered them. So I'll hand those out at the end the class, just in case more people trickle in. But it doesn't look like there's 53 people here, so maybe the TAs are going to be eating burritos for a while. There's going to be a lot left over. We'll see. I'll hold onto them. Try to make sure everyone gets their reward for responding to the class survey.

So let's start with questions about the quiz, about the material that we've covered up to this point. I'm really going to leave it to you to point out things that you feel like you need help with, you think need to be reviewed. I know you've had a review session with the TAs already. Maybe that's been sufficient. OK? Questions. It's up to you. Yeah.

AUDIENCE: Will the test contain chemical engineering material like last year's test, or will it be like Professor Green's test.

JAMES SWAN: Oh, good question. So I'll tell you that I believe the problems on the quiz this year are motivated by chemical engineering problems, OK? The reason for that is, unlike linear algebra problems or optimization problems, those can be crafted out of whole cloth. But workable, right, boundary value problems, workable initial value problems really need to stem from something physical.

You are not asked to derive anything from your extensive chemical engineering knowledge. This is not a qualifier for reactors or thermo or transport. We provide you with those details. OK? But they are-- I think they are of a chemical engineering origin. So you have equations that are reasonable to work with. OK? More questions. Yes.

AUDIENCE: I would love to briefly go over some boundary value problems, specifically like the collocation and--

JAMES SWAN: OK.

AUDIENCE: --Galerkin methods, maybe.

JAMES SWAN: Sure. So OK. So let's just discuss boundary value problems first. Turns out PDEs are a class of boundary value problem, too, largely, at least the ones that chemical engineers are interested in. So if we understand boundary value problems in general, we should understand how to solve partial differential equations too. Many classes of partial differential equations that we're interested in can also be solved in these ways.

So we talked about relaxation methods in general, which were these methods that rely on somehow discretizing the continuous function that we're trying to describe in this boundary value problem. So usually, you've got some-- linear function, some linear operator that operates on some variable that we're trying to understand in a boundary value problem.

Maybe it's temperature, right, equals 0. So it's going to be something like thermal diffusion. This equation could represent that. Could represent some advection diffusion problem in some bounded domain. Could be a PDE. It could just be an ordinary differential equation, right, but a boundary value problem.

And with this class of relaxation methods, we try not to exactly satisfy this equation. We relax that assumption. We try to somehow discretize this function. It could be temperature, could be energy, could be velocity or momentum. It could be lots of different things. Try to relax the constraint that this is satisfied everywhere. And we try to satisfy in some approximate sense instead.

So we discussed collocation methods, where we take the domain we're interested in and we divide it up into a series of points. And we try to satisfy this equation at those points. And we have some approximation for the function, its representation at those points.

That approximation could be a finite difference approximation. That approximation could be a basis set expansion. It could be a finite volume approximation. We could expand in terms of local basis functions. That's the finite element method. We covered all these different methods. That involves chopping the domain up into a series of points and trying to satisfy the function at each of those points.

There was an alternative method, this sort of Galerkin method, where instead of trying to satisfy the function at each of those points, we have some basis set expansion. So we have some approximate representation of our function in terms of these basis sets-- in terms of this basis set. And instead, we try to make it be the case that the integral of our operator applied to our approximation-- I'm going to break chalk-- multiplied by each of the basis functions and then integrated over this entire domain is equal to 0.

So we try to satisfy this equation with respect to each of these basis functions, essentially, right? There's some-- this isn't going to exactly satisfy the equation. This is an approximation. It's not going to exactly satisfy the boundary value problem we were after. But the projection of the residual, or how far off we are with this operator, on each of the basis functions, we're going to try to force to be 0.

So rather than trying to get a pointwise solution of the problem, we're going to try to sort of smooth out the error with respect to the basis functions and make this equation is satisfied as it possibly can with respect to these basis functions. And that's the Galerkin view. So it's different. It's like you're trying to minimize sort of a global error with respect to the basis functions, as opposed to a pointwise error, OK?

And again, these basis functions can be chosen in a lot of different ways. They can be chosen as sort of global functions. They can be things like sines and cosines. We may know that the solution to this equation looks a lot like sines and cosines, so that makes sense. Or we may choose local basis functions instead, little tent functions in 1D. They're actually tent functions in many dimensions as well. You can imagine sort of extending that to look like a tent in many dimensions. That's the finite element method.

So we can choose those basis functions however we want. And the goal is to just minimize this residual with respect to each of the basis functions that we include in our approximate solution. And we would call that a reasonable approximation.

Presumably, as the number of basis functions goes to infinity, this is going to converge. That's the applied mathematician's job, to show us that it's going to converge to the true solution of the equations. OK? So those are the methods that we-- those are the methods that we apply.

