QIQI WANG: So today, I kind of prepared not for the whole lecture, but a little bit short of that. So I really expect you to ask questions on this material. It's supposed to be a lecture that helps you review the material we have already covered and prepare you for the midterm.

So instead of me just going mechanically through the material, I want to ask you to initiate what do you think is more confusing, or you'd like me to clarify again, and things like that. If you feel something is confusing, it's probably confusing for the whole class. So please raise it so that I can spend more time on it.

OK, so before I do that, I just first want to finish up the finite volume scheme we have been working on for the last two lectures. We have been discussing finite volume schemes in one dimension. But applying the same concept to two dimensions or even three dimensional is surprisingly straightforward. So let me first give out what we did in 1D.

We started out in finite volume schemes-- we started out with the integral form of the differential equation. We started out with d dt of the integral of a conserved quantity inside a control volume, omega, it is equal to minus of the flux out of the control volume. The minus sign is because the time derivative is really the flux into the control volume. But the definition of our n, normally, is usually out of the control volume. So we have a minus sign to reverse the normal so that it points into the control volume.

Times the flux as a function of rho, ds. So this is like the integral form of the conservation law, right? And I can write the same thing in 1D that is the specific case of omega being just an interval. Now, this is the flux at the left minus flux at the right. Or it's really rho at left and rho at right.

Right, so in 1D, what we did was we set rho bar of k is equal to the size of the control volume of k. So it's the integral of the control volume divided by the size of the control volume. And basically, by plugging in the definition into the integral form of the conversation law, what we get is 1 over the size of the control volume, the flux at the left side of the control volume-- I'm just going to say F of k minus 1/2 minus F of k plus 1/2. That is what we use to denote the flux at the cell interfaces.

Now, let's go back to 2D, or the multi-dimensional form of the conservation law. We're going to define rho bar of a control volume, k, being the same thing. It is really the integral over the k-th control volume, rho dx, divided by-- in 2D, that is the area of that control volume. All right, let me draw a typical mesh in two dimensions.

So let's say this is a triangular mesh in 2D. And this is like the k-th control volume. Right, for example, this is part of a mesh in two dimensions. Right? And we are computing-- we found the volume scheme. We are tracking the average of the solution inside that control volume.

So this is my omega k. That's the control volume. And the area of that control volume is my Ak. And rho bar of k is the cell average there. So what is the time derivative of that cell average value? Because the area of the control volume does not change. It is equal to 1 over the area times the time derivative of the volume integral, right?

And the time derivative of the volume integral, by the integral form of the conservation law, which you can get by applying divergence theorem, right? We did that last lecture by applying divergence theorem to the differential form of the equation to get this integral form. And just by plugging in, we get 1 over Ak times the integral over the boundary of the control volume. We get a minus sign here. And the outward normal dotted with the flux, ds.

Now, let's look at what this is on the control volume. So we found the integral over the boundary of the volume. If the control volume is this angle, what are the boundaries? We have [INAUDIBLE], right?

Now, we can express that as minus 1 over Ak of summation of ni dotted with Fi times the length of each of the sides. So i is, I'm going to say, like the sides of k. So Sk is the sides, the three sides of the triangle, all the boundaries of this triangle, k. So Sk is a set of these integrals that gives you these three boundaries.

And the i the normal of each edge. So for example, in this direction, so this is the ni's pointing in this direction. These are the ni's. And the graph, the Fi is the flux on these edges, which we have gone through, again, approximately, that by applying the graph function of these two neighboring cells.

It's the same thing as we did in 1D. In 1D, we approximate the flux at the cell interface by looking at the value of the function at the neighboring two cells. In 2D, it's the same thing. We approximate the flux of the cell interfaces, so yield between two cells, by looking at the flux, by looking at the solution of the cell averages over the different sides.

If we have to do [INAUDIBLE], we do [INAUDIBLE]. We look at which direction is the [INAUDIBLE] over this edge. Is it from this side or this side? And choose the value on either side of this edge. If we go to three dimensions, it's the same. Instead of edges, we have spaces between control volumes. That is the case where we really become looking at interfaces, interfaces at each place in three dimensions instead of an edge in 2D or a [INAUDIBLE]. Now, we approximate using a numerical flux that is rho bar of k, rho bar of-- I'm just going to say neighbor. And then, we're done, right? We have approximated the time derivative.

So you know the [INAUDIBLE] of the cell average in the cell k. And the function of the cell average in k and the cell average in the neighbors of the k-th control volume. [INAUDIBLE] other point is the area, just like normal, the length of the k-th [INAUDIBLE], the n-th normal in this space. And the length of the Ses are all known quantities from the mesh.

