IntroToLinearDynamicalSystems-Lecture01

Instructor (Stephen Boyd):Yeah, I guess this means we've started. So welcome to EE263, and I guess I should say for many of you, welcome to Stanford. Well, as I said, this is EE263. I'm Stephen Boyd.

And I'll start actually just with some — I'll cover some of the mechanics of the class and then we'll start in. Today is just gonna be sort of a fun lecture, so it's not representative of the class. So by the end of the — you'll leave thinking, "Well, it was interesting, but it was kind of, like, content-free." Anyway, trust me it's not representative of the quarter.

What's that? Oh, wow. Is there finally a room so big that I'm not loud enough? How about you people in the back? Can you hear me? I can be loud. We'll try it. We're gonna go — we'll go natural until you tell me you can't hear me or something like that. Okay.

So let me just say a little bit about the mechanics of the class. I should introduce — at the moment, we have two TAs. That will grow to the end of the day to three and maybe four later. There's Jacob Mattingley over there, Yang Wang is here, and hopefully we'll have another one or two within a day or so.

So let's see. I'm trying to think what the important things are to say about the course that are just mechanics. I'll get into prerequisites and things like that soon. I do have to make one announcement. Maybe it's even legally required. I'm not sure. But Stanford is going to experiment with putting this course in particular online, open to the world. Strangely, this is gonna be after a one-quarter delay. So that means this course will go online everywhere in a quarter.

Now, that's interesting because that means if you ask — well, right now the way it works is this: If you ask a question now in the class and it's a dumb question, then of course, as you know, it's on streaming video. That means that your roommate or mates can go back and replay your dumb question multiple times and laugh, you know. But that's only 30,000 people at Stanford who have access to that. So if we go online though, of course, that's a lot more. And of course, it means it's entirely possible that you'll end up on YouTube. Maybe I will. Who knows? We'll see what happens.

So anyway, but to put you at ease over that, if the camera — I guess they were saying one of the things they have to do is they have to clear copyright and things like that before they put this on, well, open to the world. By the way, the course notes and course materials have been open to the world forever because I control that, and it's always been open to the world. And now I'm actually talking about the lectures.

So what'll happen is I think if you do ask questions like that, rest assured your face will be all fuzzed out and they'll change your voice, too. So go ahead and feel free to ask any question you like is what I'm saying there. But who knows? It's an experiment. We'll see how it works. I'm gonna forget about it. I mean, it's gonna be interesting for us.

The other courses doing this are EE261, a couple of CS courses and things like that. And so we all looked at each other and thought, what have we gotten into? We're gonna have to behave. That's gonna be very difficult. But I think the way that's gonna work is this, is they'll be edited. So if, for example, you are watching this, it's February right now. Maybe this part will all be edited out. But the way you'll know when something interesting happened in the class is ultimately my head will go like this. There'll be a little continuity glitch there, and that'll mean that something we said needed to be edited out. Okay.

So maybe I'll just jump right in and cover the course mechanics. If you can go down to the pad, that'd be great. Okay. So I'll start by just going over the course mechanics. I'll say a little bit in a very broad brush what the topics of the class are. It seems odd to not say what a linear dynamical system is, since that's what the name of the class is. Actually, what's interesting is I'll say what it is, and we won't come back to that topic for several weeks. I'll say a little bit about why you'd wanna study the material in this class, and then we'll look at some examples.

So, first of all, in course mechanics, I should say this: Everything — the course website — oh, by the way, we do not use EE class. So if you wanna go there, that's fine. You'll just find a frame that has the real website, okay? So we don't use EE class or anything like that. It's just a website where you'd expect all the course materials to be, and we try to make the website basically the most accurate source of information. After the website, I would say the next most accurate source would be the TAs; after that, only if you're desperate, me. So if I say something and the web says something different, it's more than likely that what's on the web is right. So just to let you know, what's how we do this.

We also correct typos and things like that almost in real time, and then we deny that the typos ever existed. So just to let you know, that's the policy. So you'd say, "Excuse me. There's a minus sign here." And I'll say, "Nope, no, there's not." And you can look on the web, and all you have to do is refresh. If you refresh, the minus sign will just go away, and everything will be back to right. So that's how it's gonna work. Okay.

So I should add for the reader, the PDF file for the entire reader is available online. In fact, my understanding is that the bookstore charges some idiotic fee. How much did this cost? Do you know? Thirty bucks. See, I think that's kind of ridiculous. But anyway, that's fine. Maybe we'll find another way to print it next year. But the entire reader's available as a PDF file, all the homework, just everything is available on the website. Everything will be available there.

We also fool around a little bit sometimes during the quarter to see how often people are checking the website. So we'll post things that are incomplete, links that go nowhere, just to see, like, we'll do it at 1:00 a.m. to see, like, what — anyway, we did that experiment last year, and I think we got to, like, 1:20 a.m. before somebody said, "Hey, you posted this, but the link isn't there." So we do that just for fun. Okay.

So the course requirements are gonna be weekly homework. So the homework, tentatively we're gonna be on a Friday cycle, so the first homework, which incidentally is assigned, and you'd know that by looking at the course web page, so the homework is actually assigned. I won't even say anything about it. I won't come in and say, "Oh, by the way, Homework 3's assigned." It'll just be on the web. So Homework 1, which is assigned, will be due next Friday.

So we may have to adjust that because there'll be a section for the course, sort of a problem section or whatever, which is not required, but it will also be televised. That means that we have to go to SCPD. We have to find a room that's television-ready. So we don't know exactly what day that will be, and we may have to adjust the schedule a little bit around that. So that's why I think even right now, two places on our website, we say that the homework is due Friday, and another place it mentions homework is due on Thursdays.

When we find a room for the section and announce it, then everything will be set after that. Things like TA office hours will be set. Let's see what else would be set. The section time, all that sort of stuff, will be set. Hopefully, we'll be able to do that in the next day or two. That would be great to get that ready.

For the homework, actually I would — not only are you allowed to work together, but actually I would encourage you to work in groups, in small groups on the homework. And it has to be some kind of group that makes sense for you because a lot of the homework, it's sort of — it's easy to kind of do it. It's actually — if you really wanna know if you understand it, you try to explain it to the other person or two people you're working with. If they're kind of looking at you kind of like this, that means they don't know what you're talking about. And that means — it either means you don't understand it or you haven't explained it that well, so that's a very good way.

Oh, and by the way, that means that when you're playing the other role, when someone you're working with is trying to explain it to you, don't be polite. If what they're saying doesn't make any sense to you, just say, "Interesting, but that makes no sense." So that's how I would — I'd encourage you to do that.

The homework will take a lot of — it should take a lot of time. I don't know what an average amount is, I mean, I don't know, ten hours. I don't know honestly because then you can't ask people. It's sort of a biased sample. And some people try to — they know it'll insult me if they say, "Oh, I do it in 90 minutes," or something, and others go on and say, "Oh, no, it took me 25 hours." I don't know which to believe. So it's somewhere in between those two.

It will be graded very crudely. And I think you just do the arithmetic if you multiply the number of people in the class. These homeworks, they are big, thick things. They will take you a lot of time and effort to do these. It's not gonna — it doesn't count a huge amount into the grade, but this is graduate school and grades don't really matter and all that anyway.

And they'll be graded crudely. So I think the official amount of time is something like 15 minutes. So let's just say you'd take six hours to do the homework — let's just say eight. Someone's gonna go over it for 15 minutes. Now, that might strike you as odd or something. But, I mean, you do the homework so that you can learn. That's actually I believe where you do the real learning is in the homework, so although it doesn't count a lot for the grade, it'll be looked at by a grader briefly very late at night for 15 minutes max. So but still it's very valuable just having done it or anything like that.

What that means is please don't come to us later and say, "I think, gee, the grader didn't get some subtle point in my argument here," because basically, like, I got news for you. The grader looked at this between 2:07 and 2:14 a.m. on Tuesday. So it's likely that the grader didn't, but anyway, okay.

We'll have two exams. These are both take-home exams. You may have already have heard about these. Actually, I'm just curious. How many people have heard about these? Oh, good. And what have you heard? Were you pointing down or — what was that? Oh, that was a down — I thought we had something point down. Now, wait a minute though. But then why are you here? Don't know. Okay, sort of self-destructive instinct? Okay. All right. So all right.

So actually, these are fun. These are take-home exams. They're nominally 24 hours. People rarely take over 20 to do them, but they do sometimes. So they're fun. They're now an institution. We actually even tried to change it, I think, like, last year or something like that. And students from previous years came back to protest to say that, "No, you couldn't possibly do this. It's part of the whole experience," and so on, and so forth. So anyway, so that's how we'll do that.

And we every now and then, I just ask people if they'd be interested in any other format. Like, how about 48 hours? And that, people went, "No." And I think shorter is silly, so that's how it's gonna be.

Oh, I should say that there's something very important here for scheduling. The take-home exam traditionally, and I should also say illegally, is scheduled for traditionally sort of the end of the last week of classes, okay? So there is an official exam date for this course. I forget when it is, December — I'll probably already be in Hong Kong, by the way, at that time. So we do it sort of at the end of the class. That's totally illegal according to the university, but they know where I am. They can come and get me any time. That would be good actually to have posted on the web, would be to have the — I don't know what the police unit for the registrar's office is, but that would be good actually. All right.

So what this means though, this is actually very important. I've already gotten a couple of emails from people who are making flights home or whatever, and basically as early as you wanna leave, I mean, assuming it's not in the quarter, we will work around it. If you're leaving literally, like, the week after classes finish, no problem. You're a beta tester for our final exam, okay? So we'll work around you, just to let you know that.

Okay. This is all just mechanics. Any questions about the mechanics, how it'll work? Did I forget anything? The only thing I might have forgotten is there won't be a section this week because, well, we don't have a room. So next week will be the first week there'll be a section . We don't know what day it will be or where it will be, but that will be announced on the email list, and it will also go on the website. Okay.

So now we've covered the mechanics. Let's go a little bit into — I'll say a little bit about the perquisites for the class. That's actually important. So the first is that you should have had an exposure to linear algebra. Now, these words are actually chosen very carefully. You typically in this class, there are people who have a very — it is a very wide range of backgrounds in linear algebra, all the way from essentially none from people who said, "Oh, I took a course on multivariable calculus. I think I know what a vector is and a matrix," all the way to people who've taken multiple courses on linear algebra.

In fact, what you really need is just something like an exposure to it. So, I mean, you definitely should have seen vectors, matrices, hopefully ideas like rank, range, null space. However, since linear algebra classes are by tradition extremely boring, it is natural that you should have either hated these courses and actually suppressed the memory of those courses as much as possible. That's natural. In the first part of the course, we will be going back over this, and that means of course that painful memories will be coming back, but there's lots of us here and we're all doing it together. So everything will be fine there.

I should also say one other thing here, something different this year — actually, it started last year. This course is now offered twice in the year. It's offered fall, but it's also offered in the spring, and that actually is very important. It means that if you decide somewhere into the class, a couple of weeks in or I don't know when, couple weeks in that in fact what you'd like to do is actually take this in the spring and maybe take something like Math 103 or — I don't mean really Math 103, but if you could just look on the course catalogue and see what's involved there — if you wanna defer, just sort of take that and then take this in the spring, that's an option for you. That wasn't an option in the past, and that was a problem because people sort of didn't drop out, saying they — but having said that, a lot of people actually just do fine.

I should also make a comment to those who come in with a much stronger background. So there are people who come in with a much stronger background in linear algebra who've had maybe an entire course or whatever several years ago, so that's fine, too. Actually, there the reaction will be that somewhere around the fourth week, you might be thinking something like, "When am I actually gonna learn anything I don't already know?" Actually, trust me, you will because this is different from the class you took, I promise. And in fact, all those people actually come back to me later, not all of them, but most of them come back later and say, "You're right. I learned something." So okay.

The only other real prerequisite, and this is not even really a prerequisite that much, is at just a few tiny places in the course, we're gonna use the Laplace transform differential equations. This actually, even if you hadn't seen this, but it would be difficult for me to believe that you haven't, again, we'll cover all the background material needed for this. So that's really the formal prerequisite. There's not a whole lot else.

Now, the course, this material — I'll talk about this soon — is used in tons and tons of areas. But in particular, you do not need to have taken a course on, like, control systems or something like that, which is one area. You actually don't even have to have taken any course on circuits and systems or a course on dynamics. So you could be — it would be fine.

We will look at examples sort of taken grossly from control systems, circuits and systems, dynamics, but we'll also look at examples from machine learning. There'll be examples from signal processing, communications, networking, all over the place. We'll take care that you don't actually need to know anything about these application areas. I mean, these are really more like little vignettes where you just kind of oversimplify it and show it here. So don't worry if you see things like that. In fact, today you will see things like that. Don't worry about it.

Whenever it matters, we'll make sure everything is — so the point is that although it's perfectly okay for you to have had a course on control systems, circuits and systems, dynamics, for that matter, machine learning, signal processing, it's absolutely not needed. And I know every year we have people in this class from, for example, economics, from all sorts of areas. Doesn't make any difference at all. Okay.

So let me say a little bit about the outline. The first chunk of the class is basically gonna be a review of linear algebra and applications. I'm gonna sort of assume that you've already had a class where somebody droned on and on about rank and range and the four fundamental subspaces and things like that. So this will actually be about modeling and applications. So it's actually where does this actually come up; where do you use this stuff? And that's actually the theme of the class.

Let's see. Then we'll talk about autonomous linear dynamical systems. I'll actually say what those are shortly. Then we'll move onto systems with inputs and outputs. And then we'll look at sort of basic quadratic control and estimation at the very end of the course. But in fact, this is really an application if you've understood the material before. Okay.

So maybe at this point, it's time for me to say what the class is about, although we're gonna — we'll drop this topic and we'll come back to it only in about three or four weeks. So what's a linear dynamical system? Well, a continuous time linear dynamical system looks like this. It's a vector differential equation, so it looks like DXVT is A of T. That's a matrix times a vector, X of T plus B of T times U of T. That's a vector, that's a matrix, and I'll talk a little bit about — oh, this reminds me.

On the course website, there are some extra notes. There's quick notes on matrices. You can just read them literally in 30 minutes. Please take 30 minutes and read them because, first of all, it's gonna say that these are the things I will use without ever — I will not go into them. I won't mention it. And the other thing is if the notation you saw was slightly different, this would set the notation straight. Actually, I kind of try to use sort of a high BBC level of mathematical notation. So if you saw a notation that's substantially different, that's because what you saw was weird and strange. So maybe it was in some other weird field or something. Throughout the class, I'll make fun of other fields periodically. No, I don't think I get to do it today. So okay. [Inaudible].

Okay. So here typically, T, as the choice of symbol suggests, is gonna represent time. Of course, it need not, but it will represent time. Here, X of T, that's this vector. Actually, it's a vector function, and that's called the state. That's a vector. Sometimes colloquially, the actual entries of X will be called informally the states like that. So that would be the [inaudible] state, but that's slang, and you should know that. That's XI of T. U of T is called the input or control. It depends on the field you're in. We'll talk more about that. And Y of T here is called the output.

I think this equation is sometimes called the dynamics equation, and this equation is sometimes called the measurement or readout equation. It's got all sorts of names. And then all of these matrices here have names.

You don't have to remember any of this because I'm going over this just so that you can't — otherwise, we'd go four weeks into the class and if someone asked you how's the class going, you'd say, "It's great, but we haven't even gotten to what the title of the class is." So okay.

So this matrix A of T is called the dynamics matrix. Here, B of T is sometimes called the input matrix. C is called the output or sensor matrix. And D is called the feed-through matrix. We're gonna come back and go over this again in horrendous detail when we really get to this. So you don't have to remember any of this or all that sort of stuff. Okay.

Now, this is too ugly, so it is often written in this very simple form that looks like this. It's X dot is AX plus BU, Y equals CX plus DU, just like that, where you suppress all sorts of things and have them understood.

So a linear dynamical system is nothing but a first order linear differential equation, nothing else. Now, there's other names for this. These are called state equations, dynamics equations. They have all sorts of names. It depends on the field you're in or what application area you're in.

I should also mention that A, B, C and D are traditionally — you can usually tell where someone got their PhD and so on and so forth by their choice of notation. For example, I think there was someone who taught at Stanford in Aero/Astro for a long time, and that was X dot equals FX plus GU. So turns out that's because he got his PhD at MIT in 1967 and picked this up from somewhere. So you will see other conventions here. I mean, of course, it's nothing but notation, but just to warn you that you'll see that. But if you ever see someone who writes down X dot is FX plus GU, it probably means they went to MIT or something like that or took the class from someone who went to MIT. But now they'd switch to this, too, anyway, so it's complicated.

Okay. So let me mention a couple of things. Many, but actually not all, and we'll say something about that later, are time invariant. So that means that these matrices A, B, C and D, they're constant. They do not depend on time.

Now, if there's no input, that means so there's no B or D matrices, the system's called autonomous because essentially it goes by itself and has nothing to do with what U is, which is usually interpreted as some kind of input or something like that. And very often, there's no feed-through, so you get things that are very, very simple.

Now, some notation you'll hear is this: If these inputs and outputs are scalar, the system is called single input, single output, and I think the slang for that is SISO. When you have multiple ones, it's MIMO. Now, this is a bit silly, frankly. I mean, this is kind of a holdover to when it was really a big deal to have, like, two inputs and two outputs. Even other fields, like communications and things, signal processing, are now getting used to the idea that you typically process more than one signal at a time for more than one measurement or for more than one input. So, in fact, I guess wireless communications is going through the very end of what I'd call the MIMO stage of development.

So this was very hot ten years ago, and it was a big deal, and people would get all excited, and you'd say, "Wow. What's that?" And they'd go, "It's amazing. It's totally amazing. Instead of holding up, like, one antenna and looking at the radio signal coming from it, are you ready for this? We're gonna hold up two. Unbelievable. Can you believe this? Got it? Actually two. And we'll take those two signals and by processing them right, we'll increase the capacity of our cell network." Anyway, so other fields have been there, done that decades and decades ago. There are still fields that haven't, by the way, reached the MIMO stage. They will. It'll be happening soon, sooner or later. Okay. So okay.

Let me say something about discrete time. This is what we looked at so far was a continuous time. Let's look at discrete time linear dynamical system. That's nothing but a recursion. So here, instead of a derivative, it's simply an update equation. It says that the next state is obtained by multiplying the current state by a matrix A of T, and to that you add something which is related to an input. Here, the time is an integer. So it's a discrete time thing.

And sometimes the time is either called — it could be a sample; sometimes people call it a period. For example, in economics, you would talk about periods. These could be trading days, could be anything, okay, or this could be some audio signal processing, in which case these are samples at some standard rate, like 44.1 kilohertz or something like that. Now, in this case, the signals are sequences. In the continuous time case, signals are functions, and I'll say a little bit about that. So it's nothing but a first order vector recursion.

Okay. That's done. What I just explained, there's no way anybody would have gotten any idea of what they are. But at least now you cannot say I didn't tell you what a linear dynamical system was on the first day. I've done it. We're gonna come back, and we'll be going into this in horrendous detail later in the quarter.

So let me say a little bit about why would you study linear dynamical systems. Well, it turns out the applications, they come up in — nowadays, it's everywhere, absolutely everywhere. So, I mean, sort of historically, the first application was in automatic control. This was aerospace maybe in the '60s. That would be the first sort of real application. Now, at the time, it was super advanced technology, very, very fancy.

It hit signal processing, oh, let's say, mainstream signal processing, let's say, about 15, 20 years ago is when it hit signal processing. So until that time, signal processing — and indeed, I bet your undergraduate exposure to signal processing just fiddled with a scalar signal.

Actually, for how many people is that the case? Like, how many people took signal processing? Okay. And for how many of you did you ever hear of a vector signal? Cool. Really? Where? Cool. Okay. There were just a few others. Where are some of the others who heard of vector signal processing? Where? Cal Tech. Well, okay. Times are changing. That's really cool. Okay. Great.

So what you will find actually is except for undergrad signal processing, almost all signal processing now involves linear dynamical systems in one way or another or something like it, all of it. So all signal processing pretty much involves ideas from linear dynamical systems.

Communications, I'd say it hit big time about ten years ago, although in fact in communications, there was always some stuff that went back into the '40s and stuff like that. So communication's another area.

Economics and finance, it's totally basic, if you look at evolution of an economy or something like that. It comes up all the time. In finance as well, most of the models are just linear dynamical systems. Some of the notation will be different, but it's very close to what we'll be looking at.

Circuit analysis, so it may not be linear dynamical systems, but linear dynamical systems actually plays an important role, very important role in circuit analysis, circuit simulation and also circuit design, it comes up a lot.

It comes in mechanical and civil engineering. You see it in things like dynamics of structures and all sorts of other stuff.

Aeronautics, it's everywhere. It involves dynamics, control, navigation and guidance, so, for example, GPS. So things like this are all done using this, well, the stuff you will see in this class.

A few other things you won't see, which reminds me about something I didn't say. Here's a topic that maybe should be a prerequisite but actually is not, and that's actually probability and statistics. So I think that's kind of weird that we don't have it that way, but it's just it's not a prerequisite, so just to let you know.

Occasionally, I might refer to things that involve probability and statistics, but technically that is officially just a side comment. That's just for those who know what I'm talking about. Actually, if you combine the material of this course with sort of probability and statistics, now you're getting close to actually what is used in tons and tons of fields [inaudible]. By the way, it comes up in other areas, like machine learning, as well, is another one. Okay.

Now, the usefulness of the material, it scales with available computing power, which thanks to other people in EE and materials and areas like that, has been and is and will continue to scale by Moore's Law. So that's great because that's sort of the difference between this material in 1963 and this material now. It's a huge difference.

Now, you can actually — in 1963, you could do this stuff, and there were people who did, mostly militaries. Now, anyone can do it. It's widely used; it's widely fielded. And that's basically entirely due to increases in computing power. The computing power's used for analysis and design, but it's also used for implementation and actually just imbedded in real-time systems. And occasionally, I'll make some comments about that.

And I do wanna say something about how courses evolve, so especially sort of courses on mathematical-type, engineering-type stuff. So visual signal processing, I guess maybe the first class ever given was maybe at, like, MIT in, like, 1956 or something like that, and it was this super-advanced class with maybe, I don't know, four PhD students in math in the class. And sort of for a decade or two, it was this ultra-advanced course. It was ultra-high-end technology that the only people who would field this would be, like, the military and a couple of others, maybe some banks or Boeing or something like that would do it.

But DSP, as you know, that's now, like, an undergraduate topic. I mean, it's just everywhere. I mean, so that's how that went from super-advanced, advanced PhD-level class, 30 years later, again, thanks to technology, 30 years later, it's now, like, your basic — it's your second course. In fact, in some places, they're switching it and they're making it the first course in undergraduate Cs in, for example, electrical engineering, which I think is a good idea, right, because soldering skills are, well, they're useful, sure, but maybe less so than moving forward — I can solder by the way, just if you're thinking — if you're curious. Okay.

So one of the origination in history, I can say a little bit about this, so part of it, you'll trace to the 19th century, and you'll hear names, they're names of 19th century mathematicians. So linear algebra itself, that all goes back to 19th century, early 19th century even. But it sort of blends, sort of classical circuits and systems, this would be from 1920s on. This would be the stuff done at, like, Bell Labs in the '20s. So it kind of combines that with linear algebra. So that's what the modern form is.

Now, the first really — the first time you could say here is linear dynamical systems actually either being taught or being used or whatever is actually aerospace in the 1960s, and that would be the first time it was widely used. It was used for — at that time, well, I think you'd have to say it was not used for what we would call socially positive purposes.

It was used to land missiles and things like that. So that was aerospace in the 1960s, very, very fancy, fancy stuff.

But between then and now, somewhere in the '80s, it transitioned from a specialized topic, which 12 or 15 PhD students would take, to one that basically touches all fields, and not just all fields in EE, but also other fields in optimization and finance, in economics and things like that. So that was sort of the transition time. And I've already said the story of digital signal processing, DSP, is the same.

Another story, you can go back down to the pad here, is information theory. That's the same story, that information theory, as you know, was more or less created by Shannon in the late '40s, in fact also by Kolmogorov in the Soviet Union and Moscow in about the same time, maybe even a bit earlier in that case. It was created there, and the people who actually did sort of communications just fell down laughing, saying, "Is this a joke? We could never use that." Like, "Get out of here, Mr. Theorist. We actually have work to do here." So, of course, the joke was on them because you propagate Moore's Law forward a few decades, and you'd be surprised at what you can do now.

So information theory is now not just a super-advanced esoteric topic. It's a topic, like, basically everybody needs to know about, and it is widely used. I mean, it's still also a vibrant — there's a vibrant theoretical core. But it's not just a weird thing done by the 15 PhD students who make this a topic. It's something like everybody ought to know. Okay.

I'll say one more thing about this. I'll say a little bit about non-linear dynamical systems. You hear a lot about this, and I'm deeply suspicious of these people for many reasons. I'll tell you why in a minute, but — so it is absolutely the case that many systems are non-linear. And not only that, non-linear dynamical systems is a fascinating topic. I don't deny it.

So if this is the case, why should you study linear dynamical systems? Well, I'd make a couple of arguments for it. The first is that it turns out that most methods for non-linear systems for engineering things that work for non-linear systems basically are based almost entirely on ideas from linear dynamical systems. And in fact, there's this weird thing where you design things based on linear models that are not even remotely accurate.

And unfortunately — I'll explain why I say unfortunately — unfortunately, these things, more often than they should, work. See, I don't think they — I'll tell you why I think they shouldn't: because it reinforces kind of cowboy engineering is what it — it's kind of like you walk up to someone and say, "What are you doing?" You say, "I'm designing a regulator for this thing." "Is it linear?" "Oh, no, not even close." You go, "But what are you doing?" You say, "I'm designing this. I'm pretending it's linear." So every now and then, something like that should blow up. I mean, just look, I feel — maybe just singe the person who did it a little bit.

But the point is just to send a message, which is you should kind of know what you're doing or at least respect — or at least when it works, step back and say, "Wow. That's cool because it didn't have to work, but it did." But unfortunately, these things just often work even when, like, the models are way off. It's known a little bit why that's the case, but it's just a reason to study linear systems.

The other one is that actually if you're really interested in non-linear dynamical systems, then it turns out if you don't understand linear dynamical systems, I mean, this is basically — this is the big prerequisite as far as I'm concerned. And the other thing is, it's funny, you get people who — there's like a little cult following. This would be the chaos crowd and you can go to the Santa Fe — this place and people will tell you all about this stuff. I can fool any of them. Give me five minutes with any of them. I can fool any of them. They'll say, "Oh, no. Linear systems is not interesting. You can't have bifurcations. You can't have chaos. You can't have —" blah blah blah. I mean, it will take me five minutes to put together a linear dynamical system that will totally fool them. And they'll go, like, "Oh, yeah, that's chaos right there. That's chaos." But it'll just be some linear system. That's just my ideas. You'll hear about this, okay? And actually, it's really interesting stuff, but okay.

So that finishes up my kind of overview of the course. Are there any questions? Kind of be hard to have a question because I didn't say anything. But there might be a question anyway. I guess not. Okay.

What I'm gonna do now is the rest of this lecture I'm actually just gonna go over some examples. Now, this is ideas only. There'll be no details. I do not expect you to understand any of it. All of this, we will come back and do later in the class, all of it, in horrendous detail. So don't try to get everything now. Okay. So what we're gonna do is just look at — here's a specific system.

It's X dot equals AX. Here, the state vector has dimension 16. And we have a scalar output, Y of T. So people on — the slang this would be a — this would be a 16 state single-output system. Now, it turns out this is a model of a lightly damped mechanical system, but it really doesn't matter what it is for now.

But if you want a physical picture of what this is, it is a mechanical system with sort of eight generalized positions, eight generalized memento or something like that, so it's some kind of flex structure. And the dynamics is what happens when you poke it; for example, if an earthquake excites it. And then the whole thing kind of wiggles around. And the output would be the specific displacement in some axis at some point in the structure. So that's what — if you want a physical picture, but you don't need one.

So here's sort of a typical output looks like this. And the first shows what it looks like on a short time scale or shorter time scale, and then this is what it looks like on a longer time scale. Now, the point about this is this looks very — it looks quite complicated. In fact, if I showed you, let's say, just the first 100 samples, just that right there, if I showed you that and said to you — I didn't tell you where it came from, but I said, "Here's a signal

I've observed. Would you mind predicting it?" Now, if I asked you to predict it one or two seconds or whatever the time unit is here in the future, you'd say, "Yeah, no problem, I'll make a prediction."

But what if I said, "Would you mind, having seen this, predicting what this'll be doing at T equals 1000," right? You would hopefully look at this and say, "Are you insane? I mean, I don't even know what that thing is. It's weird. I mean, it's some weird thing. And even it looks like it's changing character over the 50 — let's call it seconds — over the 50 seconds you've shown me. How on earth would I be able to predict what it does 1000 seconds later?" Everybody see what I'm saying here? Okay.

For sure — this is not long enough, but I could get one, and I could definitely get one somebody at the Santa Fe Institute to go for chaos here, no problem. They could say, "Oh, yeah, I'm seeing period doubling, bifurcation, it's all there." Okay.

So the point is you could make all sorts of stories up about this. You'd look at a longer time scale. You could say, "Well, it started off chaotic, but now you can see there's some kind of regularity emerging from this." And you could make all sorts of stories.

So out here, in fact, it would seem reasonable for someone to predict if you saw this what's gonna happen for another 100 seconds or so. It would seem reasonable because some kind of periodicity or something or pattern has set in. Everybody see what I'm saying? Okay.

Now, this is hardly surprising, but it turns out — and this will be familiar to you from the idea of, like, poles and things like that, that this output and indeed the output of any linear dynamical system, it can be spectrally decomposed. It can be decomposed into modes or frequencies, complex frequencies, and the exact one I just showed you, actually, we can write it out this way. There are eight of these. And if you were to add these eight signals up, you would get this one here, okay? Now, this would hardly be surprising to you. I mean, you've sort of seen this maybe, or some of you have seen this in other courses and things like that.

But there's actually — once you know that this is the case, it's a totally different story. If I were to take this signal and decompose it like that, and I now said to you, "Please make a prediction about what the signal will be doing 1,000 seconds in the future," you would probably say, "No problem," because each of these is just a damped sinusoid, and you have a few parameters to fit for each one — that's something else you'll learn in this class — you'd have a few parameters to fit, at which point you could probably quite easily extrapolate it, okay? So again, this is probably not too — it's not too shocking. Okay.

Now, let's look at input design. Input design goes like this. I'm gonna add two inputs. So I'm gonna add two inputs. Now, by the way, if you are thinking — if you're visualizing a mechanical structure, and we're gonna have two outputs, by the way, so you can imagine a mechanical structure, and I wanna put, like, let's say two piezoelectric actuators on it. I mean, just if you wanna make the time scale be a whole building, let's put two hydraulic actuators on it or something like that. But let's make it two piezoelectric actuators on something this small, that sits here. I have two actuators, and then I measure the displacement of the thing at two other places, okay?

So I have now a two-input, two-output system. You should have a very good idea of what happens when you put a force or a displacement input onto a structure. It will start shaking because you'll excite various modes. And eventually, if it's got damping in it — it does — it will sort of come to rest. And those two points will have been displaced if the input has a constant value or something like that.

So here's the job. We wanna find an input that brings the output to some desired point, 1 minus 2 — this is totally arbitrary because you don't know what A and B is; it's probably not relevant what the output is. So you wanna bring this output here. And so your choice now — this is a generalized — this is, like, a design or a control problem. It's choose an input or action that makes something you want to happen, happen, okay? So that's what you're being asked to do.

By the way, this is quite a realistic problem. Any disc drive has stuff like this in it where you get a command for the head to move from Track 22 to Track 236 and to do it and be tracking in some very small number of milliseconds, okay? So this is quite realistic. If you change the time scale and all that kind of stuff, this is actually very realistic, okay, the problem. Okay.