Good choices of basis function are important. So there was this discussion about quantum chemistry, where you're trying to solve the Schrodinger equation for n electrons, there's 3n degrees of freedom. So a very high-dimensional space. How are you going to do that? You do it with basis set expansions. And the choice of those basis functions is really important.

So if you read a quantum chemistry paper, you'll see they give a long series of digits and numbers that are meant to indicate which basis set expansion they used. And for certain classes of problem, some are better than others. And they make the problem actually solvable when it's not, if you were to just choose your basis functions without care for the physics of the problem. Sam.

AUDIENCE: The last homework, in the Galerkin method, where you multiplied by the complex, how did you get one basis set, basis functions? Where does that fit into this?

JAMES SWAN: Well, it should have been right here. So technically, the basis functions don't have to be real-valued functions. They can be complex-valued functions. And then this is, in some sense, a dot product, an inner product in the space of these basic functions. And the proper form for an inner product when we have complex-valued things is to take the real part of one and multiply it with the complex part of another. So that should have been right here. This should be the complex conjugate. OK? Yeah.

AUDIENCE: So when we do the Galerkin method, we necessarily have to approximate these derivatives and things using basis functions.

JAMES SWAN: Yes.

AUDIENCE: But in the collocation method, there was lots of other ways that we could do it, like find differences. I know you mentioned a few of them earlier.

JAMES SWAN: Yes. Yes.

AUDIENCE: OK.

JAMES SWAN: That's right.

AUDIENCE: That [? LOT term ?] kind of contains also the basis function in the Galerkin method, or in the collocation method.

JAMES SWAN: Well, in the-- if we have an expansion of the sort of objective that we're trying to calculate in terms of those basis functions, L is the real L, right? It is what it is. This is the model equation to be solved. We operate on our approximate solution with the real L.

In the collocation method, it may be the real L, or it may be some approximation for L. If we do finite differences, we're approximating all these derivatives. If we do a basis set expansion and we try to satisfy that at a certain number of points, we're not approximating L. We're approximating T instead. So they're a little bit different. More questions?

So that's BVPs. This picture applies for PDEs, too. So this is the integral over the domain of interest, the domain that's bounded by our boundary conditions. These basis functions don't have to be in one dimension. They can be in any number of dimensions we want. We just want them to be good basis functions in principle.

So for finite elements, you use little tents. They're multi-dimensional tents, but they're still tents. The same as these hat functions that Professor Green talked about in 1D. Just do them in many dimensions. They work exactly the same way. They're easy to take the derivatives of. These intervals are, for the most part, simple to calculate. And that's why COMSOL, for example, uses that methodology. It's based on this sort of Galerkin picture.

So PDEs, BVPs look very similar. Usually, when we're talking about spatial dimensions, they look very similar. When we go to the time domain, things get a little more confusing. That mixes together what we know from ODE IVPs with what we know from boundary value problems now.

So you talked about the method of lines last time as one way of studying, for example, parabolic partial differential equations. You have the spatial discretization like this. And then in time, you have some ODE IVP technique that you integrate forward. You rely on these robust ODE methods that know how to control the numerical error, can detect, maybe, when there's some sort of an instability and try to damp it out by altering the time step so that things stay stable. Yes?

AUDIENCE: I can't visualize the local basis. Can you give me an [INAUDIBLE] expression for the local basis?

JAMES SWAN: Sure. So first of all, there are a lot of local basis functions we could write down, OK?

AUDIENCE: This is just one of them.

JAMES SWAN: There are many of them. So yes, let's do the hat. OK? Let's suppose I have this line in x, and I target in my collocation method a series of points, column x i minus 1, xi, x i plus 1, and they extend well beyond here. It doesn't matter where they are. I choose where they are based on the physics of the problem. I'll put them where I want them to go.

And in fact, you might even try to optimize where they go. So you might start with some distribution of them, and then move them around in some way to improve the quality of your solution, right? Maybe you distribute them uniformly at first, and then you notice, oh, my solution is changing very quickly over here. I would like more points in this region.

In MATLAB, there's a function called bvp4c, which is a boundary value problem solver. It solves 1D boundary value problems. And it does precisely this. It uses finite differences and collocation. And the good thing about it is it's trying to adaptively change where these points are positioned. So it solves it, it looks where things are changing quickly, and tries to shift its points around and resolve it in order to improve the quality of the solution, hit certain targets for the error.

So what are these-- what would these local basis functions look like? So we talked about these hats. They're 1 here. They're going to be 0 on either side of this point. We could call this our hat i, which is centered at point xi. And it's going to be a piecewise sort of function. It's going to be an increasing linear function between x i minus 1 and xi.

So for x between xi and x i minus 1, it'll have one form. Then it's going to switch. It's going to be a decreasing function on the other side. So between xi and x i plus 1, it'll have a different form.