So this is [INAUDIBLE] flux, a function of the cell averages. So essentially, we have turned a PDE into an ODE. What is that ODE? That ODE is d dt of rho bar 1, rho bar 2, et cetera, to rho bar of the last control volume is equal to some joint function of all these rho bars. OK, so this n is the function that connects the time derivative of one cell average to the value of a cell average of that control volume and its neighboring control volume.

Is it clear? How we do the same thing in 2D? There is a lot-- there is going to be a lot of bookkeeping just because of the mesh and for every cell, we need to be able to find its neighboring cells, and find which edge is connected to which cell, and who is whose neighbor. Conceptually, this is exactly what we do.

OK, so we have started finding the difference, finding the volume, and all these methods are methods that are intended to turn a partial differential equation into an enormous ordinary differential equation, right? And once we turn a PDE into an ODE, we can apply exactly what we have been doing in ODEs to study these PDEs, to discretize, solve, and study the behavior of these PDEs.

OK, so this is going back to the review part. I'm going to really both the rule of the mesh and the outcomes and put particular emphasis on several of the points. The first point is order of accuracy. So the order of accuracy in an ODE scheme is found by looking at the equation, du dt equal to f of u. And we discretize this into some operator. We discretize the d dt into a finite difference using a finite difference operator.

So we discretize the d dt into some kind of a delta over delta t. I mean, this is going to be finite difference operator, operating on the u. Now, this is not going to be-- if you know the analytical solution, if you know what u is, for example, if f of u is equal to lambda u, then you know what u is, right? And you plug this into the finite difference operator. This is not going to be exactly equal to 0. Right? This is not going to be equal to 0.

OK, so for example, if you're looking at Forward Euler, if you're looking at Forward Euler, u of k plus 1 minus u of k divided by delta t, if you subtract by f of u k, it is not going to be equal to 0. If instead of u k you plug in the [INAUDIBLE] solution, you're going to get 0. But if you plug in the real, analytical solution, you're not going to get 0. And the order of that approximation is going to be the local order of accuracy. So if this is equal to O delta t to the k-th power, then k is the local order of accuracy.

Now, here, the usual confusion that happens here is that when you do the truncation error analysis in another way, if you do the truncation error analysis by writing down the tau equal to u of k plus 1, if you plug in the scheme, say, OK, u of k plus 1 minus the right hand side of the scheme, which is u k plus delta t times f of u k, I'm going to get-- tau, for example, in Forward Euler is O of delta t squared. Which, for any scheme, or for Forward Euler, it is because k is equal to 1. Delta t squared is actually delta t to the k plus 1 power.

So why is there a plus 1 when I compute the local order of accuracy by figuring out the truncation error of the update scheme? Well, I get k if I look at the differential. If I look at the approximate, the time derivative could be [INAUDIBLE]. Yeah?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Right. We still need a manipulation. You can base the finite difference approximation of the n-th derivative and the update formula. So how do you go back and forth from this and this? How do you go back and forth from the approximation or the PDE to the update scheme? You have [INAUDIBLE] to multiply these two equations by delta t in order to get to here. You need to multiply this by delta t to get the update scheme.

In other words, the approximation error of this update scheme is equal to the approximation error of the time derivative multiplied by delta t. Therefore, of course I get one more delta t in the truncation error. So that is the reason we ask you to subtract 1 when we figure out the local order of accuracy. If you figure out the order of the truncation error, it's the single step upward. You can go from the truncation error of the single step updates to the truncation error of the time derivative. You divide this whole thing by delta t. And dividing [INAUDIBLE] by delta t, you get O delta t to the k, right?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Oh, yeah, sorry. Thank you for that. Let me use p for the order of accuracy. It is not by purpose.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Ah, sorry. This is k. Those are k's. Right, I confused myself.

AUDIENCE: [INAUDIBLE]

QIQI WANG: All right. Yeah, so the k is the time step number, and p is the order of accuracy. Yeah, thank you.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Any other questions? So that is, I think, the one order difference in the local order of accuracy in that case is a big source of mistakes in figuring out the order of accuracy local order of accuracy analysis. So try to remember that.

OK, so one thing, in the PDE context, if we approximate a PDE by a set of ODEs and solve it using ODEs by [INAUDIBLE], what do you think is going to be the order of accuracy we'd get at the end? How much area are we making in that approximation?

All right, what determines the accuracy if I approximate a PDE by an ODE and solve the ODE using, let's say, a Forward Euler?

AUDIENCE: [INAUDIBLE]

QIQI WANG: The accuracy of the ODE scheme?