So here, let's just look at the simple approach. The simple approach is this: Let's assume, if you wait for the whole thing to come to equilibrium, that means everything's constant. That means X dot, which is DXDT, is zero. So you have zero is AX plus BU static. I'm putting in a constant input, U. By the way, don't try to follow this. We're gonna go over all this later. And then the output is also constant.

You can solve these linear equations by simply putting AX on the other side, so you have AX equals BU static minus BU static. You multiply by A inverse on the left. But I won't go through the details.

But what you find is you can actually simply work out a formula for what the optimal — what the input should be, not optimal, the input. The input required is this: So let's imagine then that these are, say, piezoelectric displacements. So one of those piezoelectric actuators should pull in, and the other should push out. And by doing that, that will twist the structure in equilibrium so that whatever the — one is displaced, let's say 1 millimeter to the left, and the other is 2 millimeters to the right or whatever this means, okay? So that's the idea. Okay.

Let's just go ahead and apply that input. Let's see what happens. Well, this shows you a little bit of negative time. The Us are zero. And [inaudible] to equal zero, we simply apply this thing, so you have a structure and it's equal zero. One of the piezoelectric actuators shrinks instantly; the other one pushes out. And of course, that causes everything to start shaking in this structure, so everything shakes. And this is what the outputs do. You can see they shake for a while, and about 1500 time units later, they are converging indeed to what they are supposed to: 1 and minus 2, okay? So there it is.

And you could now make all sorts of arguments about this. You could say, "Well, look, that's the basic dynamics of the system. I mean, come on, the thing shakes. It takes whatever, 1500 seconds or picoseconds depending on what application you're thinking of or microseconds. It takes 1500 microseconds to just get the energy out of the system, to have it dissipate, so how could it ever be any faster?" Okay?

Well, later in the class, we'll look at problems like that, and it turns out you can do a whole lot better. So here are some inputs. Here's one, and here's what you do at one actuator, and here's what you do at the other. And this one, I mean, this is really pretty bizarre. This ends up pushing at minus 0.6, but the first thing it does, it goes in the opposite direction; same for this one. Then it goes through some very complicated dance here, including some little squiggle at the end, okay? So those are the two inputs. Don't worry about how I got them, okay?

This is what happens. So what happens is the outputs kind of wiggle around and 50 seconds later, they hit their target values and they just stay there. So you get exact convergence in 50 seconds, 50 picoseconds [inaudible]. It depends on what your application is. Everybody see this?

Now, by the way, the time scale is this: I mean, maybe I need to draw where that is. That's about right here, okay? That's what happened, okay? So that input basically gets you convergence — first of all, it's exact convergence, but it gets you convergence something like 20 or 100 times faster, so something insane like this, okay? Everybody see this?

Now, notice that the input you're using is not particularly bigger. It's not bigger at all. I mean, the final input you have to put in is minus 0.63. You never really go outside that range much. I wanna say something very important about these two input signals here. No one can look at those and say, "Oh, yeah, I was just about to try that." Okay. There's no way, okay? So this is kind of what this class is about.

So, I mean, you would not have arrived at these two inputs by trial and error, okay? You would not have done this with a proto board in your garage and a box full of different resistor values, let me assure you, okay? You can get someone who's pretty good at wiggling, fiddling with dynamical systems, like a fancy pilot. And a pilot wouldn't do this either, okay?

So now I'll tell you the end of the — I mean, so this is pretty wild. By the way, things like this can be repeated in many, many fields. You can repeat this in finance. You can repeat it in lots of others. These things will beat the hell out of anything you can do, I mean, if it's set up properly, intuitively, okay? So this is kind of what this class is about, is this kind of stuff.

By the way, I will tell you this. This stuff right here, later in the class, by the middle of this class, maybe a little bit beyond the middle of this class, if I show you, you will — I wouldn't even ask you to do this on a homework problem because it would be too insulting to you because you would know — if I asked you, I'd say, "Look. Show me how to find the inputs that move the state from here to here as quickly as possible, and do it in, like 50 whatever or something like that," it would just be insulting. You'd just say, like, "Oh, please." It would be totally obvious to you.

But I want to point out right now — by the way, if there is anyone whose intuition was good enough to think that this was the right thing to do, please speak to me after class. But this is not obvious, okay? This is not intuition or anything like that. In fact, a very important skill in the class is gonna be to explain things like this. What'll happen is this will just come out, and it will be impressive. It'll be in this one — it'll be in other areas. It'll be in signal processing. It'll be in image processing. It'll be in machine learning. Very impressive things will happen.

The problem then is gonna shift not from doing this or knowing how to do this, but to explaining it to somebody. And I'll spend some time periodically through the class explaining how to do this. You can make a very good story about it here. You can look very smart. The truth is you wrote about four lines of code to do this. That's about what it is.

But you can get a lot of mileage out of it. You can say, "It took me a while before I realized you really had to start in the other direction. That's the key, you see, was starting in the other direction. Then for a while, I couldn't get this middle section right. That was tough. Then I realized finally, this one should bump down, sort of like it's going to the final value. But no, it's just to fake out the system because then you come back up here, you see." All right.

So anyway, that's gonna be very important, okay? You'll see. You'll see. That's an important skill. Maybe we'll put that on the final exam, something like that, or we'll have a section where you have to — you will have done something cool in maybe image processing or something, and then after that, you have to explain it to someone and we'll grade it on how plausible it sounds. Okay. It'll all be made up of course. You did it by writing five lines of code. Okay. All right.

Now, it turns out here, you say, "If you can get exact convergence in 50 seconds, how about, like, 20?" Well, you can do 20 as well. And now, if you go to 20, here's what's weird. You now are using a bigger input, okay? And now, by the way, I'll show you something cool. The first thing you do has now shifted again. Now, the first thing you do is you go down, which is gonna be the ultimate way you're going finally, and here you kind of go up, and anyway — so then again, you'd have to make a story about that. And you'd say, "Well, that's totally clear." Why? "Because now you're doing it faster, you see, and so —" anyway, you'd make up some plausible story about that, which the gullible would buy. Okay.

So this raises questions like, how do you do that? How do you find something like that? And how do you analyze this trade-off between the size of U and the convergence time and things like that? So you will know all of that. I mean, this will be, as I said, this will just be so obvious to you six weeks from now, it'll be insulting to even ask. Okay.

We'll look at one more example. This one is from estimation and filtering. So here it is. And this is, you know, if you have a background in signal processing, fine; if you don't, like, if you're in some other department, don't worry about it. So we have a system. It looks like this. It might be part of a communications system, for example, and it's gonna work like this. An input comes in. It's piecewise constant, and we'll make it a period of one second. But if it's a communications system, this might be nanoseconds or something or picoseconds or whatever. It depends what this is, maybe nanoseconds or something.

So an input comes in. It gets filtered or convolved with a linear filter. That's a second-order system with a step response that I'll show in just a minute. Again, if you know this material, great; if you don't, don't worry about it. We'll come back to this. So this is essentially a smooth version of this.

And now the interesting part comes. It's going to be very crudely quantized. It's gonna be quantized with a ten-bit — a three-bit quantizer, okay? There's only eight distinct levels that it knows, okay? So this thing will run at zero hertz, even though the input signal is constant. So here's a picture that just kind of shows how this works. So the input here is zero, and then it's got some value, but it's constant for each interval, which is one second long, okay?

Now, this runs through a filter with this step response, and those of you who remember undergraduate signal processing, if you took it, will know that this means that the impulse response is sort of positive and has a little wiggle here, but you'll look at this and you'll know what that means is it smoothes things on the order of a second or so, or maybe two, or maybe one or two, and it delays things a bit as well.

And so here's this signal after smoothing comes out here, and you can sort of see a bunch of things. I mean, it's zero here. We keep this zero just to make it — so you can see the beginning, and you can see this goes down. This goes down, sure, and it starts going up, but you can see that the individual levels have kind of been smeared out and lost, okay?

And if you know something about communications, you know this is basic communications. Basically, if you're transmitting, like, a signal with multiple layers, levels, and it's coming out where you're reading it nice and clean, it means you're not going fast enough. That's what it means. So it means you should crank up the rate until things start getting hard to guess. Maybe this is too hard to guess because now you've kind of smeared things out. But you can see things. It's a few positives here translates into this kind of big bump here, but if I gave you this, it's not an easy way to estimate what this is.

But it's gonna get worse because the next thing we do is we take this, and we quantize this to eight levels, so that's three bits, and that gives you this sequence here, okay? Each of this is just three bits. And now comes the problem. From this, we want to reconstruct and estimate that. That's the problem, okay? And that's a basic problem in communications. It basically says, let's look at what our receiver, our A to D or whatever it is, is sending us, and you wanna estimate sort of what was sent at the transmitter because from that, you can decode what the message being sent was, okay? So this is sort of very basic in communications or anything like that.

So the basic approach would be something like this. You ignore the quantization because it's — actually you just consider it some imperfection that is beyond your control. And what you'd do is you'd design an equalizer for this. Now, an equalizer is another convolution system, and it has the property — and what you want is you wanna kind of undo, or I guess in, for example, geophysics, this would be called deconvolution, so that would be another area. It'd have lots of names in different fields. You wanna sort of undo the effects of the filter. That's H. So you really want one so the GH is about one. Again, if you know that this means, great; if you don't, that's fine, too.

So once you find such a G, that's called an equalizer. In communications, it would be called an equalizer. And you'll simply apply that to the receive signal to approximate U. If you do that here, it will work hideously. It just simply won't work. You won't get anywhere near close to estimating these quantities. And then if you went back to someone who does traditional signal processing and say, "What's wrong?" and you'd say, "Well, come on, give me a ten-bit A to D, and we're talking. But a three-bit is not gonna do the trick." Everybody see what I'm talking about? So okay.

Well, now it turns out this problem can be posed as a basic problem you'll see in this course, again maybe around the fifth, seventh week, something like that, in which case it'll be just completely standard for you. This one could be a homework problem, I mean, just to set it all up and do it. Maybe it will be.

And if you do that, here's what you'll get. This is an expanded picture. If you make an estimation problem, again, you will write something on the order of five lines of code. You will carry out some simple numerical linear algebra computations on it. And here's how close you'll get to the signal that was transmitted. The dark one shows the actual signal, and the dotted one is our estimate.

Now, the one thing you notice here is that you get pretty close. I mean, you get amazingly close. And in fact, you might ask how well are you doing? And the answer is that the RMS error, the root mean square error, is about 0.03. Now, that's super-duper interesting because the quantization levels, each step is about 0.25 wide, okay? So that's means the quantization errors are, like, plus/minus 0.125. And we are predicting it better than, I mean, roughly speaking, better than four times the range of the quantizer.

Are you disturbed? Who's disturbed by this? Good. You should be disturbed. Now, for the rest of you, why are you not disturbed by this? This is weird. I think there's something wrong with your intuitions or something like that. But no, I mean, come on, you gotta admit this is kind of weird, right?

I mean, I give you this measuring device that basically makes errors of plus/minus, let's say, 125 millivolts. I give you these measurements, I mean, that's very crude. And then you go off and do something, which you'll know how to do, let's say, the seventh week. And you come back, and you actually estimate what happened from those crappy measurements. You actually get something that's typically off by around 30 millivolts. You're still not disturbed. I can tell. You're just, like, totally okay with this. Well, is this okay with you? You're, like, cool, that's fine. Why? Student:

[Inaudible]. Instructor:

Oh, okay, but you already bought the whole package and everything. Right. Okay. All right. But no, yeah, you'll know how to do this, and you'll yawn, I mean, by the seventh week at things like this. Okay. But you already bought the whole package, so you believe it all. Yeah, I'm gonna fix you guys. You know what I'm gonna do? I'm gonna come in with something that's totally wrong. I will say it with a totally straight face, and we'll just see how far I can push. You better be on your toes. That's all I have to say. Okay. All right.

Well, since you're not upset by that, I mean, if it's just, like, okay, cool, fine, theory can do anything, well, we'll just move on and we'll actually start the course proper, I mean, unless there's some questions or — this would be a good time to answer any questions about — I don't know — the basics of the class or what we're gonna do and stuff like that. So okay.

Oh, I do have one question. How many people here — we are gonna use MATLAB. I'm not a huge fan of MATLAB, but we're gonna be writing, you know, the key is you're gonna write very short things. Okay. How many people here sort of have used MATLAB before at some point? Okay. So I think that's a whole lot. We are gonna be — so you'll be fine.

The stuff we're gonna be doing is gonna be very, very basic. If you ever write a script longer than 15 lines or something like that or, I mean, if the main part of it is more than 15 lines, you're probably doing something wrong. So it's not, I mean, the key in this class is that the ratio of thinking to programming we want to be very high. So if you find yourself writing five pages and deep — if you find yourself doing anything that you could actually describe as programming, you're probably approaching this the wrong way, just to let you know. That's generally defined as more than one screen of source code, I would say, so if you're on that second or third page, something's wrong. So that's part of the class.

For those of you who, I mean, I think a lot of this you can just come up to speed on yourself, but maybe we'll convince the TAs if some of you want to just sort of have a quick introduction, if some people need this. Okay.

So let's start the course proper. I should say about the course, the first couple of lectures are gonna be review. So if you have an exposure to linear algebra, you will think it's embarrassingly and insultingly elementary. But don't worry. We'll get there. It's a little bit boring, and it picks up somewhere around, you know, it'll pick up in a couple of weeks. So okay.

So we'll start with the idea of linear functions and examples. And this presumably you've seen before. So this'll just be review, and what I will do though is focus on sort of what the meaning of all of this is, which is — and what the implications are, and that's the part that is traditionally left out of a linear algebra course, a traditional one. Okay.

So let's suppose you have a system of linear equations, so this is linear equations. So you have Y1 is A11X1 plus A12X2 up to plus A1 and XN. And then you have a bunch of these equations. In fact, we have M of these equations. So these are — everything you see here is a scalar. Well, you don't get very far, it turns out, by using complicated notation all the time. So it's actually — with matrix notation, this goes down to four ASCII characters, which is Y equals AX.

Now, here, Y is just the vector of these Ys stacked up to make a column vector. X is the column vector of these Xs, and A is this matrix, which is M by N. So that's, I mean, this you should have seen, so this is a matrix form. And you should understand — in fact, that's kind of a theme in the class. I mean, so what's actually gone on here is we're heavily using overloading here. So everyone knows what Y equals AX means when these are scalars. And here, we have overloaded that to the case where X is a vector and A is a matrix, okay? So that's the idea here. Okay.

Now, this you've also seen. I hope, presume, guess that you've seen the idea of a linear function in lots and lots of courses, I hope. But here's what it is: A function is linear if F, the function, which maps RN to RM — this notation means that this function accepts as argument an N vector and returns an M vector. Now, this says F of X plus Y is F of X plus F of Y. When you read these things, I should add, they're very easy on the eye because it just looks super simple. It kind of looks like just the plus went in a different place. It's just not that big a deal. It just looks very casual. It even looks pretty and simple. This is actually making a very strong statement, this statement.

And in fact, I think I'll do this once, yeah, I'm gonna do it. So we'll do it once. It's actually very useful in the notation we're gonna be using to make sure you can parse what you're looking at, and in fact for everything you write, you better be sure you can parse what you're writing because a parsing error, a syntax error is a bad thing.

So let's look at this. So first, let's actually talk about the data type. So I presume you've all had a class in basic computer programming, so you might wanna think about the actual data type of these things. So let me just ask a couple of questions here.

What is X? Is it a matrix? Is it a number? What is it? It's an N vector, okay? All right then. What is this plus? What plus is that? What plus is it?

Student:[Inaudible].

Instructor (Stephen Boyd):It's a vector addition. So plus is being overloaded here. I mean, everyone knows what plus is between real numbers, complex numbers. This is plus between vectors. That's an overloaded plus here. What is the data type of X plus Y? It's an N vector. All right. F is a function. Now you can do the syntax check on F. F is being passed an N vector. Is that correct? Well, sure, that's what this says. It says F has been declared to be a function that accepts this argument, an N vector. It is being passed an N vector. And so what is the return value? M vector. Okay. What plus is this?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, which vector addition?

Student:[Inaudible]. Instructor:

Precisely. That's M vector addition. Okay. And what equals is that? Student:

[Inaudible]. Instructor:

It's M vector equals. Okay. So all of this is heavily overloaded, but I'd encourage you to do this because otherwise, what happens is you start writing stuff down, very short formulas, and they look pretty and beautiful. But watch out because the point of powerful notation is precisely that you can write totally innocent-looking things down that are five lines, and they'll actually be saying a lot. I mean, that's the good and the bad news about it. Okay.

The second is that F, you can scale X either before it's operated on by F or after, and you get the same thing. So a lot of people write this this way: Alpha X plus beta Y equals alpha F of X plus beta F of Y, like that, that that should hold. And that's one of the definitions of linearity, okay?

And this is a picture, you know, basically — actually what the picture says is this: And, by the way, this is really stupid. It already had major implications. I mean, I guarantee you I can dress this up in some story where it won't look like this, but it will be about this, and it will have serious implications in that context. Okay.

So here, what it says is this: You have a vector X and a vector Y, and X ends up getting operated on and comes out over here. That's F of X. And Y gets operated on and comes out over here. And the question then is what would happen if this vector, which is the vector sum of X and Y, were passed to F. And the answer is you can find that two ways. You can either operate on this point with F, in which case it will come out over here; or you could actually take these two things and add them and come out over here. This, by the way, already has lots of implications because, for example, if F is a simulation, this says you actually can make three predictions at the cost of two simulations. Everybody see what I'm saying here? I'm assuming a simulation is expensive, but vector addition is cheaper. So okay. But this, you should have seen. That's a linear function.

And here are some examples. Here's one. Here's a function. It takes as argument a vector X and it multiplies this X is in RN, and it multiplies by an M by N matrix, and that multiplies something that's in RN. That's an explicit function. It's parameterized by the matrix A. Turns out that's a linear function you can just check very quickly. But here's the cool part. That's actually all the linear functions. That's a generic form. It turns out if a function is linear, it has to have the form F of X equals AX, okay?

Now, when you look at that, it looks very sophisticated and complicated and mathematical. It's actually not. It's actually quite straightforward. But I think I'll say one thing about it, and then we're gonna quit.

So it's actually quite straightforward to work out what A is, and this you should think of. I think this is on your first homework, you should think of this as someone putting on the table in front of you a black box with a bunch of BNC connectors on the left, in fact specifically N on the left, and a bunch of N BNC connectors on the right. And then you're just in a lab. And the question is, how would you make a model of that box in your lab slot so that later when you're not in the lab, you can actually predict what it will do, actually for any input, and of course given that it's linear? So okay. So I think we'll quit here and then continue on Thursday.

[End of Audio]

Duration: 77 minutes

IntroToLinearDynamicalSystems-Lecture02

Instructor (Stephen Boyd):Looks like we've started. Let me start by making – there were a couple of announcements I should have made last time and forgot to. Actually, some of it involves stuff we didn't know then. The first is we have to apologize. The readers that you have – it's a very minor error. The homework exercise numbers after lecture ten have the wrong numbering. We apologize for that. We caught this a week ago, silently updated all the PDF files on the website and had the readers not been made would, of course, deny that we ever did this. But with all the readers out there, we can hardly deny it. All the materials there are in the same order. It's just the numbering gets weird.

The second announcement I wanted to make is for contacting me or the TAs. Please use the staff email address. It's on the webpage. That goes to all of us, and that's a very good way – that way we can see which emails have been responded to and which have not. Please do not email the TAs' personal email addresses. That means that other people in the teaching staff haven't seen your email. Please use the staff address.

I think I did ask if people had seen mat lab, and it seemed like a lot of people had. That's great. I do want to say that in fact you are perfectly welcome to use octave. We'll do whatever it takes, which should be, theoretically, nothing, to make anything that you have to do in this class run under octave. The difference is octave, of course, is a new open source thing. Mat lab runs on a bunch of Stanford computers. By all means, please do not go out and buy mat lab for this class.

You're also welcome, by the way, if you're crazy enough or whatever, you're also welcome to use python or something else. Jacob will help you with that. Is there anybody who is tempted? By the way, the "programming" you'll be doing in this class will be limited to one screenful. We're talking 15 lines at most. Any python – really? No one's gonna take us up on that? Usually there's one person. We never hear from them again, of course.

There was one more thing I wanted to say. The last thing – I wanted to say a little bit about how this class compares with CME 200. You may never have heard of CME 200. You should know about it. It is a class. It's taught also this quarter. If you were to compare the material listed in the abstract for that course and this one, you would see a bunch of overlap, enough to make you have some questions. I think I can explain a little bit about what the difference is.

CME 200 actually focuses on computation. It actually focuses on – it's a good compliment. The classes are meant to be complimentary. CME 200 will tell you how to actually carry out the numerical calculations. This class will focus more on modeling and applications of the materials. They're complimentary. Is there anyone here taking CME 200? It goes the other way in the other class.

Now some administrative things. We have progress. The first is we have a section. That's good. Let me repeat that the most valid information you'll always find on the website, but now that we have a section, which is going to be Mondays 4:14 to 5:05, it will be televised. That's the section, and that sets the whole timing or phasing diagram for the whole class.

That says that the homework will actually be due Thursdays, and that includes homework one. It's true, we did at one point say homework one was due on Friday. We changed our mind. It's going to be next Thursday, a week from now, so we don't feel guilty pulling the homework due date up one day. The homework will be due Thursdays.

The section is on Monday, and then immediately following section, there will be TA office hours, and then on both Tuesday and Wednesday, there will be TA office hours from 4:00 to 6:00. Where is the section? It's Gates B3. Since Jacob said that, it's likely to be correct.

Another thing is that we coordinated with some of the other classes many of you are taking and we worked out a tentative date. Let's just say it's the dates for the midterm. The midterm is gonna be either October 26 or 27 or 27-28. That's your choice. In other words, midterm and final will be something like that. Each one will be 24 take-homes and you'll have a choice of two days, so whichever fits better in your schedule.

I should also mention that if you have a moderately good reason why neither of these dates works for you – it doesn't have to be a superb reason or anything like that, but if it's a moderately good one or it's funny or something like that, we'll make arrangements for you to take it another time, generally speaking beforehand. That means you become a beta tester. You might want to mark these dates in your calendar.

The final is going to be – this is to protect my legally so the registrar's police unit doesn't come pick me up. This would be the final, but you can't hear in my voice the quotes. If anyone asks you, it's not the final. I think it's the last homework assignment and it just counts for a lot. Actually, I think that's illegal, too, because you're not supposed to assign homework on the last day of class. I'll figure something out. I should probably put in my cell phone an attorney's number so when I do get arrested by the Stanford registrar's office I know who to call.

That's traditionally the last day of class. These things usually go out at 5:00 in the afternoon and then people come back a day later, tired but alive. We haven't lost anyone yet. That's gonna be the last day. Once again, if these don't work for you, we can work something out.

By the way, if there's some weird thing where you have to take it after by some significant number of days, that's something you need to let us know really early so that we can work around that. We try to release solutions as soon as possible, but we can't release any solutions until everyone has taken the exam. If there are any quite strange scheduling things, people should get in touch with us soon.

Now I think I have covered all of the administrative stuff, including what I forgot to mention last lecture. Sorry. Today and for the next week or so, what we're covering should be review. What I'm mainly going to focus on – if two weeks from now you find that we haven't covered anything that you don't know, don't worry. We will.

What I want to focus on is really the meaning of all this, because that's something that's conspicuously absent in your traditional linear algebra class. Last time, we looked at the definition of a linear function, a function that takes its argument and N vector, returns an M vector is linear if you can form a linear combination of the inputs and apply the function, and it's the same as if you had applied the function to each of the arguments separately and formed the same linear combination.

This formula, which looks very innocent, in fact involves – it says a lot and it involves a lot of overloading. What I mean by that is that all sorts of symbols are doing multiple duty here. This plus is technically not the same as that plus. This Alpha X is not the same as this Alpha X. Those are basically in different dimensions and things like that. There's a lot of overloading there.

At the very end of the last class, we got to the matrix multiplication function. It's a very important function. Very simple. It's two ASCII characters. It's this. You've taken argument X and you multiply a vector and you multiply it by matrix A and that's what you return. That's the matrix multiplication function. It's linear. You just have to check these things. It's easy to do.

It turns out actually it's more than that's just an example of a linear function. That is actually the generic form. All linear functions can be represented as F of X equals AX. People make a big deal about this. It's actually not that big a deal, although it's useful to think about how given F – if F were a black box or just a function or operator which you could call but cannot see the internals of, the question is how would you reconstruct A, a model? That's essentially the question. That's why you're going to be seeing F of X equals AX a lot.

Now let me say something broad about what does Y equals AX mean. It turns out there's lots and lots of applications of this. In fact, really much of the course is going to be about banging practical problems into a form like this or a variation on it that might be more complicated, but something like that. Here, there are huge, broad categories of interpretation. One is this. In one interpretation, X is a vector of unknowns or parameters that you want to estimate – things you don't know.

Y is a vector of measurements or things you do know or, in communications, this would be something like a transmitted signal and Y is a received signal. That's one very broad category of linear functions. In this case, there will be all sorts of interesting things you want to do. The things you want to do would be something like this.

Suppose you're given Y. You'd like to say as much as you can about X, ideally saying what X is. That would be undoing this linear operation. In another broad category, X represents an input or action. In this interpretation, X is something we can mess with. It's something we can change. X could be all sorts of things. It could be a signal we transmit. Then this Y is an outcome in this interpretation.

X could be all sorts of things. It could be the thrust that you put on your engines. That could be X1. X2 could be the deflection of your [inaudible] and your elevators and your control surfaces and things like this. So in that case, it's an action. The general idea is we control it. Y is now a result. Y might be a vector of climb rates and things like that. We'll see more specific examples soon.

Another very useful interpretation is that you think of this abstractly as a function or a transformation. It's just something that takes a vector, does something and gives you another vector. An example of this would be something like a Fourier transform, or it could be a rotation. Something that takes a vector, rotates it 27 degrees clockwise around some axis. That would be – there, you'd be thinking in terms of a transform. They're all going to look like Y equals AX.

What I'm about to say is very simple and totally obvious, but I think it still needs to be said. Here's Y equals AX written out component by component. This is the – I'm going to call Y the output and X the input. I'm neutral on whether it's an input we can mess with or it's some parameter we want to estimate or something like that. I'm just – input and output are just neutral here. This says that the I output is a sum over all inputs. It's weighted by AIJ. That's what it says. What that means is you can interpret AIJ immediately. It's the gain factor from the J input to the I output.

Again, what I'm saying is not exactly complicated, but it's very important to understand. A lot of these are very important things to know. That means that the I row of a matrix – that's AI1, AI2, AI3 – you're going across the row. All of those things concern the I output. Those are the gains from the inputs to the I output when you scan across a row. If you look at a column of a matrix in Y equals AX, a column concerns an input. So the J column – the third column concerns the third input. In fact, as you scan down that column, what these are is these are the gains from the third input to all the outputs. That's the idea.

That's fine to say, but now what it means is when you see a matrix and you see something in it, you should interpret things immediately. So for example, if you see that A27 equals zero – that's the 27 entry in the matrix. It's zero. That has a meaning, and the meaning is the second output, Y2, doesn't depend at all on the seventh input. Again, this is obvious, but it needs to be said.

If, for example, you see that the 31 coefficient to the matrix is much bigger than all other entries in that row, that has a very specific meaning. What it says is that Y3 depends mainly on X1. I'm assuming here that the coefficients in X1 are scaled, so they're all about the same. You could have something weird where the X1 entries are on order of units and the X3 entries have numbers like 10 to the minus 9, but I'm assuming they're all about the same. Then you could say this.

You can see all sorts of other stuff. For example, this goes on and on. If you have a column – if you find that A52 is a lot bigger than AI2, that tells you that the second input affects mainly Y5 among the outputs. You can just go on and on and on. For example, if A is lower triangular, that means AIJ is zero for I less than J. It means that YI depends only on X1 up to XI.

By the way, this means – when you have lower triangularity, you're going to want to make a vague symbolic link in your mind to causality. If the index here represented time, this would be something like causality, and that's what lower triangularity means. You shouldn't just dispassionately look at a lower triangular matrix and say yeah, it looks pretty. It has a very specific meaning, and it talks about what inputs affect what outputs and it's very specific.

Here's a very special case. Let's say the matrix is diagonal. If it's diagonal, what that means is that the I output only depends on the I input. In general, you have the idea of a sparsity patter of a matrix. The sparsity pattern is the list of zero and non-zero entries. In a lot of cases, a matrix has no interesting sparsity pattern.

All the entries are non-zero, in which case you call it a dense matrix. In a lot of interesting cases, a matrix will have a non-trivial sparsity pattern. So blots of entries will be zero, and that will have a meaning. You should never, ever look at a matrix that has blocks of zeros and not think what does that mean? A lot of this is simple, but this is the idea.

How would you find, for example, if I gave you Y equals AX and I asked you something like this – which input makes the greatest contribution to the seventh output, what would you do? Yeah, so you'd look at the seventh row, you'd scan along it and if the entries were all about the same, you'd say they all affect it about the same. If one stood out, you'd say it effected.

What does the sign mean? What does it mean to say that A35 is positive? It means the following. If A35 is positive, what it means is that if you increase X5, the fifth input, the third output will increase. These are obvious things. Don't think about them too much, because there's not that much here to think about. When we see it in action, it'll make more sense.

Now what I'm going to do is go through a bunch of examples to give you a rough idea – just so you have some pictures in your mind of this thing. So the first one is from mechanical engineering. We look at a linear elastic structure. So I have a structure like the steel frame of a building or something like that. Here, XJ is gonna be an external force applied to this structure at some point and in some direction. It could also be a torque. For example, these could be all the various forces here.

Here I have the first four forces – we can imagine these are wind loadings on the building. These are wind loadings. These might be dead loading in the building. YI is an output. That's going to be the deflection of a point in a building in a certain direction. It might be the deflection of this floor here. It has to be oriented, so for example, I might say that downward is positive deflection. Upward is negative.

There are names for these. If you get into civil engineering or structural engineering, the name, for example, for the difference between the horizontal displacement from one story to the next is called interstory drift. Any time you have some specific area, you'll find all sorts of colorful language for it. We'll say Y is one of these deflections.

It's a basic fact. It's roughly true. It says that if these forces are small – of course, in a steel frame building, small does not mean five Newtons. It can mean quite a lot. If they're small, which depends on the application, then it says that the deflections are approximately a linear function of the applied forces. That's actually quite interesting. Now I can ask you some questions. I have to show some displacement.

I'm going to make the displacement here Y1, the displacement here Y2 and so on. That's Y4. Now I'm going to ask you to tell me about some entries of the matrix. Tell me what you can say about A11. It's positive. What's your intuition behind that? You push here with a unit of force. The building will deflect this way, and then whatever the ultimate equilibrium deflection is – it looks like Y1. By the way, what can you say about A21 compared to A11? It's smaller. In fact – what sign do you imagine it has? Positive. Okay.

By the way, only those of you who have had a class on structures are actually allowed to say that. The rest of you, like me, have to say we guess it's positive. Tell me about the column that associates with X3. Tell me about the third column of the matrix. What do you think it might look like? So Y3 would be big. How about Y1 and Y2? I think they'd also be big, because you'd shift it over and there'd be nothing up there. You get the idea.

By the way, the units of this matrix – in general, it's sometimes useful to think of the units – the units here are in meters per Newton if those are the units I'm using for displacement and force. It's got a name. It's called the compliance matrix. If the deflections and the forces are measured at exactly the same place, then the inverse, because the matrix is actually square – the inverse is called a stiffness matrix for example. That's just one simple example here.

Second one is we have a rigid body. For example, let's say a satellite or something like that. Here's the center of gravity. Let's say that I'm going to apply forces and torques on it. I'll apply a torque here. X1 is a force in a fixed direction. X1 measures the size of it. X1 negative means that I have a force in the direction opposite to whatever my reference direction is. XJ is gonna be my vector of these external forces and torques. What I'll do is if I give you the forces and torques, there is a net force and a net torque applied to this rigid body. In fact, those are linear. You have Y equals AX.

