PROFESSOR: OK, now we're recording. I think about half of my lectures are up on Stella, because about a quarter, I forgot to push record, and another quarter, something went astray with the microphone, and all you hear is like, blegh-egh-egh. But there's about half of them up there, the most important ones.

OK, so the residual is the remainder when an approximate solution is substituted into the governing PDE. And remember, what's so powerful about the residual is that we can take an approximate solution, we can see how close the PDE is to being satisfied, which gives us some idea of how good that solution is, but we don't actually need to know the true solution, right?

We don't need to know the exact solution. So this is different to an error. But it is some measure of how far our PDE is from being satisfied. And so it turns out that this is a really useful thing when we're looking for finding good approximate solutions.

And the example that we worked with the most was the diffusion or heat equation, which we wrote as K dT / dx, all differentiated with respect to x, equal to minus q, where q is the heat source. So I can just write the x's down here. This is notation for differentiation with respect to x.

So if that is our governing equation, and we have an approximate solution that we call T tilde of x, then the residual would be [? r. ?] It's a function of that approximate solution. It's also a function of x. And it would be defined as bring everything over to the left-hand side. So K, then substitute in the approximate solution. So d T tilde / dx, all differentiated, plus q.

So that would be the definition of the residual. And if the approximate solution were exact-- in other words, if we put not T tilde, but T in here, the residual would be 0.

And so remember, we define the residuals. And then we said, let's say that this approximate solution, T tilde, is discretized and represented with n degrees of freedom. And there are different ways.

We could use a Fourier-type analysis and assume that this thing is a linear combination of sines and cosines, or when we get to finite element, we could assume it's linear combinations of these special hat functions. But whatever it is, if we have n degrees of freedom representing our approximate solution, then we need n conditions to be able to solve for that approximate solution. And we talked about two different ways to get to that.

One was the collocation method, where, remember, we set the residual to be 0 at the collocation points. So required the residual to be 0 at, let's say, capital N points, where this approximate solution here is represented with n degrees of freedom. So that was the collocation method, pinning the residual down to be 0 at n points, which gave us the n conditions that let us solve for the n degrees of freedom.

Or the method of weighted residuals, which said, let's instead define the j-th weighted residual, where what we do is we take the residual, we weight it by another basis function, and then we integrate over the domain. So that means the j-th weighted residual-- still a function of our approximate solution T-- is some weighting function, Wj, times our residual, integrated over [? our ?] 1D domain.

So define the j-th weighted residual in this way. That would be the first step. And then the second step would be to require that n weighted residuals now are equal to 0. You guys remember all of this? Yeah?

And then remember, the key is now, OK, what are these weighting functions? And you remember, we talked specifically about a kind of weighting called Galerkin weighting, which says, choose the weighting functions to be the same basis vectors, the same functions that you're using to approximate the T tilde and its n degrees of freedom. And so that's what's key about a Galerkin method of weighted residuals, is that the weighting functions are the same as the functions used to approximate [? as ?] a solution.

And again, then requiring n of those weighted residuals to be equal to 0 gave us, remember, the n-by-n system, where each row in the system was the equation that said Rj equals 0. And each column in the matrix system corresponded to the unknown degree of freedom describing our approximate solution, T tilde.

OK, so that's the method of weighted residuals. But actually, I think once you understand that, you have, more or less, all the building blocks to then move to finite element. And so what's the jump in terms of finite element is that finite element uses a very specific kind of basis function that represents a solution in this special way that then turns out to have really nice properties. And there are a variety of basis functions that you can use, depending on what kind of an approximation you want, whether you want a linear approximation in each element, or quadratic approximation, or even higher order.

And what did we talk about? So we talked about finite element basis functions in 1D and 2D. And we talked mostly about the nodal basis. And I think we talked even mostly about the linear nodal basis. I'm not sure.

Did Vikram talk about the quadratic basis for 1D in the lecture that he gave? No. So I think maybe you saw only the linear nodal basis. So e.g., if we look at the linear nodal basis in 1D, remember that these hat functions, they have the property that they are 0 at all nodes except their own node, where they're equal to 1.

So this is a plot of [? bi. ?] And it's got a value of 1 at node xi. It goes to 0 at node xi plus 1. And then it stays 0 for all nodes to the right of xi plus 1. It also goes to 0 at xi minus 1 and stays to 0. So the basis function is actually defined on the whole domain at 0, and then it's up to 1, down to 0, and then stays at 0 for the rest.

And then, just to run by the finite element notation, we talked about element i as being the element-- the piece of the x-axis that lies between xi and xi plus 1. The element to the left is i minus 1. The element to the right is element i plus 1.

And what's key about these basis functions is that they're 0 in most of the domain. The only place where Ci is non-zero is on element i minus 1 and element i. And we use that fact when we get all those integrals that come out of the definition of the weighted residual, that a whole bunch of integrals go to 0, right? Because you're multiplying things together that are 0 on big parts of the domain.

All right. Questions about [? your ?] [? advisor ?] remembering this? Yes. Any questions about either of those two?

So I will next just remind you about integration in the reference element. What's the idea? Well, first of all, in 1D we have things in the physical x domain. So we have xj, say, and xj plus 1.