So here, it's going to increase between 0 and 1. So it's going to have to look something like x minus x i minus 1 divided by xi minus x i minus 1. So it goes from 0 to 1. And then on the other side, it's going to have to have the opposite sign. So this is going to have to look like-- let's see, x minus x i plus 1 divided by xi minus x i plus 1. Will that do it? I think so. OK?

You want me to make it look like this one. So we could have plus 1 xi, and then put a minus sign in front of it if you want. You decide how you want to write it. But it describes this line here

and this line here.

But there are other choices. So it doesn't have to be piecewise linear necessarily. A lot of times, people choose different types of splines, different types of interpolating functions. They might not be localized to one point. They might be localized over a few points. Maybe something about the physics of a particular problem that tell you, you can't get away with these piecewise linear functions. They're not good enough to describe, say, higher-order derivatives in your equation.

So you might instead interpolate with, I don't know, like a quadratic basis function that spans a few more points. That'd be fine. It would be easy to do the integration in your Galerkin method. You just do these integrals out. They're complicated, but you do them once for a generic point xi, and you know their value for every point in the domain. So you choose them. You choose the basis set. But that'll give me an equation for the tent function centered at xi. That's the tent function centered at any point in the domain, in principle.

In many dimensions, right-- so if I have two dimensions, you see in COMSOL, you have a sort of a triangular mesh. And what does my tent function do? It sort of stands up, stands up above the triangle. It's a real tent now. It's coming out of the board. Can you see it? Put on your 3D glasses. You'll see it.

But it's coming out of the board. It has height 1 above this triangular domain. It sits at the center of the triangle, for example. It's one way to draw these elements. There are actually many ways to draw these elements. This is just one way of drawing these elements. OK?

Good. Other questions?

AUDIENCE: Could you please explain why central differencing is second-order accurate and not--

JAMES SWAN: Oh, good question. Good question. So why is central differencing second-order accurate? So the central difference formula says that the derivative of a function x could be written as-- let's see, f at x i plus 1 minus f at x i minus 1, divided by 2 delta x. Or let's say these aren't equally spaced points. Well, let's make these equally spaced points for now. Let's make it easy.

So let's say x i plus delta x and x i minus-- it'll make our lives easier if we do it this way. It can be generalized, but let's do it the easiest possible way. And this is the derivative at the point xi.

Graphically, here's x, here's xi. Here's my function. I want to know what the slope here is. And

I'm going to approximate it by looking one point upstream and one point downstream. And I'm going to connect those points with a line. You would agree that as the spacing gets smaller and smaller, it will be a good approximation for what the derivative is.

So why is it second-order accurate? So you can show that. It's not too hard. You can show that using Taylor series expansion. Like everything else in this course, the Taylor series is really a key.

So let's look at it. So f at x i plus delta x. I want to write this as a Taylor series expansion about the point xi. So tell me the first term. f of xi. OK, tell me the second term.

AUDIENCE: f prime at xi.

JAMES SWAN: f prime at xi. Times?

AUDIENCE: Delta x.

JAMES SWAN: Delta x. Tell me the third term.

AUDIENCE: 1/2 f double prime.

JAMES SWAN: 1/2 f double prime at?

AUDIENCE: Xi.

JAMES SWAN: Times?

AUDIENCE: Delta x squared.

JAMES SWAN: How big is the third term?

AUDIENCE: [INAUDIBLE].

JAMES SWAN: OK. Is this always true, by the way? This? Not always, right? There are lots of functions, non-analytic functions that don't have these sorts of expansions. So we're sort of-- we've got to have a function that has these derivatives. They need to exist at the point xi. Let's assume that's true, OK? Good.

OK, now I'm going to do something really simple. Let's write f at x i minus delta x. And let's just point out the differences between these two formulas. So this is the same. This switches sign because delta x switched signs. It was plus delta x. Now it's minus delta x.

This is the same, delta x squared, same as minus delta x squared. Doesn't matter. This one down here-- well, OK. It's order delta x cubed. It'll actually look the same but have the opposite sign. OK?

We can't resolve it, though. That's sort of the tricky part, right? We don't know. We don't know for sure what value that takes on. This just gives us a bound for how big the error is in this expansion.

So let's now compute what's in that formula. So if you take the difference between this and this, these two will cancel. These two will combine. I'll get 2 f prime xi times delta x. These two will cancel. So I'll get 0. These two, I can't rely on them necessarily canceling each other out. In fact, they want to combine in principle, right? We've just bounded the error. Here, we don't really know what it is, but it gives me something that's order delta x cubed in size.

So this is the-- this is 1 minus 2. It's 2 f prime xi times delta x plus the error, which is order delta x cubed. If I divide by 2 delta x-- I know you guys hate it when the instructor does this, but you know, you can't help it. Divide by delta x, the error becomes order delta x squared.