AUDIENCE: [INAUDIBLE]

QIQI WANG: And the accuracy of the discretization, actually, both. Actually, both. So if that happens even for some advanced graduate students when they look at the order of accuracy of a PDE discretization, for example, it's quite usual for them to have a plot of the area versus the grid. And often what they find out, as I refine the grid further, I no longer improve my accuracy. So what happens?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yes, because then I'm replacing my grid spacing. And if I'm not also decreasing my cell step size, if I'm not decreasing my delta t, even though my spatial discretization is extremely accurate, has no error, I feel that a truncation error would be proportionate to delta t. So if I don't decrease my delta t, I'm not going to reduce my discretization error further.

So the ODE discretization scheme makes an error. And that error is actually sometimes overlooked when you do PDE analysis. So remember, there is still going to be an ODE component, always, even if you look at PDEs, time dependent PDEs.

OK, so this is the local order of accuracy. How does that relate to the global order of accuracy? So what if I do global order of accuracy? How does-- I mean, it's a very simple answer, so please answer me. How does local order of accuracy relate to the global order of accuracy? Global is one more than local?

AUDIENCE: They're the same.

QIQI WANG: Or are they the same? If it is consistent, yes.

OK, global is equal to local if the scheme is zero stable. That's the Dahlquist Equivalence

Theorem. Yes. Right, so local and global order of accuracy are the same, right? There is no plus one or minus one. The plus or minus one happens over here when you figure out the order of the one-step truncation error. Then you can get local order of accuracy, you need to subtract one from the delta t. But there is no plus or minus one in the global and local order of accuracy.

AUDIENCE: [INAUDIBLE] one or greater [INAUDIBLE]. You can't have like zero order of local accuracy?

QIQI WANG: Oh, yes, this is provided we have a consistent scheme, provided we have at least equal to 1 here. Right, yeah, thanks for that. So the whole thing is under the assumption that p is greater than or equal to 1. I mean, that's usually the case. We usually only look at vertical schemes that are consistent, right?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, the definition of consistency is p greater than or equal to 1, local. That is a consistent scheme. Yes?

AUDIENCE: And the practical way of finding that is the sum of the--

QIQI WANG: The practical way of finding what?

AUDIENCE: If p is greater than or equal to 1 [INAUDIBLE] discrete surface [INAUDIBLE] equals 0.

QIQI WANG: A practical way of finding out if a scheme is consistent is by doing the global truncation error analysis. You have to look at the Taylor series expansion to find out if the scheme is consistent or not. I don't think there is an easier way to find out if a scheme is consistent or not.

AUDIENCE: There was something in the notes about the coefficients of the non-derivative terms sum to 0. Might be a little bit [INAUDIBLE].

QIQI WANG: Yeah, OK, so there is a condition that the coefficients of the constant terms sum to 0. But I think that is a necessary but not sufficient condition.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Right, right. So if I flip a coin of [INAUDIBLE], we're going to get a sum of 0, that is not a consistent scheme.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, the sum to 0 only means that you get consistent approximation of the d by dt equal to 0 equation.

AUDIENCE: [INAUDIBLE]

QIQI WANG: OK, again, if you look at a PDE, the global order of accuracy is going to be determined both by the spatial and temporal discretization. So even though if you look at how the area decays as you refine your grid. You still need to be careful with the time derivative term. You need to refine your time step at the same time. And maybe you have to refine your time step even more if your order of accuracy in time is less than the order of accuracy in space. All right, any questions on accuracy?

And accuracy, if you need to figure out what a scheme is, the order of accuracy is a good way to check what the scheme is. Because different schemes have different orders of accuracy. Forward Euler, Backward Euler are first-order accurate. Trapezoidal rule and midpoint are second-order accurate. And there are more advance schemes. Many of them have an even higher order of accuracy. So that's a good distinguisher of different schemes.

OK, eigenvalue stability. OK, can somebody tell me? We looked at zero stability, right? That is, what makes a scheme have a global order of accuracy equal to local order of accuracy? How does eigenvalue stability, how is that different from zero stability? Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Hm? If the general case--

AUDIENCE: [INAUDIBLE]

QIQI WANG: Right. So eigenvalue stability, in some sense, is an expanded version of zero stability. So it basically says that my solution, Vk, is bounded for the question of du dt equal to lambda u. So this eigenvalue stability, while zero stability says that the solution is bounded for du dt equal to 0.

So when you talk about eigenvalue stability, you have to give me two things for me to determine if a scheme is eigenvalue stable or not. You have to give me the scheme. What scheme are you are using? Are you talking Forward Euler, Backward Euler, midpoint? You have to also give me something else for me to determine if I'm going to have eigenvalue stability or not. What is that?

AUDIENCE: The governing equation?

QIQI WANG: Hm?

AUDIENCE: The governing equation.