And the idea is that the elements could be all those different [? links, ?] depending on how we decided to make the bridge, based on what's going on with the physics. So the j-th element might be a lot bigger than the j minus 1 element, or smaller, or however we'd like to do it. And what we'd like to do is map to the [? xc ?] space where the element goes from minus 1 to 1.

So every element in the x space, element j going from xj to xj plus 1, could be mapped to the reference element in the [? xc ?] space that goes from minus 1 to 1. And the mapping that does that is x-- as a function of [? xc ?] is xj plus 1/2, 1 plus [? xc ?] times xj plus 1 minus xj.

And you should be able to see that when I set [? xc ?] equal to minus 1, I get xj. And when I set [? xc ?] equal to plus 1, these terms cancel out and I get xj plus 1. And then everything in between is just the linear mapping. The middle point here is the middle point here, and so on.

So that's what it looks like in 1D. In 2D, we need to map to the reference element, which is a standard triangle. So in 2D, we have a generic element that might look something like that, where it's got coordinates. Now I'll say x1,y1, x2,y2, and x3,y3. Those are the xy coordinates of the three nodes in this element.

And I want to map it to the reference element, which again, is in [? xc ?] space, which, remember, looks like this right triangle with this node at 0,0, this guy at 1,0, and this guy here at 0,1. This is coordinate direction [? xc1 ?] and this is coordinate direction [? xc2. ?] And again, we can write down the generic mapping just like we did it here. We now have to map x and y, and they are given by x1-- the mapping at x1,y1 plus a matrix that looks like x2 minus x1, x3 minus x1, y2 minus y1, y3 minus y1, times [? xc1 ?] [? xc2. ?]

Just in the same way as here, we can see that varying xc from minus 1 to 1 gave back xj and xj plus 1. You can see the same thing here. If you vary [? xc1 ?] and [? xc2 ?] from the 0,0 point, clearly 0,0 is going to give you back x1,y1. 1,0 should give you the x2,y2, and 0,1 should give you the x3,y3.

So you can define these transformations. And this is generic, right? I can now give any element, any triangle, and you can always get me to a reference triangle. And what that means is that, now, when you do the integrations that show up in the finite element method-- so an integral over, let's say, a domain, over element K, some function, function of x, these are the kind of integrals that show up in the 2D finite element method.

We can now do an integration over the reference elemental, [INAUDIBLE] omega [? xc. ?] We're now writing f as a function of x of [? xc. ?] And what we need to do is to introduce the

Jacobian of the mapping so that we can now integrate with respect to the reference element. And the Jacobian here, this is just the determinant of this matrix. Can't remember, but I think you guys must have done this with Vikram, because I think I was away [INAUDIBLE] yeah?

So we end up with this j, and this j here is just the determinant of this x2 minus x1, x3 minus x1, y2 minus y1, y3 minus y1. So basically, this accounts for the change in area, because you're integrating over this guy That gives you the reference element.

Sometimes I think it seems a little bit messy when you see it written down. But the reality is it makes you code really clean, right? Because you just introduced these mappings, and then every single integration in the code takes place over the reference element. You just go account for the mappings from whichever element you happen to be in through the determinant of this matrix. Is that good?

OK, so somewhat related to that, because we'll still talking about integration, was Gaussian quadrature. A really simple idea, but quite powerful, and you had a chance to practice that on the last homework. So remember, for Guassian quadrature, for the simple examples that we worked through in class, we could do the integrals analytically, because we were integrating constants, or linear functions. So we were integrating things like e to the x, or xc to the x.

But in practice, for real problems, you may end up with integrals that you can't do analytically, either in the stiffness matrix or in the right-hand side system, on the system. So if you can't do the integrals analytically, then you resort to Gaussian quadrature.

And remember that Gaussian quadrature in 1D looks like this. If we're trying to integrate some function, g of xi, over the domain minus 1 to 1, we approximate that as the sum of the function evaluated at quadrature points xi i weighted by weighting coefficient alpha i, and it's the sum from i equals 1 to nq.

So nq here is the number of quadrature points. Number of quadrature points. [? xci ?] here is the i-th quadrature point, and alpha i here is the i-th quadrature weighting.

Do you remember-- where do the rules come from? How do we figure out the [? xci ?] [? alpha i ?] [? pairs? ?] Perfect integration of a polynomial.

So if, for example, we wanted to just use one quadrature point, then we would say, with one point we can perfectly integrate a linear function, right? Because if I take any linear function, and what I'm interested in is the area under the function, I can get that perfectly by just evaluating its midpoint, and the area of the rectangle is the same area of the-- what is this thing called?

AUDIENCE: Trapezoid.

PROFESSOR: Trapezoid, yeah. So with one quadrature point, I can typically integrate a linear function. And I would do that by setting alpha 1 equal to 2 and xc 1 equal to 0. So again, midpoint and the area.

If I am willing to have two quadrature points, it turns out that I can integrate linear functions exactly, I can integrate quadratics exactly, and I can actually also integrate cubics exactly. And it turns out that the weights I should choose are alpha 2 equals 1 and quadrature points at plus or minus 1 over root 3. And again, those points are chosen to give perfect integration of polynomials. And of course, when we apply those quadrature schemes to non-polynomial functions, we introduce an approximation, but typically that's something that we're willing to tolerate so that we use not too many quadrature points in the scheme.