A depends on geometry. You don't actually often write it this way. All of that is obscuring the fact that the net moments and torque on this rigid body is a linear function of these applied functions. This comes up. This would be, for example, a vehicle. These could be thrusters or a control surface – anything that creates a force or torque on the vehicle. This would be a very interesting matrix because it would actually tell you how what you do maps into the total force of torque. There's lots of questions you can imagine you might want to answer here.

So for example, here, the J column of A is a pattern, and it tells you the first three components in the J column give you – let's imagine the J column is a force applied. Then the first three components – A13, A23, A33 – what are the units of those? If the third input actually is a force, what's the unit of these? These are unitless, and the reason is they map Newtons to Newtons. These are unitless. How about A43, A53 and A63? What are they? Meters. They have units of meters because they map Newtons to Newton meters to torques.

If you want to know if that is an actuator or a thruster and you want to know what does the thruster do, you get a rough idea by scanning that column. Looking at the geometry here, tell me something about the second column that corresponds to this input. Let's go right down the second column. What can you say about A12? That's gonna be the X component of the net force contributed by this thruster. What is it? The magnetite is less than one.

Let's be more specific. Zero. Is it really zero? It's not perpendicular. It tilts a little bit to the right. So it's small and it's positive. Let's go for a number. Let's say it's .15. It's very important to know that the number I made up was not arbitrary. If I'm off, I'm not off by too much. How about A22? It's large. In fact, how large? It's close to one. It's .85, .95. Something like that. That's enough on that.

Let's look at a linear static circuit. Here, I have a linear circuit. It's an amplifier, but it doesn't really matter what it is. It's got linear resisters here. It's got an external current source here, an external voltage source here and it's got a dependent current source here. That's how that works. That's a current controlled current source here. Here, I'm going to let X be the value of the independent sources. In this case, that's X1 and X2. Y is going to be any circuit variable. For example, it could be the voltage across a leg. It could be the current through a device. It could be the potential – the voltage at a point.

It doesn't matter what it is. In this case, it's definitely linear. You have Y equals AX here. You have Y equals AX and that's interesting. By the way, depending on what YI is and XJ, all of these entries have interesting units and names. For example, if the Xs are all currents and the Ys are voltages, then A is called the impedance or resistance matrix. In this circuit – this is again only for people in electrical engineering.

What is A11? A11 has a meaning here. I want the street name. It's the input resistance of the amplifier. Again, if you're not in EE, don't worry about it, but that's what A11 is. It says when you pump current in here, you develop a voltage here. The gain is A11. It's in ohms and it's the input resistance of the amplifier.

Next one is a simple dynamic system. It's your basic frictionless table here. You have a frictionless table and you have your standard issue one kilogram mass, and what's gonna happen is this. It's gonna be at a position zero and it will be at rest at T equals zero. What we're going to do is we're going to subject it to a P wise constant force. So for one second, we apply a force, which is X1. Then we apply a force X2, then X3 and so on. So now the interpretation of X is like a force program. Program means it's your plan. It's a force. You could call it a force program, a force plan, a force trajectory.

You can call it all sorts of things. That's what it is. That's what X is. We'll make this up to N. There are going to be two outputs or outcomes, and it's simply going to be the final position and the velocity of that mass. The point is that's linear. That's Y equals AX. What is the size of A? Two by N, because it's got two outputs and N inputs. I think I mentioned this last time, but I do encourage you to read some things on the course website that are very elementary but set up the notation of matrix multiplication. Just scan it.

You'll also find another PDF file there, which has a title like "Crimes Against Matrices." You might want to scan bits and pieces of it. I'll come back to that later. It means you should always at an instant's notice be able to identify the dimension of something. If you can't – if someone can point to something you're doing or looking at and say what's the dimension of that and you don't know, that means – it's very easy with this stuff to sit back and go yeah, sure.

It's very easy to get complacent here, but you should be checking yourself whether you know exactly – that's related to these things. Y equals AX of course only makes sense if the number of columns of A is equal to the dimension of X. If you ever write down Y equals AX and that's not the case, we call this a syntax error, and it's bad. This is not a good thing. Don't do that.

Here you can interpret stuff. The first row gives you the influence of the applied force at different times on the final position. Actually, I'd like to ask you about that row. What does it look like? I don't want the actual number. I just want intuitively to tell me about that row. It's ascending. What you're saying is a Newton applied at the beginning has less of an effect than a Newton applied at the end. If you say that first row increases as you go down the row, for me, that's this way. If you say it's increasing along the row, it has a meaning.

It basically says – what are the units of the entries of the first row? Is it even linear? Yeah, it wouldn't be in the notes if it weren't. By the way, that's not entirely true. There have been things in the notes that were mistakes for three years at a time. Let's just say I'm pretty sure this is okay. If not, you don't have to do the homework problem where you work out A, but it is.

Assuming that it is linear and the entries of the first row are in meters per Newton because they tell you meters of final displacement per Newton applied, and if you say that that row is increasing, it has a meaning. Do you believe it now? The initial velocity is zero here. The initial position is zero. What do you think? It's decreasing, somebody said. It's decreasing. What does that mean? It says that if you apply a Newton for the first second, you will get more final displacement than you will if you apply a Newton at the last one second. Is that true? Why?

There's no friction here, so what that means is something like this. When you apply a Newton in the first second, you accelerate the mass to one meter per second. There's no friction. It now coasts for N minus one seconds with that velocity you just gave it. That gives you – you cover a lot of distance. In the last time when you apply one Newton in the last one-second period, what happens is in fact you're merely accelerating it from whatever speed it has to one meter per second more. In fact, the total displacement you get is half a meter in that case.

The details of this don't matter. The point is to think about these things. What about the second row of A? Yeah, the second row of A is going to be constant because if you apply a Newton for one second to a mass on a frictionless table, it has the exact same net effect. You will increase – we can even say what that row is. It's all ones, because if you apply a Newton for one second to a one-kilogram mass, it will be moving at the end one meter per second faster.

These are simple examples, but you should – this is the kind of thing you should do without even thinking about it. We can put a negative force no problem. The matrix A doesn't care and has nothing to do with whether you push to the left or right. When you push to the left or right in your force program, what you're doing when you push to the left or right is that's changing X. In fact, let me be very specific. Here's an X. Let's make N equals 4 and here's my X. 1,0,0,-1. That has a meaning. That means you push one Newton, you coast for two seconds and you pull. What is Y for this, roughly? It's about three.

In fact, it happens to be exactly three. Y is a two vector, so it's first entry is three. What's the second? Zero. What does that mean? This means that the final position is three meters to the right, and the final velocity is zero. So this is, in fact, a force program that will transfer the mass three meters to the right. It will take it from stationary to stationary three meters.

This is all very simple, but this is, for example, a very simple version of exactly what your disk drive head does when you initiate a seek. This is what happens. It moves from track 25 to track 150. It does exactly this. By the way, the force program is not quite this, but it's roughly like this. Later, we'll find out what it really is.

This one is from geophysics. It's gravimeter prospecting. It works like this. This is underground. It could be seawater or include some air. These are little voxels, and these voxels have a density row, and we'll make XJ the excess mass density over the average earth density. I forget what that is. If one of these has gas in it then X is negative. If this has got some really dense rock, X is positive. When you have an array of things at different density and you actually want to know what is the gravitational acceleration here, you get it by integrating all these things and using the gravitational force business.

Mostly, it's going to be pretty [inaudible] as it's going to be pointing down and it will be about 9.8 meters per second squared. It turns out it actually can be deflected. It's very subtle. It's out there in some digit – something like the third or the fourth digit or something like that, but it will point somewhere else. Its magnitude will change. If you're right next to a giant mountain over there, down is now ever so slightly deflected that way. If you're standing over flat ground and there's a giant, giant cavity under there filled with gas, then in fact the same thing might happen.

This would be deflected slightly that way, and the magnitude would be slightly different. What's often measured is the difference of G with the average around there. You calibrate it somewhere and then you move it around, and in fact, this is approximately linear. Here, A is quite complicated. The formula for it is horrendous and involves all sorts of things. It's got positive and negative things. Very complicated. This is going to involve all kinds of crazy things with sines and cosines and distances and all sorts of other stuff. That will just obscure this fact.

Now you can say lots of stuff. You can say, for example, that the J column of that matrix shows you the sensor readings that would be caused by a unit density anomaly at voxel J. That's literally what it means. The row shows the sensitivity of a sensor. If you make a measurement here, it tells you as a function of voxel what the sensitivity is. You'd expect this to be – the matrix wouldn't be sparse, but it might have a lot of small entries. You'd expect that the gravitational anomaly here would have not a who lot to do with voxels that are way, way off over here. There would still be some structure, if not exact zeros in the sparsity pattern.

Next one is a thermal system. Here, I have a system, and I have some sources of heat in the system. These could be heaters where in a control problem, I want the temperature to do something I want, so these are little resistive heaters. Or these could be processors on a multiprocessor chip where they're just operating and then the temperature is whatever it is. Lots of applications. I inject heat at these sources, and XJ will simply be the heat source, and that's measured in watts. I assume that the thermal transport here is linear, so it's via conduction. It could be by convection, but it would be a very simple linear model of convection.

It's not by radiation, which is going to involve these temperature to the fourth terms. Again, if you don't know what I'm talking about, it makes no difference. It's fine. It's linear. There'll be some appropriate boundary conditions, like they'll be isothermal or they'll be isolated or something like that. Then this whole thing will come to some thermal equilibrium temperature distribution and I'm going to let YI be the temperature at a certain point I. That's the idea.

If you look up thermodynamics, you'll find out – people will tell you that's a [inaudible] equation. It's very complicated. PDEs will come out and you'll pretty quickly – you have to do that if you really want to understand this. But in fact after all the PDEs go away and everything else, you will find that, in fact, Y equals AX. The vector of equilibrium temperatures will be equal to a matrix times the vector of the power dissipation in these things. It's that simple. Let's talk about A. What can you say about the sign of the entries of A? They're all positive. What does that mean?

It means that if you pump heat into something, at some point, the only thing you can do is increase the temperature everywhere. Here, what can you tell me about A41? Small. The idea is this – A41 is the gain, and it's in degree C per watt from this heat source to that location. It does have an effect, but it's small. That's kind of the idea. You can imagine all sorts of cool things you might want to do with this. These might be things under your control, and you might say find me an X that makes the temperature distribution something I want.

Let's say for some kind of experiment you want a nice, uniform temperature gradient or you want it uniform. AX is about equal to some desired temperature to the extent possible. We'll be able to answer questions like that. Another one would be estimation, which is I give you 57 measured temperatures. I want you to estimate the power being deposited at these five locations. That's an estimation problem. You want to deduce it from the measured temperatures. There are all sorts of things.

Next one is illumination with multiple lamps. Here I have some surface like this and I have a bunch of lamps at these points up here, and what I can do is each lamp has a power XJ. These things – they go down to this patch here and here, I can actually say what AIJ is. Here, I say YI is going to be the illumination on patch I, and XJ is the power in lamp J. One thing to notice when you look at matrices and things like this, and you get used to it after awhile, but when you have A sub IJ, I indexes the output for effect. J indexes the input. So really, the pair IJ is really indexed output, input.

That's weird because most people think of – if you just walked up to someone on the street, they would probably index things by input, output. Matrices are organized as output, input. Too bad, that's the way it worked out. You'll get used to it. By the way, these are dummy variables. I could switch them and be sick and just turn it around and make it JI. That will happen. But right now, I'm just trying to stick to a reasonable – they can be anything you like.

Here, the illumination of a lamp on a patch is given by the inverse square – it's one over the square of the distance and then it's multiplied by this cosine factor, which is basically how much of the light is caught by the angle. By the way, if it's all the way over and the light is below here, you get nothing because you're obscured. That's what the max is here.

Once again, the vector of illumination levels is a linear function of the vector of lamp powers, and that tells you it has the form Y equals AX. Again here, A is non-negative. That's clear intuitively, but it's also clear from this. You get a rough idea. For example, if you look at a column of A, what you are looking at is the illumination pattern generated by that lamp. You're looking at the third column. That column gives you the illumination pattern. If you look at a row, you're looking at a patch. You're focusing on a patch and you're asking what the different gains from the different lamps to that patch are.

We'll look at another couple of these. This one is from communications. Here I have N transmitted receiver pairs. The idea is transmitter J wants to transmit to transceiver J. Unfortunately inadvertently it also transmits to the other ones. We don't want that. P is going to be the power of the J transmitter. S is going to be the received signal power of the I receiver. Z is going to be the received interference power of the I receiver. GIJ is going to be the path gain from transmitter J to receiver I. That presumably will depend on how far they are apart. It may depend on all sorts of other things in between.

G is non-negative. You have S equals AP and Z equals BP where S – A is in fact a diagonal matrix where you just take the diagonal part of G. That gives you the vector of signal powers. You take the rest of G – the off diagonal part of G and you shove that into matrix B, and if you multiply that matrix by the power vector, you get the vector of total interference powers. I'm assuming that the interferences are going to add incoherently. The powers are going to add. It's not coherent addition.

Ideally, what you want is you want A to be large and you want B to be small or zero. That means you want this matrix G to have a very strong diagonal and lots of little off diagonal entries. If I asked you questions like this – the third receiver is most susceptible from interference from which transmitter – how do you find that out given G? The third receiver is most susceptible to interference from which transmitter? What do you do? The answer is you walk across the third row of the matrix G. I guess for you, that's kind of like this.

You walk across the third row of G. The three entry in G is very important. That's actually the gain the transmitter you want to listen to. You look at the other entries, and in the other entries, you look for the largest entry, and that tells you which transmitter you are most susceptible to. These are simple things, but this is the kind of thinking you need to do.

The next one is from economics. It involves things like cost of production. Here you have a bunch of production inputs like materials, parts, and labor. You combine these to make a bunch of products. We'll let XJ be the price per unit of production input J. AIJ is gonna be the units of production input J that you need to manufacture one unit of product I. So that means if you go cars the row of that matrix, it corresponds.

If you go across the third row, it basically tells you how much – what are the inputs you need to make one unit of the third product. If there's a zero there, it says you don't need that. If it is a large entry, it says you need a lot of whatever that is. Could there be a negative number? Strangely, it depends. Generally speaking, no, but in fact, yes, you could have a negative number. What would it mean if A23 were equal to minus one? What could it mean? It's a byproduct. Exactly.

It says that when you make one unit of product two, not only will you not need input three, you will actually as a byproduct of making that generate one unit of input three. A lot of these things where normally you think of it as something that's positive, there usually is a really interesting interpretation of what happens when it's negative.

I'll show you an example where that's not the case. Transmitting negative powers – people have tried to do it, but so far, it just hasn't worked out. How about this one? What is it? You could have active cooling. They have these things and yes, you can pump one watt out. I just mention this because it's good to keep these in mind.

If Y is the production cost per unit of product A, you have Y equals AX. This is beautiful. This tells you something like this. This tells you how the cost of making your family of products depends on the vector of input prices. For example, I could ask a question like this. Among all the products you make, which is most sensitive to the price of energy? How would you answer it? You look for the energy column. It says energy is X3. You scan down the third column and you look for the biggest entry and you say that product is the one most sensitive to a change in the price of energy.

Let's move on to the next example. These do get a little bit boring, but it means that when we do talk about stuff that's abstract, at least it has meaning in all of these contexts. The next one is from networking. I have N flows in a network. A flow is something that passes from one node across an edge to another node to another node. These are going to have rates F1 through FN. It doesn't matter. These could be in bits per second. They could also be, for that matter, in liters per second. It can also be electricity. It could be anything. It could be goods. These could be transported by trucks. These could be packets.

It doesn't matter. They pass from a source node to a destination or some fixed route. The traffic on a link – some of the routes will go over each link. If no route goes over a link, then it's utterly unused. But every link will have some routes go over it. It may be one, two or 100. The total traffic on that link is the sum of the flows of the routes that pass over it. You can write this as exactly the same thing. If you have T, that's the traffic vector, and that is a vector that tells you – its index refers to a link, and it says this is the traffic on all the links. It's a linear function of the flow rates. It looks like AF.

A is, in fact, a very simple matrix. People call it a zero one matrix or something like that. It basically encodes which flows pass over which links. Now, I can ask you a question. A bottleneck is a link that has a large number of flows going over it. If I gave you the matrix A, how would you find a bottleneck? You look for a row, and a row corresponds to an output – in this case, a traffic. A row corresponds to the contribution to a link from all the flows. You look for a row that has a lot of ones in it. There we go. That's what you said, right? You're right. That's it.

By the way, what's the meaning of a column with a lot of ones in it? It's a long – it's a flow with a long route. That's the idea. Interestingly, we can do this. Let's say that each link has a delay on it. In other words, when you're going to go over a certain link, you actually arrive at that node. There's a queuing or transport delay. It doesn't matter. These are just applications to give you some context for all of this. Each link has a delay, and that's the delay it takes – it might be waiting to get queued up and transmitted. It might be the transmission delay. It doesn't matter. It's whatever the delay is across it.

Therefore, if you have D1 through DM are the delays on each link. That's the link delay vector. The latency of a flow basically is the sum of the delays along the route. If these were packets, it literally tells you if I inject a packet here how long it will take before it emerges at the other end. That's the latency, and it's simply the sum of the delays along the route. It turns out this is very easy to write down. It's L equals A transpose D where this is the transpose of A. In other words, you simply take – you can work it out.

There's some very interesting things here. For example, if you work out what F transpose L is, F is a vector of flow rates. Let's just say it's in bits per second or packets per second. L is a vector of the same size. For each flow, it tells you the latency, which is basically the delay. When you inject a packet, how long it takes before it emerges at the destination. FILI is exactly the number of packets in transit. That's what it is. F transpose L, which is the sum over FILI over I is exactly the total number of packets in the network. That's what this is. This you can write out lots of different ways.

You can also write it, by the way, as F transpose A transpose D, because L is A transpose D. This is – just doing some simple matrix arithmetic, I can rewrite this as AF transpose, but AF is the traffic. So it turns out it's the same as this. This is an inner product. Let's talk about what this is. What is TI times DI? It is exactly the number of packets in transit or waiting on link I. This is the sum this way. You get the same – these are two ways to get the total number of packets in the network. You either sum over the flows or you sum over the links and you get the same thing.

By the way, if there's any of this you didn't get, you should go back and make sure you believe it. Don't spend too much time on this because I've had people come back to me later and say yeah, but just the way you were saying it, it sound like it was very deep. Nothing we've said today is complicated or deep. If you think you don't get it, you do. I've had people come and say things like I understand everything, but I think there are subtleties I'm not getting to which I'd respond there are no subtleties. What we covered today was trivial – interesting and important, but trivial.

Here's a generic source of linear mappings. It is linearization, which you've seen. There's another name for it. It's sometimes called calculus. Here it is. I have a function that accepts an N vector and returns an M vector. You say it's differentiable at a point. Whenever X is near X0, the function value is very near F of X of zero plus DF of X0 X minus X0. You have to be careful here. That is gonna be an M by N matrix. This is DF. That's the derivative of F at evaluated X0. It's entries are these partial derivatives. Very near is a technical term. If I did this, this is the definition of F being continuous.

It says basically if you're near one point and evaluate F, you're near the image of that point. That's the definition of continuous. The definition of differentiable is this. Very near, by the way, means it says that the error here is like the square of the error here. That's what the very near is. Very, very near means it goes like the cube of the error, by the way, and it goes on from there. You can read this informally or formally. In many cases and in lots of contexts, people focus – what they do is they focus – lots of contexts have different names for this. For example, in a circuit, X0 would describe the bias or operating point.

You'd have something like – these would be the bias voltages, the bias inputs or whatever, and then the deviations would be called the small signal values. You've seen that. In aeronautics, you would have the so-called trim condition. The trim condition is that your thrust is at such and such a level. Your elevators are at this level an so on and so forth.

You're in level flight at 40,000 feet at such and such a speed. That's your trim condition. Delta X represents a small change from that and a change in your elevator deflection. Delta Y would be a difference in, for example, the net moment in torque on the airframe. Different fields have different names for a base operating condition and then wiggling around it.

A lot of people introduce a notation like Delta Y is Y minus Y0. Delta X is X minus X0, and then you can write it this way, and this basically says the deviations in a response or output is a linear function of the deviations in the input. That's the generic example. You've seen this. That is literally calculus. The problem is all these stupid multivariable calculus classes make everything complicated by bringing up things like radiance and curls and things like that. This material was super useful in the late 19th century and maybe up until the 1930s.

It's not really much anymore. You need a few people to know all those things, but not really. The problem is that if you look back at your multi-variability, it's unbelievably simple. It's just DF. In fact, that is – this is an approximation. Just by syntax, this has to be – this is an N vector. This side – what you're approximating is F of X, which is an M vector, so the only thing you could multiply an N vector by and get an M vector is an M by N matrix. So that's just an M by N matrix. It would be, for example, in the case of the gradient, it's actually a row vector, which is the right way to write it.

This is something like this. This is the derivative of F evaluated at X0. That's a matrix and then this is indexed at IJ. That's what the parsing is here. The left and right hand sides of this are numbers. INJ actually are the indices. I index actually the component of F. F returns an N vector, so F3 or F of X sub three or something represents the third component of it. J indexes into the input.

We'll look at a specific example of this. I won't do that. I'll just wrap up a little bit. The things we looked at again – this is the last day where you'll be subjected to me looking at stupid examples of Y equals AX. On the other hand – we didn't cover any actual material today. We just looked at a bunch of examples. Don't worry. The rest of the class is not going to go this way, but still, it's an important thing to do. I guess we'll wrap up next time. Remember, there is actually a section on Monday.

[End of Audio]

Duration: 81 minutes

IntroToLinearDynamicalSystems-Lecture03

Instructor (Stephen Boyd):Are we on? Okay, hey, I guess we've started. Any questions about last time – if not, we'll continue. I think they're – I wonder if there's some announcement. Let's see, one is that Homework 1 will be due Thursday. We're probably gonna post Homework 2 later today. So I know some of you are very excited about that. The reason is we had at least a couple of people who are gonna be gone next weekend and asked us to figure out Homework 2. So we did it, and it's figured out.

So we'll post that. Of course, it'll be due a week from Thursday. That's how that's gonna work. I don't think there are any more announcements. Oh, there is one more – we now have a third TA, Thomas, who has a class at this time, so he's not here, but he sneaks in for the last half hour. So maybe if I remember, I'll point to him and he can wave his arms when he comes in or at the end of the class, something like that.

Okay, well, let's just continue. I need to go down to the pad. Last time we looked at linearization as a source of lots and lots of linear equations. So linearization is you have a non-linear function that maps RN into RM. And you approximate it by an affine function.

Affine means linear – sorry, that's not linear. There we go – that's linear plus a constant, so that's an affine function. You approximate it this way. In the context of calculus, people often talk about a linear approximation and what they really mean is an affine approximation. But after awhile, you get used to this.

So linearization is very simple. You simply work out the Jacobean or the derivative matrix, and what it does is it gives you an extremely good approximation of how the output varies if the input varies a little bit from some standard point, X0. That's the idea.

And in fact, in terms of the differences or variations measured from these standard point, X0 and F(X)0 that's Y0, this relation is linear, so the small variations are linearly related.

Okay, so let's just work a specific example of that. It's an interesting one; a very important one, too. It's navigation by range measurement and of course, this is roughly to give you a rough idea or is actually how, it's part of how GPS works.

We'll get more into detail. We'll see this example will come up several times during the course. So here you have X and Y, two variables unknown coordinates in the plane and we have a bunch of beacons at locations PIQI, so these are X and Y coordinates of the beacons. And what we measure is a range. And a range, because the beacons can only measure range, ranges to this point, it could be of course, be the other way around, that the point can measure its distance to the range.

But for now, we'll assume everybody has all the information, so here the beacons get the range to this point and that's nothing but the distance and so you have a bunch of points here and each one has a range and its not hard to figure out, for example, the ranges you could figure out where the point is.

In fact, if you know the range from a beacon it means that the point lies on a circle of a fixed radius and if you have two of those, it means it's in the intersection, now it's ambiguous; it's down to two points that are possible, and if you have a third beacon it's completely determined. This is in the plane.

So, let's see how this works – let's see how linearization works. Well, here the mapping, here we have four ranges and it's a non-linear function of the point, XY of the location of the point, and of course, it's the square root of XI minus pi squared. It's the Euclidian distance.

So it's this function, which is of course, non-linear in X and Y. Let's linearize around the point XY0 or in this case, you got the change in the range, that's a four vector is equal to the change in X and, Y, multiplied by Matrix A. The entries of the matrix A actually can give you work out geometrically exactly what they are, they are unit vectors. They're unit vectors – the row is a unit vector pointing from this point to the beacon, like that.

So basically it says actually it's the other way around, it's the negative of that, so, sorry. So if you step in this direction, you would reduce the range. If you step in a direction orthogonal to a beacon, actually to first order the range doesn't change at all. And if you step opposite direction, the range increases. So what that says is that this Matrix A which is 4 by 2 in this example, that these entries tell you how much small changes in position translate into small changes in the four ranges like that.

And so that's the – and in many, many applications of this, for example, if you know where X and Y is at one step, it's now a little bit later, so it may have moved, but it hasn't moved far in a linearized model would give you an excellent approximation for how the ranges are gonna change, okay?

So this is sort of the picture. So that's an example, and you'll I guess in Homework or something, you'll see plenty of examples sort of like this.

Let's just work out and actually just do this intuitively for this case. Actually for the specific case we have here, let's estimate, just by eye, some of the entries of A. So in fact, let's figure out for what I have right here, I would like to know what the second row of A looks like. So I'll start with that entry. What's the number; what's it look like?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):I'm not hearing anything clear, so I need someone to –

Student:[Inaudible].

Instructor (Stephen Boyd):Here, I'm only asking you to do a thought experiment with your eye. The thought experiment with your eye is to move X in here, X and Y a little bit and to see what happens to the range. If you move X a little bit, suppose you increase it. I'll take my coordinate system for where it goes like that, if I increase X a little bit, what happens to the range to Beacon 2? It goes down –Okay.

And it goes down – if that were actually horizontal, it were exactly horizontal, if it moves to the right a meter, let's say, the range is gonna go down by a meter. Let's assume that's 20,000 kilometers or something like 20,000 meters, okay, as it might be in GPS.

So if it goes down – so this something like about minus 1, okay? And now tell me this entry. It's about 0, okay, and that has a meaning here. And so that's just a quick exercise just to do it, and of course, you should always be doing this and then of course, checking it against the formulas, just checks to make sure what you think and what the formula thinks is right. Okay.

Okay, that's a whirlwind tour of various applications, and now I want to talk a little bit of one level above all these applications, kind of organize them. So there are a lot of applications or models or categories of Y equals AX and in fact, you're gonna find out that much to use, you know, the material of the first four weeks of the class, everything is gonna hinge on actually hammering or pounding with more or less, you know, sometimes with little violence, sometimes with lots of violence, hammering your problem into a form that looks like Y equals AX, okay.

So it's gonna be that; that's what you're gonna be doing. And of course, in a real application it won't look pretty like this. No one will ever walk up and say Y equals AX. They'll actually walk up to you and they'll go on for hours and hours about their particular application and they'll tell you about the stein model of the electron density and the troposphere. They'll go on; they'll be cosines and sines flying around. There may be some PDEs and things like this. That's because they're in that field.

It'll be your job to step back, ignore all of that and calmly step forward later and write it a Y equals AX. So that's the way this is gonna work, just to give you a rough idea.

Now, what does Y equals AX mean – well, one area is estimation or inversion. I've already talked about that and when we looked at other examples, well when we actually get into methods, we'll see more.

In these ideals, X is something you don't know. It's something you want to guess or find out or something like that. These could be parameters in a model. They could be a transmitted signal in communications, something like that. It could be something that you really want to know, but can't go and directly measure.

Y represents what you measure. So Y is something you do know or you can measure or something like that and from that you want to deduce X. That would be the type of thing you would want to do. A, in this case, is represents your measurement set up or in the communications context, it’s your channel. So it’s what maps, what’s transmitted to what’s received. That's what A is in that case.

All right, in a design problem, X actually is, in fact, it’s the opposite. X is something is what we can control. X are the knobs we can turn; it’s the design parameters; it’s the thrust; it’s the, that we can command an engine to give. It is control surface deflections. It’s things we can mess with, we have control over.

Y is an outcome. So here, in this interpretation, so Y is a vector of outcomes and in a case like this, you know, the problems are obvious. You would want to do things like this. Please achieve this desired outcome.

That means find an X for which Y equals AX. If you can’t achieve that desired outcome, come as close as you can. These are the types of things you’ll do. You’ll say, “What outcomes can I achieve?” These are the types of questions in this interpretation we’ll see.

Now, in another interpretation, Y equals AX will be simply a mapping or transformation, so it might be a geometric transformation or just some transformation. It might be a coding or a decoding or something like that and you think of X as an input, Y as an output. That simple.

It could be geometric; could be anything. Now, there actually even are combinations of all of these things, where it’s a problem that it’s partly inversion and partly involves something in design or something like that. You can have weird applications where some components of the X are actually primers you want to know. Other components of X are things you can mess with, okay.

That's every common, for example, when you have a system and there are two things affecting the outcome. First of all, when what you do; that’s the part you can mess with, and the other part is what noise or other people of interference does.

So you get all sorts of variations on this, but we’ll come back to these models many, many times. Okay. So let’s talk about estimation or inversion. So here YI is interpreted as the ith measurement or sensor reading, which you know. That’s the idea.

XJ is the jth parameter or to be estimated or determined and AIJ now has a very specific meaning. It is the sensitivity of the ith sensor to the jth parameter, okay. So that’s the meaning of this.

A as a matrix describes the measurement setup, or if you like to think of this as a communications problem, it’s the communication channel. Here’s some sample problems. The most basic one is this – given a set of measurements, find X. That’s the most obvious thing you could ask. Then you could be more subtle and you could say, “You know what, actually find all Xs that result in Y,” and that means find the all parameter vectors consistent with the measurements.

That’s a very interesting thing to do. For example, suppose it’s empty. There are none, right. The answer is there are no Xs for which Y equals AX. Then it says this model is wrong and that could be for various reasons. It might be that in fact, the model is not this, but actually there’s a bit of noise added, all right.

But in any case, it’s better to know that there’s some noise there than not, and the other option is that the noise really is insignificant, but a sensor has failed. That’s another option, in which case this would be a very important thing to know that no X is consistent with the measurement you just make.

That means something is wrong with the measurements or with the model. And that could mean one or more sensors has failed, for example. And that’s a whole area that is widely used, fielded and so on. Health monitoring, sometimes called – okay.

Now, if there is no X that gives you Y equals AX, and maybe that's because of noise and not sensor failure, you might say, “Find me an X for which the outcome,’ if, in fact, the parameter had been X and you believe the model, you’d get AX and you’d like that to match very closely what you observe, okay.

So that says find a set of parameter vectors, which is almost consistent with what you’ve measured, okay. So we’ll spend a lot of time looking at that. These are the types of problems we’ll look at.

Now, when you flip it around and look at the control or design problem, it’s quite different. Here X is a vector of design parameters and inputs. For example – an [inaudible] example would be the mass force example where X is a vector of forces that give you a whole program of what you’re gonna do to [a mass, for example.

It could be the input [inaudible] a vehicle over time. Here Y is a vector of results or outcomes and the Matrix A describes how the input choices affect results. So here’s some sample problems. Find an X for which Y equals Y desired. In other words, here’s desired outcome, please achieve it. That would be one example.

The next would be find all Xs that give a desired outcome. That’s find all designs that meet the specifications. Now, I want to point something out. If you’re doing estimation and the answer to find all Xs that result in Y is a big set. Suppose there’s a lot of Xs that give you Y.

In an estimation problem, that’s not good, obviously, because it basically says you cannot say what X is given the measurements. You can’t. If that set is big, it says there’s a lot of ambiguity in the measurement system, okay. Now, on the other hand, mathematically it’s the same question. It’s identical. It’s find all solutions of A equals Y.