QIQI WANG: The governing equation, or more specifically, lambda times delta t. Right? You have to give me these two things in order to determine if a scheme is eigenvalue stable or not. And if you search for MIT math links, eigenvalue stability, the first thing you're going to get on Google at least is this thing. Always run, OK, run. You have to do a bunch of Java things. Come on.

Yeah, that is going to give you the eigenvalue stability. So you need two things. One thing is the scheme, right? For different schemes, you are going to get a different plot of eigenvalue stability. So if I get Backward Euler, that is my stability region. And this plot is a plot [INAUDIBLE] depending on the distance lambda delta t, right? That's the second thing you need to tell me in order to [INAUDIBLE] eigenvalue stability. You also need to tell me lambda delta t, and that determines one point in the complex set.

And depending on [INAUDIBLE] and eigenvalue stable [INAUDIBLE] or the eigenvalue unstable [INAUDIBLE]. Right? Any questions? So questions over there? OK, so yes, tell me a scheme, and tell me lambda delta t. [INAUDIBLE] I'm going to tell you I'm stable or not. And one thing is for Backward Euler, as your delta t decreases, whatever is happening, for the same lambda, as my delta t decreases, I'm more and more zooming into small regions and in all regions. And the smaller I get, the more I'm approximating the stability of the true ODE.

The value of the true ODE is the [INAUDIBLE]. If lambda has a [INAUDIBLE], then that's unstable behavior. If lambda has a [INAUDIBLE] value, then it's stable. Must be stable, right? If I zoom in [INAUDIBLE], that's going to be more and more of what I get.

And same thing for Forward Euler, right? As I zoom more and more into the [INAUDIBLE], that's what I get. And the same thing for trapezoidal. It's really depending on if I zoom in or not, but the same thing for RK2. As I zoom in [INAUDIBLE], I get [INAUDIBLE] stable, right term being unstable. And same thing for [INAUDIBLE]. I get the behavior that on the left hand side is stable, on the right hand side is unstable. Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: For the Backward Euler, I can easily [INAUDIBLE] delta t, and it actually is going to make an unstable equation stable, yes. That actually happens with some PDEs. Now, you can even sometimes, for example, [INAUDIBLE]. And a lot of the fluid flows are actually unstable. So if you're looking at, for example, water shedding behind the [INAUDIBLE], what you see there is an unstable flow field. But you can actually get a stable flow field if you use Backward Euler. You can use Backward Euler with a really [INAUDIBLE].

If you do that, you can actually converge, force yourself to converge to an actually unstable closed solution.

AUDIENCE: [INAUDIBLE]

QIQI WANG: The question is, are you still consistent? The question of consistency is for ODEs. If we find that lambda delta t goes to zero, consistency means the [INAUDIBLE] behavior of the ODE as lambda delta t goes to 0, in this case, that's always consistent because if your lambda delta t goes to 0, and approximating the true behavior [INAUDIBLE]. Consistency has nothing to do with if I give you a really big delta t. If I give it a really big delta t, then it has nothing to do with consistency. Because consistency is behavior of the linear [INAUDIBLE].

AUDIENCE: [INAUDIBLE]

QIQI WANG: Oh, OK, yes. I said converges in a different sense. So the convergence we are talking about here, as I decrease my delta t, as the numerical approximates the analytical solution, what I mean over here is that when [INAUDIBLE] delta t is low, as I've [INAUDIBLE], grows closer and closer to a solution of the time independent equation.

So that's what I was saying when I spoke about convergence earlier. It has a-- I think that this is not a very good equation, but convergence in [INAUDIBLE]. One thing is when I decrease my [INAUDIBLE] time and space, I might converge in the analytical solution.

The second thing is in an iterative scheme, I guess you're going to learn more when you go through more advanced classes. When you apply an iterative scheme, trying to compute the solution to a steady state differential equation, these states are [INAUDIBLE]. The steady state never stops closing. And as I increase the iterations, do I get closer to the solution of the steady state equation? It's more like a convergence in the sense of we can apply Newton-Raphson. We can apply Newton-Raphson to solve the founding equations. That happens to be more [INAUDIBLE] of your iterative solution. Convergence moves, as I do more Newton-

Raphson steps, do I converge to the solution for the [INAUDIBLE] equation?

So there are two completely different concepts of convergence. One is, as I decrease both t and delta x, I get closer to a solution. The second is as I why iterate more, do I converge to a solution?

AUDIENCE: [INAUDIBLE] What about [INAUDIBLE] So what it's saying is that [INAUDIBLE].

QIQI WANG: Right.

AUDIENCE: [INAUDIBLE]

QIQI WANG: OK, I'm--

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yes, so I'm going to draw what [INAUDIBLE]. So if you look at one where there's [INAUDIBLE] over here, that means a lambda that is that type of property, the real part-- and if you look at the analytical solution, the analytical solution here is something that oscillates sinusoidally while growing in magnitude exponentially. Right? So that's an analytical solution.