OK, questions about reference elements or Guassian quadrature? Is that good? Yeah, Kevin.

AUDIENCE: [INAUDIBLE] just slightly [INAUDIBLE] quadrature. This is on [INAUDIBLE] assuming there's no quadrature that can approximate [INAUDIBLE]

PROFESSOR: Yeah. So the question is that-- so it's not only for polynomials, but these are the points that are derived by considering exact integration of polynomials. And yes, absolutely, there are other famous quadrature schemes. There are Chebyshev points, and there are all kinds of points that are derived in different ways. Some of them are derived by looking at sinusoidal functions. Yeah, absolutely.

Yeah, it's a whole study of math, quadrature points and quadrature schemes. It's actually kind of neat stuff. Yeah.

All right, so let's move on and talk about boundary conditions, and then we'll close by forming the system. And then, that will be finite element in a nutshell.

Number five is boundary conditions. And the main types of boundary conditions that we discussed were Dirichlet, where, remember, for a Dirichlet condition, you specify-- what do we specify? Yeah, the value of the unknown. The solution at a point.

And for a Dirichlet condition, we are going to replace the weighted residual equation. So we're going to form the system, we're going to look at forming the system in just a second. But we already said that setting each weighted residual equal to 0 gives us one row in our matrix system, right? One row corresponds to the equation [? Rj ?] equals 0.

So if there's a Dirichlet condition in a node-- that's usually at the boundary, maybe at the first one-- we're going to come in and we're going to strike out that weighted residual equation that we got from multiplying by [? c1 ?] and getting the weighted residual to 0. We're going to strike that out and replace it with an equation that might look like 10000 equals to whatever the Dirichlet condition is.

Then we're going to fit the value. And we have to be a little bit careful and make sure that we're setting truly the value of the unknown. If we're using the nodal basis-- I guess I didn't state explicitly when I drew the nodal basis, but what's so special about the nodal basis? What does it mean for the unknowns?

Yeah, that's right, that the unknown coefficients that you solve for turn out to be the values of the approximate solution at the node. And that's because, by construction, those basis functions are 1 at their own node and 0 everywhere else. So if you're using the nodal basis, the unknown coefficients correspond to the value of the approximate solution at the node. Which means that for a Dirichlet condition, you can directly manipulate that coefficient.

And then the other kind of boundary condition that we talked about was a Neumann condition, where now you specify the flux. And what we see is that when we do the weighted residual, when you integrate by parts, you get a term out that is a boundary term that's where the flux shows up exactly. And so a Neumann condition contributes to the weighted residual. You'll see its place exactly in the derivation of your weighted residual, where you can go in-- if the Neumann condition is flux equal to 0, it basically strikes out that term. If it's a non-zero, a non-homogenous Neumann condition, then you could go in and put that contribution as the flux.

So in this case, you leave the weighted residual equation, but you add the contribution. So two different boundary conditions, two different implementations. And so what this means is it's an additional contribution to the stiffness matrix. So it's getting a little low. It says, add contribution to weighted residual, additional contribution to stiffness matrix.

OK, so last thing. I'll just kind of put all those pieces together for the 1D diffusion, and then finally, end with just showing how the matrix system gets formed. Yep?

AUDIENCE: [INAUDIBLE]

PROFESSOR: That's right. The Robin condition is a combination of the two. So there's a Dirichlet component and a Neumann component. I don't know if you actually did-- were they discussed in the finite element? They were mostly discussed in, I think, the finite volume, finite difference section. But yeah, it's just a combination. We've put something in the right-hand side. What's that?

AUDIENCE: We had them in the project.

PROFESSOR: Oh, you had them in the project. So then you know how to do them, right? Or you know how not to do them.

[LAUGHTER]

Yeah. OK, so just as the last bringing it all together, let's think about finite element for 1D diffusion, and we need to make sure that you are clear that you can go through all these steps. So starting off with the governing equation, K dT / dx, differentiated again with respect to x, equals minus q on the domain minus L over 2 to L over 2 [INAUDIBLE] x.

So let's see. We would write the approximate solution, T of x, as being the sum from i equals 1 to n of sum ai times Ci of x. These here are our finite element basis functions. We might, for example, choose to use the linear nodal basis, those hat functions. So again, those ai, the unknown coefficients that we're going to solve for, are going to end up corresponding to the value of our approximate solution at the nodes in the finite element mesh.

We want to write down, then, the j-th weighted residual. We'll call it [? Rj. ?] And what is it? It's take everything over on the left-hand side, substitute in the approximate solution.

So we're going to get a K d T tilde / dx, all differentiated, plus q. Weight it with the j-th spaces function. And because it's Galerkin, we're going to use the same C. So we weight it with Cj, and then integrate over the domain, integrate from minus L over 2 to L over 2. I could write out all the steps, but I think you've seen it many times in the notes.

So we get that, and then what's the next thing we do? What do we do? What's the first thing-- what are we going to end up with? We're going to end up with [? Cxx. ?]

We can assume K is constant, if you want. We're going to end up with [? Cxx's ?] multiplied by C, so we're going to end up with [INAUDIBLE] the integral of [? C ?] times the second derivative of C with respect to x.