If in fact, in the controller design case there are lots of Xs that reproduce Y. That’s good. Why? What makes that – what is that good? Why is it bad for estimation and good for design?

Student:[Inaudible].

Instructor (Stephen Boyd):Exactly. It means that lots of designs to choose from. Many designs meet the specification, achieve the desired result and it means you can now start optimizing over some other objective. For example, some costs, or the size of that. You could say, "Find me a small X," or if there's an additional cost, you would say, "Find me a cheap X that achieves my goal." So in fact, in this case, ambiguity is good. In fact, ambiguity in this case is design choice, is what it really is. Okay.

And the example here would be among the Xs that satisfy Y equals Y desired if there's many. Find a small one. That's something that generally speaking has the interpretation of an efficient choice of X, okay? By the way, what I'm saying, there's nothing to it, so don't imagine that what I'm saying is deeper than the say it sounds. It's actually less deep than what I'm saying. I just want to point these things out explicitly once.

The third broad category is mapping your transformation. Now you think of this as a mapping and you ask questions like this, "Is there an X that maps to a given Y?" Find an X that maps to Y or you might say, "Find all Xs that map to a given Y." And you might say something like this, "If there's only one X that maps to Y, find it." That means a de-code or undo the transformation. So these are kind of ideas.

Okay, so that's just to summarize the broad categories. You can go back and look at the examples we did and ask yourself in each case which of these three it is. By the say, some of them are not any of these three. They're just – they don't categorize neatly.

Okay. I want to do a little bit of review of matrix stuff. I want to remind you to please read all of this and you can scan it very quickly, all of the notes on sort of matrix, matrix arithmetic and things like that, matrix using vectors on the website. You're supposed to know this material. This is just to make sure the – this is to make absolutely certain that you know the notation and things like that.

But here I'll review some of it very quickly. So we'll look at matrix multiplication. Then we'll interpret it out of any practical context and we'll look at it just as ways to interpret Y equals AX, so one way to interpret Y equals AX is this; when you multiply AX, one way to think of it is you are forming a linear combination of the columns of A; that's what you're doing.

Because if A is written as A1 up to AN and these are the columns, then AX which is Y is nothing more than X1A1 plus X2A2. And so in Y equals AX in this interpretation, this is –there's many ways to say it. It's basically your formula, linear combination of the columns of A. X is the mixture parameters. It tells you how much of each column to mix in.

So if X4 is 0, it says don't throw in any 4 in any A4 into the recipe. Okay. So that's sort of the picture there.

Now, here, this is quite useful. You should know this. If you have X is EJ, the jth unit vector, the jth unit vector is a vector with 0s everywhere and a 1 in the jth position, I do want to warn you about one thing. In some fields, I won't even name them, the vector e without a subscript is meant to be the vectors of all 1s.

I think that's weird and sick. There's probably some people here in those fields. I'm not gonna name those fields. We're gonna use the vector, which is a bold 1, like that, and that's gonna mean all 1s.

Now, this is sort of a transition as we move towards more overloading. After all, we don't write 0 and then make it bold when it's a vector because we've moved to the next stage of overloading where it's not a big deal. So I presume in five years, I'll come along and these bold 1s will become just ordinary 1s. We'll see how that works. Hopefully, the context will disambiguate it, but for right now, that's that.

I just mention this because there are places where E is used to represent these vectors of 1s. Okay. But EJ, I think everyone kind of knows what that means. I think that's quite standard. These are the uni-vectors. If you multiply the jth unit vector by A, if you take the column interpretation it's absolutely clear what it means. It means you are making a mixture of the columns, and what you should use is no column except the jth columns and you should use the unit amount.

So AEJ gives you the jth column of the matrix. Surely all of you have started working on Homework 1, I guess. Anyway, but I'll just note that this is relevant to a problem, I think. Did we assign that problem? Okay, we did, yeah. Good, all right, it is relevant to a problem. Fine.

Okay, so it turns out there's a duel interpretation, a row-wise interpretation. The row-wise interpretation goes like this: When you multiply a Matrix A by a vector, you actually write the Matrix A as rows and now when you multiply that by a vector X, what you're really doing is you are taking the inner product of each row of the matrix with the vector X, okay.

By the way these have different interpretations if you go back to our, like, you know, control or estimation or something like that, this is basically saying that these As for example, in a measurement set up, each A is actually the sensitivity pattern of 1 – let's see if I can get it right, or an input, it's not the sensitivity pattern; it's the effect on all the sensors of an input at jth input. That's what this is.

And this says the overall input you see is a blend of this. In fact, you know what, these would be called signatures sometimes. So this would be the signature of the jth input. It's the pattern of sensor readings you see due to that input, and this says that the overall output is nothing but the sum of the weighted signatures according for each of the inputs, okay.

Here you focus on a row, a row of a matrix in a measurement set up, a row focuses actually on a sensor and so basically this says, this explains it by saying you actually for each sensor, you calculate the inner product, this gives you the sensitivity of a sensor.

So this is just this way, so it's a way to calculate in batch a pile of inner products, so that's another way to interpret matrix multiplication. Now the geometric interpretation of this again, this should be review, is something like this. If my row is AI tilde here, then these dashed lines give you the level sets of the inner product with AI, okay.

So for example, this line gives you that line gives you the set of all vectors, orthogonal to AI tilde. They're the ones that have 0 angle with AI tilde. All the ones that have an inner product of 1 are here – that's 2; that's 3 and in the general case, these are not lines; they're actually hyper planes. So for example in R3, these are planes and you can actually think of the planes as stacked up parallel and the normal to the plane is AI here.

So in this case, what it means to say if I tell you, for example, if I tell you that YI is 1, it tells you about X, that it lies on this line. That's the picture. Now, if I had another column, another row pointed in another direction, that would give me another plane and I could get the intersection and so on.

So this will be useful. Now, the stuff I'm pointing out, now these are all totally obvious things; however, you'll see that soon you'll start putting together obvious things, the chain of 2 or 3 or 5 or 6 and you start getting things that are then not obvious. In fact, it's shocking how few obvious things you can chain together and get something that is not immediately obvious.

Okay. Now, another very useful way to think of a matrix multiplication, matrix vector multiplication is this: when you multiply a matrix by a vector, you think of a transformation and you could write down something called the signal flow graph or block diagram. Now, here's an example with the 2 by 2 matrix. You could think of it as a transformation, two input signals produce two output signals.

Notice it says you get a mixture here and what you do it you write it this way. Now the semantics of a signal flow graph, you don't have to know it, but it's actually quite straightforward. Each node is a value; in this case it's an input value and this is delivered to something that scales by A11 and A12, and a node like this when two things come into a node and the semantics are that you should add the signals, okay.

So that's what this is, and now you can interpret all sorts of things. You can think of like A11, well indeed, that's the game from X1 to Y1. A21, you can interpret as sort of a cross game. It's how much the input X1 affects Y2. If, for example, and it might have an interpretation, might – depends on the application of an interference term.

Right, if for example, these are knobs and these are outcomes, if this were diagonal or nearly diagonal, it means that Knob 1 affects Output 1 mostly and Knob 2 affects Output

2. By the way if it’s a measurement system that’s also good because it means the first thing you’re trying to measure mostly affects the first measurement – that kind of thing.

So sometimes, not always, the cross terms have an interpretation of interference or something like that, but certainly not always. Now it’s not interesting in general to write this out, but it becomes interesting when the matrix actually has a space, non-trivial sparcity pattern. So a sparcity pattern is of course, a substantial set of entries that are 0 as a meaning.

By the way it’s also interesting – it’s traditional when they gain is 1, to not draw it and to simply to draw a straight line – makes sense, in fact. So here’s an example. A matrix is block upper triangular, so it looks like that. What does it mean? Actually I'm hoping sort of after this lecture or actually maybe by the end of this week, you will never, ever look at a matrix as a spark without – you’ll have an autonomous reaction that you cannot help.

When you see a matrix with a non-trivial sparcity pattern, you’ll know what it means. You won’t just look at it and say, “Oh, yeah, sure, whatever,” or something like that. You’ll look at it. This has a meaning. This 0 has a very specific meaning. Well, it’s very easy to write it out. If you write it out in block form, it means this and what this means if you stare at it, what’s missing here is the A21 term and it’s missing right over here.

And what it says is that Y2 depends only on X2, so the second block of outputs don’t depend on the first block of inputs; however, the second block of inputs do affect the first block of outputs and that’s what that means and you should just – when you see that, that’s what it should mean to you. They’re the same thing.

Now, when you draw the block diagram, well it’s kind of obvious, you get this. And you know, you look at this and your eye is now telling you what you knew anyway which is the following, X1 doesn’t’ affect Y2. Why? There’s no path in the signal flow diagram from X1 to Y2. So that’s what it means, okay?

Oh, these are kind of obvious things. Okay. Let’s look at matrix multiplication. You know what matrix multiplication is; if you have an M by N matrix and an N by P matrix, the inner dimensions have to agree. You can multiply them and you’ll get a matrix which is M by P and the formula is this; it’s CIJ is the sum over K, intermediate variable AIKBKJ, like that.

Now, matrix multiplication comes up a lot. It has lots of interpretations. We’ve been looking at a special case where B is N by 1. So matrix multiplication, though, has lots of interpretations. That’s one of them. Now, 1 is the composition interpretation. Supposed you have Y equals CZ where C is AB. What this really means is something like this, Y equals AX and X equals BZ.

So let’s see Y equals – did I get this? Y equals AX and should I put a Z in there? No, no, I have it right. Sorry, Z is the input. Sorry. I confused myself. It represents a composition and in terms of all-block diagram, you write it this way, that Z gets mapped, first it goes into B; that produces BZ, which is X. That's here. And I've noted the dimensions of these signals here. X then goes into A and it comes out as Y, like this.

Now, the one thing you'll notice immediately is that block diagrams in algebra are backwards. Sorry; it just worked out that way. Somewhere around 1850, somebody made the terrible mistake of ordering the indices the wrong way or writing from left to right on paper or whatever. Anyway, it just didn't work out. You know, it's kind of like driving on the left or right, I guess except there's no intrinsic difference there.

Here, it just sad that algebra – it goes backwards from block diagrams, but after awhile you get used to it and in fact, the way you can really make this really understand it is when someone walks up to you and you think of AIJ, the J indexes the input and I the output, so you can think of it this way, as input to output, so A32 is the gain from the second input to the third output, whereas had this all been done right 150 years ago, everything would have worked out.

Now, by the way, there's weird subfields that have tried to do something about this. A friend of mine at Cal Tech decided about ten years ago, he had some kind of religious conversion he decided that he would henceforth write all block diagrams going from right to left. He stopped after a year or two, but anyway, it just – I don't know. It was fine, but I mean so what? So he did that.

There are fields, probability is one of them, where they decided instead of using column vectors as your basic vectors, you'd use row vectors, and then everything works out again, because vectors are row vectors and you multiply and block – all your intuition works there the right way.

So and that's even arguably kind of right, but it's not the way other people do it, so sorry. So anyway, you just remember this picture. Just block diagrams and indexing in algebra as practiced by the vast majority of people who do algebra sadly are backwards. Okay.

Now, there's lot of other interpretations of C equals AB. I'll show you some. Here's one – you can multiply AB. It's actually a batch matrix vector multiply. So basically it's this; you take B and you write out its columns, and then it's actually A operating on the first column and AB. This is very useful. It means that if you ever need to multiply a matrix by a whole bunch of vectors, you can put those vectors into a matrix and do matrix multiplication.

I'm going to say something about that in a minute. It's not obvious and there might only be a handful of people here who actually know what I'm gonna tell you.

Okay, now you can also write it out this way and it's block, things like that and I'll give one more interpretation which is an inner product interpretation and here it is. The inner product interpretation says that when you multiply two matrices, the IJ entry is in fact the inner product of the ith row of A with the jth column of B. That's what it is.

And you know this because that is after all, how you think about when you multiply two matrices and you want to get some entry, you go across the ith row, like this or I don't know how you do it, but this is how I do it and you go down the jth column like that and you go down and you're doing a cumulative sum.

Or if you like, you just think of that as an inner product of these two vectors; okay? So that's it, and you get all sorts of things. If you have a bunch of vectors, this thing called the grand matrix, that calculates actually all the inner products of a set of vectors, okay? Sop that's the grand matrix.

So let me ask you a quick question on this right away. Suppose that A and B are square matrices and AB equals I. Now, you know what that means; it means that B is A inverse and it also means that A is B inverse. A and B are square, okay? Now, I want to ask you about this. Tell me, what is the inner product of, let's let AI tilde be the rows of A and let's let BJ be the jth column of B and I would like to know, what does that equal to?

Student: Zero.

Instructor (Stephen Boyd):Zero – provided I is not equal to J, okay? So you've just deduced the following; the – let's see if I can get it right. The ith row of a matrix is orthogonal to the jth column of its inverse provided I is not equal to J. What's the inner product if I equals J – 1 exactly.

By the way, this fact will come up in lots lf cases and it will look totally magical and it will look like it came out of the blue and it's totally elementary. Okay. Let me say something about matrix multiplication via paths. You can – if you understand the block diagram B composition. AB is actually, and the way to think of it, of course, is this, is when you see Y equals ABX here, you should think of it, you should immediately write that down.

In fact, of course you can associate it anyway you would like. But this is the way, as an operator, you should interpret it first and what this means is that B is first, even those B is on the right and that's why this diagram goes over here, like that, okay?

So this is AB and what's very interesting here is this term, AIKVKJ; that's the gain of a path from X1 to Y2, but it's the path that goes via Z2. And you simply multiply this gain and this gain, okay?

There's one other path, by the way. That's this one and if you add these two path gains, you will get exactly the, now let me get it right, the 2-1 entry of C, which is the product. So again, what I'm saying is obvious, but you put a bunch of these things together and all of the sudden, things are not totally obvious or something like that.

But in generally CIJ is the sum of the gains over all paths from Input J to Output I and in fact, specifically, when you sum over the K there, the K, if you wanted to put a comment in your code or whatever, K has a meaning. K is the intermediary node.

In fact, you would even literally say it's the sum over all paths from Input J to Output I via Node K. That's exactly what it means. So things like this should not be just definitions. They have a meaning, and this is the meaning. Okay, now I'm gonna say something; maybe some of you know this, maybe not, though, because they don't really teach this.

Supposed you have [inaudible] matrices, all right. Everybody knows the formula, CIJ is sum on K. AIKBKJ – here we go; there's the formula. All right, now supposed you were gonna write some code for that. Doesn't look too hard. Looks to me like three loops or something like that for, your loops on I, you loop on J and then you run a sum on K. So it's like three lines of C or something like that, maybe four.

Okay, but whatever language you like, it's like four. Everybody understand that? And the number of operations you do, is gonna be on the order of, let's just make them square so I don't have to think about it, so they're all N by N and the number of operations you're gonna do [inaudible] is on the order of N cubed, okay? Okay.

Now, I'm gonna ask you some cool questions – so here's one. Could you do it faster – maybe some of you know. Does anyone know? Yeah.

Student:[Inaudible].

Instructor (Stephen Boyd):You got it. Okay, this is totally bizarre. There's actually an algorithm which is order ended at 2.8, whatever and Jacob you could type in or you have a browser and find out what the current number is and the number, you know, varies. It goes down and it goes down by small amounts. You know, it's a huge big deal if you make a 2.799, or I don't even know what the number is.

It means nothing to you, I can tell. This is a hard – you know, I remembered that from the first day. It was like we were estimating things with the three big quantize and we were getting like, you know, six-bid accuracy and you were like, "Yeah, that's cool."

All right, so there is in fact, an algorithm that will multiply this out and it's something. I'm gonna leave it there because.

Student:[Inaudible].

Instructor (Stephen Boyd):It's 2.41? Wow, okay, wow – in progress. Okay, 2.41. All right, by the way this method actually at the moment has no practical use whatsoever. Now, it's an extremely interesting topic in computer science. It doesn't have, the reason is that the constant in front is absolutely enormous and you'd have to have matrices so big that you can't store them all that kind of stuff but you never know, some day it might actually come up.

I'll tell you a little bit in a minute how it's done, although it's related to the – now let's go. That's the theory; let's talk about the practice. So here's my – I have a practical question for you and it’s this. Everyone here could write a four-line C or whatever program to multiply two matrices. Let’s say thousand by thousand. Everyone here could do that and it would compile and it would work fine and it would produce the product.

It would produce C, given A and B, right? Now, the question would be, oh and everyone would be doing because no one here is gonna write the super-fancy stuff that does this while doing that formula right there. That’s what we’re implementing. We’d all be doing the same amount of floating point operations and the question would be could someone else, me or someone else or something like that, write a function that multiplies two 1,000 by 1,000 matrices faster than you? That’s the question.

What do you think? This is not exactly a complicated – it doesn’t look like a complicated thing to do, right? It doesn’t look to me like there’s a lot of room for programming creativity here. Anyone know the answer? What’s the answer?

Student:[Inaudible].

Instructor (Stephen Boyd):There we go. Okay, perfect – so I told you; I told you a few people in here would know. Most of the people don’t. Guess what? Someone who knows what he or she is doing will beat you so badly you won’t even know what happened in a thing like this. It could be if you’re lucky, you would be slow only by a factor of 4. If you’re unlucky, you’d be slow by a factor of 100 or more, okay?

So, now that’s pretty bizarre. You are doing exactly the same floating point, exactly the same calculations. You’re doing 1 billion calculations, so that’s not the difference. The difference has to do with memory and the cache hits and misses and things like that. Let me say how that’s done, just for fun. How many people knew this or knew about this – so certainly some.

Okay, so by the way, do you have to write this code? No, it’s all automated; it’s all open source. Public domain has been used for 15 years and in fact, when you go to [inaudible] lab and you type in A times B, guess what happens? It calls this open source library; it’s called LA path, okay?

So let me tell you how that works. The way it works it this. It’s another interpretation of matrix multiplication. I’ll just say a little about this because it’s fun and the stuff we’re doing now is so boring, that it seems like I should introduce things so people don’t know. So what you do is this; suppose your matrix is, I don’t know, 1,000 by 1,000 and I block it up into, I'm gonna make it actually 1,200 by 1,200 and I block it up into little chunks of 400 by 400 matrices, okay/

If I multiply them, the same formula works. In other words, I multiply this matrix times that one plus this matrix times that one, plus this matrix times that one and that gives me the one-one block of the product. So you can multiply matrices in a block way. Okay? Everything works. You can just; you can multiply matrices. Now, if you add up the number of operations you just did, congratulations. It's exactly the same, so that didn't change and you know, how could it have changed. That's kind of stupid.

Actually, if you want to know where does that 2.4 comes from, it comes from this. It's the observation that when you multiply two 2 by 2 matrices, there is a way to multiply them. This is the most basic one to get less than 3 is when you multiply them, there's some weird way to do it like you know, with 7 floating point operations, as opposed to 8.

Now, you [inaudible] that, so you take a big matrix, you divide it by block 2 by 2 and you do the 7, you do that smaller and smaller and you're gonna actually find out you get M to the 2. something; however, it's the same idea that works in practice. So the way this works in practice is why might it be faster to block a matrix than multiply it? I know there's ten people in here who know. How about some of the others?

Student:[Inaudible].

Instructor (Stephen Boyd):What?

Student:[Inaudible].

Instructor (Stephen Boyd):Precisely, exactly. Right, so if for example, one of these guys fits in L1, registers or something like that, it's gonna be way fast. The whole thing's gonna get pulled into L1. All the operations will be L1 and you won't have to wait for that, or if this is really big, disc stuff which is like an eternity.

Okay, so actually with the memory locations are adjacent. You have a very nice memory flow and data flow and stuff like that and something like this. So, the way this really works is this, on a processor, you run something that's called ATLAS, which is Automatically Tuned Linear Algebra Software. It runs on your machine and it determines optimal cache sizes for your processor, your L1, your L2 and things like that.

And when you then say in effect, in that lab when you multiply two 2,000 by 2,000 matrices, you will be using an optimized BLAST. BLAST is Basic Linear Algebra Software and it will block it up into, I don't know, 30 by 30 and then below that, something smaller and I don't know. But the point is, it will be way, way faster than if you wrote out in your four-line C program.

Okay, so, I said something that now, especially those who have a strong mathematical background, and are currently bored out of their mind, didn't know, for the record. How many people care? Oh, well, that's another story.

Okay, what I'll do now is, this is gonna be on a review, this lecture. We're gonna cover a lot of stuff that you should have seen before, but every year 20% of the people in the class actually haven't seen it before and they generally do just fine.

So I'll define things like vector space and sub spaces, but I'm not gonna go into the horrendous detail that you would in a normal linear algebra class. So a vector space – that's sometimes also called a linear space over the reels, although occasionally we will look at vector spaces over other fields like the complex numbers.

We won't look at it, but for example, bouillons are finite fields like that. And it consists of a set of the points and they're called points or vectors. You have a vector sum and that's a function that takes this argument two vectors and returns another vector. And you have a scalar multiplication, but the vector sum is denoted – you don't write it this way, you know. You don't write it as V-sum XY, although you could.

It's actually traditional to write it this way. But in fact, it's really that. And there's a scalar multiplication that takes the scalar argument in a vector and returns a vector and again you know, you don't write it this way, following this thing, alpha X. You simply write it as alpha X for compact notation.

And you have a distinguished element 0 in the vector space. That's not the 0 number and in fact, 15 years ago, you might have made that 0 bold or something like that to sort of give you a hint that you're overloading it and it's not the number 0. Now, these have to satisfy a list of properties. They are – and I won't go over, you know, your model for vector space is something like M vectors, and I won't go over this in too much detail.

Has to be commutative. Has to be associative and that means that when you add them, you can add them in any order. Zero has to be an additive identity. There has to exist a negative for each vector. Scalar association is associative; there's a right distributive rule, a left distributive rule and the scalar 1 multiplies by X has to give you X.

So it has to satisfy these properties. And let me ask you a couple of questions, how about doubles in a, if you write just the arithmetic of I triple e floating point standard, let's say doubles. Is that commutative?

Everyone know what I'm talking about? I'm talking about writing a one-line program where you add two numbers.

Student:[Inaudible].

Instructor (Stephen Boyd):Okay, this is associative?

Student: No.

Instructor (Stephen Boyd):No, exactly. It is not associative. It's awfully close, but it is not associative. Okay, so you can get round-off error easily in this case, okay. It's not exactly, so it's very, very close, but it's not exactly. Okay. I won't go into any of the others? How about – is scalar multiplication associative? Just for fun.

Student: No.

Instructor (Stephen Boyd):No, it's not because you could underflow or overflow something with these things. There are things here that you really don't have to worry about too much, but it's good to know that in fact, these are already not true for doubles on a computer. These would be true.

You can – there actually are systems for algebra and other things, which are infinite precision, in which case, all these things hold exactly. So for example, if you store numbers, as rationals as ratios of integers, then you can arrange for all of these things to hold exactly. And there are systems that do that.

Okay, let's look at some examples. Well, the [inaudible] example is just RN, so that's the set of N vectors plus the column vectors and you have standard component wise vector addition and scalar multiplication, so that's the standard. Here's another one; it's a big silly.

It's a single vector in RN that has all 0s in it. Here's one. It's the span of some vectors where the span of some vectors is the set of all linear combinations of them. And these are some particular vectors in RN, and so that's – now you have to check that these are, in fact, that these are actually vector spaces and you have to check various things.

The way people, and the way you think of this checking process is something like this. You check that for example V3 is closed under-scalar multiplication and you check whether it's closed under addition; that would be the thing you do, so you'd ask yourself if a vector is a linear combination of V1 through VK and another one is as well and you add those vectors, is that a linear combination?

The answer is yes it is and you can construct the linear combination from the other two linear [inaudible] by adding the corresponding coefficients, okay?

Now, a sub space is a subset of a vector space that is itself a vector space. So that's a sub space and it means something like, you know that it's close under vector addition and scale in multiplication and these examples that we just looked at were sub spaces and again, this should be review.

And let me mention a few, something that you may not have seen, which are infinite dimensional vector spaces, so let's look at here's a vector space. It's gonna be this. It's the set of all functions that map R plus, that's the interval of 0 infinity into RN for which X is differentiable, okay?

Now, this is quite complicated because points in V4 are themselves functions. So actually if you like, you could think of things like a short, but complicated C declaration or something like, for something like V4, so it's elements are themselves functions that take as argument, a non-negative real number and return a vector.

So by the way, you can think of these as trajectories if you like of something, like a trajectory of a vehicle where X1, X2, and X3 are the positions and something like that and X4, X5, X6 are the velocities or something like that; it could be anything. It's a trajectory.

Now, you have to – you can't just say, "Here's the set." You have to say what the sum and what the scalar multiplication is. So how do you add two functions? Well, we have to define that and the way we define it is this, so this plus is what plus? Is that the plus of two scalars, two vectors, RN?

That's not RN. That plus is what plus. When this is resolved by context what plus function actually does this refer to? Is the plus in V4; it's the plus that adds two functions because the data type of X is a point in V4 which is a function. That is function addition. So what's the data type of X plus Z?

It's a function. It's a function taking as argument a non-negative element and returning a vector. You then call it with the arc T and what is the data type of that? That's RN because that's the return type of X plus Z. That's RN, okay.

That is also RN, that is RN, what plus is that?

Student:[Inaudible].

Instructor (Stephen Boyd):That's the plus in RN and that equals what is the equals in RN, exactly. Okay, so I won't do that much more, but the problem is when you see things like this, you know, in fact you look at it and it just looks so innocent. You just rearrange the same stupid, or whatever; it's like 8 ASKII or 7 ASKII characters or 9 or whatever it is, you just rearrange these characters a little bit.

So watch out because these little rearrangements of 9 ASKII characters on the left and right-hand side to which it's very easy to look at and go, yeah, yeah. It can mean a lot because of overloading, so watch out.

Okay, scalar multiplication is you have to say what it means to multiply a scalar by a function, and that's defined this way, okay. Here's a sub space. If the set of differentiable functions or trajectories here that satisfied X. equals AS; that's a linear system, so you now know that the set of linear, the solutions of an autonomous linear dynamic system is the sub space, okay?

Okay. There's the concept of an independent set of vectors. You say that a set of vectors is independent if the only way to make a linear combination of 0 is by putting in all 0 in the recipe. That's when it needs to be independent. Now, what I'm saying is very, very important here.

Independence is an attribute of a set of vectors. It is not an attribute of a vector. Okay. So it makes no sense to say, let me tell you the slang. On the streets, people say this is a set of independent vectors. Anyone would understand what you just said if you said it's a set of independent vectors.

But there's a big difference between – you'd say that the same way you'd say this is a set of non-zero vectors. When I say this is a set of non-zero vectors, it means it's a set of vectors, each of which is non-zero.

If you use the same expansion, for this is a set of independent vectors, you mean something like it's a set of vectors, each of which is independent. I mean that has a meaning and it is absolutely not the same as what the person meant.

So we'll try, I will slip into it. After a few weeks, I will talk about an independent, so let's say it right. The correct way to say it, maybe we'll just practice it all together is to say, to say, I can't even get it – let me try. I'll try very hard. You're talking about an independent set of vectors. That was correct. You don't talk about a set of independent vectors.

Now having said that on the streets, you'd talk about a set of independent vectors. But I want to make very clear, it makes no sense, the argument of independent is a set of vectors; it's not a vector.

Okay, so what does this mean? To say that the only way you can add some vectors up and make 0 is by plugging in 0 coefficients. It says basically that the coefficients are uniquely determined. That's very interesting. It says that basically if I make a vector with one mixture of the Vs, and if I make it up with another mixture of the Vs, it turns out the recipes are identical.

That's what it says. How do you show that? Well, you subtract this from this and you get sum alpha I minus beta I times a VI equals 0 and then if they're independent, you conclude immediately by this that alpha I minus beta I is 0, so the recipes are the same, okay.

Here's another way to say it, no vector, if you have an independent set of vectors, it's like if you can watch me, when I slip back into the slang you can start complaining or something. I really should just should always say independent set of vectors. So, all right, so no vector in an independent set of vectors, can be expressed as a linear combination of the others and that's if that's true it's an independent set of vectors.

Okay, and you can sort of check. I'll let you work out to convince yourself of these things, okay. Basis and dimension – well you say that a set of vectors is a basis. Basis is like independence. It's argument, it's a set of vectors; it is not an attribute of a single, of the vectors individually, like for example, non-zero or normalized which means the norm is 1.

So you have a set of vectors is a basis for a vector space if they span the vector space, and what that means is anything in the vector space can be written as a linear combination of these vectors and if they're independent and you know what that means. It means every element in the vector space can be uniquely expressed as a linear combination of these basis vectors. Okay, so that's what it means to be a basis.

We’re gonna look at a lot of examples and things like that, so if this is going by too fast, basically I'm going fast because first of all, it’s review and second at this level, it’s kind of boring. Okay, so here’s a very basic fact. If I have a vector space, there are many bases for a vector space – many.

But it turns out there’s an invariant among those bases, so in other words, there’s one attribute of a basis that doesn’t change, and that's its cardinality. So the number of elements in a vector space, never – it’s not the same. You can’t find a basis with six elements where you found one with seven. It’s impossible.

If one basis has six elements, all bases have six elements and that’s called the dimension, that unique number. Now if there’s no basis, then we say the dimension is infinite, okay. And I want to mention something here that’s something a bit different. You will hear the word basis used in other contexts. For example, you might be taking a course in Fourier transforms or something like that.

And people will talk about something like this. They’ll say that sine, you know, KT and cosine KT where K equals you know, 0, 1, blah, blah, blah. That’s an infinite number of periodic functions, but these form a basis for sort of periodic functions, 2 pi periodic functions. So you’ll hear people talk about that. That’s actually not a basis in this algebraic sense.

It’s a different concept I just mentioned. So there’s a question?

Student:[Inaudible].

Instructor (Stephen Boyd):That’s right.

Student:[Inaudible].

Instructor (Stephen Boyd):That’s correct. This is the definition of basis. In other contexts, for example, in some infinite dimensional spaces, actually I'm not even sure they use the word basis there. I actually have to go back and check. They might hedge. So what this is is, I guess a basis there means that any element can be written as an infinite sum. For us that’s not the case. That’s not what a basis means. It means a finite sum.

And actually for someone doing just linear algebra, it means finite sum – actually just algebra, not linear algebra, just algebra. It means a finite sum. Just to warn you, you hear basis in different contexts. Another weird one is this, which I find very strange, but this comes up in signal processing now. I don’t know how they got away with this and even statistics. Now, let me say what it is.

Suppose I have a vector space of Dimension N. Let’s say it’s R10, okay. So the dimension is 10. And let’s say – if I come up with, you know, a basis V1 up to V10, just make these the unit vectors. So there, E1 through E10. That is a basis for R10, okay?

Now, suppose you add some other weird stuff. Suppose someone walks up to you and says, “Here’s my basis, V55,” okay. If I say, “Here’s my basis for R10.” You would say, “No, no, no, come on. That’s not a basis. You’ve got 55 elements and the dimension is 10; it’s not a basis.” And they go, “Oh, you’re still stuck in that old idea of basis.”

This is, they call this an over-complete basis. I'm not kidding; this is not a joke. This is actually intelligent people on this campus. They say stuff like this. I mean it’s weird. There’s a name – and you say, “What the hell does that mean, over complete basis?”

For hundreds of years a basis has meant independence and spans, and they go, “Oh, yeah, that’s what it is, except they’re not independent.” They go, “Okay, but we have a name for that; that’s called expand the vector space.” That’s what we’ve called it the last 300 years.

And they’re like, “Oh, oh, no, this is like wavelets and blah, blah, blah.” And you know, this is why image processing and modern this and that, and it just, anyway makes no sense. But that’s fine; they can do whatever they like. They can take a term that’s been used with no ambiguity for several hundred years, turn it around and use it some sick way for their stupid sub field, that’s fine with me.