If they used Backward Euler with a small time step, so that is like when your lambda delta t is going to be 0.1, with a small delta t, you're [INAUDIBLE] lambda to somewhere close to the origin. So you may get a solution that maybe you wouldn't get exactly the right behavior because unless your delta t is infinitely small. But you are also going to get something that grows exponentially larger.

So that is when you have a small delta t. Now, if you use a large delta t, like you are scaling the lambda delta t to somewhere larger along the same line, because the delta t is real, what happens is that we get a stable solution. So although analytically the solution grows larger, you are expected to get a solution that looks more like this. So it is going to get to the wrong answer qualitatively even. And of course, this is because you're using a super large delta t. You're using a delta t that is actually much larger than 1 over the magnitude of the eigenvalue. Right? Does that make sense? Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yes.

AUDIENCE: [INAUDIBLE]

QIQI WANG: So you're asking a very good question. So for a system that is analytically unstable, what is a good way of telling my numerical scheme is doing a good job or not? This is a much deeper question than I can answer. Yeah, there is a [INAUDIBLE]. If the solution analytically is unstable, that means to approximate using numerical methods is extremely difficult. If you make a small error in the beginning, even if you have a small error let's say in terms of [INAUDIBLE], even though we'll assume you are doing everything exactly starting from time step 0, you're still going to get a [INAUDIBLE] error as you come to here. Just because the equation itself is unstable. The equation itself being unstable means you can make a small perturbation over here. That perturbation will [INAUDIBLE] grow larger and larger as we integrate more and more.

So treating a case like that numerically is very difficult. And if interested, let's talk more about that after class. All right, any other questions? OK. And by the way, that's something I'm actually looking at in my research right now. So very good question. I'm very impressed.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, that's why people can't predict the weather, right? I mean, they try to solve a PDE to get the weather seven days later. But you know it's going to be a not very good solution by experience. So that's exactly what they're trying to do, solve an unstable system forward in time for something like seven days.

All right. OK. I got a question of, what is the advantage, what is the main advantage and disadvantage of explicit versus implicit methods? Let's do a comparison here.

You all did the project. So you all kind of know the disadvantage of an implicit method. What is that?

AUDIENCE: [INAUDIBLE]

QIQI WANG: A lot of coding, right? Why lots of coding?

AUDIENCE: [INAUDIBLE]

QIQI WANG: You have to solve a non-linear equation. Solve non-linear equation, right, in every time step. And the way we solve it is using Newton-Raphson. So we have to apply Newton-Raphson equation within every time step. That means a nested loop within every time step. So you have an outer loop that is looping through the time step. Within the outer loop, you need to have the inner loop that does Newton-Raphson iteration. So of course, it's much more complicated than explicit schemes, right? Where you don't need to solve any non-linear equations. That's why it is explicit, right?

OK, but now, what is the advantage of implicit schemes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: It's way more accurate. But I wouldn't hold that as a rule. I mean, in the problem, yes. We used an implicit scheme that turns out to be way more accurate. But the reason may just be our implicit scheme is eigenvalue stable, right? The explicit scheme we were using was [INAUDIBLE]. That turns out to be eigenvalue stable only along the imaginary axis. So accuracy is actually not the main driver of people adopting implicit schemes over explicit schemes.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yes, the main driver is a larger stability region. Larger stability region, as we were just looking at. OK, just for example, compare Forward Euler with Backward Euler. Forward Euler, tiny, right? Backward Euler, the region where it's unstable is tiny, right? That's kind of an extreme comparison, but gets the point through. Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: OK, good point. What happens if Newton-Raphson doesn't converge in the implicit method? And now, when you say converge, you don't mean as delta t and delta s go to 0, right? You mean as my iteration goes to infinity, doesn't converge?

So it's a very good point. Because yes, if you have a very non-linear problem, if you use a super high delta t, it's quite possible your Newton-Raphson doesn't converge. So if you use implicit methods, there is actually an implicit restriction on delta t in which you cannot get through this eigenvalue stability analysis. It is actually a delta t that is going to enable it to converge rapidly with Newton-Raphson equation.

So one thing I think you can-- by just a little bit of analysis you can find out is that as I decrease my delta t, my Newton-Raphson is going to have a much easier time to converge. In fact, if my delta t is very small, then my Newton-Raphson, I'm going to have a very good initial guess of my Newton-Raphson. Because if my delta t is very small, then my next step solution is going to be pretty close to my current time step solution. Right?

The change of the state over the two steps wouldn't be that large. And Newton-Raphson will always converge if your initial guess is close enough. So that's the nature of Newton-Raphson because it uses a linear approximation to get the root of that linear approximation. I can talk more about that.