What do we do? Integrate by parts. [? Usually you ?] always [INAUDIBLE] integrate by parts. That's going to move one differential off the C, onto the other C. Yeah.

So get the approximate solution, substitute it in, bring everything to the left-hand side, multiply by Cj, and integrate. Integrate by parts, and when you integrate by parts-- I can write it out, if you feel uncomfortable, but again, I think you've seen it many times, and it's in the notes.

I'm going to keep the T tilde, rather than writing the sum. This thing is really the sum of the [? ai's ?] times the [? Ci's, ?] right? But I'm just going to write it as T tilde, because I think it's a little bit clearer. You can [INAUDIBLE] if you're working through things, feel free to leave the T tilde in there until you get to the bottom, so you don't have to write too many.

So there's the-- we're integrating v to u, so there's the [? uv ?] term that comes out on the boundaries, out of the integration by parts, minus now the integral where we have moved-- we're going to differentiate Cj, and this is the Cj that's coming from the weighting for the j-th weighted residual. And then our other one has been integrated, so now we just have a K d T tilde / dx. dx.

And then we have the q term, which looks like an integral from minus L over 2 to L over 2. And again, it's the Cj times q dx. We don't have to do any integration by parts. This side just stays the way it is.

So I jumped one step there, but again, the [? first line ?] of the residual would have a Cj times this full term with the T tilde there. One step of integration by parts gives us the boundary term minus the integral with one derivative on the Cj, the weighting Cj, and one derivative on the approximate solution. Yes? No? Yes? OK.

So where do all the bits go? Here is if we had a Neumann condition. This is where this contribution would go. Yeah?

This term here is going to give us the stiffness matrix, and this term here is going to give us the right-hand side. Yep. Let's just remember how all that goes together, conceptually. This guy here is going to go to our stiffness matrix. This guy here is going to go the right-hand side.

And if we think about just integral xj minus 1 to xj plus 1 of Cj x K d T tilde / dx. So why did I change the limits here?

AUDIENCE: [INAUDIBLE]

PROFESSOR: What is it? Mumble a bit louder. Yeah?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Yeah, because we're looking at the j-th weighted residual. We've got the Cj x sitting out in front here, and we know-- again, if you sketch the Cj x, you know that it's only non-zero over elements that go from xj minus 1 to xj and xj to xj plus 1.

So all the other parts of the integral go to 0. So that's the only bit that's left. And again, I'll skip over a few steps, but it's all derived in the notes. If you then substitute in the T tilde [? has been ?] the sum of the [? ai ?] times the [? Ci's, ?] again, a lot of the [? Ci's ?] are going to be 0.

The only ones that are going to be non-zero on this part are going to be this one, so Ci equal to Cj, and then this one, right? Ci equal to Cj minus 1. So itself and its left neighbor are the only ones that are going to be non-zero over this. Because in the next one-- oh, and this guy here. And this guy here. That's right, there you are. Three of them. Itself, its left neighbor, and its right neighbor are all going to contribute to the way I've written it, this full integral.

And so what does it end up looking like? It ends up looking something like there's an a j minus an a j minus 1 over delta x j minus 1 squared, and here are I've left in the K. If K were a constant, it would just be coming out. And then there's another term that looks like a j plus 1 minus a j over delta x j squared integral x j to xj plus 1.

OK, so I mean you could work through all of these, but what's the important concept? The important concept is that-- let me draw this a little bit more clearly. The important concept is that when you're an-- x j-- always draw the picture for yourself, because I think it really helps.

Don't get too embroiled in trying to integrate things and plug things in. It's easy to make mistake. Just think about where things are going to be going. So there's Cj. There Cj plus 1. And there's Cj minus 1.

So when we think about this element, the j minus 1 element, which goes from x j minus 1 to x j, which two get activated? It's j minus 1 and j. So we expect the coefficients aj and aj minus 1 to be the ones that are going to get coefficients put in front of them in the stiffness matrix, right? Nothing else.

And when we think about this element, xj and xj-- from xj to xj plus 1, this is the j-th element, it's Cj and Cj plus 1 that are going to get activated. So we expect to see coefficients aj and aj plus 1, with something [? multiplying ?] them.

And so before we went through we worked out all the integrals, again, you could do it by just plugging and substituting, but you could also just think, conceptually, what is the system going to look like? At the end of the day, you're going to end up with a big matrix, the stiffness matrix, multiplying these unknown coefficients, which correspond to the approximate solution at the nodes in the finite element [? match, ?] and then the right-hand side.

And again, remember that the j-th row corresponds to setting the j-th weighted residual equal to 0. So each row in the matrix comes from setting a weighted residual equal to 0, and then each column-- so i-th column multiplies [? an ?] [? ai. ?] So how does this system start looking?

Well, we know if this were the first element, we would be getting what? [? c1 ?] and a [? c2, ?] so we might get entries here and here. Although, if we had a Dirichlet condition, we'd see 0 [AUDIO OUT] entries. First, second, and third. The matrix starts to look like this.

And again, it's because when we do the j-th weighted residual, which is here, we're going to trigger contributions to j minus 1, to j, and to j plus 1. And that's because the j-th weighting function, again, triggers two elements, the j minus 1 element and the j-th element.