So if the derivative is this thing plus some order delta x squared piece, it's second-order accurate, we would say, right? The error is going to be reduced like the square of the spacing.

Now you can imagine all the finite difference formulas out there in the world are constructed in this way. You use Taylor series expansion. You've got as many terms as you need to go for the level of accuracy you're trying to achieve for different points based in different ways. And you try to dream up some linear combination of these things so that you get a certain level of accuracy.

They're also not unique. I can construct these in a lot of different ways and have the same sort of error level out here, but different coefficients associated with the error, but the same overall scaling. And then you might choose different approximations for other reasons. Sam.

AUDIENCE: What would stop me from going on further and then just subtracting f prime from both sides? I mean, it wouldn't really tell me anything, except maybe the order of [INAUDIBLE] approximately 0.

JAMES SWAN: Well, I mean, yeah. So let's say you go out further. What's the next term here? It's going to be 1 over 6 f triple prime xi delta x cubed. And here, you'll have minus that because I cubed the

delta x. So when I combine them, I'll actually get 1 over 3 f triple prime.

So I could do a little better. If I knew what my function was, I can tell you exactly how big the error was. I can tell you that it was as big as the third derivative of f. My function doesn't have a third derivative at that point. The error might be huge. And it's possible to construct functions that lack that particular property, OK? So you can go further, and you can use it to sort of quantify exactly how many you think the error might be in the approximation.

But it's all constructed from linear combinations of different Taylor expansions about nearby points. And the related problem, actually, the equivalent problem is polynomial interpolation. So the sort of-- if you can somehow, in your mind, transpose this problem, it looks a lot like polynomial interpolation. So these two are intimately related to each other. OK? Yes.

AUDIENCE: How much do we have to know about polynomial interpolation?

JAMES SWAN: How much do you have to know about polynomial interpolation?

AUDIENCE: Yeah, like, do we have to be able to compute the primes and polynomials?

JAMES SWAN: Say that again.

AUDIENCE: Do we have to be able to compute the primes of polynomials? [INAUDIBLE]

JAMES SWAN: What do I want to say about that? I wouldn't expect that level of detail. I think it's quite difficult to construct Lagrange polynomial expansions by hand. I don't know anyone who does that professionally or for pleasure. I think that's a bad idea.

You should understand polynomial interpolation. You should understand the context in which it's important. What is polynomial interpolation important for, numerically? Do you recall? Hersch.

AUDIENCE: Numerical integration.

JAMES SWAN: Numerical integration, right. So this was really the key to numerical integration, was trying to construct appropriate approximations of our function from a finite set of data and figure out the area under the curve. So you should understand how one uses polynomial interpolation to do numerical integration. But I wouldn't expect us to ask you to actually compute a polynomial interpolant and numerically integrate it by hand. I think that's too much. OK?

AUDIENCE: OK.

JAMES SWAN: OK. More questions. Hersch.

AUDIENCE: Now do you mind showing how the trapezoid rule is over delta t squared accurate?

JAMES SWAN: Sure.

AUDIENCE: For ODE IVPs.

JAMES SWAN: For ODE--

AUDIENCE: IVPs.

JAMES SWAN: ODE IVP, OK. OK, so we have to have an ODE we want to solve. Can I just do-- can I do a one-dimensional unknown? Do I need to do a vector here? Can I just-- would you be satisfied if I wrote this as f of y and t and that it's sufficiently general? That way, you don't have to blame me when I forget to underscore things along the way. And we don't have to think about the Jacobian or anything like that. OK.

So oh, time derivative, right? So there's the ODE we're going to solve. The trapezoid rule. OK, so let's do this the following way. It's actually going to wind up looking very similar to what we just did, which is useful. I might even be able to pull that board down and reuse it at a certain point. So that'll be good.

Let's do this. So what is-- ultimately, what is ODE IVP solving like? What are you trying to do when you solve one of these things?

AUDIENCE: Extrapolate.

JAMES SWAN: It's extrapolation, right? I know the value of y initially. I'm trying to figure out where the hell this thing is going. I've got to extrapolate for it. And you should be very uncomfortable with the notion that I can do that, right? How good a job can I do predicting the future knowing how things are now? It's really quite complicated.

In fact, it's so complicated, there are certain circumstances in which it's not really possible. Discovered here at MIT, actually. So there was weather research going on at a certain point at MIT. It's a very famous case. A researcher, a very famous researcher was doing numerical simulations of a weather model in an ODE IVP.

He ran it for a while, and he stopped. He looked at his end state, the last value of y in his simulation. He put in three digits, used that as the initial condition, and started his simulation again, and it went wacky. Didn't look anything like the initial solution. OK?

How did that happen? It seemed fine up to that point. Solution's cranking along. He stopped. He took three digits accuracy out of the computer, and he started again. And it, I don't know, did something else.