But if you have a close enough initial guess, Newton-Raphson will always converge. Therefore, by decreasing, if Newton-Raphson doesn't converge, a very straightforward recipe is decrease your delta t. And that is going to make the change between one state and the next time step state closer, and therefore give you a much better initial guess to Newton-Raphson. And that is going to allow you to converge much easier. So yes, Newton-Raphson actually can diverge if you have a super non-linear problem and you use a super large time step.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Oh, yes. By the way, I can make the same argument for [INAUDIBLE]. If I get an unstable system, I can always decrease the time step so that I get into the stability region of the implicit scheme. That is still except for the time step restriction for Newton-Raphson, we said over here, is set by a different mechanism as the largest delta t I can handle up here.

The largest delta t I can use in the explicit step is done by how large, by a larger eigenvalue. It is really-- we can really obtain it from a linear analysis. And linearizing the equation [INAUDIBLE] the largest eigenvalue, I can find out what is the largest time step I can take here. So [INAUDIBLE] the implicit method is governed by if Newton-Raphson converges. And Newton-Raphson will always converge in one step if I have a linear equation. Newton-Raphson will converge in one single step if I have a linear equation.

So the time step restriction here is set by the convergence of Newton-Raphson, not by the eigenvalue step, right? How non-linear the equation is. So there are some integration methods where people actually separate out the linear part that has super large eigenvalues and the non-linear part.

So for a linear part that has super large eigenvalues, they do it implicitly. And for the non-linear part, they do it explicitly. And there's a class of methods for the [INAUDIBLE]. Fancy name, but short for implicit, explicit methods that creates different parts of the equation, either explicitly or implicitly. And as you can guess, the part they treat implicitly is going to be the linear part that has super large eigenvalues. So that you converge in one step, you can do Newton-Raphson, and you avoid the time step restriction set by the large eigenvalues.

All right, OK. Right. So this is really especially in stiff problems. And what is a stiff problem? A stiff problem is actually defined in terms of if you use an explicit integration scheme.

If you find yourself having to use a super small delta t not because you want super high accuracy but because it's still going unstable if you use a larger delta t, then you know you have a stiff problem. That is really the easiest way, I find, to define a stiff problem. That is like, you are forced to use a small delta t by the stability region, not for accuracy reasons, not for wanting higher accuracy, but simply trying to avoid [INAUDIBLE].

OK, so then you get a stiff problem. Any questions? Yeah, so any question, please raise it. Otherwise, I'm going to review Newton-Raphson a little bit, and there is no more things I'm planning to cover. So I'm kind of expecting questions from you guys.

AUDIENCE: Finite difference [INAUDIBLE]

QIQI WANG: Yeah, finite difference, and finite volumes I include in the scope of the exam. As you saw from the previous year's questions I posted, sometimes we include it in the actual exam, sometimes we don't. So it is included in the scope, but because we just did it, I think it's a good idea to review some of the earlier stuff that you may have forgotten. And another benefit is that we are also going over this now that we have [INAUDIBLE] to give you a sense that all the stuff we learned applies to PDEs because what we did in PDEs is just to approximate the PDE using a big ODE. Right?

And you can apply implicit methods on the PDE also, except for the Jacobian. You get it's going to be something close to the matrix form of finite difference, we did in the finite difference vectors. Yeah?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Oh, OK, the example format is that it's closed notes, right? It's closed everything in the period from you get the exam to when you come to our office. So it's closed everything, closed computer, closed cell phone. Closed Wikipedia.

We are actually letting you work out, really think about the problem during the time you are looking at the problem. But even though you don't get the answer, or during our face time, we are going to also interact with you when we ask you questions that you may not have expected, or we may actually help you going through some of these. So that's kind of how [INAUDIBLE] goes. Do you have something else you want to ask?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, right. You can write things down and bring whatever you have written into our office. And in the office, you're expected to use a whiteboard or blackboard and explain to us like you're the professor, the other students, explain to us what you got. Any other questions? No?

The last thing I want to go through is Newton-Raphson method. And it's another sort of confusing-- as I said, Newton-Raphson method is a method that simply solves regular non-linear equations. And a non-linear equation can appear in 16.90. It can appear anywhere else in your future career.

So what you're learning about solving non-linear equations really goes very far, even if you don't deal with numerical methods later on. So as I said, it's a method of solving non-linear equations if the set of non-linear equations is relatively small. Say you have two equations or three equations, you can go to Matlab and use F solve to get it solved like brute force.