OK, so I think [INAUDIBLE] two sets of skills you have to have. One is that you have to be comfortable doing the integration, writing down the weighted residual and doing integration by parts. And then the other is not to lose sight of the big picture of forming this matrix system. Columns correspond to the unknowns, rows correspond to j-th weighted residuals, and it has a very specific structure because of the way we choose the nodal basis.

Questions? You guys feel like you have a good handle on the finite element? Is that a no, or a yes, or you will have by Monday and Tuesday?

So the last thing I want to do is go back, and I want to remind you that I've given you a very specific list of measurable outcomes, which represent the things that I want you to take away from the class. And so when I construct assessments like the final exam, that's what I'm doing, is I'm trying to measure those outcomes.

So if-- when you are studying for the final exam, going back to those measurable outcomes and going through them I think is, hopefully, a really helpful guide for you. Because it really, again-- let me get plugged in here-- really, again, tells you exactly what it is that I want you to be able to do.

So let's go through them. And I just-- so I've pulled them. And I'll go back to the [INAUDIBLE] site, or the MITx site where they are. But starting at Measurable Outcome 212, this one is, "Describe"-- and I also want to point out that the verbs are also important-- "Describe how the method of weighted residuals can be used to calculate an approximate solution to a PDE." Hopefully, you feel comfortable with that.

"Describe the differences between method of weighted residuals and collocation method." We didn't actually talk about least squares, so just the first two. "Describe the Galerkin method of weighted residuals." And we just went over all of that.

Then under finite element, describe the choice of approximate solutions, which are sometimes called the test functions. So that's the choice of the [? Ci's ?] that you use to approximate the solution. Give examples of a basis, in particular, including a nodal basis.

And I've crossed out the quadratic, because I don't think [? Vikrum ?] actually covered it in that lecture. But if you're comfortable with the linear nodal basis, that will be good. To describe how integrals are performed using a reference element. Explain how Guassian quadrature rules are derived, and also how they're used, in terms of doing integrations in the reference element for the finite element. We went through all of these.

Explain how Dirichlet and Neumann boundary conditions are implemented for the diffusion equation, or Laplace's equation. So Robin not actually necessary for the finite element. And then describe how that all goes together and gives you this system of discrete equations.

Describe the meaning of the entries, rows and columns. So again, this is what I was meaning by, this is kind of the big picture, right? Now, to do the integrals and come up with numbers is one thing, but explaining what the system means, and what all the pieces are, and how they go together, is important.

So I think that's essentially what we just covered. And I also want to remind you that if you go to the MITx web page, so there's the Courseware where the notes are, but remember, there's this Measurable Outcome Index that has all the measurable outcomes. And they're also linked to the notes.

And I know one piece of feedback I've heard is that it would be really helpful to be able to search on this web page. I know you can't search, but this is actually one way to navigate, that we could come in here and click on one of these. And I think our tagging is pretty thorough, but as you click on that, here are all the places in the notes where-- in particular one, describe how integrals are performed in the reference element-- these are the sections in the notes that relate to this measurable outcome, and then these are the sections in the notes that have embedded questions in them.

So again, if you're looking for a study guide, going through these measurable outcomes, and going back and looking at the notes, and making sure that you can execute whatever it says here. Yep. OK, any questions about finite element? You guys look worried or tired. OK.

All right, so I'm going to do the same thing, then, for probabilistic analysis and optimization. I'll go through a bit more quickly, just because hopefully that stuff is a little bit fresher. But again, I just want to touch on the main topics. We'll hit the high points on the board, and then we'll just take a look at the measurable outcomes. And particularly, I just want to make sure you're clear on a lot the last bits, which are a little bit introductory, what it is that I expect you to be able to do.

Does it feel like you've learned a lot since spring break? Remember on spring break, wherever you guys were, in Florida or whatever, you didn't know finite element. You didn't know Monte Carlo. You maybe didn't know anything about optimization. It seems like a long time ago, no?

AUDIENCE: It's a long time, but a short time.

PROFESSOR: It's a what?

AUDIENCE: Long short time.

PROFESSOR: A long short time?

AUDIENCE: [INAUDIBLE]

PROFESSOR: All right. So over there are the six main topics that we've covered in probabilistic analysis and optimization, the basics of Monte Carlo, the Monte Carlo estimators. We talked about various reduction methods. We talked about design of experiment methods for sampling. We did a very basic intro to design optimization, and then we talked about responsiveness models in the last lecture.

So let's just go through and-- where are my notes? So this is probabilistic analysis and optimization, so I'm going to start numbering from 1 again. So again, Monte Carlo simulation, I hope by now, after the project, you feel pretty comfortable with the idea.

I think the little block diagram says a lot. If we have a model, we have inputs. Maybe we have x1, x2, and x3, and maybe the model produces one or more outputs. So inputs, in this case, maybe just a single output is always considered. That we want to represent our inputs as random variables, and so if we propagate random variables through the model, then our output will also be a random variable.

And the idea of Monte Carlo simulation is really quite simple. Remember, we broke it into the three steps. The first step is to define the input PDEs. So once you make the decision that you want to-- not PDEs, PDFs. Input PDFs.