So you will hear things like that, over complete basis, which is just like a joke, I don’t know what it would be. It would be something like a non-true theorem. It’s a new concept really and it describes a theorem. It’s like a theorem; it makes a specific statement, but it’s false. Or it’s not always true and so that would be a new thing, and we’d call it a theorem, but it’s like a theorem, but it’s just not always true, you see.

So but anyway, you’ll hear it there and you’ll hear it in other places, too, but I think they call it frames. That’s another name for it in signal processing, okay.

But all of these things you’ll find a local dialect will emerge when they start using these terms and then the local dialect, it depends on how isolated it is from the rest. You know it’s kind of like evolution in Australia or something like that. And some of these fields are like pretty weird, right and they go out there and they go off and they have a whole theory and they make whole new names for things like basis, independent and so on.

All right, so we’re now gonna get to – we’re gonna keep going on this review of linear algebra. And this is the part in linear algebra that really I mean I guess most people when I think back, when I talk to people and I say, “What do you remember of your linear algebra class,” most people say as little as possible.

But when, in fact, the memories come back, they remember somebody at the board droning on and on about the four fundamental sub spaces. And I’d say, “Well, what are those,” and they’d say, “I don’t know.” Or they’d remember some of these things; so that’s what we’re gonna talk about now.

A quick overview – I know a lot of you have seen this; this is review. So I'm gonna emphasize is not actually the technical aspects which I presume was covered in linear algebra class you saw. I'm actually gonna concentrate on what the meaning is.

So the null space of a matrix, of an MIN matrix is defined as this. It's the set of points in RN, that's sort of the – I think of that as the input space that gets mapped to 0 by the matrix. That's the null space, okay?

So null space is actually itself an operator; it's a complicated one. It takes its argument and M by N matrix and it returns, the data structure, the data type it returns is a set. In fact, it's a sub space – thus its name.

It returns a sub space of the input space, okay. So it's easy to just look at this, but these are very complicated operators and that's what the null space does. It takes its argument a matrix and it returns a sub space of the dimension of the input of that matrix, if you want to call it the input.

Well, it's obviously, it's a set of vectors that's mapped by 0 by Y equals AX and is also interesting. It's also the set of vectors that are orthogonal to all the rows of the matrix A. Why – by the row interpretation of the matrix vectors multiplication. Because to say AX equals 0 says the first row of A in a product with the vector, that's gonna come up backwards for you, but you'll – I should learn to do that backwards, but I probably can't so is 0, and that would be true for all of them.

Now, the meaning of it is this. It gives exactly the ambiguity in Y equals AX. The ambiguity in X in Y equals AX, because it says this. If you have Y equals AX and you have any element of this null space and you add that element to X, then A multiplied by X plus Z gives you, if you expend it out, AX plus AZ, but AZ is 0 and that's AX, so you get another solution, okay?

That means if you have a solution to Y equals AX and you add to X anything in the null space, you're gonna get another solution. By the way, there is, the null space is never empty. It always has the 0 vector in it, okay, because the 0 vectors is always mapped to 0, okay?

All right, so that's the null – so it says that if there's more than a 0 vectors in the null space, it basically says, there's gonna be ambiguity in X if you have Y equals AX, all right. Now, turns out it exactly characterizes it. And that means this – suppose you have two solutions, Y equals AX and Y equals AX tilde, then it turns out that in fact, X and X tilde are related because they're difference is in the null space.

So it basically says this idea of adding something from the null space into a vector to give you a new solution that's completely general. All solutions come about that way, okay. And that's easy to show.

But let's think of some examples, just real quickly. Here's one. Tell me this. Let's think of the forced mass example. Let's make for ten steps. So X is the forces you apply over ten seconds, each for one second and YAX is gonna be the final position in velocity. That's a 2 by 10 matrix A, okay and what I would like to know, is tell me about the null space, I mean not a lot, but what does it mean?

First of all, is there any element in the null space other than 0 and what would it mean? What's the answer? What is the meaning – what have I said? This is in the null space and I told you [inaudible]. There's like 1 minus 3 plus 5 minus 8, 0, 0 plus 4; how would you check me?

Student:[Inaudible].

Instructor (Stephen Boyd):Yep. You would think of the product with both rows, which means you would actually multiply A by – you'd take my F; you'd multiply it by A and you'd get 00. If you didn't get 00, you'd say, "It's not an element of the null space." If you get 00, it's an element of the null space.

Now, my question to you is what does such a forced program mean?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay, so I would say that my interpretation of such a forced vector is it's a joy ride. It means – no, the condition is this. It's any vector of forces that has the following property. It acts on the mass, okay. It can shoot off to the right. It can shoot farther to the right. It can shoot off to the left and go back and forth, but after ten steps, it must land exactly at the origin and with 0 velocity.

In other words, it's a set of forces that ultimately does nothing, okay? Now, by the way, is that a good thing or a bad thing? Well, if you're paying a bill for the fuel, that's probably a bad thing, but it may not be a bad thing. I mean it's neutral. Okay, so now the question is somebody give me an element of null space.

Student:[Inaudible].

Instructor (Stephen Boyd):Zero – that was good. That was, okay. Somebody give me a non-zero element of a null space. That was very good; that was very quick, very good, and it was correct. How about a non-zero element of a null space?

Student:[Inaudible].

Instructor (Stephen Boyd):So you're forced, you wanna go 1 minus – 1, and what 0s, after that?

Student:[Inaudible].

Instructor (Stephen Boyd):I don't believe you; you know why? Your final velocity will be 0, but your position will be 1, so that's not gonna work.

Student:[Inaudible].

Instructor (Stephen Boyd):Another minus 1? No, no, no, no, then your velocity – how about this, minus 1, 1, and then how about that. Does that work? That works. This moves you, so you're now stationary after two steps you're stationary, one meter to the right and then this undoes it and moves back. There's an element in null space, great. Any others – or is that it?

Student:[Inaudible].

Instructor (Stephen Boyd):5 minus 1s?

Student:[Inaudible].

Instructor (Stephen Boyd):And five 1s. No, no, I don't think so. No, no, no, no. That's – your velocity will be 0l but you're gonna be way, way off to the left. So I don't like that one. Maybe we could shift this one. We could put – we could shift this to the right and generate a lot, because the point is all this discussion suggests the following. There's lots of elements in the null space, okay?

And actually it already means something interesting. It means this. It says when someone says you have ten forces to apply to the mass, I want you to be at a certain position in velocity at the end of the this ten seconds, there's a giant pile of force programs you can use to affect this; that's what it says.

They're characterized in fact, by the null space, along with a particular solution. So that's what it means, Okay. That was just to kind of see what this looks like. That's what the null space means. Okay. All right, so if a matrix has only the 0 vector. If the null space, right, so that means the null space is 0, by the way that's the set consisting of the 0 element. I probably don't have to say that. That is not this and it is not this, okay?

This makes no sense. This one makes sense, but is impossible and wrong, okay? So this is, I guess we would call this a semantic crime. This would be a syntax crime, right, because it doesn't even make sense. That's a set and that's a single element. So I mention this, right. Okay. So that's just a little, a small syntax rant there.

So if the null space of A consists – is the set consisting of one vector. Now, by the way, the way you say this on the streets, is you say 0 null space or the null space is 0. Okay, that's the correct way to way it, just so you know. If you're deposed or there's lawyers present, that's what you say.

But otherwise, you know, informally, you say, "Yeah, a null space is 0 or something like that. That's fine; but you would never write that. Now, this has a huge meaning. It says this. It says the vector X can always be uniquely determined from Y equals AX. Now that's very interesting.

It says basically that if you think of this as a transformation, no information is lost. It means we can undo it. By the way we do not get no how to undo it. We'll get there, but the point is it means at least in principle, no information is lost. It can be undone. If I give you Y, I can reconstruct X. We will very shortly see how to reconstruct X.

Okay, so you don't lose information. By the way in an estimation problem, this or reconstruction problem, this would be very good. It means your sensor sweet or the measurements you proposed to make are good enough that collectively from your measurements you can deduce the parameters you want to deduce.

Okay, that's what it means. It's a good thing in that context. In the context of design, it's maybe a bad thing, possibly. Maybe not – in terms of your leisure time it's good because basically it says there is no design problem. Once someone specifies why, there's only one X; it satisfies it. There's nothing for you to do. So that's bad, by the way, from an employment perspective.

Now, this also means the mapping from X to AX is 1 to 1. It means basically that if you draw one of these pictures like this, and you show elements here in RN that get mapped to RN, it says that different things go to different ones and if you don't have this, you can't have two things landing on the same element.

Or another way to say it in a context like one of the things we look at, it would basically say something like this. You can't have – let's go to geophysics, you can't have two subterranean patterns of densities that produce the exact same gradiometer readings above earth. That would be great if it were true. It's not, by the way.

But if it were true, it would be great. It would mean in principle, you know, using some big computers you could reconstruct exactly what was below you, okay.

I'll mention just a few things and then we'll quit for today. Another way to say that a matrix is 1 to 1 – that's also slang for it, that the – it's not slang; it's okay. It's 0 null space or 1 to 1. It says that the columns of A are independent, so it just rewording the same thing.

Why? It basically says to say that the columns of A are independent, says that whenever you make a linear combination of A of the columns and it adds up to 0, then the linear combination of the columns, that's linear vector multiplying. So whenever you matrix vector multiply A by a vector and you get 0, the only possible way that could have happened is if that vector was 0. That's the vector of coefficient. That's just a restatement of the same thing.

We'll start on this next time; this is kind of where it gets interesting. And this is not obvious, this one. Not remotely – this says that if a matrix is 1 to 1, it has a left inverse.

That’s really cool. That’s a matrix. You multiply on the left by the Matrix A and you get the identity. Okay, so that’s the picture.

By the way, this is extremely important. Let’s think about what it does. If you have Y equals AX and let me multiply both sides of this equation by Y, I get BY equals BAX which is IX which is X, okay?

That says if you have a left inverse of a matrix you should think of it – what it is is it’s a decoder. It is a perfect decoder, a left inverse is what it is. If this is a channel in a communication system, this is a perfect equalizer.

If this is some kind of measurement – if this is a measurement system, this B is the matrix that processes your measurements and gives you what the parameters are. It’s a perfect thing that undoes A. It’s also linear anyway. So we’ll continue with this next time.

We haven’t said how to find a left inverse, but you will know very shortly how to do that.

[End of Audio]

Duration: 80 minutes

IntroToLinearDynamicalSystems-Lecture04

Instructor (Stephen Boyd):I guess we've started. Let me start with a couple of announcements. The first is I guess I should remind you that Homework 1 is due today at 5:00. Several people have asked what they should do with their homework, and I should have said this maybe on the first day, although I guess its most relevant now. One thing you should not do is give it to me. I am highly unreliable. I will accept it by the way, that's why I'm warning you. You'll give it to me. It'll be in my hand. By the time I get back to Packard, who knows where it will be?

The TAs are more reliable. You can give your homework to a TA if you personally know them and trust them. But really, the correct I.O. method is that there is a box; it's actually a file cabinet; it's explained on the website. But it's near my office in Packard. That's the right place where you should submit the homework.

Okay, the next is that we have gotten more than just a handful of sort of requests or suggestions to do something like a linear algebra or matrix review session. That's for people who either never had a detailed course on this or have successfully repressed the memories of the course they may have taken. So for those people, the TA's and I have been talking about it, and we'll probably do something. It won't be a formal session. It might simply be one of the office hours. One block of office hours will simply be devoted to this topic. And we would of course announce that by email and on the website. So that would sort of be the idea.

Just out of curiosity, by a show of hands, if such a session were to be put together, how many people would go to it? We'll have a session, that simple. That's what we'll do. Okay, any questions about last time? If not, we'll continue.

So, last time we started looking at this idea of the null space of a matrix. Now the null space is a subset. It's a set of vectors, it's a subspace, in fact, as its name implies. And it is a subset of what I would call a dimension of the input dimension. That's what it is. So, it's the vectors that are mapped to zero. Now what it gives is, this gives exactly the ambiguity in solving Y = AX. So, it tells you exactly what you know and don't know when I give you Y, about X when I give you Y in Y= AX.

In particular, anything in the null space is going to map to zero. So for example, in a sensor or estimation problem, X corresponds to some pattern or input for which your sensors all read zero. So for all you know, it's either zero or it's something in the null space. If the only point of the null space is zero, then that means that's the only way that can happen. This is what it means.

Now in a design problem or in a control problem something in the null space is a very interesting thing. It's something like this; it's something that has, you might say, zero net effect. So in the case of the mass problem, it would be a sequence of forces were non-zero, so the mass goes flying off to the left, right, goes all over the place. But the point is that N seconds later, it is back at the origin with zero velocity. So that means that basically nothing happened. Okay? So that's the idea.

Now one case of great interest is when the null space – I can't see if – It's not me but that's twisted at an angle, so maybe you guys can rotate your camera there a little bit. There we go. Too much. There we go. Actually, that's still too much. I'll let you guys figure it out back there. So you say that you have zero null space, that's the slang. That means, simply, not that the null space is zero. The null space can't be zero. The null space, the data type is it's a set of vectors. So when you say its zero, you mean it's a set whose only element is the zero vector. But the slang is the null space is zero. And some people call this 1:1, and what it means is this, it is the same as saying that X can always be uniquely determined from Y = AX. It basically says when does the transformation A not lose information? And 1:1, of course, means that different X's get mapped to different Y's. That actually, literally, what 1:1 means. Otherwise you might call it many to one, would be the other way.

In fact, one too many is not a function, because the definition of a function is that only one thing comes out. Another way to say this is that the columns of A are independent. So when you say something like zero null space, you're really just saying that columns of A are independent. So, it's two different languages for exactly the same thing. And you can check that. Why does it mean that? Because when you say AX = 0 implies X = 0, AX, one interpretation of AX is it's a linear combination of the columns of A where the X's give you the recipe, the coefficients. So basically it says that the only way to add up a linear combination of the columns and get zero, the only way, is if actually the recipe is trivial, if all the X's are zero. That's the same as saying the columns are independent.

Okay, now this part, everything up here is easy to check, or it's basically just a restatement of – I mean, there's nothing to it. This is not obvious, and so a few things here are not going to be obvious, I'm not going to show them now. I'll state them as unproven fact, later we'll come back and you'll be able to show all these things. This is an unproven fact. It says if A has zero null space, if and only if it has a left inverse. Now a left inverse is a matrix B, which when multiplied on the left – Sorry, when it multiplies A on the left it gives you the identity. That's what BA is.

Now you may or may not have seen this. You have probably seen the idea of a matrix inverse, but maybe not a left inverse. Actually, how many people have even heard of the concept of a left inverse for a non-square matrix? That's good because it used to be basically no one. So a left inverse means you multiple on the left and you get I. By the way, the meaning of the left inverse is actually, it's great, it's actually a decoder is what it is. Because if BA = I, Y = AX then if you multiple by B, what you get is you get X. So XB, when you see a left inverse, it means it is a decoder or in communications, it's a perfect equalizer. In any kind of signal processing it's a perfect deconvolver or something like that. That's what a left inverse is.

Now what's interesting about this is that, sort of, of course, is a very strong statement about this first one. It says yeah, you can uniquely determine X, given Y = AX. This says, actually, not only can you determine it, but you can get it by applying a linear function. That's not obvious right? It could have been that you could uniquely get X, but you had to jump through all sorts of hoops and do all sorts of signal processing or something like that to get X. It turns out no. In fact, you can decode with a linear decoder.

And the next one is this. It turns out this one is relatively easy to show. It says it's the determinant of A transposed A, is non-zero. At the moment, that means nothing to you, but it will soon. And I think this is one of the ones that's – this one, I have to be very careful here, because some of these are extremely simple. They're either three lines or they're actually very difficult. And all of the difficult ones, by the way, are equivalent to each other.

Shall I try showing this? What do you think? So I reserve the right to absolutely give up one minute into this, and to pretend that I never thought – that I knew ahead of time this was actually one of the tough ones. Okay, so we're just gonna see. We'll see if this is one of the easy ones. What we want to show is this. Let's start by saying suppose that A has a non – We're going to show that AX – Well, here, A is 1:1, if and only if, det A transpose A is non-zero. Okay, let's try that. All right, so let's try a few things here. Let's start by saying, assume A were not 1:1. Okay? That means, well by definition, it says that its null space is not just zero. That means that there is some X which is non-zero, but AX = 0. That's what it means. It means there's a non-zero element in null space.

Now if AX is zero, then surely A transpose AX is zero because I could write it his way, that's the zero vector, and A transposed times any zero vector is zero. But I could also write it this way and write A transpose AX. I can re-associate. Ah, but now we have a problem. We have a square matrix here multiplying a non-zero vector and we get zero. And that implies that the determinant of A transpose A is zero. Okay? So we just showed that if this fails to hold, this fails to hold.

And now we do the opposite. Now let's assume – that's one way. Let's assume now det A transpose A is zero. Assume that's zero. That means the matrix A transpose A is singular. That means there's some non-zero vector, which gets mapped to zero like that, where X is non-zero. Now let me ask you a question here. If I had scalars, and I wrote down – Actually, can I just cancel the A transpose off of here? That's an excellent question. Can I just write this as A transpose A = 0, as A transposed times AX = 0, and now say A is non-zero, so I can just A transposed non-zero and just remove that. Can I do that? Can you cancel matrices? No, in general, no you absolutely cannot. So it's entirely possible you can have the product of two matrices, both non-zero, and you get zero. So you can't cancel matrices. Okay?

That's not going to help, but watch this. This is actually – Here I'll show you a trick, you may have to pan up here. It's going to set the pace for the class. I can move it up. All right, what happens is this. We know this, and here's the trick. If this is true, I can certainly multiple on the left by A transpose and get that because X transpose times a vector that's zero is surely zero. That's the case. Now I'm going to rewrite this in a different way. I'm going to write this as AX transposed times that. And that's just one of the rules for transposition. You transpose a product by transposing each thing and reversing the order. Okay?

Now you may know that the – Well, I guess its part of the prerequisite. I think actually we're going to cover it in a few slides anyway. But in fact, the transpose of a vector times itself is actually the square of its norm. That's actually this. This is actually nothing more than the sums of the squares of the entries. This is AXI squared sum on I. Now when a sum of squares of numbers is zero, every single number is zero, period. And in fact, what this says is, therefore AX is zero. That's just what we wanted, because now I have an X in the null space of A, and its non-zero, and therefore A is not 1:1. Okay?

By the way, I won't do this very often. I would actually encourage you to see how much of – Most of the statements in the notes are things you should check. These things get boring after a while, but you should actually try to check as much as you can. Most of it is quite straightforward. Some of this stuff for now is not.

Okay, so let me say something about the interpretations of the null space. Well, suppose Y = AX describes a measurement set up or something like that. Then what it says then is something in the null space is undetectable. It is a pattern that is undetectable from the sensors. Because when you have that pattern, all the sensors just give you flat zero reading. And you don't know if maybe what you're looking at is zero. In fact, it gives you the exact same signature, zero. It also says this, it says that X and X – is you take any vector and you add anything in the null space, they are completely indistinguishable according to your measurements, totally indistinguishable because they give you the same readings, the same Y, the same AX.

Now in fact it characterizes the ambiguity in the measurement. That's what it does. Now if Y = AX represents something like an output resulting from an input or action, X, then basically it says that something in the null space is actually an action. It is a non-zero action that has no net result. That's what it means, and in fact in this case, the null space characterizes the freedom of input choice for a given result. So actually having a big null space in a measurement problem is, generally speaking, bad because it means there's lots of ambiguity, if what you're trying to do is recover X.

If on the other hand, X represents an action that you can take, and Y is an outcome, having a big null space is great because it says there are lots of X's. And you have an organized way, in fact, of finding X's that will satisfy the requirements. So that's good, and it means that you can drop back, and you can optimize something else. You can drop back and find a small X or whatever you like.

Now the range of a matrix is another one of these fundamental subspaces, or whatever. And it's denoted with a calligraphic R or a script R or something like that. It's range of A. This is also a subspace. It is not a – it is a subspace of the output dimension. So it is actually a subset of Rn and it is simply the set of all vectors you can hit. It is the set of all vectors that have the form AX where X ranges over all of Rn. So this is a subset of Rm.

Okay, unless you're matrix is square, the null space in range are sets of vectors of completely different dimensions.

Now you can interpret it this way, it's a set of vectors that can be hit by the linear mapping Y = AX. That's what it means. It's the set of all things you can achieve, or something like that. But it's also, in terms of the columns of A, it's very simple. It's the span of the columns, so null space zero means that the columns are independent. If the range – Here the range is simply the span of the columns. Okay? And you can also think of it this way. It's the set of right hand sides, if you write it as AX = Y for which AX = Y has a solution, so that's the idea.

Now here it has all sorts of interesting interpretations and applications in different areas. So for example, Y = AX represents a design problem where X is an action, Y is an outcome. Range of A is the set of all things you can actually achieve. So if someone says, "I would like this outcome." But if that outcome is not in the range, it means, "Sorry, I can't do it. I can't get that outcome. Maybe I can get close." We'll talk about that. "But I can't get that outcome." In the case of a sensor reading, it is also very interesting. Y = AX, the range means it is a possible sensor measurement. What would be a practical use for the range in a case like that?

Student:[Inaudible]

Instructor (Stephen Boyd):Got it, that's it precisely. So what would happen is this, if in fact the measure – If you believe the system is modeled by Y = AX, Y is a vector of measurements. If Y comes and the first thing you do with Y is check, is it in the range? If the answer is yes, no problem, you can then proceed to whatever you're doing. If the answer is no, you want to throw an exception. You want to throw an exception to whoever passed you the Y, and basically says sorry, that Y could not have possibly been something of the form AX. By the way, that might mean one of you sensors is off. So this could be a sensor detection routine or something like that. And you could do all sorts of interesting stuff there. How would you check? Suppose you knew that one sensor was bad, what do you do? What's the algorithm? How do you check it? That can be a homework problem, by the way. Want to make a note of that? That's simple.

So let's talk about it. How do you do that? If I give you Y = AX, X is in R10, Y is in R20. Roughly speaking, you've got about a 2:1 redundancy of sensors over parameters you're estimating. But there's a trick. One of the sensors has failed. How would you find it? So, I want you to tell me how to find out which sensor. I want you to come back and say, "Sensor 13 is no good." How do you do it?

Student: You put in two inputs.

Instructor (Stephen Boyd):You know what, actually it's passive. You don't have – You can't do that. You don't have the – it's passive. You can't control X, you're just given readings, that's all. You had a suggestion.

Student: Take the [inaudible] of Y [inaudible].

Instructor (Stephen Boyd):I don’t think you can take the inner product of Y with the rows of the sensor matrix because the have different dimensions. The rows of the sensor matrix are of size Rn, the input. And the output is of size – if it was a ten in our example, the other’s R20, but you’re close. What would you do? [Crosstalk]

Instructor (Stephen Boyd):Find the null space of what? Student:

[Inaudible]

Instructor (Stephen Boyd):Let me ask you this. Suppose I have a sensor suite, and I say, “Please take out sensor 13, what does that do to the matrix? How do you express that in terms of what you do with Y = AX, Y used to be 20 high, now it’s now 19. How do you get that y and what do you do with A? What do you do?

Student: You delete the 13th row.

Instructor (Stephen Boyd):You delete the 13th row of A. Okay, so that’s what you do. So basically, if everything is fine, if it’s Y = AX that’s what it’s been for a while, and I come along and say, “Nuh, uh, sensor 13 is going out for maintenance.” Then you have a reduced A red and you remove the 13th row. That’s what it means. Does everybody have this? So now you have 19 x 10 matrix. Okay.

You can calculate the range of that 19 x 10 matrix. Let’s suppose – Let’s say you can do that. You will within one lecture. You can calculate the range of that matrix, and you can check if the reduced measurement is in the range. If it’s in the range, what does it mean after I remove the 13th row? You have to be careful. What does it mean? It’s kind of complicated.

Student:[Inaudible]

Instructor (Stephen Boyd):I’m going to turn it around and say that it means that the 13th – It means, if now you remove a row, and you look at the reduced Y, and it’s in the range of the reduced matrix, what it means is this; the 13th sensor might not have been bad. Sorry, it’s a strange statement, but that’s exactly what it means. What if, I remove the 13th row; I look at the range of that matrix. I look at the range of that matrix or something like that, and actually now the reduced Y is not in the range. It means you still have a bad sensor. It means you’re still wrong. It means the 13th was not the probably. I’m assuming there’s only one sensor failure.

I have a feeling that I’m just making things more confused. This is what we call a homework problem. You will be seeing this shortly. Trust me. Okay, so. Or you can go back and see if any of this makes sense later. All right?

Now, there's a special case, which is when you can hit all vectors in the output space. That's when the range is Rm. That means you can hit everything. That says, you can solve AX = Y for any Y. That has all sorts of obvious interpretations. We'll get to that. Another way is to say that the columns of A span Rm. That's what it means. Now in this case, this is not obvious. It says there is a right inverse of A. So, this says, for any Y you give me, I can find and X for which AX = Y. This third bulla here is much more specific. It actually says, not only that, but in fact I can construct the X as a linear function of Y. That's what it means.

So here, if in fact there's AB with AB = I, that's a right inverse. What it is says is this, it says that ABX is X, well of course because IX is X. On the other hand, it just says – You know what, I shouldn't have used X. Sorry. There we go. It was perfectly okay, just horrible notation.

What this says is – this is quite specific. It says, "Oh, you want an Ax for which AX = Y, no problem. Let's take your Y and multiply by B." So B actually is an automated design procedure. That's what it is. If this is a design problem, it's a design procedure. It says, "You want this velocity and position? Multiply by B that gives you a force program that does it." Okay, so that's what it is.

This is the same as saying that the rows of A are independent. Again, you can check. And it's basically, now it gets tricky. This is the null space of AT is zero. Now, some of these look complicated. Some actually are, most are not, but I am going to defer a lecture until we show all of the details. So this is the null space of AT is empty. And you can say this, these two are actually basically the same thing because to say that the null space of a matrix is equal to just zero is basically saying that the columns are independent. But the columns AT are the rows of A. And so, this one and this one are simply the English equivalent of that statement. Okay? And this one would probably follow up from one on the previous page or something like that.

I want to warn you, some of these are not obvious. But by the way, I would not – You should sit down and sit down and see how many of these you can actually show. You won't be able to show some of them, like that one. Maybe that one, I'm not sure, right away. But within a lecture you will be able to.

Okay, so let's look at various interpretations of range. Suppose a vector V is in the range of a matrix, and a vector W is not in the range. We'll see what does this mean. If Y = AX represents a measurement. Then if you observe V that means it's a possible, or maybe a better name is a consistent sensor signal. We just talked about that a few minutes ago when I confused all of you, most of you any way. So if however, you get a sensor measurement which is not in the range, that basically says it's impossible, if you really believe Y = AX or it means it's inconsistent. That's what it means.

Now if Y = AX represents the result of an output given an action, or an input or something like that, then it says, if you're in the range, it's something that's possible. It's achievable. However, something not in the range is something that simply cannot be achieved. That's what it means. So, in this case Ra characterizes possible results or achievable outputs. That's what it does.

Okay, now so far we have talked about the size of A has been arbitrary, it doesn't have to be square. And in fact, once you find my greatest complaint with, aside from the fact that they are profoundly boring, linear algebra classes. The first linear algebra classes, my first complaint is that they focus on square matrices too much, which actually in engineering has essentially no use. We'll see why, but in fact, most times in engineering when you have, in any kind of engineering application, you have a square matrix, except we'll see some examples later, but in general it means something is deeply wrong. We'll see why. You'll know why later, why in fact matrices should either be fat or skinny or something like that.

It means kind of that you set things up so there are just enough knobs to turn to do what you need to do. That's silly. The whole point is you either want to exploit redundancy or something like that. There are cases where you want to do that. If you want to measure something, if you want to measure six things, and sensors are super duper expensive, and for some reason you can only afford six, okay fine. But in general, this makes absolutely no sense. It's not robust either. In fact, if you have seven sensors to measure six things, at least then you can actually add a nice layer of software that will actually detect and remove a sensor that fails, automatically. Using the method which I deeply confused all of you with, and which you will see on the homework shortly.

Hey, what do you think? Do you think we're allowed to, even though we've posted Homework two? I think we can do it, we'll see. All right. We just have to make – it may have to wait until Homework three, we'll see.

Now we're going to look at a square matrix. You see, it's invertible or non-singular if its determinant is non-zero. And this is equivalent to many, many things. The things like the columns of A are a basis for Rn. That means they span Rn; that means its range is everything. And it means that they're independent which means that, in this case, the other half of being a basis, so leave that. It means a null space is zero. That's the slang. The null space is the set consisting of the zero vector. I don't think – I quit. I'm going to start saying null space is zero.

It also says the rows of A are basis for Rn. It says Y = AX is a unique solution for every Y in Rn. So, it means it has now – there's a matrix that is both its left and its right inverse, and it's the only one. So, if you're a left inverse of a square matrix, you're also a right inverse. Okay? It's the same as saying the null space of A is just zero, and the range is everything. And it's the same as saying the determinate of (AT) A and det (A) AT is non-zero. By the way, for a square matrix, this is easy to show because here, I can write this, det (AT) A is det (AT) x det (A), right? That I can do. Can I do that here? If I wrote here, that's det (A) X det (AT), what would you say to me?

Student: Syntax error.

Instructor (Stephen Boyd):You'd say syntax error, exactly. Why? Because det takes as argument only a square matrix, and in this problem there is M and there is N and they need not be the same. So this is a syntax error, which is a terrible, terrible thing, right? Because it means the most casual parsing would reveal the error. We'll see later that there are some deeper semantic errors. That's where the syntax works out but the meaning is wrong. Strangely, a semantic error is actually less of a crime. Okay, but in this case, they're square, so we can write these out, and these are very straightforward.

So what's the interpretation of an inverse? Well, I mean, this is kind of obvious. It basically undoes the mapping associated with A. Actually, the cool part is its both – it can work as equally well as a post equalizer or a pre-equalizer or a prefilter, and I'm using some of the language of communications here. In other words, you can in this case, you can send something through a channel, and then apply B to reconstruct what happened. That's a traditional equalizer. Or, before you send the signal, you can process it through B, and that has lots of names like, one is like a pulse shape or pre-equalizer, or something like that, in communication. So that's the idea.

Okay, so that's that. That you've seen. I'll mention one other thing involving that. One interpretation of inverse, in fact you can also use it for things like left inverses and right inverses, but let's take B is A-1, and let's look at the rows of the inverse of a matrix. Well, let's write this, this is Y = AX. If you write it out column by column is Y = X-1A-1 = XnAn, of course X = BY. This first one is Y = AX and this is just X = BY, because that's the inverse here. Now, if I write this way, I want to write out Y column by column, this one, and I'm going to write X out row by row. This is XI is BI until they transpose Y, because when you multiply a matrix B by a vector Y, if you do a row interpretation it says you take the inner product of each row of B, with Y. Did I get that right? I did.

Hey, that says look. These XI's I can plug them in here and I can rewrite it this way. And now you see the most beautiful thing. You see what the rows of the inverse of a matrix are. They are things that extract the coefficients of – so this basically says something like Y is a linear combination of the A's. Now if you want to know what linear combination, or how do you get the mixer, what's the recipe? It says you take the inner product with the rows of the inverse matrix and these give you exactly the coefficients in the expansion of Y in the basis A-1 through An. Okay?

Sometimes people call A1 through AN and B1 through BN, dual bases. That's actually kind of common now in some areas of signal processing. They call it – this goes back a long way, and it's interesting, actually to work because we're going to look at some other stuff later. But let me ask you something. What can you say about this? What is that equal to? What is it?

Student:[Inaudible]

Instructor (Stephen Boyd):Yep, that's one. If I = J, and its zero if I ? J. Okay? By the way, actually very soon we're going to talk about things like orthogonality and things like that. So some people refer to these as biorthogonal sets or something like that. In other words, it's kind of weird, it says that if the B's are orthogonal to the other A's but it doesn't – what do you know about this? What can you say about (AIT) AJ? Not much. There is only one thing I can say. I can say something intelligent about that. What can I say intelligent about that? It's what?