But if you have a large set of differential equations you have to solve simultaneously, like what we're going to have if you use an implicit time integration method, apply it to a discretized PDE, then you're going to get at least 100 ODEs you have to-- which, if you have to solve using an implicit scheme, you're going to get hundreds, maybe thousands, maybe millions, or maybe trillions of algebraic equations you need to solve.

OK, so imagine you have to solve f of u equal to 0 where u, instead of writing down a simplified version of what you would get in an implicit scheme, your implicit scheme you would get u minus u k, which is the stuff you already know, over delta t is equal to some right hand side of u, and uk, and maybe something else, right? And instead of writing the same f here, I'm just going to write a big F of u equal to 0. So that big F in this case is really defined as u minus u k over delta t minus f of u, u k, et cetera.

So this is what happens if we solve, let's say, a discretized PDE [INAUDIBLE]. U is the next time-step solution we want to get. And u k is the previous time step solution you already know.

Now, if F is non-linear, you have to find the root of its big F, which is essentially the left hand side minus right hand side of this implicit update [INAUDIBLE].

Right? You need to find a u which is a vector of the solution that makes this f 0 for all the components of f. And f has the same dimension as u. If u is a 3D time step, f is going to be a 3D time step.

And you need to find these 3D numbers that make the [INAUDIBLE] f equal to 0 simultaneously. That's not an easy task. And I think F solve is going to have a hard time dealing with this if u is a high dimensional vector.

Now, what does Newton-Raphson do? Newton-Raphson starts with an initial guess. So u, for example, I'm going to use parentheses to denote the information [INAUDIBLE]. Parentheses 0 is the initial guess. I'm going to set it to u k. I'm going to set it to the solution at the previous time step.

Then, I'm going to approximate this non-linear function using a linear function. Can I approximate F of u by something linear? Let's do it one by one. Let's approximate the first component of F. Think of F having components. We're just going to approximate the first component and have all the other components follow.

OK, so if you just approximate the first component, I'm going to use Taylor series. And I'm going to use a Taylor series of a function of the [INAUDIBLE] variables, all right? So a function-- the periodicity of a function of a [INAUDIBLE] variable is pretty complicated. But the lucky thing is that I don't need to keep all the terms. I'm going to throw away anything that involves more than the first derivative.

In other words, I'm only going to keep the zeroth order term, which involves no derivatives, and the first order term, which involves only the first order derivatives. OK, now what is the zeroth order term?

The zeroth order term is F1 at my initial guess, right? What is the first order term? Actually, in this case, because it's a multivariate function, I have more than one first order terms. If I have as many first order terms as there are many u's, I'm going to be summing from i goes from 1 to the dimension of the system, let me call big N, partial F1, partial u of the ith dimension times ui minus u0i. Right? Does that make sense? That's the Taylor series expansion of F1.

We are only keeping the zeroth order terms as opposed to all the terms. If I start to write all the other order derivatives, it would get too complicated. But I'm not going to write them, so I'm going to truncate them. And I'll approximate [INAUDIBLE] method, it's a linear function. It's a linear function because this F1 of u is probably a linear function of u. But now, I'm truncating. I'm removing everything that happens over here, so if it's quadratic, or cubic, or anything. I'm only keeping the constant part that is independent of u, and this part, which is 0.

Now, I get a linear approximation of F. We all know how to find the root of a linear function? Even though it's a million dimensional? Right, that's the reason we have linear algebra, right? That's why we have matrices. That's why Matlab is called Matlab. That is because we can write functions like this, we can write linear functions in matrix form. And to solve linear equations like this, even ginormous ones, we just use linear algebra, right?

OK, so we are also approximating all the components of F, all the way to the n-th component using Fn u0, plus summation i goes from 1 to N, partial FN partial ui, ui minus ui0. So there are big N equations. We have big N of these variables. How to solve them? How to solve these coupled, N-coupled linear equations? Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, it's just write it into a matrix form, right? First, write it into a matrix form. So I want to try to say, OK, if I want to set all these things to 0, I'm trying to set like 0, 0, 0, 0, is equal to F1 at u0, et cetera, to FN of u0. This is this term. And how about this term? This term is just a joint matrix vector multiplication. The matrix is other derivatives, which is called the Jacobin when I put that into a matrix form.

The first row is my partial F1, partial blah. The first column is my partial blah, partial u1. The last column is partial F blah, partial uN. So each row corresponds to one [INAUDIBLE] the residual. Each column corresponds to one thing in the independent variables, in the u's. And multiplying that by order u1 minus u1 0, et cetera, to uN minus uN 0. Do you all see these linear equations are the same as this matrix?

All right? So what I did is the Taylor series expansion for all these non-linear equations and write them all into a matrix form. And what I'm trying to do is I'm trying to solve for these u1 to uN. How do I do that? I just invert the whole Jacobin. If I call this as J, then my u1, et cetera, uN is going to be equal to u1 0, uN 0 minus J inverse times my F1, FN evaluated at the initial guess.