So there are problems that are inherently chaotic and very sensitive to their initial conditions. They're hard to resolve. So the entire point of doing this sort of analysis of ODE IVPs is, at least in the best-case scenarios, we want to try to manage the error associated with these kinds of extrapolation.

In some case, it's really not very possible. You don't know the initial conditions well enough to predict what the final state is. So there are problems that are inherently unstable or sensitive to their initial conditions.

They don't have to blow up either. They can find themselves moving through the phase space in some way that's very, very sensitive. And two points that start very close together can wind up very far apart from each other. So that's deterministic chaos. But outside of those problems, we try to do this extrapolation and figure out sort of a best approximation for where we think our target function is going.

So what are we trying to do? In principle, we know y at a certain point. And we want to know where it's going to go. And so you might say, well, maybe I know dy dt initially. And so I should just move a small increment along that slope. And that's a good approximation. That's the forward Euler approximation.

Trapezoid rule-- or equivalently, the midpoint rule-- says, well, maybe that initial slope isn't really the right one to use. Maybe y does something like this, and we should actually look sort of halfway between the point we're trying to extrapolate to and the current point, and use that value as the best approximation of the derivative and extrapolate for it along that. And it turns out that's more accurate.

So what does that look like? So one typically says, you know, y i plus 1 is equal to yi plus delta t times f y i plus 1/2 comma. And we won't worry about the t here, but it's going to have to be t i plus 1/2.

So I look halfway forward in time. And I say, that's the slope, that's the derivative of y. So let me step forward along that line. It turns out this is going to be reasonably accurate if I can get a good approximation for y i plus 1/2. OK?

OK. So let's-- [COUGH]. Excuse me. Yeah.

AUDIENCE: Is this the midpoint method or--

JAMES SWAN: They're functionally equivalent to each other. They work in precisely the same fashion. So whether it's the trapezoidal rule or the midpoint method, they have fundamentally the same pictures there. OK?

Here, if you do forward Euler, you're doing numerical integration, assuming the slope is the value at this point. If you're doing the midpoint method or the trapezoid rule, you're trying to get a better approximation for the slope, right? So you're trying to build a trapezoid and integrate the area under that trapezoid. So they're fundamentally the same thing.

Now the question becomes, I don't know f at y i plus 2. So I need to use a Taylor expansion of this quantity in terms of the value of f that I know. So I need to expand this in terms of f at yi. And I'll know the derivative of y with respect to t at yi. I know it from this formula.

So if I take this Taylor expansion and I carry it out to the first term I don't know, which is the order delta t squared term, and I carry through all of the errors, I'll find out that there's a numerical error proportional to order delta t squared.

So I need to do exactly this and substitute in terms of quantities I know and quantities I don't know. And the leading-order contribution of the quantities I don't know will give me this order delta t squared piece. And that'll give me the local truncation error in the solution. Does that make sense? So it's still Taylor expansions.

Good. Other questions. Yes.

AUDIENCE: Why is the central difference 0, not [INAUDIBLE]?

JAMES SWAN: Why is the central difference--

AUDIENCE: No, why's the outlook [INAUDIBLE] the one we did [INAUDIBLE]? That one is not implicit, right?

JAMES SWAN: I don't think I said anything like that. So implicit versus explicit really depends on what you're trying to do. So if I'm trying to use a central difference formula of some sort to solve an ODE which progresses forward in time-- it has a direction associated with it that things always progress in-- then this might be an explicit formula. Because I could solve for f at x i plus 1 in terms of what's going on in the past.

If we tried to do finite differences in a boundary value problem instead, and that boundary value problem doesn't have some direction associated with it anymore-- so it's not like space has some preferred direction. Time goes in a preferred direction. Space does not. You may find that the pointwise equations you write down cannot be solved strictly in terms of one of these. You can't just move this to the other side of the equation and solve in terms of everything else.

So you'll have a coupled set of equations that can't be diagonalized, can't be simply reduced. They won't have the form of, say, of triangular matrix, for example. This will happen. And in that case, you would say this formula is implicit.

So it really depends on which problem you're trying to solve, whether you label them explicit or implicit. Explicit is going to mean that the data I want can be found with a simple equation solve in terms of the data I know. Implicit is going to mean I've got to do some complicated equation-solving methodology. I've got to go do Gaussian elimination, or I need to go to fsolve and solve this thing. The data that I want is not available through some simple scheme where I can just write explicitly a thing I want equals some function of things I know. OK?

Other questions. OK. Yeah. Yeah, yeah.

AUDIENCE: [INAUDIBLE], how many questions will there be? And for each question, how many subquestions? Could you just roughly--

JAMES SWAN: Oh, man. The question was, how many questions and how many subquestions on the quiz? Yeah. The quiz is-- in my opinion, it's now shorter than the last one that you took. We've been pretty sensitive to time concerns. We know that was something that was an issue on the previous quiz.