Student: It's non-zero. Instructor:

How do you know its non-zero?

Student:[Inaudible]

Instructor (Stephen Boyd):So I would have a column of A0, what's wrong with that? I could remove it. So, remove it.

Student:[Inaudible]

Instructor (Stephen Boyd):Thank you. It's all the same thing. Yeah, A could not be invertible if it had a column that was zero. Right. Okay. All right. All of these things are very abstract now, but we'll see them in applications and it will – Yeah?

Student:[Inaudible]

Instructor (Stephen Boyd):No, they're not. Ah, but a basis doesn't mean that they're orthogonal. We'll get to that. Yeah. We'll get to that. By the way, if your background is physics or its something like applied physics or something like that, most, a lot of the matrices you encounter will be not only square but symmetric. And so, a lot of operands will be self-adjoined. And that means a lot of things because that's what you've always seen. If that's the case, how's my diagnosis? That's a diagnosis there. Okay, so in that case this class is great for you. You're going to see all sorts of stuff, and you're going to learn all the things that are almost always true, like eigenvalues are real for you, right? Yeah. Got it. Good. You'll be fine, everything will be great.

Actually, then it comes up. You'll see it's going to come up in physics, too. So, we'll see. There will be cool cases where it's not symmetric. And then weird things will happen and all your fellow physicists will be deeply and profoundly confused because for them everything is symmetric, all bases are orthogonal and so on.

Rank of a matrix. The rank of a matrix is simply the dimension of its range. And here are some facts, these are not obvious. Things like the rank of A is the rank of AT. Another way to say this is the rank of A is the maximum number of independent columns or rows of A you can have. And that means immediately the rank of a matrix cannot be any more than the minimum of its input or output dimension, number of rows or columns. I actually said those backwards, but that's fine. That's what it means. If you have a 3 x 5 matrix, its rank could not possibly be more than three. It is at most three.

Here’s a very basic fact, some people actually make a big deal about this and they circle it, and it’s in some fancy box and everything and it’s called sometimes, the fundamental theorem of linear algebra or something like that. They make a big deal. They have a theme song, and anyway. This is sometimes called the fundamental theorem of linear algebra. It’s beautiful, but it’s not that big a deal. Actually, you’ll find out it’s actually much cooler to be able to do stuff with linear algebra than to celebrate pretty formulas. You know, it’s like Ei? + 1 = 0, you get over that and everything’s fine.

What this says is very interesting. It says that the rank of the matrix, that’s the dimension of the set of things you can hit. That’s this, plus the dimension of the null space, that’s the dimension of sort of the ambiguity set. The size of the set of things that get crunched to zero is equal to N, which is the input dimension. My interpretation of this, well I’m sure lots of people interpret it this way, is this is something like conservation of dimension or if you like, something like degrees of freedom. I mean, maybe that’s a better way to say it. Don’t take any of this too seriously, but it’s just to give you the idea.

So the rank of a matrix is actually the dimension of the set hit, but let’s make this very specific. Let’s take A is NR, well we just had something that was 20 x 10. That’s it; so by the way, the name for that technically is that’s a skinny matrix because it’s got 20 rows and ten columns. I’ll use that terminology. It’s going to come up a lot. So I have a skinny matrix, R 20 x 10, and I’m going to tell you that the rank of A, by the way could it be 11? No, but I’m going to make it eight. Okay? So there it is, it’s a 20 x 10 matrix whose rank is eight.

Now what does this mean? It means that if you ask, what can you hit, what’s the set of things that have the form AX? Well, the answer to that is this. First of all, that’s a subspace of R20, so you have twenty long vectors and there’s a subspace. It could have a dimension up to ten, but in fact it only has a dimension of eight. That’s what it says. It’s got a dimension of eight. So basically it says, if X is an action and Y is a result, it says that the set of things I can effect or do is actually eight dimensional. That’s what I can do. I can do eight dimensions worth of stuff.

But wait a minute, in this case I had ten input knobs, so roughly speaking, you would look at that and say, if what you can do is only eight dimensional, you have ten input knobs, you have two redundant knobs. Guess what. That’s the dimension of the null space, so the dimension of the null space of A must be two because they have to add up to ten. That’s the idea here. It basically says something about the total number of degrees of freedom going in minus the degree of ambiguity is equal to the total dimension of what you can affect. That’s kind of what it says. It’s a good idea to have a rough idea of what this is, and it will get much more specific in specific cases.

This one is very loose. Each dimension of input is either crushed to zero or ends up an output. That’s boy, I wouldn’t put that in anything but notes. Because it’s actually, in some sense, I didn’t even know what the dimension of an input is. What I’m saying is that if any of you said this, boy, we’d come down hard on you. I mean, unless it was casual, in casual conversation, but in anything other than that you would not want – On the other hand, as a vague idea it kind of explains what's going on.

Now there's another interpretation of rank, which is actually very, very nice. It's the idea of coding, and it comes up in ideas like transmission and actually it comes up in compression, all sorts of things. So here it is, if you have a product of matrices, then the rank is less than the minimum of the ranks of the matrices, of the things that you multiply. This sort of makes sense because if you look over here, and you think about what you can do, it basically says – this one is actually very, very easy to show. This one up here. But now what this says is if I write a matrix A in factored form, if I write B as M by R matrix multiplied by an R by N, you can think of R as an intermediary dimension, the intermediate dimension. It basically says, I have something that looks like this, that's A, but I'm going to write it this way, and I'm going to factor it, and that means basically I'm going to write it as a product of two things. Actually, if you like instead of one piece of code, it's gonna call two, one after the other. Okay?

And what this says is, in this case R is the intermediary dimension, it's the intermediate dimension here. That's what it is. Now if you factor something like this the rank is less than R, but in fact, if the rank is R you can factor it this way. So I can write it this way. Now this doesn't look like it has any – It's interesting, it's actually extremely interesting, and tons of super practical stuff is based on just this idea. But we'll just look at one now. Actually later in the class we're going to do something even cooler which is this. Very soon you'll know how to factor a matrix into one to make a factorization like this.

This has lots of applications immediately. Like, suppose I want to transmit Y, given X, but I pay for the number of real numbers I have to transfer between them, right? If N is 100, and M is 200, but R is 3, this makes a lot of sense because what happens is first I do this processing on this transmit side. I transmit three numbers, or whatever I said the rank was, and then they're reconstructed over here. Already these ideas should have ideas of efficient transmission, compression; all that kind of stuff is close to here. It is very close to this.

Let's just look at one application. Let's suppose I need to calculate a matrix vector product Y = AX. Now you would think that can't be that big a deal. Why would that be a big deal? And it would not be a big deal if A were, for example, 5,000 x 5,000. Right? Or it depends on what we're talking here, A could be 20,000 x 20,000, and you could do this doing NPI and a whole bunch of processes, or something like that. Or A could be 100 x 100 and you could do this in a – actually, 40 x 40 and you could do this in a shockingly fast way. Okay? I'm just multiply Y = AX. This could be – What are you doing when you're multiplying by A? This could be an equalizer. It could be precoder. It could be some estimation you're doing, it could be anything. So we're not worried about that.

Now, suppose you knew that A could be factored, and let's suppose – The interesting case is when the rank is much smaller than either M or N. But let's look at this. Now, if I compute

Y = AX directly, that's MN operations because I calculate the inner product of each row of A with Y, and each of those requires N multiplies N-1 adds. You know, roughly. Okay? That would be MN operations. With a typical sort of gigaflop machine – well my laptop comes out faster than that, so it's pretty fast. That's okay because as fast as you can do this, there'll be some application where you want to do it faster, or you want to handle bigger things. That would be MN operations.

Now suppose you do it this way, you write A as BC, and you associate it this way. You first operate by C, and then by this. Well, if you multiply by C first, it's going to cost you RN operations, then it costs you MR, and that's (M + N) x R. Okay? Just to plug in some numbers here let's suppose – Let's make both M and N equal to 1,000, and let's take R = 10. So it's a 1,000 x 1,000 matrix and its rank is 10. The saving now is pretty obvious. In the first case it's 106 and in the second one it's 102,000, so it's a million versus 20K. Okay? That's a 50:1 savings in multiplying AX. Has everybody got this? This is the idea.

By the way, later in the class we're going to look at very good methods for not just – very soon you'll know how to factor a matrix this way. If it is low ranked you'll factor it. Later in the class, we'll do some things that are very cool. We'll actually look at how to do approximate factorizations. That's even cooler. You will be given a matrix A; you will know how to find, for example, an approximation of A that's ranked ten. Okay? This has tons of uses all over the place. Okay, that's what it means. If you see a matrix that's low rank, it says actually, you could operate on it quickly or it means it's got something to do with compression or something like that.

Student: How do you know if a matrix is low rank in the first place?

Instructor (Stephen Boyd):How do you know if a matrix is low rank in the first place? You will know, the sort of exact answer, if you want exact rank, you'll know in one lecture. If you want to know – much more interesting question, much more practical is this, given a matrix, how do you know if it's approximately rank 10. You'll know that in about five weeks. By the way, did you notice that I didn't answer your question at all? Okay. But I did it forcefully. So, that's okay. We put your question on a stack, it will be popped. Okay?

One last topic is change of coordinates. Now, the standard basis vectors are these EI's like this, and you obviously have this expansion. I mean, that's kind of silly, but it's a pedantic way of writing this out. But you would say in this case that the XI are the coordinates of X in the standard basis. Normally, you just say XI are the coordinates. Okay? Or the entries of the coordinates or something like that. That's how you write it.

Now, if you have another basis in Rn, if T one through ten is another basis, you can expand a vector X in another basis, and we'll call the coordinates ~X one up to ~X N. Okay? And of course, the fact that that's a basis tell us that – That tells us two things. It says that the T's are actually independent, which means there's only one way to write it this way. And it means, also, that any X on the left hand side here can be written in this form. Okay? It turns out we can get these ~X very easily, by a couple of matrix operations. So here ~X and the coordinates of these, and what we're going to do is we're going to write this out as a matrix, so we'll write that as X = T[~X]. Now, this is basically a character representation of this, so you write it this way. But now, immediately we get the answer. T is invertible, and therefore ~X is simply T-x, and indeed it turns out that the inner product of the Ith row of the inverse of a matrix extracts the TI coordinates. Okay?

For example, if someone says – I mean actually, where would this come up all the time? It comes up in things like vision and robotics. And actually, it comes up in navigation as well. So, for the rest of we just use a fixed coordinate system, and seem to go through life just fine, but for seem reason in things like dynamics and navigation and things like that, everyone has to have their own personal coordinate system, and so you have to transform things from one to the other. For example, in robotics, every link of the robot arm has its own personal coordinate system, and you have to transform them. I don't know why people do this, but they do. This is kind of the thing that would allow you to do that. Now you know what it is, it's sort of the inverse of a matrix and it the rows extract the coordinates, and things like that.

Now, you can also look at what happens to a matrix. If you have Y = AX, this is all in the standard coordinate system, and now suppose you write X in the T coordinates. And then we're going to write Y also in the T coordinate, and the T coordinates will be ~X and ~Y, then just by working it out you get ~Y = (T-AT)~X , that's what it looks like. So, this thing is something that's going to come up a bunch of times, not a bunch of times, but it will come up several times, and it should light up a light in your brain. First of all, it's got a name, its called similarity transformation, that's the first. For the moment this is all very abstract, but you should think of it as basically representing Y = AX but you represent both X and Y in a different basis, and you'll get something that looks like that. So we'll get back to that later.

Now for some reason at the end of this lecture, there's some stuff that's more elementary than all of this and should have been at the beginning. But that's okay, we'll do it now. It has to do with just a quick review of norms and inner products and things like that. Now, the Euclidian norm of a vector is actually the square root of the sum of the squares. Some of the squares you can write as X transpose X. That's the inner product of a vector with itself. And it's the square root of the sum of the squares. And it's supposed to measure the length of a vector. And indeed it measures the length of a vector with N = 1. It's the absolute value. When N = 2 it's actually the Euclidian distance in the plain, and for three it's Euclidian distance in R3. But it makes sense to talk about then norm in R500. It makes perfect sense, and people talk about it all the time.

Now there are various things related to this, but I'll mention some of the basic ideas. So, it's supposed to measure length. It's supposed to be a generalization of absolute value. By the way, some people have actually – the double bars are to distinguish it from the absolute value. But presumably if you were super, super cool, and kind of just grew up from a child onward using vectors and matrices, you would probably write this this way, and there are people who have reached that level of development, of linear algebraic development that they write it this way.

The same way we've got to the level where zero, we don't mark it; we don't put a doo dad on it. We don't put an arrow on it, like our friends in physics. We don't make it bold or something like that. So for zero we're fully developed. Zero is totally overloaded for us. It's the same thing for the number, the complex number, matrices, vectors, everything. This is a hold over. Actually, it's 150 years old, maybe more; something like that. There's only one minor catch here. Unfortunately, there's a bunch of fields where people – it's useful to interpret this as the element-wise absolute value. That's the hitch there, but it's supposed to be a generalization of this.

So it satisfies a bunch of properties, these are all easy to show. This one, I believe you did show, or will be showing. Is that right? No, it's close to something you will show or will be showing, which is because showing, which is the Cauchy–Schwarz inequality. Is that right? On this homework? I'm always a couple of homeworks ahead, so I can't remember. Okay, this is a triangle inequality. It's kind of obvious. It says basically, if that's X and that's Y, that's X + Y, and it says that the length of this is no bigger than the length of X plus the length of Y.

The picture makes it totally obvious when they would be equal, and that's when X and Y are aligned. This is called definiteness. It says if the norm is zero, the vector is zero. That's easy to show here because is the square root of a sum of squares is zero; the sum of the squares is zero. But if a sum of numbers that are non-negative is zero, each number is zero. That means each XI2 is zero. Therefore, each XI is zero.

Now, there are some things related to norm. For example, it's extremely common in a lot of applications where N varies. By the way, if N doesn't vary in an application, like if N, if you're doing control or something like that, you don't – when N is fixed, it's like six and it's the three positions and the three momentums. You don't divide by N or something like that. But in other areas, like signal processing, where you get vectors of different lengths. For example, an audio clip is a vector, but N depends on what the length of the audio segment is. And you want to talk about different things like that. Or it could be any other signal.

It's extremely common to normalize the norm by square root N. That's the same as taking the sum of the squares of the elements, dividing by N. This is called the mean square value of the vector X. And that is obviously the root mean square value, that's the RMS value of a vector. And, it just gives you a rough idea of the typical size of that the entries would be. There's actually some things you could say about it. But so if the RMS value of a vector is one or one point three, it means that basically if you look at a random entry it should be on the ord. of one point three.

By the way, if a vector has RMS value of one point three, how big could the entries be? Could they be 1,000? Could you have an entry that's 1,000?

Student: It depends on [inaudible].

Instructor (Stephen Boyd):I'll simplify your answer. Ready? Yes. That's his answer. That's correct. You can have a vector whose RMS value is one point three and you could have an entry in it that's 1,000. That only works if you have a really big vector. You would have a lot of zeros and then a really huge one. You couldn't have a four long vector that had an RMS value of one point three and have an entry of 1,000. There's things you can say about that, but that's the rough idea.

Now, norm is length from zero. If you take, since you have a difference, it says the norm of a difference gives you a distance. This is very important. This allows you to talk about the distance between two things. This is what allows you to do things, talk about the distance between, for example, two audio signals, or two images. You can say, and in fact here you might even give an RMS value, and in both cases it would be very interesting. What would you do? You might have the original and some encoded and subsequently decoded signal, and you would say, "I was within in 1 percent." The RMS error is 1 percent that means the norm of the difference divided by the norm of, say, the original signal is about point-zero-one. That's what that would mean.

Now, the inner product, of course we've been using it all the time. There's lots of ways to write it, but one way is this. It's actually just X transpose Y, nothing else. In fact, we won't write this. Actually, it's kind of cool. There's other ways people write this. I'll show you some of them. Here's one that you see. Again, we're going toward physics now. This would be the notations there. There are others. You could write X, Y that's one way. And I've seen this. You could also do dot, and depending on what subfield they're in, your dot is either little or big thick thing. These are different notations for inner product. Does anyone know any others because there's probably some others too?

All right, so the inner product, you can work out these properties is quite straightforward. In fact, I don't think I'll say much more about that, I'll zoom on. Because these are kind of obvious.

The Cauchy–Schwarz inequality is very important and it says this, that the inner product of two vectors in absolute value is less than or equal to the norm of the product. From this, you can immediately derive, for example, the triangle inequality right away. But that's what this says. And this is something, I guess you show on the homework, or have shown, or will show in the next half day or something like that.

The angle between two vectors is simply the arc cosine of the inner product divided by the product of the norms. Now, this number here is between plus and minus one. And the arc cosine, let's draw the cosine. And we usually take it between zero and 180, or something like that. So we don't distinguish between minus 10 degrees and plus 10 degrees. It's the unsined angle, is the way to say it. So it's the angle between things. And this allows you to do all sorts of cool stuff. To talk about two images, and say that they have an angle of 3.7 degrees with respect to each other. It makes perfect sense. Or an image can come in and you can have a database of 10,000 images and you could ask which of the 10,000 images is the one that just came in. Have the largest inner product width, which would mean the smallest angle. And this would make sense, and you can already start guessing these things have applications.

It's kind of weird to talk about the angle between two vectors, in for example R one million because that's what you're doing if you're looking at 1K by 1K images. After a while – don't try to visualize, by the way, R one million. Or at least work your way up there because you could be in big – yeah. Start with R4, when you're comfortable at R4, maybe R5 or R8, something like that. There also might be some drugs you might have – some pharmacological things are needed to really, really understand R.

Actually, Tom Kover gives a wonderful talk, basically showing beyond a shadow – makes it completely clear. You know, we pretend, so people ask, "What are you doing here?" When we're talking about the angle between two images, "There's no such thing as the angle between two photos." And I go, "Yeah, sure, its 3 degrees." Then I'd say, "You don't understand, we do this kind of applied math stuff, and the way it works is we talk about distances and angles, but in huge dimensional spaces." And they say, "Can you really visualize that?" At that point you have the choice of either lying and saying, "Oh yeah."

Or what you would really say is something like this, you'd say, "Well, no not really." The truth is not even close, right? But you'd say, "Well not really, but we use our pictures that we draw on two dimensional paper or in our offices, you get a grad Student: to say hold your hand up here, and then you get in there and you poke around this way." You say, "We use that intuition from R3 to guide us in making image compression algorithms in our one million, right?" So you might ask how useful is your intuition from R3?

Tom Kover has a series of examples showing that you know nothing. He'll show you all sorts of things. They'll be true for N = 3, 4, 5, for six on they're so false it's amazing. Thing are like wow – I should collect some of these from him because they're really good, and there are about four of them. One is like if you pick a point at random in a unit ball in arc 20, it's basically; I think they call it a sphere hardening. There's something like a 99.99999 percent chance that you are within 1 percent of the boundary of the sphere. Just totally insane things, but okay, just to let you know, so that at least we were honest about it.

It's very important to understand what it means to have a positive inner product or a negative inner product or something like that. Is should say this, if X and Y are aligned, that means they point in the same direction, that says that the inner product is the product of the norms. If they're opposed, that means that the inner product is as small as it can be. If they're orthogonal, it means that there's a 90 degree angle between them. It means X transpose Y is zero. And it's very important to understand what it means to have a positive inner product or negative inner product.

So positive inner product basically means that the angle between them is between zero and 90. If you have two things, as you twist them out, the angle goes from 90, and the inner product is quite positive. It's as positive as it can be when they're aligned. As they twist along like this, and when they get to 90 degrees, the inner product is zero, then the inner product starts getting negative, like that. So a very, very crude thing to say would be something like this. If the inner product between two vectors is positive, they point roughly in the same direction. And I mean really roughly. I mean they're not fighting each other. If you add them, that's actually one of the characterizations, that the norm of X + Y is bigger than the minimum of the two. You make progress.

If you have a negative inner product, it means the angles between 90 and 180. This allows us to combine things like a half space. If you have a set of X's who have a negative inner product where they're give vector Y, that's this. You draw the vector Y and the vectors that have a zero inner product with Y, basically those that are orthogonal to Y, that's the hyper plain with Y as a normal vector. That's here. This shaded region, these are actually vectors whose inner product is negative with Y. So they have an angle more than 90 degrees, that's sort of the picture. This is called the half space, and these things come up and all sorts of things. A half space is the solution of a linear inequality.

I'll start in on the material of the next lecture, which you've probably seen in one or two other contexts, maybe in the context of Fourier or somewhere. So we'll talk about orthonormal sets of vectors. The Graham-Schmidt procedure or when you right it out as a matrix, in matrix terminology it's called the QR factorization. And then we'll talk about various things we can show with this.

We start with the idea of an orthonormal set of vectors. So a set of vectors is called normalized if norm UI is one. So normalized just means the length is one. And by the way, some people call those unit vectors. I don't know why. Other people call them direction vectors, vectors whose norm is one. So that's very common notation for it. Unit vectors is a bit weird because for most people unit vectors means eI, e3, e5, but still.

A set of vectors is orthogonal if different vectors in the set – another way to say this is mutually orthogonal. It says that any two different vectors in the set have a 90 degree angle between them. Okay? And it's orthonormal if both. Now, watch out because people will simply say, you will say on the streets, U1 through UK are orthonormal vectors. That is basically completely wrong. By the way, there's nothing wrong with saying U1 through UK are normalized vectors. That's cool because to be normalized is an attribute of a vector alone. To be orthogonal, it makes no sense to say – if it made sense then you could have your set of orthogonal vectors and mine, and we could throw them together. If you have five and I have seven, now we have 12 orthogonal vectors. It doesn't work that way.

Normalized vectors works, no problem, we can combine your normalized vectors with mine, and now we have twelve normalized vectors. Orthogonality refers to a set of vectors. It is not an attribute of a vector. Okay. How do you write that in vector notation? Very simple, if you have some vectors U1 through UK, you put them in a matrix like this. You just simply make them the columns of a matrix, call that U. And to say this, is to simply say that U transpose U is I, period. That's it. Why? Because U transpose is U-1 transpose up to UK transpose like this. And then you multiply by U-1 through UK. And you can see now, the II entry of this matrix is basically UI transpose UI. That's one and the off diagonals are zero.

What would be the matrix way of saying that a set of vectors is orthogonal but not necessarily normalized? How would you say it?

Student:[Inaudible]

Instructor (Stephen Boyd):U transpose U is diagonal. That means they are – that's a mutually orthogonal set. How do you say – what's the matrix way of saying a set of vectors is normalized? How do you say it?

Student:[Inaudible]

Instructor (Stephen Boyd):In terms of the U transpose U, it would be the diagonal entries of U transpose U would be all ones. How about the off diagonal entries? Who knows? That depends on the angles between them. That's what it would say.

Now, you can quickly show that orthonormal vectors are independent. How? Well, if you write something this way, and you multiply it by UI, you will get that I is zero. And that says, if you have an orthonormal set of vectors, U1 through UK, it turns out it spans our view. You have to be very, very careful here, because if I say that I have an orthonormal set of vectors, that's exactly the same as saying U transpose U is I, exactly the same. This is the matrix way to say that.

Watch out because it is not the same as that. Although, that will hold in some special cases. Okay? In general, this is not true. An actually, there's a quick way to remember it. If you have a bunch of vectors and you stack them up. So here's three vectors in R10, so I write them this way, and I write this and so you get a 3 x 3 matrix here. Right? Like that. That's U transpose U equals I. Okay?

Now, suppose I got confused and switched them around and I wrote, (U) U transpose equals I. By the way, there's no syntax crime here. None because this would be a 10 x 10 matrix. This would be a 10 x 3 matrix. And that would be a 3 x 10 matrix, and there is no syntax crime. So a casual parsing doesn't reveal any problem with this. The parser goes right through it. Okay? But as a quick way to know that you're in deep, deep trouble, and that this, when you multiply two matrices together, the rank of this matrix is three. The rank of that matrix is three, and the rank of the product, therefore, is at most three. And what's the rank of the identity matrix in R10 x 10? It's ten, so we have a problem here. So this is a semantic error if you do this. It's a semantic error.

By the way, we're going to find out that that 10 x 10 matrix is not the identity. What it is is really interesting, and we're going to see it later. But for the moment, just remember it is not the identity. Okay, I'll mention the geometric properties, which are actually very interesting. Suppose you have an orthonormal set of vectors, which is to say a matrix whose columns are orthonormal. Ooo, that was slang, did you hear that? But I'm trusting you that you are cool on that and aren't going to make semantic and other errors. That was slang. That's how, by the way, people would say it. Otherwise, you'd sound like a pedant right? If you're like, "Excuse me don't you mean the set of its columns is orthonormal." You shouldn't talk to people like that. You should avoid them.

So suppose the columns of U are orthonormal. It says the following, if you take W = UZ. It says the norm of W is the norm of Z. Now note that the norm of W is different – W is possibly a different size from the size of Z, so what this says is, when you multiply by a matrix whose columns are orthonormal, you don't change the norm. That's what it means. Another way to say it is, people say it this way, W = UZ is isometric. "Iso" means the same, and "metric" means that it has to do with the metric or distance properties. And it basically says if you have two points like this, and the distance is three between them. If they get mapped by you, it says that the distance between if that's like X and Y, the distance between UX and UY in possibly a higher dimension will also be three. So it preserves these things.

It's very easy to show this. That's just a quick calculation. I think maybe this is a good time to quit for this lecture.

[End of Audio]

Duration: 76 minutes

IntroToLinearDynamicalSystems-Lecture05

Instructor (Stephen Boyd):Oh, looks like we're on. Well, I have an announcement. Yesterday, Jacob, the TA – somewhere, lost him. Oh, well. He's around somewhere. Jacob and SEPD put together a prototype of the publicly available lectures for 263. I mean, not that you care, but just so you know, it's on the website now. You can go take a look at it, and it should be just open to the entire world, the lectures. So we haven't figured out yet how to, like, buzz your head out if the camera, you know, decides to point to you, or if we have to – I suppose we do have to; we'll figure that out later. But anyway –

We'd be interested if you find any problems with it, or if your friends somewhere else have any trouble with it, we'd be curious to – I mean, we know that it actually works from many places, but a lot of different platforms and things like that. So just to let you know, that experiment has now reached the next step. It's up and running. It's in Flash, so just like – should be not much different than YouTube. So it's up and running. We might also make available some WMV streaming, and then actually WMV download as well. We've already piddled with that internally and may also make that public soon. So – okay.

Let's continue with orthonormal sets of vectors. Does anyone have any questions about last time? If not, we'll continue. Our topic is, of course, orthonormal sets of vectors. So a set of vectors is orthonormal. So I have K vectors in RN. They're orthonormal if they're normalized. That's an attribute of the vectors separately and then mutually orthogonal, and that's an attribute of the set of vectors, and it's orthonormal if both.

Now, you can write that very compactly in matrices. It's U transpose U = I, and U transpose U = I is nothing but the matrix statement of normalized – that's the diagonal here, the ones, and the zeros on the off diagonal of I tell you that these vectors are mutually orthogonal. That's what this says. Okay. Now, last time we worked out several properties of these. We'll look at the geometric properties of a matrix whose columns are orthonormal. So let's suppose we have a set of orthonormal vectors. Oops, I said it. That was slang. An orthonormal set of vectors. After awhile, in fact, some time later today I'll just stop worrying about it.

So you U1 through UK are an orthonormal set of vectors. If W is UZ, then the norm of W is the norm of Z. In other words, if you multiply by a matrix whose columns are orthonormal, you preserve lengths; you preserve norm. Let's see how that works. It also means that people say the mapping Z to UZ is isometric. Iso means the same, and metric means that the lengths or distances are preserved because, for example, U of – if you have Z minus Z – or I should write it this way. UZ minus UZ tilde norm is equal to the norm of Z minus Z tilde.

So if two vectors in RK have a certain distance, then their images under U will have the same distance. This is very easy to show. If you look at the norm^2 of W, that's UZ, that's norm UZ^2, but the norm^2 of a vector is nothing but the transpose of the vector times the vector. So I'd put UZ, and I transpose it here. I use the transpose rule to get Z transpose, U transpose U, and now I use the fact that this is the identity, and I get this, so a very quick derivation.

Now, inner products are also preserved. So if I have two vectors, Z and Z tilde, and I calculate their inner product, and then I calculate the inner product of the images under this mapping; you'll find out that they're the same, and it's the same argument. You simply say the inner product of W and W tilde – I just substitute UZ and UZ tilde, that's UZ transpose times UZ tilde. You switch it around and you get the same thing.

Now, because norms and inner products are preserved, that means angles are preserved. In other words, if two vectors have an angle of 15 degrees, then when you multiply them by this matrix U, their images will also have an angle of 15 degrees. It means, for example, orthogonality is preserved. Having a positive inner product, that means acute angles mapped to acute angles and so on. And, in fact, it really means that it's something like – well, let me careful. It's something like a rigid transformation. In R3, there are two types – I'll get to that in a minute; I'll wait on that.

It basically means that geometry, metric geometry, is preserved – distances, angles, inner products, lengths, all that. Now, a very important special case is when you actually have N of these, and these have a height of N. So I have N, a set of N – this is slang, orthonormal vectors or an orthonormal set of N vectors in RN, and they, of course, form a basis in this case because they're independent, and they span RN, and in this case U, the matrix is a square. Up till now, it's been skinny. It can't be fat; it's been skinny, a pot which includes a square as a special case.

Now that's square, and in this case it's got a name, and it's called orthogonal, and, sorry, these should have been called orthonormal, but language is weird, and orthogonal is the name given, maybe sometime in the mid-nineteenth century, and it stuck, and that's absolutely universally used, except maybe in some weird, you know, who knows. There may be people who do some weird branch of signal processing or something that people who talked about over complete bases, they might have redefined orthogonal and have some special meaning, especially they might've given it some other name.

So that's an orthogonal matrix, so it's square. It satisfies U transpose U = I. Of course, in this case, this means here that U transpose is actually the inverse of U. So it means transpose and inverse coincide. That's what it means to be orthogonal. Now, that also means that U, U transpose is I. Now, in general, it's absolutely false that if AB = I, that implies BA = I. If AB = I, it means only that B has a left inverse which is A, and A has a right inverse which is B. This absolutely does not mean, imply, that BA = I, but if they are square, it's the same. So in this case, we include U, U transpose is I.

And that's a very interesting formula; it says this: it's the sum of UI, UI transpose is I. Now, some people actually give a fancy name to a matrix of this form. This is often called an outer product, so there's lots of slang for this. I should've put this in the notes. One is it's called an outer product, and it makes, sort of, sense, right? Because the inner product is when you reverse the two and you get a scalar. This is an end-by-end matrix, right? When you reverse them you get a scalar. It's called an outer product.

Another name for it, also very widely used is dyad. So this, in fact, when you have a sum of dyads, it's sometimes called a dyadic expansion. You will hear that very commonly, and that's the high language of mathematics. Anyone would understand what you're talk about if you say that. So this is a dyad. Oh, some people also just refer to it as a rank-1 matrix, which is actually – we'll see, in fact, soon, in fact, you already know, although we don't have the derivation, that this is the case, that it's not true that it's rank-1, just rank-1; I'll get to that later, but symmetric rank-1, something like that. These are all the synonyms for an outer product like this. By the way, if you have PQ transpose where P and Q are vectors, what is the IJ entry of this? What is it?

Student:[Off mic].

Instructor (Stephen Boyd):No, it's PI, QJ. So basically what the outer product does is it takes two vectors, actually, of different sizes, possibly different sizes, and it simply forms all possible products of one entry of one with an entry of other, and it stuffs it in the matrix. So that's what PQ transpose is, like that. Okay.