Once you decide you want to model these things as random variables, you have to come up with some way where you can probabilistic analysis, some way of describing what kind of distribution these guys follow. And for pretty much everything that we did, those were given to you. Remember I said that, in practice, this is a really difficult thing to do. You can use data that you might have collected from manufacturing, or you might query experts, but you somehow have to come up and define the input PDFs.

Then, once you have those PDFs defined, you sample your inputs randomly. Each random [? draw ?] you solve by running through the model, and so what this means is that if we make capital N samples, then we have to do n solves. So each solve is going to be one run of our model, which might be a finite element model or whatever kind of stimulation it would be.

So now we've taken N samples, we've run the model n times, we have n samples of the output, and the third step is to analyze those resulting samples. So analyze the resulting samples of the output, or outputs, if there's more than one. And in particular, we might be-- well, we're usually interested in estimating statistics of interest. Statistics of interest. And those statistics might include means, variances, probabilities of failure, or whatever it is that we want to calculate.

And if these inputs are uniform random variables, then drawing a sample from them is pretty straightforward. You can draw them just randomly [? under ?] [? 01. ?] If these things are not uniform random variables, then, remember, we talked about the inversion method, where you use the CDF, you generate a uniform random variable, which is like picking where you're going to be on the CDF, go along, invert through the CDF, and that gives you a sample of x.

And in doing that, by sampling uniformly on this axis, remember, we graphically saw that that puts the right [? entity ?] of points on the x-axis. And again, you implemented that in the project with the triangular distributions. What?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Nothing awful. Yeah. I think the triangular is the only one that's really analytically tractable.

AUDIENCE: [INAUDIBLE]

PROFESSOR: Yeah. I'm not going to have you do Monte Carlo simulation by hand.

[LAUGHTER]

Although it could be interesting. We could correlate your grade with how many samples you could execute in a fixed amount of time. See what the [? process ?] of power is. But yeah, no. Triangular is fine. Triangular is fine, as long as you understand it generally.

AUDIENCE: Can we build [INAUDIBLE]

PROFESSOR: If you can do that in the half-hour you have to prepare with the pieces of the paper that you have with you. That would be impressive. OK, so I think the more conceptually difficult part of Monte Carlo are the estimators. And let me just-- I know, I feel like I've been harping on about these for a while, but let me just make sure that they're really clear, because I think it can get confusing.

So we talked about three main estimators, the mean, the variance, and probability estimators. So if we want to estimate the mean, and it's the mean of y, y being our output of interest-- so let's call it mu sub y.

Then what do we do? We use the sample mean as the estimator, and we denote the sample mean as y bar. And it's equal to 1 over n, the sum from i equals 1 n of y i, where this guy is the i-th Monte Carlo sample. The y that we get when we run the i-th Monte Carlo sample through our model.

So we're trying to estimate mu of y, which is the real mean. To do that, we use the sample mean, y bar, which is just the standard sample mean. And what we know is that for n, as long as we take a large enough n-- large, 30 or 40-- that this estimator-- so let me back up a second.

This estimator-- this thing is the true mean. It's a number. The estimator is a random variable. And it's a random variable because we're sampling randomly, so any time we do this, we could get a slightly different answer.

So this is a deterministic quantity. This is a random variable. We need to be able to say something about the properties of this random variable to be able to say how accurate it is. What we know is that for large n, where large means of the order of 30 or 40, this random variable, y bar, will be normally distributed. And the mean of-- let's call it mu sub y bar, the mean of x-- mean of mu sub y bar, and variant sigma y bar squared.

OK, so this is the mean of y bar and a variant of y bar. And what we can show is that the mean of y bar-- the expected value of our estimator, y bar, is in fact equal to mu of y. It's equal to the thing that we're trying to estimate. And so this is what's called an unbiased estimator, because on average, it gives us the correct result. So that's great, on average.

But now, the variance of that estimator is important, because it tells us how far off could we be if we just run it one time. And we could also derive-- that variance of the mean estimator is given by the variance of y itself divided by the number of samples that we drew.

And the square root of this thing, sigma y bar, is sometimes called the standard error of the estimator. I tend to not use that term so much, because it confuses me a little bit. I just like to think of this as the variance of the estimator. So I really recommend that when you write these things, use the subscripts to denote mean of y bar, mean of y, variance of y bar, variance of y.

Keep it straight. Yes, Kevin.

AUDIENCE: [INAUDIBLE]

PROFESSOR: This is not an estimator. The estimator is y bar. This is the mean and the variance of that estimator. We haven't yet talked about the estimator for the variance, which could biased. Yeah, so this is not an estimator. This is just the variance of this estimator. OK? Yep.

OK, so that's the mean estimator. And the other ones we talked about were variance and probability. So let's say we want to estimate the variance of y. And again, y is the output of our code. And so this is sigma y squared.

Then, here we have a variety of different options. I think Alex presented three to you. Maybe the one of choice is f y squared, which is given by 1 over n minus 1, the sum from i equal 1 to n of the [? yi's ?] minus the y bar squared. OK, so again-- and I guess the notation is a little bit unfortunate, because this isn't what people use.