I think there's only going to be two problems on it. They have many parts, but we've really tried to refine the questions in those parts. That's what you should expect. It's comprehensive. It Covers all the things from this section. OK? Yes.

AUDIENCE: Also, DAEs problem--

JAMES SWAN: Yeah.

AUDIENCE: It says that as index increases, we have less degrees of freedom.

JAMES SWAN: Yes.

AUDIENCE: Could you explain?

JAMES SWAN: Sure. Sure. So higher-index DAEs come up all the time. And the degrees of freedom refers to your ability to choose initial conditions for the problem. And we saw several examples in which, if we tried to convert that DAE to an ODE, we found along the way that there were certain hidden conditions that all the initial conditions would need to satisfy in order for this ODE, the system of ODEs that we built from the DAEs, to follow the same solution path or solution manifold as the original DAE system would follow. The higher the index is, the more of these hidden constraints you find. And those hidden constraints tell you that the initial conditions have to be very particular.

So there are-- I would encourage you to look at the specific examples in the notes. And you'll see there was a simple CSTR example that was index 1 and almost unconstrained, basically. There was an index 2 DAE in which the consistent initial conditions, you actually couldn't specify any of the initial conditions. They were fixed by the equations themselves.

But remember, that sort of happened because certain DAEs are pathological in nature. They often ask us to do something we can't do physically, like try to control the input by-- or try to predict-- let's see, how do I say this? It's easy for us to predict the output knowing the input. It's very hard for us to predict the input knowing the output. And these higher-index DAEs were essentially asking us to do that.

There's a lag in how we measure things. And it's easier to go forward through a process than backwards through a process and make these predictions. There's inherent sensitivity in doing the former. It's the same sort of sensitivity you get from these systems of ODEs, in fact. A small change in the initial conditions can lead to big changes in the outlet. So if I measure the outlet, it's hard to predict the inlet. It's just very inherently difficult to do.

And so formulating a model which has this high index, you will naturally find that your choice of

initial conditions for the solution of that model are constrained. And they're constrained because of the way you formulated the model.

If you're actually trying to do control of a chemical process, it's going to be very difficult to try to somehow control the-- try to make the input be a certain thing by controlling the output. That doesn't make sense almost, right? It seems very, very strange. You would really want to change the input in order to hit a target in the output. How do you change the output and hit a target in the input? The process doesn't actually care about things in that way.

And you saw that. Because in that particular CSTR example, you couldn't fix any of the initial conditions, actually, right? There was no free choice in the problem. Everything was automatically fixed, and you had to have perfect knowledge of everything in the system. I wasn't actually able to do control in a sense there, right? I could just measure the output, and I can tell you what the input was supposed to be. And it was super sensitive to my measurement.

So these high-index problems tend to come up when we want to constrain things in a way that's, let's say, physically quite challenging. High-index problems are incredibly common in mechanical engineering, things like robotics where you want to make sure the robot's arm doesn't whack someone in the head. This is important.

And so you're trying to constrain all these degrees of freedom and the paths that it's going to move on. And all those constraints-- I mean, there's actually a very narrow envelope in which this thing gets to operate. So accurately simulating the motion and then making the robot move that way is very empowering.

In chemical processes, that's true, too. But oftentimes, we can make choices in model formulation, right? Like how you try to control the process that can ease those constraints. If I had some control or it doesn't control in an instantaneous fashion, even if it's a very high-bandwidth control, there's still a finite time for signals to propagate.

There's dynamics associated with the controller that can convert that DAE system into a system of ODEs, for example. You don't have-- they may be stiff ODEs. DAEs are the limit of sort of infinite stiffness in ODEs. But they may not be as troublesome to solve as the DAE system would be. OK? It's about model formulation.

OK, can I talk briefly about PDEs? Is that OK? I'll tell you a little bit about it. I like this finite

volume method. I think it's very interesting. You've got some notes on it. I think I like it because I think it comes very naturally, actually, to chemical engineers. Because the problems we like to solve are of the type that finite volume eats for breakfast. These are problems of conservation.

So we have a conserved quantity b. Is that what Dean uses in his book? The new version still uses b as the conserved quantity, right? So the time rate of change of b is equal to minus the gradient in the flux plus some, I don't know, some volumetric change. Maybe a reaction, maybe there's some injections somewhere of mass or momentum, or you know, some source.

And our numerical solution, oftentimes, we would like it to also conserve b, right? Conserve mass, conserve momentum, conserve energy, conserve species. These collocation methods in general, or relaxation methods in general, don't do that. They're not designed to do that. You might be able to construct, for example, basis functions that have that property that they will satisfy conservation.