All right. Back to our main story, so our main story is that U, U transpose = I can be written this way. I mean, you could check on this. This is just one way to expand this thing is to write it out as sum UI, UI transpose. Okay. Let's see what this means. There's lots of ways to say this. If X is U, U transpose, X – well, because U, U transpose is I, we can write that this way. We can write X = sum UI, UI transpose, I = 1 to N × X, and I'm gonna reassociate things here. I'm gonna put a bracket around that. That's a scalar.

Now, a scalar can actually move to the front, and, in fact, in some cases, I guess when we recast, at the moment, that's formally a 1 by 1 matrix, but if I simply put in front of scalar, parentheses this, or parenthesis scalar to recast this as a scalar, it can move up front here, and then you get this formula here.

Now, this formula is actually quite pretty, and let's see what it says. It basically says X is a linear combination of the UI's. What are the coefficients? The coefficients actually are found in an incredibly simple way, by simply multiplying X by UI transpose. So people call UI transpose X, the component of X in the direction UI. Actually, sometimes people call this the coefficient, and you would distinguish – you would call UI transpose X the coefficient and UI transpose X × UI would sometimes be called the component of X in the direction of UI.

So here, the multiplication U transpose X, which is nothing but this, that's equal to U1 transpose X down to UN transpose X. This thing, when you make this multiplication, it's called resolving X into a vector of its UI components because that's what this says. It's basically calculating the coefficients here. When you form the operation X = UA, that's actually reconstituting X from its UI components. It's rebuilding X from the recipe; the recipe is A here.

So if you write this out, that's called the UI expansion, and you can see that it's really two operations. When you write X = U, U transpose X, if you think of it this way as U, then U transpose X, these things have real meanings. U transpose X resolves X into its U expansion coefficients. You multiply U by this vector of coefficients; this is the recipe. That reconstitutes X. So that's the meaning of this. I mean, you have to, kind of, go over this just to think about it, but there's nothing really too complicated here.

Now, this identity, which is I as U, U transpose, which is the sum. Actually, there's lot of names given to it, but in physics it's sometimes written this way, and that's mirroring this idea here. If you take X, you multiply this on the right by X like that or something like that. So I write I, and if I put this here, then you see it, sort of, completes the inner product symbol here. So I'm just curious, is anyone – you want to guess what this is called in physics? There's people here who know.

Well, that's called a bracket. So this is called – I couldn't make this up, okay? It's called a Keptbra, although I don't know how you pronounce it because this is a bracket. Well, I got news for you, that's about as fun as it gets in physics, so that's the best they can – actually, have people heard this? Cool. So they stick to it, huh? Okay. Do they think it's funny? No, they don't.

Student:[Crosstalk].

Instructor (Stephen Boyd):No, so they didn't even get it. You said when you saw it, what, did they just smile or what?

Student: Yeah, they smiled.

Instructor (Stephen Boyd):They smiled.

Student: Yeah.

Instructor (Stephen Boyd):But nothing more?

Student:[Crosstalk].

Instructor (Stephen Boyd):Okay. So let's just say they found it mildly amusing. It was, sort of, like a private joke kind of a smile or what are we – was it – what are we

Student:[Off mic].

Instructor (Stephen Boyd):Oh, other people – no, no, no. No, I'm not. I'm just asking. Okay. All right. We'll move on. Okay. By the way, you'll see this in lots of context, I presume, and already have. You may have seen the Fourier – or will see Fourier Transform described this way, and so this should look very much like many, many things you've seen, if you haven't seen exactly this. Okay.

Okay. Let's take a look at the geometric interpretation. If you have an orthogonal transformation, then the transformation, when you multiply it by an orthogonal matrix, it preserves a norm of vectors. So we already know that. It preserves angles, and examples would be something like this. If you rotate about an axis, so if you fix an axis somewhere and then rotate 22 degrees, that's a linear operation, and it's given by multiplication by an orthogonal matrix.

Another example is reflection. So in reflection you specify a hyper plane, and then you say – and then the operation is to take something on one side of the hyper plane, basically calculate the projection onto the hyper plane, and then go the exact same distance on the other side. That's a reflection. That's also linear, and it's also given by an orthogonal transformation.

Let's look at some examples in R2. So in R2, this is, kind of, pretty straightforward. If you just want to rotate by theta degrees like this, a good way to work out the matrix is quite simple. You simply figure out the first column of that matrix is the image of E1 because we know AE1 gives you the first column. So it's gonna be this thing which is cosign theta, sign theta, and, indeed, that's the first column, and the E2 rotates to the vector of -sign theta, and then cosign theta; it's the height, and that's this one, and you can check. The U theta, transpose U theta is I, and it's just a trigonometric identity. The ones on the diagonals return $\text{cosign}^2 + \text{sign}^2$ which will be 1, and the off diagonal ones will just cancel away.

If you reflect across this line, that's a line of theta over 2, if you reflect across it, you can work out what happens to E1 and E2, and that would give you this matrix here. It looks very similar to this one. It's also orthogonal, and this one gives a reflection, so that's the thing. Now, in fact, you can show the following: any orthogonal matrix, 2 by 2 matrix, is either a rotation or a reflection.

So what this tells you is that orthogonal matrices now, just from knowing what it means geometrically, it's gonna come up anywhere where you have isometric changes of coordinates. So it's gonna come up in areas like dynamics, aeronautics, anywhere where you're changing coordinates. GPS, it would come up all the time. It's gonna come up in navigation, and it will come up in robotics all the time.

So it'll just be matrices filled with cosigns and signs, and that'll be the orthogonal transformations that map, you know, the coordinate system of the third link in the robot arm back to the base coordinate system, or in GPS, you'll have the world fixed – whatever the earth-centered, earth-fixed coordinate system and some other coordinate system, and there'll be rotation matrices, orthogonal matrices that affect the transformation. Okay? So you'll see them in lots and lots of fields like that. Okay.

So now we're gonna look at something called the Graham-Schmidt Procedure, and before we start, let me point out where we are in the class and where we're going. So this may be a time to have a little road sign to say where we are and where we're going. So far, we've rapidly reviewed these ideas from linear algebra that you may or may not have seen before, but, at the moment, you actually have, so far, we have introduced no computation teeth into what we're talking about here, right? So, so far, you have no idea how one would actually, sort of, compute a subspace, how would you calculate the range? So, in fact, so far it's all been talk.

That's gonna change in the next two lectures, and it's not a big deal, but it's gonna change in the next two lectures, and all of the stuff we've been talking about will be given algorithmic and computational teeth. So you will actually be able to do things with all the things we've been talking about, which actually will make the course a lot more interesting, I assure you, for me, anyway. It's gonna be a lot more interesting and a lot more fun. So that starts now. That's where we're going.

Okay. Graham-Schmidt Procedure, this you probably have seen somewhere. It's not exactly a very complicated thing, so that's okay. Graham-Schmidt Procedure goes like this. It's an algorithm, and it accepts a set of independent vectors. Ooh, that was slang. I really have to work on this, right? Because independence is not a property of vectors individually; it's a property of sets of vectors. Okay. I'm gonna try it again, and then I'm just gonna say forget it. I gotta collect my thought.

It's given an independent set of vectors. Yes, it came out. So, actually, you could help me, and whenever I slip into slang you could hiss just a little bit or something like that just to help me here, but after awhile you just get used to it. All right. So given an independent set of vectors – yeah, that's good, okay.

A1 through AK and RN, Graham-Schmidt procedure actually finds a bunch of orthonormal vectors with the following property. They have the same span as the vectors you started with, and, in fact, that's true no matter where you stop the algorithm. So if you look at the Q1 up to QR, these are orthonormal, and they all have the same span as A1 through AR. Okay.

So what this says is that Q1 to Q4 is an orthonormal basis for the span of A1 through AR, and the idea of the method is, first, what you do is you orthogonalize each vector with respect to the previous ones, and then you normalize it to have norm 1. I mean, it's not particularly complicated. So let's see how it works. So it's actually quite straightforward. The first thing you do is you're given A1, and your task, or at least the specifications, are to produce – well, a Q1, which has the same span as A1 and should be orthogonal. Well, it should be a singleton which is orthonormal which is the same as saying it should have length one, okay?

That's easy to do. You simply take A1 and you divide it by its norm, and that gives you a unit vector that points in the same direction as A1, and we'll do that in two steps. We'll say Q tilde 1 is A1, and then Q1 is Q1 tilde divided by Q1 tilde, norm. This is the normalization step. This is the orthogonalization step except there's nothing to orthogonalize against.

Now it gets interesting in Step 2. You take Q2 tilde is A2 minus Q1 transpose A2 × Q1. And this is what you do, now when you subtract this, you are actually – the name of this step, if you asked me to describe it, is this: You are orthogonalizing with respect to Q1. That's what you're doing in this step. You're orthogonalizing with this – you can check that Q1, Q2 transpose, Q2 tilde is zero, and you do it by subtracting the appropriate component of Q1 – the appropriate multiple of Q1, okay? So that's what this does.

This ensures that Q1 transpose, Q2 tilde is zero. That's what it does, and you can check because Q1 transpose, A2 is the same as this. You could work out what this is times – and then you have Q1 transpose. Q1 transpose, Q1 is one; it's normalized and so is zero, okay?

So this is really the orthogonal, in fact, this is called orthogonalization, okay?

Then when you orthogonalize, there's no reason to believe that what you are left with has unit norm, so you normalize. So basically it alternates between orthogonalize, normalize, orthogonalize, normalize. Now, you always orthogonalize with respect to the stuff you've already done, and that involves this, okay? So this is the next step is you simply orthogonalize and so on, and you keep going.

And there's lots of questions you have to ask here. In fact, what you have to ask in an algorithm like this is if you fail, why? And you'd have to argue why you don't fail. If I failed here, if I had a divide by zero exception here, what would it mean? Suppose here we fail, as we really can't fail here. Here you can fail by a divide by zero exception. What would it mean?

Student:[Off mic].

Instructor (Stephen Boyd):A1 is 1; do you have any problem with that? It violates the contract here because we were allegedly given or passed a set – ooh, an independent set of vectors. Caught myself that time, okay? If A1 were zero, that contract was violated that we were not sent an independent set of vectors, okay?

Now, the second one is actually really interesting. What would happen if we – oh, not here, here you go. What if there is a divide by zero exception at 2B; what would it mean? It would mean that Q2 – well, it'd mean Q2 tilde is zero. That means A2 is equal to this thing times Q1. Q1 is also a multiple of A1. That says A2 is a multiple of A1. Once again, it means this – you get a divide by zero exception only if A2 is a multiple of A1. That means the pair, A2 A1, A1 A2, is not independent. It means that the preconditions didn't hold. Okay.

By the way, some people use – you can use a Graham-Schmidt-like procedure to check, in fact, if a set of vectors is independent. That came out right, by the way. It checks because if it has a divide by zero exception somewhere, instead of, sort of, jumping core, it can throw an exception saying – in fact, it can be quite explicit. It can actually say these are not independent. In fact, it can say the following: The last vector that I processed is actually the following specific linear combination of the previous ones. So it actually returns a certificate proving that the set of vectors is not independent. Everybody see what I'm saying here? So these are the methods, and we'll see how that works.

All right. So here's the geometry of it. You start with A1, that's over here, and that's also Q1 tilde. The only thing here to do is to divide by the norm, so apparently that's our unit length. The next step is you take A2, which is like this. You subtract off the Q1 component, that's you subtract off this. This vector is this thing here, right here. I subtract that off, and what I'm left with is this. Oh, by the way, there's a name for this. Some people call that The Innovations or something. In a statistical context, in signal processing context, it's called The Innovations, this thing.

I mean, it makes sense. This is, sort of, what's new in A2 that you haven't seen already in A1; it's what's new. There's other reasons why it's called The Innovations, but, I mean, that, sort of, makes perfect sense even in this geometric context. So that's the innovations, and you simply normalize the innovations. So this is the picture.

Now, if you do this for awhile – if you run up our steps, you have the following: At each step here, I take these terms here, and I put them on the other side, and you see something very nice which is that A3 is equal to Q3, which is, in fact, nothing but – Q3 tilde is going to be Q3 × norm Q3 tilde or something, and times a scalar, and you see that this expression here gives A3 as a linear combination of Q1, Q2, Q3.

Oh, by the way, that's, kind of, what we're required to do. This says that at each step, you express AI effectively as a linear combination of Q1 up to QI, like that. And we're gonna give these names. We're gonna call these R1I, R2I, and RII, okay? And they just come right out of the Graham-Schmidt Procedure. And RII is not zero because RII – in fact, not only that, you can even say this. RII is positive because it's actually this thing, okay?

Well, if you write this set of equations out in matrix form, let's see what it says. It says that each AI is a linear combination of Q1 up to QI, and if you write that on the matrix form, it's an embarrassingly simple formula. It's four ASCII characters; it's A = QR. So A = QR. Here A is RN by K. That's a matrix; you've concatenated the vectors into a matrix. Q is of the same size; it's N by K, and R is K by K, but what's interesting – oh, let's see how this works out.

This says A = Q × R, and you can check that by the rules of batch, batch matrix multiplying, and if you interpret column by column, it says basically that A1 is this thing times that, and that's Q1 × R11, exactly like we said. A2 is this thing times the second row, and that's gonna be Q2 × R12 + – Q1 × R12 or R12 + Q2 × RQ2. These are scalars, so they, kind of, they rotate forward. That's the picture.

The lower triangle of – the fact that this is upper triangular here comes from the fact that when you apply this procedure, you express AI only as a linear combination of Q1 up to QI. It doesn't involve QI + 1, QI +2, and so on. There's, sort of, a causality here, which is, sort of, what you should think of automatically when you see an upper triangular matrix or a lower triangular. Suitably interpreted in the meaning is something like causality here.

And, in fact, I guess I can say what the causality is here. This algorithm, the QR algorithm, spits out Q7 before it has even seen A8, A9, A10. Those may not even exist. That's the causality, that when you calculate Q7, you only need A1 through A7 to calculate Q7, not A8, not A9, which may not even exist. So that's the causality that you have here. Okay.

Now, it's a beautiful decomposite that says you can write a matrix as a product of a matrix whose columns are orthonormal – that was slang – times a lower triangular matrix which is actually invertible, upper triangular matrix is invertible. And this is called the QRD composition, and it's also, maybe, sometimes called the Graham-Schmidt Decomposition or something like that, but this is it. Okay.

And that's called a decomposition or a factorization of A, and then you'll find, actually, in terms of English, there are two verbs. I don't know which is right, but you will say that the verb to do this is either to factor or to factorize, and I think it depends on whether you, your advisor, and your advisor's advisor were, maybe, educated, like, at Cambridge, or Oxford, or something like that. I'm not exactly sure, but you see it about half and half. People will say, "You should factorize QA to get QR or factor." I think, actually, factorize is winning out now, so – all right. So that's called a factorization. We'll see several factorizations before the class is over. Oh, by the way, I've now shown – by the way, we have a big stack of unproved things, and we just popped one, for the record.

Earlier I said that if a matrix – well, not quite; we didn't get that. This is one of – this is – well, okay. No, we didn't. Sorry. Scratch all that. So it'll be fun. When we can actually edit the videos it's gonna be fun because I'll start and then my head will go like this, and it'll be a little bit shorter. It'll be good. I can't wait to do that. All right. Forget all that. Let's move on.

By the way, the method I showed here is – the Graham-Schmidt Procedure the way I showed it is actually – and obviously it's completely correct. It is not the way it's actually done in practice. In fact, people use a variation on it called Modified Graham-Schmidt Procedure, which is less sensitive to numerical rounding errors, and, in fact, they don't use the Modified Graham-Schmidt Procedure either. I mean, if you really want to know how it's done, it's done by blocks and things like that.

So it's fairly complicated, the same way, I think, at one point, when I was ranting last week, maybe, about how do you multiply two matrices. Anyway, it's not the four line C program. It's actually done by blocks and things like that, same for QR factorizations. If you were to look at the actual code that runs, for example, in Matlab, if you do a QRD composition – and let me point something out.

Math Works had nothing whatsoever to do with any of the numerics. All of the numerics in Matlab – just to make sure this is absolutely clear because it irritates me beyond belief when people think otherwise. Every numerical method in Matlab relies on open source public domain software written by professors and graduate students, just so you know. So please never say, "Oh, the numerics in Matlab, beautiful." Because you, and your friends, and your professors, and colleagues, and peers wrote it, and it's free for everybody. Sorry, just had to have that slip out. Okay. All right.

Okay. So what happens is this is actually an algorithm for computing an orthonormal basis for the range of a matrix. I mean, that's what it does; it creates an orthonormal range for the basis. By the way, as a method right now, it doesn't really – you've got a little bit of algorithmic teeth. A modified version of this would actually allow you to check if a skinny matrix is full rank. It would allow you to check if a set of vectors is independent. How would you do it?

Well, using this method, as I said, you would modify the algorithm to not dump core on a divide by zero exception, but instead, to return something saying they're not independent. That's what you would do, okay? So at least you have a little bit. You have an algorithmic method to check the rank – I'm sorry, to check if a matrix, so far, is full rank. We're gonna get to rank in a minute.

Now, the General Graham-Schmidt Procedure – General Graham-Schmidt procedure works like this. In the one I just described, we assumed the input set of vectors is independent. Notice that's slang in writing, but you're allowed, by the way, to write slang on lecture notes, not in papers, by the way – oh, unless it's obvious. Unless it's in a paragraph in which it's clearly, kind of, fuzzy and giving the idea, then you're allowed to use slang, but in lecture notes I'm allowed to use slang.

Now, if they're dependent, what'll happen is you'll get a divide by zero exception in the algorithm we just looked at. Now, here's the modified algorithm. It's really simple. It does this – let's go back to the algorithm and see what we do. Here's what happens. What happens is this. You do the same thing as before. So, for example, you remove the previous Q components from the current vector, all right? Now you attempt to normalize. The only thing that can fail here is that Q3 tilde can be zero.

So you simply put a test in front that says if Q3 tilde = 0, by the way, that has a very specific meaning. That will only happen – it means, specifically, that A3 is a linear combination of Q1 and Q2, which is the same as saying it's a linear combination of A1 and A2. That's the only way this will fail. If it fails, you simply fail to do this and you actually just get – you say get me another column. You just say, instead of dividing, you call get call or get next vector, and you pull another vector in, and you do the same thing; you normalize. You didn't spit out a Q.

So this is the algorithm. It runs like this: U form AI tilde is this thing, so you actually orthogonalize with respect to the previous things here. If there is any innovation because A tilde you can interpret as the innovation. It's basically, sort of, what's new that I haven't seen already? That's A tilde. If this is zero, you just do nothing. You just go right back, and you get another column, and you keep going. You don't generate a QI. Otherwise, you generate a Q.

So the number of Q's you generate actually is gonna tell you something about the rank. Okay.

Now, by the way, this works perfectly nice. Let's see how this works just for fun. All right, let's do it this way. Suppose I put in vectors that look like this: 0, E1, 2E1, and E2. Actually, you know what? It doesn't matter. I'm gonna call that A1, A1, and A2, and A1 and A2 are independent. What does a Generalized Graham-Schmidt do here? Well, it checks zero. It orthogonalizes it with respect to the previous Q's of which there are none, and it attempts to normalize.

You're gonna have some serious trouble normalizing zero, so this doesn't happen; this fails. So you simply skip over it, and you get a new vector which is A1. You orthogonalize with respect to previous Q's; there are no previous Q's, so you are already orthogonalized, and now you normalize with respect to A1. So we divide A1 by its norm, and that's gonna be Q1. And so notice that the span of the first two vectors here, that's zero and A1, is now, in fact, Q1, which has dimension one, okay?

At the next step you get 2A1. You orthogonalize, and you'll get zero because 2A1 is in a linear combination of A1 – well, of A1. So here, once again, this fails again, and you don't spit out a Q. You pull an A2, and then you get a Q. So as you process these four vectors, you will, in fact, generate only two Q's. Reason, the rank of this is exactly equal to two, okay?

So you can actually say much more. You know, it's not here. It's, kind of, silly, isn't it? Oh, it's not Q1 through QI; they're indexed by a different way. Q1 through QR is an orthonormal basis for the range of A, but notice what happened here. R can be less than K, okay? So that's what happens. R can be less than K. Oh, there's another name for this. This thing is sometimes called the Rank Revealing QR Factorization.

So that's a great name for it. Rank Revealing QR, and you will hear this. People will talk about a Rank Revealing QR Algorithm. That's basically a Q algorithm that's modified like this, doesn't dump core when what you pass it is not independent, but an interesting thing is when it terminates, it has actually calculated the rank; it's got the rank.

But there is one thing, by the way, that we're not looking at and, in general, don't care about in this course. They're important issues. It turns out that you have to be a little bit, you know, when you have numerical codes, this is not A tilde = = 0. In other words, it's actually exactly zero as a double vector. Now, more likely than anything, this is something like the norm is less than some tolerance which is really small, okay?

So whenever you see something like a rank revealing QR, be forewarned that there is a tolerance in there, and that you may have access to it. You may want to change it. You may want to know what it is. It is probably set very low, so low that, for engineering purposes – it may be inappropriately low for engineering purposes, but I mention this just because it's an issue. You really don't have to worry about it, but you should be aware that there is a tolerance in a real rank revealing QR algorithm. Okay.

Now, in matrix form, you can write it this way. You can write A = QR where Q transpose Q is I. Now, you should simply process these five ASCII characters. You don't need the R because you're sophisticated now. You know that's just simply the width of Q. When you see Q transpose Q = I, that should simply be an alias or a macro for the columns of QR orthonormal. So that's what this is.

And R is an R by K, and it looks like this. The R's get spit out this way; it's quite interesting. I think it's called Upper Staircase or there's another one you'll hear like Upper Echelon or some – I don't know. You'll hear various names for this type of factorization – upper staircase, upper – I don't know. Who knows? Reduced Row Echelon – I don't know what it is. Anyway, but it looks like this.

Now, the corners on the steps occurred exactly when you spit out a new Q, which occurred when the orthogonalization, the innovation, turned out to be non-zero. So for this picture, we can say a lot about the rank of various bits and pieces. So I'm gonna ask you some questions. The fact that a Q1 was generated on the first step is equivalent to what statement about the columns A1 up to AK? We generated a Q1 on the first step, why?

Student:A1's not zero.

Instructor (Stephen Boyd):A1's not zero. We generated a Q2 on the second step; what does it mean?

Student:[Crosstalk].

Instructor (Stephen Boyd):A1, N2, H over independent. Okay. On the third step, nothing was generated. It has a meaning; what's it mean?

Student:[Crosstalk].

Instructor (Stephen Boyd):Okay. A3 is the linear combination of A1 and A2, exactly. Okay. So you get the idea. Okay. So actually the structure of this is quite revealing. It, kind of, tells you – well, it certainly tells you in ascending order, it, sort of, tells you which vectors are dependent on previous ones, and so you get a lot of information and structure here. Okay. Oh, I don't know. Let's see. It's only count out a couple. What is the rank of A1, A2, A3, A4?

Student:[Crosstalk].

Instructor (Stephen Boyd):It's three, and, in fact, we have an orthonormal basis for it; what is it?

Student:[Crosstalk].

Instructor (Stephen Boyd):Q1, Q2, Q3 – Q3, I heard Q4, but I'm just ignoring it, okay. That's it. It's Q1, Q2, Q3, right. Did I do that right? Yes, I did. All right. It's okay. Okay. Now, there's another way to write this and another way you'll actually find this – this is one method. What you can do is actually you can permute the columns. Now, permuting the columns is the same as multiplying by a permutation matrix on the right.

I used to get all confused about this, and I still do, sometimes, but I just go – actually, the one thing you do actually if no one's looking, you can write down a 2 by 2 matrix and do this secretly, but not if anyone is looking. Don't ever do that because it's – well, it's humiliating, really, if someone catches you. I'm talking about whether permuting columns or rows corresponds to permutation multiplication on the left or right.

The other way to do it is say well, look, columns are associated with inputs; you know that. So, basically, if you mess with the inputs, that's something that should happen first before you apply the matrix. If it applies first, it's on the right. I can see this was extremely unhelpful. Well, okay, fine. Make up your own mnemonic. I don't care. Okay.

So I take this matrix here, and I apply a permutation on the right that rotates these columns forward, right? And takes all the columns when there were no innovations and rotates them to the back. If I do that, it's gonna look like this: A is Q, R tilde, S – that's, kind of, whatever's left over – × P. Now here, Q is, again, a matrix with columns that are orthonormal. R tilde is upper triangular and invertible, so now it looks like this. It's just like the old QR factorization; that's that. S is – well, who cares what S is, and it looks like that. So you have this picture. Okay? And then Q goes here, and A is the same size as Q, okay? So that's the picture. Yeah, you've rotated everything forward – or rotated the ones where you had something. Okay. So this is another form you'll see this in. Okay.

All right. Let's talk about some applications of this. Well, it directly yields a orthonormal basis for the range of A. Now, I'm gonna go back. Remember, I went on a little detour, a little ahead of time. Now it's actually time. So, now, we're gonna actually pop something off our stack of unanswered questions, or something like that, or unproved assertions.

Earlier in the class, there was a factorization interpretation of rank, and the factorization went this. It says that if a matrix A has rank R, then you can factor it as a matrix, you know, you can make a tall, skinny factorization, okay? So you can get a tall, skinny factorization where the intermediate dimension here is the rank. I asserted that without proof, and we also had interpretations of this right then. Okay.

Now you have one because here is one, right there. So that gives you one right there. That's exactly the size. Here's A is Q, that's the first part of the factorization, that's what I called B, and you call this thing C, and it's got exactly the right sizes, so it gives you that factorization.

By the way, this means – and you can do, sort of, a pseudo application. Earlier we found out that a factorization of a matrix, especially if it's a stunning factorization, like if it's 1,000 by 1,000 matrix, that, for example, has rank 15, and it factors as 1,000 by 15, and 15 by 1,000 matrix.

We found out many things you can do now. One is you can multiply by that matrix super, way faster than you could before, okay? And let's not make it 1,000. Let's make it a million, million by million matrix, okay? So if A is a million by a million, it says that you can do this – it says you can factor it this way. You couldn't even store it, a million by million matrix, let alone multiply by it, okay?

Factored as a million by 15, 15 by a million, it's no problem and would be shockingly fast. I can do that on my laptop. So, now, the question is you're given such a matrix. Of course, you can't store it, so the story gets a little bit fuzzy here, and you're asked to see if you can do such a factorization. You can apply something like this Modified Graham-Schmidt, and you can keep getting columns and stuff like that.

By the way, you don't actually have to ever store a whole lot when you do this algorithm, right? All you need is a method to get – you just request columns, and you check against the other one. You could basically say you know what? If the ranks more than 20, forget it; this idea is not gonna fly. So you keep processing the columns, and you see how many Q's you pick up. If, in fact, you process all the columns, and Q only got to 15 or 16, you win. You have a factorization, low rank.

That work was an investment. You can now multiply that matrix by that matrix some absurd about of time faster. If that's a simulation, or some signal processing calculation, or something like that that's gonna happen a lot of times, you just won big. By the way, this is not so useful because these are exact factorizations. Later in the class, we're gonna see approximate factorizations. That's much more powerful, is an approximate factorization, but for now, we'll leave it this way.

Student:[Off mic].

Instructor (Stephen Boyd):No, it doesn't. No. I'll repeat the question. In Matlab, when you multiply big matrices, does it attempt to do anything like this? And the answer is no, it doesn't. In fact, I don't really know any linear algebra system, although one could be developed, that would do cool stuff like that. That would basically flag a matrix as low rank and – now, ideas involving rank are widely used in numerical linear algebra, but, as far I know, there's no general system. It'd be quite cool to build one, actually. Okay.

So let's see. We've already checked a couple of things. Oh, here's one. If you want to check if a vector is in the span of a bunch of matrices, which is basically saying, for example, suppose you want to check if B is in the range of A, no problem. All you do is you could run the Modified Graham-Schmidt on A1 up to AK and B, and what you check on is this. You only check that when B was pulled in, you generated a new vector. If you did generate a new vector, a new Q, that means B is not in the range. If you didn't – if on the last step, you didn't generate a new Q, then B is in the range of the previous ones. Actually, the algorithm automatically solves, by the way, AX = B in that case. It finds an X that gives AX = B.

Now, one interesting fact is that because of this causality, the Graham-Schmidt Procedure yields a QR factor is A and yields QR factors that if you stop it early, and if you send a terminate signal to the QR process, and it stops, it actually has calculated a QR factorization of a leading – if it's processed P columns, it's the leading sub-matrix of A, has already been processed.

So what's, kind of, cool about that is this. It says that when you actually calculate, for example, the QR factorization of A, you've actually simultaneously calculated the QR factorization of every sub-matrix of A. I have to say, it's not every, but it's every leading sub-matrix. So here's A, and I'm gonna call a leading sub-matrix, what happens if I truncate this way, okay? And you'll actually have – if you truncate Q and R appropriately, in the same way, you get the QR factorization of a – oh, I don't know. What you call that; what would you call the first chunk of a matrix? What would you call it?

Student: The head.

Instructor (Stephen Boyd):The head of the matrix? Sure, yeah. Yeah – no, no, no, that's fine. Okay. What do you call it when you have a string at the beginning of a string?

Student: Prefix.

Instructor (Stephen Boyd):Pre – what, prefix? Prefix of a matrix; what do you think?

Student:[Crosstalk].

Instructor (Stephen Boyd):Head, prefix? Listen, we gotta step in. If we don't, the physicists are gonna step in, and it's gonna have some totally idiotic name, so – sorry. That was just for you. Yeah, I presume you're in physics?

Student: No, I was.

Instructor (Stephen Boyd):You were.

Student: Yeah.

Instructor (Stephen Boyd):Okay. What are you in now?

Student: Aero.

Instructor (Stephen Boyd):Aero, all right. I have plenty to say about those people too, sorry. You had a question? No, okay. All right. So prefix or head at the moment. I'm just saying we have to be involved because, otherwise, you won't believe the names other people will assign to them. So all right.

I'll mention one more thing which is the full QR factorization. Full QR factorization is this. You write A = Q and R1. Let's call that the QR factorization we just worked out, without the permutation, just assume it's full rank or whatever. It doesn't matter. Actually either way it doesn't matter. You write Q – here you put Q1, and what you do is you fill it out to make it square, okay? And you put a zero here, so it really, I mean, it doesn't matter, and let me explain a little bit about how you do this.

This is a very common procedure. It basically says given a, like, let's say K orthonormal columns, that was slang. I'm gonna stop saying that, but anyway. Given an orthonormal set of K columns, the question is – and these are in R10. It's very standard to fill it out, which is to generate something like a Q8, Q9, and Q10 that makes the whole thing an orthogonal matrix, okay?

Now, it sounds weird. The question is how do you do it, and the way is actually quite straightforward. What you do – let me just show you how you would get this Q2. It's really, kind of, dumb. It works like this. Let's feed in to our rank revealing Q4 the following: A and then the identity, okay? So that's what we're gonna feed in. Now, that matrix, sure, it's definitely full rank because of this I here, okay? No doubt about it, that's full rank.

But what we're gonna do is I just want to set a flag that the first time our QR algorithm gets a column, has to pull into the I matrix to get a new column, I want to set a flag. So here's the way it happens. Our modified QR starts working on A, pulls in column one, processes it, two, three, four, five. Let's say it finishes up A, and it doesn't matter what the width of A is. Well, let's just put some numbers on here for fun.

That's ten. Let's say that A is, you know, seven vectors, and let's say I've run this Graham-Schmidt QR Factorization, or Graham-Schmidt, or algorithm, or whatever on this, and let's suppose the rank of A is six. So what will happen is I will have pulled in A7 and processed it, okay? And how many Q's will I have at that point? Six, I'll have an orthonormal basis for the range of A. Okay.

Now, normally that's where you stop. That's where our old factorization stops. Now we're gonna do the following though. I'm just gonna go get E1 because that's the next thing here, but I'm gonna set a flag, which is by I've overrun – I'm indexing out of my matrix, okay? I'm no longer processing A. I'll take E1. Now, E1 might be in the range of A, in which case, I won't generate a Q7, okay?