The f y squared is the estimator for the variance sigma y squared. And again, this is a random variable, and we want to know what its expectation and its variance are, because that tells us how good of an estimator it might be. And what we can show is that the expected value of this particular estimator is actually equal to the variance we're trying to estimate. So this one is unbiased.

And Kevin, to your question, if we had 1 over n there, that would be biased. And I think [INAUDIBLE] Alex also showed you 1 over n minus 1/2? Oh, 1 over n minus 1.5. That one, I think, turns out to be unbiased estimate of the standard deviation. Yep, yep.

OK, and so that's the mean of it. It turns out that the variance, or the standard deviation of this estimator is generally not known. So the mean estimate, we know by central limit theorem, follows the normal distribution, and we know its variance. For the variance, typically it's generally not known. How could we get a handle on this?

We could do it multiple times. Maybe we can't afford to do it multiple times. How else could we get [INAUDIBLE] what's the cheating way of doing it multiple times? Bootstrapping. Bootstrapping, which is redrawing from the samples we've already evaluated. That could be a way to get a handle on how good that is.

Turns out there's a case that if y is normally distributed, then you do know what this is, and this estimator follows a chi squared distribution that's mentioned in the notes. That's a very restrictive case. So in general, we don't know what this variance is, the variance of the variance estimator. Yep.

And then, the third kind of estimator that we talked about was for probability. So estimate mean, estimate variance, and then estimate probability. So let's say that we're trying to estimate the probability of some event A, which I'm just going to call p, little p for short. And so we would use as an estimator, let's say, p hat of a, which is na over a-- over n. The number of times that a occurred in our sample divided by the total number of samples.

Now, again, for this one, we know that by central limit theorem, for large n-- so again, this estimator, t hat a, is a random variable. For large n, it follows a normal distribution, with the mean equal to the probability that we're actually trying to estimate, and the variance equal to the probability we're trying to estimate times 1 minus the probability we're trying to estimate divided by n. So this guy here means that it's unbiased.

AUDIENCE: So you cannot say that in general, an estimator of [INAUDIBLE] variance, true population variance, you have to include for each type of estimator--

PROFESSOR: That's right. Yeah, and I think that's a place where some people were a little confused in the project. It turns out that this is a variance of n. But this is the variance of the Bernoulli random variable that's the [? 01 ?] indicator that can come out of the code.

And so that's a way-- if you feel more comfortable thinking about it that way, you can think about it that way. But I much prefer to think that this is the mean and this is a variance-- the mean of the estimator and the variance of the estimator. And the variance of the estimator is this whole thing, just like the variance of this estimator is this whole thing here.

And there do turn out to be other variances divided by n, but that relationship is a little bit murky, and I think you could get yourself into trouble if you think of it that way. It doesn't work over there. Yeah. So it's definitely-- it's not a general result. Yeah.

Maybe the key is that here you can write a probability as a mean, because it's the average with that indicator function. And so you can use the same result that you have used here to get to this one. But again, that seems a little-- if the ideas are clear in your mind, you can think about it that way, if you want to.

OK, so once you have this, then specifying an accuracy level with a given confidence is pretty straightforward, right? If we say that we want our estimate to be plus or minus 0.1 with confidence 99 percentile, then what we're saying is that this distribution's standard deviation-- square root of this guy-- has to be within plus or minus, if it's 99 percentile, 3-ish times that 0.1, or whatever I said it was. Yep.

And so that will tell you how many-- should make this a bit of a curly n-- how many samples you need to run in order to drive-- so the more samples you run, the more you're shrinking this distribution of p hat a, and making sure that the one run you do-- shrink, shrink, shrink-- falls within the given tolerance that you've been given, with a given amount of confidence.

Now, of course, the trick is that you don't know p, in order to figure out what this variance is. But like in the project, worst-case, variance is maximized when p is 1/2, so you could use that. Or you could do some kind of an on-the-fly checking as you go, get an estimate for p hat, and then use that in here, and keep checking. And so then again, it would be approximate, but if you built in a little bit of conservatism, then you would be OK.

I posted solutions for the project last night, where that's explained. OK, questions about Monte Carlo estimators? Three different kinds of estimators that I expect you to be comfortable with. OK? You guys are very talkative today. Very talkative.

All right, so moving on, variance reduction methods. The main one that we talked about was importance sampling, where again, it's kind of a trick to help control the error in our estimates. So in particular, we looked at estimating a mean, mu of x, which is the expectation of some random variable x. And remember, I was putting the little x down there to denote that we're taking expectation over x, because we're going to play around with the sampling.

And remember, we said that we could write this as the integral of x times f of x dx, where this f of x is the PDF of x. And then, remember, we just play this trick where we divide by z and multiply by z, which means that we don't actually change anything. But then we define z such that z times f of x is actually another density, f of z. Which means that we can interpret the mean of x as being the expectation over x with respect the density f x, or we can also interpret it as being the expectation of x over z with respect to the density z.

And why do we do that? We do that because now we have two ways to think about this mean that we're trying to estimate. We can think about it as the expectation of x under the density f of x, which means that we would sample x, again, under this density, f of x. And then we would get our mean estimate. And more importantly, we would get our estimator variance that we just saw a couple minutes ago as being the variance, again, under f of x of x over root n.