You might be able to construct certain meshing methods, certain collocation methods-- and this is one of them, the finite volume method-- that respects conservation. And here's how you do it. So you worked really hard in transport to derive these differential forms of the transport equations. But finite volume actually likes the integral form. So you've got to go back.

So you've got to take the integral of these equations over some volume V. And then the time derivative of the amount of b in this volume is related to the flux of b through the surfaces of that volume plus the amount of generation or consumption of the conserved quantity within that volume V, right? And this volume is my finite volume.

I take the domain of interest, and I chop it up into all these little volumes. And then I just need to conserve b within that volume. So I have the first equation you learned in chemical engineering, right? The amount of accumulation of the conserved quantity is related to the fluxes in and out of the control volume plus the amount of generation or consumption within the control volume. This comes very naturally to chemical engineers. It fits perfectly in the problems we like to solve. Yeah.

AUDIENCE: Is the nominal vector pointing outwards or inwards?

JAMES SWAN: Well, it all depends on how we choose to write j, actually, right? So is j the flux into or out of this volume? So I won't prescribe that here. It depends on what you chose as the flux.

So maybe, I don't know, what does-- Dean writes the stress tensor with the opposite sign of Bird, Stewart, and Lightfoot. So these guys, that's the flux of momentum, right? They choose a sign. Then the normals have to respect that sign. Doesn't really matter. The physics doesn't care. You just have to choose the flux in the right direction. If we think of j as the flux into the volume, n should point in.

OK, so you have this conservation equation. You've got to solve for each of your finite volumes. And the finite volumes are coupled together, because a flux out of one volume is a flux into another volume, right? So you can think of these volumes as being little CSTRs. They're all sitting next to each other. And they're all linked together by these volumetric flows from one to another. And that's, in fact, what they are.

The key is, how do I model these fluxes, basically? The accuracy of the method is going to be set by how big a volume am I going to model. Can't have infinitely small volumes. There are going to be too many of them in my domain to solve the problem accurately. And then, do I have good models for these fluxes? And you can think about these terms in this way. That sometimes is useful.

So this big B, the amount of b in the volume, is like the volume times the density of b, the volumetric density of b in the volume. The average density in the volume. R is like the volume multiplied by the average rate of generation or consumption. F is the flux in or out. It's the sum over all the faces of the flux through each of the faces, which is the area of the face multiplied by the average flux through the face. So this is the normal to the face, and this is the flux through the face.

And so I just-- I'm going to solve this equation. I try to figure out, in each volume, what's the average b? That's what this tells me. On average, what is b? On average, what is the concentration? On average, what is the amount of momentum? On average, what is the total energy? That's fine. So finite volume, very simple.

What's nice about it? Well, you know, it's really suitable to any sort of geometry you want to look at. We want to solve this equation, say, in this square domain. We divide the domain up into a bunch of little square volumes. And in each volume, we just make sure that the amount of this b balances the fluxes coming through the volume and the amount of consumption or generation of that quantity within the volume.

But it didn't have to be a square domain, right? A square domain, that's OK. But maybe I want to solve it on some circular domain. Well, I can divide that up into volumes, too. And actually, these equations are the same. They look identical. I've just got to calculate the flux through each of the faces. OK?

But maybe I'm interested in, I don't know, pollution in one of the Great Lakes. And I want to know how concentration of some pollutant is transformed. I'd divide this up into a bunch of little volumes, too. And now I've got to calculate the flux through each of these faces. And that links all these domains together.

Maybe I'm interested in transport within some porous medium-- say, a bone. I think it's someone's hip joint here. OK, well, I divide that up into a bunch of volumes. And I still solve the exact same conservation equation.

So I've sort of decoupled the problem of meshing and equations, the governing equations associated with the meshes. As long as I have a way to divide the domain up. It may be very complicated. I'm going to go ask a buddy in applied math to do that. But I'm going to come to him with the right engineering equations for each of those domains, a little equation for a CSTR. OK?

Turns out this-- you know, one of the most common fluid mechanical codes out there that people use now is OpenFOAM. So it's very common. Lots of people use it. It's a very general solver. Does finite volume. That's the method it chooses to use.

Why does it choose to use it? Because it wants to conserved momentum? A guaranteed momentum conservation. Can't lose momentum, can't lose energy. Can't lose mass. I just have to pass the right amount between each of these faces. As long as my calculation of that is accurate, I've conserved the quantity. Never evaporates.

Finite difference, finite element, different basis set expansions-- they will have this problem that at low levels of accuracy, you can have sort of artificial consumption of, say, the quantity, or artificial production of the conserved quantity of interest. So this can be a very nice way to solve these problems.

It's another relaxation method. It's a type of collocation. I divide the domain up. And then I try to satisfy conservation equations on each of these little volumes. Hence finite volume. OK?