Right? So this is the solution [INAUDIBLE]. This is one step in the Newton-Raphson, right? In Newton-Raphson, we compute the residual [INAUDIBLE] at the initial guess. And we do the Jacobian. [INAUDIBLE] the Jacobin inverse times the residuals.

Then, you stop at the result from the [INAUDIBLE]. And get your next step solution. And what is the next step solution? The next step solution is the zero of the linear approximation? All right? Which hopefully gives you a better initial guess than u0. And what you do next is you call this set of u's to be u parentheses 1. We call these as u parentheses 1. And then, you linearize again on these.

So the one dimensional analog is this. You try to find F of u. OK, you want to find F of u. You start over an initial guess. This is my u0. I construct my first order Taylor series approximation, which is what in this case? If the zeroth order term is going to be this, this is the zeroth order term. It's a constant, right? It's just F of u0.

The first order term is a derivative at this point, which is going to give you this. So this is my first order term is going to give me the tangent line. My u1 is the solution of the Taylor approximation equal to 0. So this is my u1. And then, I'm going to linearize around this again. I'm going to construct another first order approximation, and go to this point, et cetera, et cetera, until I converge to the zero at the blue line?

Now, it's hard to draw this picture when both u and F are million dimensional. But the concept is the same. I construct a linear approximation. And the linear approximation can be arranged into matrix form using the Jacobin. Jacobin is just a matrix containing all the derivatives of Fu's. And I keep iterating. I keep iterating. I solve the 0 of the linear approximation, linearize again at that new point, and then solve, get the zero of that linear approximation. I linearize again at the new point, I get the zero of that linear approximation, and then linearize again at the new point. Any questions on this?

General technique of solving large systems of non-linear equations. Using the fact that we are already good as solving large systems of linear equations. OK, now I'm all out of things. Just time to answer your questions. Yes?

AUDIENCE: Will we be expected to [INAUDIBLE]

QIQI WANG: No, you don't need to quote anything. You don't need to have your hands on the keyboard.

Yes?

AUDIENCE: [INAUDIBLE]

QIQI WANG: What does big F represent in this case? Big F represents the function of u if we want to get the zero. If I want to solve [INAUDIBLE], if I use an implicit scheme and I want to evolve from this case to the next u, to the next time step, then my F of u would be the left hand side of the scheme minus the right hand side of the scheme, right?

Imagine if I have Backward Euler. Then my big F would be u minus uk divided by delta t minus F of u. That is the Backward Euler. If I trapezoidal, my big F would be u minus uk over delta t minus half of F of u minus half of F of uk. All right? And the difference is implicit schemes are going to get a different value. I have a Backward Euler, [INAUDIBLE], half of that F of u plus half of that uk. And that is in the derivative.

Whatever the scheme is, F is the thing along with [INAUDIBLE] zero that big F is [INAUDIBLE].

AUDIENCE: [INAUDIBLE]

QIQI WANG: It's a [INAUDIBLE], yes.

Questions? After this, we are going to [INAUDIBLE] the midterm.

AUDIENCE: [INAUDIBLE]

QIQI WANG: All right. Anything else I can talk to you about? All these materials? Oh, and just to remind, we are really looking at the measurable outcomes. And don't forget to go [INAUDIBLE] something that [INAUDIBLE]. Analytical solutions [INAUDIBLE] what is the solution of u [INAUDIBLE]? And things like that. And if the solution is stable, when it is unstable, you need to know that and things like that. So this is important. Yeah?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Should d of dt be equal to lambda u? You need to solve it analytically. Huh?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah. For d of dt over the lambda u, you need to be able to solve it analytically, yeah.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Huh?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yes, yes. D of dt equal to au, yes, right. Any matrix, A, right? You need to be able to solve it.

AUDIENCE: [INAUDIBLE]

[LAUGHTER]

QIQI WANG: Yeah. My A can be a trillion by a trillion. And you need to know how to solve it. You are not expected to--

AUDIENCE: [INAUDIBLE]

QIQI WANG: Hm?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah, of course, you won't be able to do the eigenvalue problem of A in your head if A is a 1,000 by 1,000 matrix. But you need to know how to get the analytical solution of the ODE.

AUDIENCE: [INAUDIBLE]

QIQI WANG: Hm?

AUDIENCE: [INAUDIBLE]

QIQI WANG: Yeah.

AUDIENCE: [INAUDIBLE]

QIQI WANG: All right. Anything else? And homework here. If you didn't [INAUDIBLE] homework for [INAUDIBLE], the previous homeworks, you can always come to my office and figure it out.