So that would be the variance of-- the mean [INAUDIBLE] so that's just the standard Monte Carlo that we've been talking about. But what we just saw over there was that we can also think about this thing as being the expectation of x over z taken with respect to the pdfz. So this would mean, conceptually, sample now x over z, under now f of z.

And then, the real reason that we do that is that we can then play around with the estimator variance. And it now is the variance of x over z with respect to z over n. And while these guys are the same, because the mean is the linear calculation, these two are not necessarily the same.

So then, remember, we talked about different ways that we could come up with a clever f of z that would put more samples, for example, in a probability region that we're trying to explore, that could have the effect of reducing the variance of the estimator, meaning less samples for the same level of accuracy, while giving still the correct estimate of the probability in that particular case. And we also talked about how you would modify the Monte Carlo-- remember, the Monte Carlo estimator had that x over z, and then the z scaling back [? inside it. ?] I think, unfortunately, that is one of the lectures where the audio got scrambled. But I scanned in my notes, and they're posted online.

OK, how are we doing on time? [? 46. ?] So we talked briefly about variance-based sensitivity analysis, and I think I'll put it up on the board. But I really just introduced you to the idea of doing variance-based sensitivity analysis of defining those main effects sensitivity indices that gave us a sense of how the variance in the output changes if you learn something about one of the inputs.

We talked about different DOE methods, the idea being that what we're trying to do is sample the design space, but not do full random Monte Carlo sampling, maybe because we can't afford it. So in DOE [INAUDIBLE] that the idea is that we would define factors, which are inputs. And we would define discrete levels at which we might want to test those factors. And that we're going to run some combination of factors and levels through the model, and collect outputs, which are sometimes called observations in DOE language.

And that the different DOE methods are different ways to choose combinations of factors and levels. And we talked about the parameter study, the one-at-a-time, orthogonal arrays, Latin hypercubes. So again, different DOE methods are different ways to choose which combinations of factors and levels.

And then we also talked about the main effects, averaging over all the experiments with a factor at a particular level, looking at that contingent average of all experiments. And remember, the paper airplane experiment, where we looked at the effects of the different design variable settings. That gives us some insight to our design space.

OK, and then lastly, in optimization, not too much to say here. Just a basic idea of unconstrained optimization, what an optimization problem looks like, the basic idea of how a simple algorithm like steepest descent, or conjugate gradient, or Newton's method might work. We talked about the different methods to compute gradients. And we saw particularly that finite difference approximations, which you saw when you were approximating PDE terms, can also be used to compute gradients, or to estimate gradients, in the design space.

And then, we also talked about how we could use samples to fit a polynomial response surface model, if we needed to approximate an expensive model, and use it in an optimization. So that was the last-- really, the last two lectures. All right, so let me put up-- quickly look at the outcomes. Everybody got-- what's that?

AUDIENCE: [INAUDIBLE]

PROFESSOR: We'll go look at the outcomes for that last section, the probabilistic analysis and optimization. And again, these are specifically what I expect you to be able to do. I'm not going to ask you to do anything that's not in these outcomes. If you can do every single one of these outcomes, you will be 100% fine.

So 3.1 and 3.2 related to some basic probability stuff. 3.2 described the process of Monte Carlo sampling from uniform distributions. 3.4 described how to generalize it to arbitrary univariate distributions, like triangular. Use it to propagate uncertainty. You've [INAUDIBLE] this one in the project-- [? horray. ?]

Then, a whole bunch of outcomes that relate to the estimators. Describe what an estimator is. Define its bias and variance. State unbiased estimators for the mean and variance of a random variable and for the probability of an event. Describe the typical convergence rate. So this is how does the variance, how do the standard deviations of these estimators behave as n increases?

For the estimators, define the standard error and the sampling distribution. So this is what we're talking about with the normal distributions that we can derive for the mean and for the probability. In the variance, we typically don't know what it is, unless we have that very special case that the outputs themselves are normal.

Give standard errors for the sample estimators of mean, variance, and event probability. This one should really only be in that very special case. And then, obtain confidence intervals, and be able to use those to determine how many samples you need in a Monte Carlo simulation.

Then on Design of Experiments and Response Surfaces, so just describe how to apply the different design of experiments methods that we talked about-- parameter study, one-at-a-time, Latin hypercube sampling, and orthogonal arrays. Describe the Response Surface Method, and describe how you could construct it using least squares regression.

Remember, we did the points. We construct the least square system. Use it to get the unknown coefficients of either a linear or a quadratic response surface. And then, we didn't talk about this in too much detail, but I know you've seen the R2-metric before, and you understand that that measures the quality of the [? search, ?] of the Response Surface to the data points.

Then lastly, on the Introduction to Design Optimization, be able to describe the steepest descent, conjugate gradient, and the Newton method, and to apply it to simple unconstrained design problems. Describe the different methods to estimate the gradients, and be able to use finite difference approximations to actually estimate them. And then, interpret sensitivity information, so things like the main effect sensitivities that we talk about, and explain why those are relevant to aerospace design examples.

Remember, we talked about how you could use sensitivity information to decide where to reduce uncertainty in the system, or where-- there was called factor prioritization, or factor fixing, where there are factors that don't really matter, and you could fix them at a value and not worry about their uncertainty.