OK. So how do you solve PDEs by hand or write your own code? There's a nice little tutorial

here. We'll see how far we're able to get through it before I've got to give you guys your cards. But I'll illustrate some key points that I think are really important.

So step one, no matter what method you choose, whether you choose collocation, whether you choose Galerkin, whether you're using basis sets or you're using finite elements or finite volumes, you've got some domain you're trying to solve your PDE in. And you've got to discretize it. You've got to figure out how to decompose the domain.

So for finite differences, you might decompose it into a series of nodes at which you want to know the value of your function, whatever that function is. For finite volume, you would decompose it into cells in which you would like to know the average value of your conserved quantity. For finite elements, you divide the domain up into some points, across which local basis functions span the region. We would like to know the weighting of these local basis functions near each of these points. But you've got to divide the domain up.

And the most important thing is you always choose the spacing of the nodes, or the dimensions of the cells, to match the physics. A very natural thing to want to do is to pick-- say I've got this square grid-- to pick the number of grid points to discretize with. I say, I'm going to use 100 points in this dimension and 100 points in that dimension. Actually, that tells me nothing. That's an irrelevant number. The physics has no idea how many grid nodes you put in there. Doesn't care.

There are length scales that are intrinsic to the problem you're trying to study. And the discretization should resolve those physically relevant length scales. So it might be the case that I only need a resolution of one part in 10 along this dimension and one part in 10 along this dimension because the physics are such that that's good to get a description of the solution to this problem. It might be that I need one part in a million in this dimension and one part in 10 in that dimension because the physics are that way.

So you really want to think about length scales, not numbers of nodes. It's a very common mistake. You say, well, 100 and 100, that makes sense, or 10 and 10. Or you know, you pick these randomly. But you want to think about physics instead.

Here's a good example. This is my fault. You'll solve a problem like this. You already saw it in some sense in the finite element or COMSOL tutorial. But there's a little drug patch here near this red zone. It's leaking drug into a sheer flow, and it's all being advected down.

Here's my solution to the problem. I think it's a pretty good one. Here's a solution from when I taught of course two years ago that the TAs produced. And I didn't talk with the TAs about this issue with length scales. And it sort of got the right structure. You can see, here's the patch, and there's some advection and diffusion. But it's not a very stable solution, it turns out. It's also not very accurate. It doesn't reflect the essential physics.

And the point is actually the scales. So here, the patch was 1 centimeter in diameter. And so here, you got many centimeters long this direction, where you get advection. Here, you've got 2 centimeters in this direction. But actually, the solution is really confined down to this small space. The physics all happens at fractions of a centimeter.

You took Transport, so you know that this is a problem in which there's a boundary layer. And all the change in concentration is confined to that little boundary layer. So really, when you do your numerical solution, you'll do like was done up here. Now the lengths are made dimensionless on the width of the patch, but they're-- you know, you can think of these as measured in centimeters.

But look at the dimension here. We need to resolve the solution down at the 0.1 centimeter and smaller scale. I want to put all my little discretization or my nodes down here so that I can see how the concentration field changes as a function of distance. I have to know something about the physics to do that, though, OK? Can't just solve this blindly.

It may or may not be the case that COMSOL recognizes the physical situation and remeshes your problem to put mesh points down where they matter. It may do that. It may not do that. It just doesn't know. People build heuristics to try to solve these remeshing problems, but they may not always apply. So it's good to understand the physics of the problem.

This drug patch problem, this is a pretty common transport problem. I mean, did Professor Bazant give you guys a talk during your visit weekend? Was that this last year? Was that two years ago? It was two years ago.

He does problems with bromine flow batteries. And they worked exactly this way. You have flow of bromine in. An electrochemical reaction takes place at the surface. This entire thing is premised on the fact that you have a very thin boundary layer in which the electrochemical-- in which the reagents are confined. And if that boundary layer gets too thick and reaches the other electrode, it screws up the reaction and doesn't work as a battery anymore.

So being able to resolve concentration gradients in this small domain, it's not just important for this drug problem. It's important in general. It's a common sort of transport-y type problem. And the numerical solution is really going to hinge on your ability to recognize the physics and resolve the solution over the length scales that are important in the problem. OK? So that's the number one most important thing for doing these PDEs.

The rest of the notes will take you through the construction of a numerical solution to the diffusion equation in 2D, and some different scenarios that you might try to employ in MATLAB to solve them, and how you can accelerate the solution using different MATLAB functions. It's fun to try out and play with. You'll see that I can make the solution to this problem go very fast for a domain that's quite highly discretized by recognizing certain forms of the equations, choosing certain solvers to solve with.

OK? Good. All right, I'll meet you guys in back, and we'll hand out these gift cards. And good luck on your quiz.