No problem if it isn't there. I pull E2 in, and I process that, okay?

Now, when I finish here, I must have Q1 through Q10. I absolutely must, why? Because the rank of this matrix is 10. So, however, the last three, Q8, Q9, Q10 were not generated from A; they were generated from this I or whatever other matrix you put here, okay?

However, the Q8, Q9, Q10, they're all normalized. They're mutually orthogonal, and they're orthogonal to Q1 through Q7, and I'm gonna write it this way.

So this would be that first – this is the chunk of Q's that I got out of A, and this is what happened when I actually – I called get call on A, and it returned an EOM. I guess that's end of matrix flag, okay? So it returned an EOM, and I said no problem, and I got a column from – I just got E1, let's say, and kept going. So that's what this is, okay? So this is how you do it, and this is very cool because this is finishing off, extending an orthonormal basis from a given basis out to a basis for the – a basis for a subspace to a basis for the whole thing.

I should add, all of this, at the moment, since we're not looking at applications, it probably doesn't make a whole lot of sense, or, even if it made sense, is not that interesting. It will be very interesting when we actually start doing stuff. There was a question?

Student:[Off mic].

Instructor (Stephen Boyd):E2, you go to E2. Then you go to E3, but the point is, by the time you finish processing this, you have to have ten because the rank of this matrix, I guarantee you, is ten no matter what is, including A = 0. That's about a low as ranks go, okay?

So in A = 0, let's find out what would happen here is A would generate no Q's anywhere. So you'd stop. Then it would be correct. It'd be, like, an empty matrix or something like that, no Q's, okay? Then the whole thing – then I would actually, in fact, be Q2 because they're already orthonormal, and in that case, this would be zero – not zero, sorry. No, and Q2 would be I. Does that answer your question? Okay.

So this is called extending an orthonormal basis. Now, the two subspaces, the range of Q1 and range of Q2, they're actually very interesting. They're called Complimentary Subspaces. They do two things. First of all, they're orthogonal. We overload – right now, you know what it means to say X perp Y. It means X transpose Y is zero. They have zero inner product. That's what that means. We overload the perp to sets of vectors.

So you can say a set A is perpendicular to a set B, and the meaning is this. Any vector in A is orthogonal to any vector in B, okay? So here, you would say that range of Q1 and the range of Q2 like this are orthogonal like that. It means that anything in here and anything in here, and, by the way, you need to check that. You really need to check that; you can check it.

It means that anything in here is a linear combination of, let's say, Q1 has got columns Q1 through QR, and this has columns qR + 1, that's little q, you can tell by the way I say it; it's a little bit softer, the lower case q. It's, like, qR + 1 up to qM. This is a linear combination of the first R1's, and it's a linear combination of the others, but they're all mutually orthogonal, so the inner product is zero. Okay.

Now, the other thing is we overload sum for sets. When you overload the sum for a set of vectors, it means the following. It's a new set, and it consists of all sums, all possible sums, of a vector from this set plus a vector from that set, okay? So we're gonna overload plus as well to sets of vectors, okay? Now, so you write this this way. If you do that, you'd say that RFQ1 + R of Q2 is equal to RN, like that, and what that says is the following. It says if you take all possible sums of a vector from range of Q1 plus all possible – well, if you pick a vector from here and here, add them, and then do that over all possible choices, you get RN. Yes, a question?

Student:[Off mic].

Instructor (Stephen Boyd):What's what?

Student:[Off mic].

Instructor (Stephen Boyd):The bar, you mean this thing?

Student: Yeah.

Instructor (Stephen Boyd):I'll tell you what that means in a minute. Okay. So this is what people would write. And, by the way, the way you'd say this in English would be something like this: oh, range of Q1 + range of Q2 is all N, or you'd say something weird like together they span RN. That would be the English phrase. That's how you'd say that informally.

Now, this bar, this is a symbol that says R1 – this terminology means R1, the range of Q1 + the range of Q2 is RN. That's what the plus means, and that little perp, it saves you writing this: And range of Q1 is perpendicular to range of Q2, so all the perp means is that. You will see it. There's no particular – it doesn't save you a whole lot of trouble to write out both, but you have to admit, it looks cool, and it's a very advanced notation, and if you do that, people won't know what you're doing, and you can say – well, you'll see I'll give you some words you can say, and your roommates or whatever will be very impressed. So you asked what it's for; that's basically what it's for. Okay. All right.

So another way to say that is this. In this case, you say that they're orthogonal compliments, the two subspaces, and another way to do it is the perp symbol actually acts as a post-fix superscript on a subspace, and it is, in fact, the orthogonal compliment. It's the set of all vectors orthogonal to every vector in this set here, okay? We'll have a homework problem or something on this soon to, kind of, help you get straight about this, and we'll look at some more of this in another context.

So now, I want to say one more thing. This is, at the moment, gonna look a bit bizarre, but later it'll make sense, I promise. If you do a QR factorization of A transpose, now A and A transpose have exactly the same rank. So I do a QR factorization of A transpose, I get something – sorry. If I do a QR factorization of A, which is a full one, and I transpose it, I get this. Sorry, I should've said A is Q1, Q2 × R1 N0, like that, okay? That's the full

QR, and I transposed this QR factorization, and you find the following. You can now imagine what A transpose does to a vector by first operating here and then operating here.

Now, this is interesting because basically it says that when you form A transpose Z, you first for Q1 transpose Z and Q2 transpose Z. So you produce two sub-vectors. The first one – let's see if I've got this right. No, I've got it totally wrong, completely and totally – there we go. All right. So it's really this. Q1, Q2 – let's try that again – is U form R1Z and zero here. When I multiply – no, no. I had it completely right.

I'm wondering if this posting the lectures on the web is such a great idea. We better look into editing them pretty fast, I think. That's my conclusion, okay? This happened just because they've been posted. All my friends at MIT will be watching this, chuckling. I'll get you, don't worry. Okay. Let's try it again.

Okay. We're just gonna go slow here. R1 transpose zero, okay, there we go. A transpose Z is this. There we go. So the first thing we do is we calculate Q1 transpose Z and Q2 transpose Z. Q1 transpose Z gets then multiplied by R1 transpose and actually causes something to come out. Q2 transpose Z then gets zeroed out. So you calculate Q2 transpose Z, but then nothing happens, and now you see an amazing thing. You see that basically A transpose Z = 0, actually if, and only if, Q2 transpose Z = 0, which is the same as saying that Z is in the range of Q1.

Student:[Off mic].

Instructor (Stephen Boyd):Where? Thank you. It's just not – this is what happens. I knew we shouldn't have done it, see? Okay. There we go. Thank you. There we go. All right. Okay. Now, what this says is interesting. It says that the range of Q2 is the null space of A transpose. It's the null space of Q1 transpose, which is the same as the null space of A transpose, okay?

And what you find out is that this second – when you fill out this second part of the QR factorization here, this Q2, what it really is is the columns are an orthonomal basis for the null space of A transpose, okay? And what you include is this, the range of A and the null space of A transpose are complimentary subspaces, okay? Indeed, one of these is the span of Q1; it's the range of Q1, and the other is the range of Q2 obtained in this full QR factorization. So they are complimentary subspaces.

That means, again, this is the method used to impress your roommates and so on. It means that they add up to all of RN, and it's says that they're also orthogonal subspaces. By the way, one consequence of this is the following, is that every vector in RN can be written uniquely as a sum of a vector which is in the range of A and a vector which is in the range of A transpose, and more over, those two vectors are orthogonal, okay? That's what it means.

And so people call this the Orthogonal Decomposition of RN induced by the matrix A. That's the name for this, and once you know this, and we have, now, a constructive method for doing this. In fact, we simply apply the Modified Graham-Schmidt to AI, that's the algorithm. If you do that, you can generate this Q1 and Q2, and all of this will actually – and now it's constructive.

Now you can actually prove most – I think maybe actually all of the assertions from the linear algebra review once you know this. It'll actually be a little bit easier in a couple of weeks when we do more material, but this is it. But if you switch A to A transpose, you get a decomposition of RK, which is the other side, and you get this, the null space of A is an orthogonal compliment of the range of A transpose. And so these are the various – these are, by the way, the four fundamental subspaces. There's whatever people glorify them as, these four.

Actually you don't have to worry about them too much. All that really matters is what they mean in the context of applications, I think. So that's the importance, so you don't have to – these are just silly symbols that are interesting. It's much more interesting when we start doing estimation, or channel decoding, or input design. Then these things are very interesting, and they have very real consequences, and that's what we'll be doing in the next bit.

Let me show you a couple of things here. When you show your friends – when you assert this they say, "Wow, that's a – " oh, by the way, some people consider – I think this is sometimes called – this is also sometimes called one of the – this is also one of the major "results" in linear algebra. So it's a big deal. It's treated with a lot of reverence. You're never supposed to joke about this because this is – I mean, it is cool, all right? But the point is it's – I want to point out a couple of things. Bits and pieces of it are trivial, but one-half of it is not. But want to see people look at that and go like, "Wow, I need to really go to a quiet place, and sit, and actually think about what that means." That's pretty serious stuff, but I want to show you one thing. I hope this is gonna work out. We're gonna try.

I'm gonna at least – why don't I show you this part? Range of A is orthogonal to the null space of A transpose, right? That's one-half to the assertion. That's why the little perp is sitting above the plus, that's what you asked. So let's see. Let's show that. What do you have to show here? Well, really, I have to show that any element of this thing is orthogonal to any element of this one. I hope this works, by the way, but considering that we just posted these things online, it's almost certain I'm gonna end in a deep, deep hole, but let's just see what happens, all right?

So a general element of the range of A is a vector of the form, let's say, AX, and then I'll put a question mark here because this is what we have to show. Let's call a general element of null space of A transpose, I'm gonna call it W, but we put a little note, which is that A transpose W = 0. Oh, yeah. It's gonna work, okay? So this is what we, kind of, have to show here, and let's see. Well, I mean, to check orthogonality, you write this. The question mark means it's open; it's not shown yet. Okay.

Well, that's X transpose, A transpose W, and if you don't mind, I'm gonna just reassociate right now and skip a step, and I'm gonna look at that and put a question mark there, and I can erase the question mark because A transpose W is zero, okay? So now, the way you do that – this is the way you derive it. The way you hand in your homework or make your argument is the other way around.

You start and you say well, let's take W to be in the null space, that means this, and then you say let's let – here's the general element of range of A, it's AX. Then you say, well, let's just form the inner product this way, and you say let's start by writing this down. You first say to someone, "You'd agree with that, wouldn't you?" And they'd say, "Yes." And you go, "Well, no problem, but A transpose W is zero, right?" "Yes." "So I can then write this. You're not putting the question marks there." And they'd say, "I guess." And then you go, "Cool, then I can reassociate it and rewrite that that way." And they'd say, "Yes." And then you'd say, "Oh, I'm done." So that's the way that works. Anyway, what I wanted to point out is that in all of these things, some of them are very easy, and the others are not that easy. Okay. All right.

So I will start saying a few things about the next topic because it's the first one that is actually real and useful. It's least squares. I'll say a little bit about it, maybe just now because we're just finishing up, I'll just say a few things about it. It's not a new topic. It goes back to Gause who wrote, actually, a book about it and may or not have been the inventor of it, but as far as most people know, he was or something like that, and he wrote a book about it in Latin, which was actually just translated, which is, kind of, cool, and you can get the book. I mean, not that you're required to do so, but you get the book, and then on the left is the Latin, and on the right somebody translated it, so you get to see it.

And, by the way, if you read the book, you'll be very, very grateful you were born in a time when we actually, A.) Had notation because imagine what happens when you have to write out all of this stuff. Y = AX, that's, like, a page of Latin, okay? The transpose goes on, and on, and on. I mean, this is not simple stuff. That's No. 1, so that's the first thing you can be grateful for. Also, that you don't have to read it and the lecture notes for the class are not in Latin, another thing you could be grateful for.

The other one, actually, is that you were born in a time when computers will actually do all this stuff for you, and you can actually use this stuff, and embed it, and things like that. So anyway. All right, that's just history. I'm just telling you that least squares is not exactly a new topic. Okay. But I'll tell you what it's about. It's about approximate solution of over-determined equations. So let me say just a little bit about that so that Thursday we can do this.

Let's suppose you have Y = AX where A is skinny, okay? So that means something like you have more equations than unknowns. It depends on the context of what exactly it means. You have more specifications than you have designed degrees of freedom, lots of ways to say it. You have more measurements than things you have to, you know, it's a redundant measurement system, or at least, I should say, it's a dimensionally redundant measurement system. It would appear, based on the sizes that you have more measurements than you need.

So people call this an over-determined set of linear equations, and basically for most, Y you can't solve that equation, obviously, because the range of A is a subspace of dimension M in RN. That's very small. In fact, even if M is N minus 1, in some, kind of, uniform distribution – with any probability density, if you draw Y from it, the probability that you can solve Y = AX is zero, right? Because a subspace, any strict subspace will have probability zero under any probability distribution, if you know about that kind of stuff. Okay.

So there's a compromise, and the idea is to not solve it but to approximately solve it. So you approximately – now, what does it mean to approximately solve? It means, instead of finding an X for which AX = Y, which is impossible, we're gonna find an X for which AX is approximately Y, and we'll measure proximity by the norm of the error. So that's the basic idea, and I think we will quit here.

[End of Audio]

Duration: 77 minutes

IntroToLinearDynamicalSystems-Lecture06

Instructor (Stephen Boyd): Are we on? We're on. Let me go down the pad here, and I'll make a couple of announcements. The first one, is that Homework 3 is now posted and Homework 2 is due this afternoon. I want to make one other comment, this is just in case I forget to announce this at the end of the lecture, you should read the notes on – there's several notes in the notes section on the course website. One is on notes on least squares, least norm, and solving linear equations. I think that's two different sets of notes. We won't have covered all of the material, we will not have covered all of the material by today, but there's no reason that you can't read these. These are each two pages or something like that, and you need to read these.

I'm trying to think if there's any other announcements. I should probably remind you, I'm sure you know, that the midterm is coming up actually in two weeks. And, yes, I'm as shocked as you are at this. We've scheduled a day, or days, there are two days. You can take it on either Friday to Saturday or Saturday to Sunday. If that's not gonna work for your schedule, like you're in Antarctica or something like that on those days, you need to contact us, and we'll work something out. Generally, it means you take it early, and that makes you a beta tester. Whatever you do, don’t just not show up those day, because we'll come looking for you, and we know where you live or we can find out where you live.

Okay. Any questions about last time? If not, we'll start in on – actually, the first part of the course, I think I hinted at this last time, which is actually quite useful. Everything else, so far, has just been sort of background, really leading up to this and its infrastructure. We're actually now at the payoff point.

So we're gonna talk about least squares. It's something you've probably seen in a couple of different contexts, and it concerns overdetermined linear equations. So we have a set of over determined linear equations. Now, here we have y=ax, where a is we'll make strictly skinny. It's overdetermined because you have more equations than unknowns. And, of course, unless y is in the range of a, which if you pick y randomly, and rm is an event of probability zero, you can't solve y=ax. So one method to approximately solve y=ax, and it's very important to emphasize here we're not actually solving y=ax, is to choose x to minimize the norm of this residual. So ax - y, by the way, some people call the residual y – ax, it doesn't make any difference because mostly you measure yourself by the norm, and the norm is the same. So the residual is ax - y. It's the amount by which ax misses y. If you take the norm of that, that's the distance between the point ax and the point y. If you minimize that, that's called the least square's approximate solution. Now, here, I'm going to insist on using approximate. I hate to tell you that, on the streets, you would call this the least squares solution of y=ax. This drives me insane. Because whenever somebody says the blah, blah, blah solution of y=ax, I presume, in particular, x solves y=ax. And it's very important to understand in this case, it does not solve it, that's the whole point, it's an approximate solution. Of course, it can solve it, that's possible. And that, indeed, would be the least squares solution, because residuals don't get any smaller than zero, norms of residuals. So I'll call it least squares approximate solution, but you will here people, if you go to Wikipedia, or whatever, and type it in, you'll hear people talk about the least squares solution of y=ax, which is not a solution of y=ax. So the geometric interpretation is something like this. This is very childish, that I can only draw this in r2. In this case a , of course, is 2 by 1, it's kind of silly. The range of a is a line, so here's the range of a, and here's a point not on the line. And so the question is to find the point closest to this target point y that's on the line. And there's a name for that. When you find a point in a set closest to the point in the set, closest to a given point, that's called the projection of the original point on the set. So here ax, ls is the projection of y on the range of a. And you'll see all sorts of notation for that, but maybe the most standard would be this. So ax, ls would be a projection of this point on that set. So that would be probably, if you looked at ten books, you'd probably see this in eight. So that's the geometric interpretation. Let's work out what it is. It's not too complicated to actually just work out a formula for it. So a is full rank and skinny, that's our assumption here, and what we need to do is minimize the norm of the residual. Well, that's the same as minimizing the square of the norm of the residual. The square of the norm of the residual is r transpose r, and you simply plug in what that is and you get the following function here. Now, that's a constant, that's the norm of y squared, that's a linear function of x, and that's a quadratic function of x, a pure quadratic function of x. The whole thing all together is just a quadratic function of x. This is actually called the quadratic form, the pure one. How do you minimize this? Well, we'll set the gradient with respect to x to be zero. And what you'll find is the gradient of this – is that on the current homework? No, it's on the one we just assigned. Of course, you can work out the gradient yourself by taking the partial derivative of this with respect to xi. But what you'll get is the gradient of this function is 2 a transpose ax, it should look very, very much like the formula in the scaler case, minus 2 a transpose y, and that's equal to zero at a stationary point, in this case it's the minimum. This only has one solution. There's only possible solution of this equation because a transpose a is invertible. It's invertible because a is full rank and skinny. That's one of these basic properties. So if I reduce the two and put it on one side, these are very famous equations, they're called the normal equations. You'll see why in a minute. Actually, the picture before explained that. Since a transpose a is invertible you have this: You have x least squares is a transpose a inverse a transpose y. So I don't how many ASCII characters that is, let's say it's ten or something like that, watch out because the whole point of overloading notation is that a very small number of characters can pack a lot of meaning. So this looks very simple, looks very innocent, it means a lot. By the way, it's also unbelievably useful. So it's a very famous formula, you probably shouldn't forget it. You probably won't forget it because if you look for it you'll find it in lots of other contexts in classes and things like that. So here it is, it's a transpose a inverse a transpose y. You can do all sorts of checks here just to scan this and make sure it's okay. You can check things like, well, if you're gonna invert something it should be square. And, indeed, a transpose a is square. That's fine. You should check, in fact, that this is full rank. For example, what can you tell me about that? Why not? You're shaking your head.

Student:

[Inaudible.]

Instructor (Stephen Boyd):Exactly. This is square but not full rank. So that's a semantic error not a syntax error. How about this one? That's a syntax error. Right. That was vigorous and violent headshaking, which is the correct reaction to this.

So there's the formula for it, it's quite simple. Actually, one very interesting thing here is this is actually a matrix. Let's call it bls times y; it's linear. So calculating this approximate solution is actually a linear operation on y. So that's the first thing you note.

Now, if a happens to be square, then it's invertible, a transpose a inverse is a inverse times a transpose inverse. By the way there's some slang for a transpose inverse, it's not really slang, people write this as a – 1 and then a - t, that's standard. That's what this is. And then if I multiply on the left by a transpose, let me do that, a transpose, I get this, and you get a inverse. So you can see that when x is square, this is just a pedantic and silly complicated formula for the inverse again. So you can think of this as, actually, a generalization of the inverse, and you'll want to think of this as a generalization of the inverse of a. Now, watch out, because in this formula a is not square. But if a were square, this would reduce to the inverse of a. Of course, this calculation here is completely wrong when a is not square. In fact, you have syntax errors already here, right here.

Now, suppose if y is in the range of a, that means there is an x that solves ax=y. In this case, it turns out x least squares is it. How do you know that? It's actually by the definition, x least squares minimizes the norm of ax - y over all possible x. But if there exists an x for which ax=y, then this thing can be as small as zero. You're not ever gonna do any better than that, so that x has got to be x least squares.

That's a good property of an approximate inverse to have, or a method that gives an approximate solution, is when the equations actually have a solution, it will return the solution.

Now, there's a name for this matrix, it comes up all the time, and it's called a dagger, that's a very common, that's universal notation. So it's a with a superscript dagger. And that's called the pseudo-inverse of a, and it's got other names, too. If you're British you'll here it called the Moore-Penrose inverse, and there's probably some other names for it, too. In fact, some people, who like long names, call it the Moore-Penrose pseudo-inverse. And there's other names, but they don't matter.

Here's what's interesting about it, it is a left inverse of a full rank skinny matrix. So it's this, if you work out a dagger a, that's a transpose a inverse times a transpose, that's a dagger times a. And I re-associate this this way, and I see a transpose a inverse times a transpose a, that's the identity. So you have the following: It is a left inverse of a. By the way, the dimensions of this pseudo-inverse here are the transpose of the dimensions of a. So if a is skinny, it is in this context, I shouldn't say if a is skinny, in this context a is skinny, and a dagger, or a or pseudo-inverse, is gonna be fat here, and it's a left inverse. By the way, I could have said this early on. One of the statements on this linear algebra review was that, if a is skinny and full rank, then it has a left inverse. And I could just have said, here it is, a transpose a inverse a transpose, and you could have just plugged in the formula. But, of course, you'd be scratching your head thinking where on earth did that come from. So that's why I deferred it. So you now know, here's a very explicit way to construct a left inverse of a matrix, it's just the pseudo-inverse. It is not the only left inverse, we'll find others, and, in fact, we'll find all left inverses soon. Now, if you want to look at projection on range of a, if you multiply a by xls, by definition it's the point in the range of a that's closest to y, and it's the projection of y onto the range of a, that's denoted this way, and that's ax, ls. And the projection function is actually linear, and it's given by this matrix here, it just multiplies by that matrix. And that is an m by y m matrix; that's what it does. It's very interesting matrix. By the way, it's pretty, it's got a nice a symmetry, you've got some a's and some a transpose appearing, and an a transpose a inverse, and all that sort of stuff. You're actually gonna see a lot of formulas like this; you're gonna have to get used to checking these very carefully to make sure that they're, first, past syntax, and then whether they're up to the semantics as well. And if you look at it, it has the look of a matrix that sort of would want to be the identity. It is not the identity, unless, by the way, a is squared. In which case, when all the smoke clears, this thing just goes away and it's just the identity. But it's not the identity, it's projection, like that. By the way, this already has lots of uses. It basically says, you can actually imagine something like this, you can say you receive a signal with noise, a vector signal, a bunch of, say, multiple antennae communication system, but you happen to know that the signal should lie in a certain subspace to the range of a. You can simply then project onto the range of a, and what you've got now is the closest signal to the one you received, which is sort of consistent, that's in this known range, and this is used.

Student:

[Inaudible]. I don't get why a transpose a is full rank.

Instructor (Stephen Boyd):Oh, why is a transpose a. So that goes back to my slide, which said that if a is full rank and skinny then a transpose a is invertible. So I'm referring back there. I'm not showing it. Actually, you could now, with the qr factorization, we could actually establish that connection.

Student: Excuse me, question.

Instructor (Stephen Boyd):So this is a matrix you will also see quiet often. It's a, a transpose a inverse a transpose, it's actually a projection onto the range of a, that's what this matrix is.

Student: Excuse me.

Instructor (Stephen Boyd):Let's look at a couple of things here. There's a so-called Orthogonality Principle, and that says this: That the residual –

Student: Question. Hello.

Instructor (Stephen Boyd):Yes.

Student: All these thing that you said is –

Instructor (Stephen Boyd):This is so cool.

Student: Sorry?

Instructor (Stephen Boyd):This stuff hasn't been used since 1969. Oh, my God.

Student: No, actually, the next class here is using it so often.

Instructor (Stephen Boyd):Really?

Student: Yeah.

Instructor (Stephen Boyd):You mean more recently than 1969?

Student: Yes.

Instructor (Stephen Boyd):Cool.

Student: Much more recently, like last Tuesday.

Instructor (Stephen Boyd):Really?

Student: Yeah, really.

Instructor (Stephen Boyd):That's so cool.

Student: Yes, it is.

Instructor (Stephen Boyd):Okay. I didn't know where it was coming from. Anyway. You have a question I bet.

Student: Yeah, I still do. So all these things that we are saying is about a skinny full rank matrix, right?

Instructor (Stephen Boyd):Right.

Student: But what if we have a, like, sensory and it's not full rank? That's really bugging me. If we have a matrix and it's not full rank what should we do?

Instructor (Stephen Boyd):It should bug you. We're gonna get to it. So our discussion right now, here's our discussion, is skinny full-rank matrices. Why? Because I said so, for now. I promise you we're gonna talk first from here, will talk within next week, we'll be talking about fat full-rank matrices, then a week or two later we will talk about what happens when you have non full-rank matrices. But I promise you we're gonna get back to this. By the way, just to warn you, a lot of times when I say that we're gonna get back to it later, of course we don't. But in this case we really are.

Student:I'll remember that.

Instructor (Stephen Boyd):Okay. Good, you should. And it's a good question. And, by the way, the answer's it's a bit complicated, and that's why we're doing the simple thing first.

So the Orthogonality Principle says that the optimal residual, that's ax, ls - y is this thing, it's a transpose a inverse a transpose - i times y, is actually orthogonal to the range of a. It's right here. That if this is the range of a, and here is ax, ls, and here's y, the residual, like this, is orthogonal to this. Now, to really make this more interesting, you might imagine at least r3, and in r3 the range of a you could make a plane. And then you have a point, and then you have ax, ls; ax, ls is the projection of that point on the plane, and if you go from that point, the point on the plane, up to the original point, you're gonna get something that's normal to that plane. So that's the picture. And it's quick arithmetic to just calculate r transpose az, and you'll see that it all just works out here. I'll let you do it because this is just arithmetic to work out.

Now, by the way, there's also a way to understand this. In this case, suppose someone said, "Well, that usually happens. Sure that happens in the university, but when you get out in real life, that's not right, that doesn't really happen." And you can have things, "Oh, we found things where the angle can be 92 degrees." Something like this. There's actually a pretty simple way to know that this can't happen. And I can turn this into a formal mathematical argument, but I don't want to. If someone produced a point that was alleged to be the projection of y on a subspace, and this residual weren't normal to the subspace, how would you prove him or her wrong quickly? I just want a quick geometric argument, which, by the way, would easily be turned into a mathematical argument. What would you do?

Student:

[Inaudible.]

Instructor (Stephen Boyd):I can see your fingers are – I'm going to interpret your hand motions. What you do is you take this point and simply move it in the direction where your residual's leaning. If you move a small distance in that direction your angle goes up and your distance to that point goes down, and you've just proved the other person a liar. Because you've just moved closer, and you're still in the range. The original point was alleged to be the closest point and you just moved closer than the point they had, so it's impossible.

This is a very good [inaudible]; for example, Fourier series and probably other things as well. Okay. Let's make a connection with QR factorization. If a, is again, full rank and skinny, we can factor it as a=QR. Now, in this case, here R is square because it's full rank and skinny. So we write it, a, which looks like this, as Q, and then R, where this upper triangular, like so. So that's the picture. And the columns of Q are orthonormal. Well, the pseudo-inverse is just this, it's a transpose a inverse, and I'll just plug in for a, QR. So this becomes a transpose is of course R transpose Q transpose, that's here, QR, and now you have to be very, very careful with these things. So when the columns of Q are orthonormal, then Q transpose Q becomes i. So this just goes away then, and you have R transpose R inverse. Now, R is upper triangular and invertible, because its elements on the diagonal are positive, so it's invertible. In fact, its determinant is the product of the elements on the diagonal, which are positive, so it's invertible. So R transpose R is therefore invertible. In fact, the determinant of R transpose R is the determinant of R squared, because the determinant of R transpose is the determinant of R. So this is invertible, and of course it becomes R inverse R - t, that's slang for R inverse transpose, which is the same as R transpose inverse, these two operators commute, times R transpose Q transpose, that goes away, and in the end you get R inverse, Q transpose. So a very simple formula in terms of the QR factorization for the least squares approximate solution of y=ax. And it's kind of weird. And let me give you kind of a complete hand waving, and utterly wrong argument, but kind of just as a rough idea, it works okay. You say, well, you have a=QR. If a were square, and Q were square, and R were square, then you would have a inverse is of course R inverse, Q inverse. Ah, but Q is, in this case, orthogonal, so I can replace Q inverse with Q transpose. This is if a is square. But now I look at this for a minute, and I say, "Hey, wait a minute, what if a is not square?" Of course a doesn't have an inverse, so I want some kind of approximate inverse, and I just use the exact same formula, which is R inverse, Q transpose. So roughly speaking, the transpose of a matrix with orthogonal columns is something like an inverse. I can tell you're not buying it. But anyway. Well, it was just I just threw that out there in case that made any sense to you. But I guess it didn't. Now, projection on the range of a, that's a, a transpose a inverse, but we just worked out what all this stuff is, this is just that. So if I multiply a, which is QR, the R's go away and you've got Q, Q transpose, and now you know what Q, Q transpose is, it's actually projection on the range of a. Now you know what it is. So let me remind you, that last lecture we talked about Q skinny matrix, it actually has to be, because I'm gonna assert that the columns are orthonormal. So the columns are orthonormal, that's Q. And in this case, that is exactly the same as saying Q, Q transpose = i. And let's make it M high, and My, it doesn't matter, something like that. And I mentioned at the time that Q, Q transpose is not i. And I said I'd tell you later what it is. And this is one of those times where I wasn't lying then, because I really am telling you what it is later, which is now. Q, Q transpose is not the identity, it's actually the projection, it projects onto the range of these columns, that's what it is. And it's sort of like an inverse. Projection sort of gets you as close as you can in the subspace. By the way, if you're in the subspace it's the same as the identity. So now you know what happens when you take, for example, three orthonormal columns in R10, and form, call that Q, form Q, Q transpose. You get a 10 by 10 matrix, and what it is, is it has rank three, and what it does is it projects onto the range of those three columns. That's what it does, it takes an argument x, and it returns a y, which is the closest point in the range of those three columns, that's what it is. I should mention one thing here, least squares is not – actually, you can, and I would even encourage you to do that, in math lab, although, I mean, this is not supposed to be a class about math lab, but I'll say just a little bit about that. You are more than welcome to type something like this, in fact, I would encourage you do this, this times a prime times, let's say, y. You're more than welcome to type that in, and you will get something up to numerical precision, which is basically a transpose a inverse a transpose y. Now, in fact, generally, this is not how this is calculated. It's actually calculated using the QR factorization. And I'm not talking about math lab, I'm talking about how people do numerical calculations, which is not what this class is about, but I thought I'd mention it. So you have to distinguish between the formula and then actually how it's calculated. In fact, it's computed, typically, using the QR factorization. So what would really happen is you would actually do this, you'd form Q transpose and then R inverse here. R inverse because it's upper triangular and it's very easy to carry out. I should mention one thing. This is so common, that in math lab this is just this, it's backslash. And that, I believe, actually, is a math works innovation. It might be just about the only one, but there it is, as far as I know. They were the one to popularize it basically to say, a transpose a inverse a transpose is so important it should have one ASCII character assigned to it, that's how important it is. And I'm pretty sure that was, in fact, these guys. You could write this, but I guess that would be at least two if you type it in in text. So that's it. Now, I also have to warn you about this, this does other stuff. So you asked what happens when a is fat, when it's skinny, when it's skinny but not full ranked; a backslash y will do something. So be very careful, that's a symbol that is overloaded, and it does not just work out this. It will not warn you or you'll get very strange warnings, and things like that. And it could very happily return an answer, and it won't be what you think it is. So I just wanted to make mention of this. There's some notes on the website about this and you have to read those. So let's look at full QR factorization. So here, you simply extend out the columns, Q, you extend out, you complete an orthonormal basis, and you append on the bottom of R a bunch of zeros. So this now has the same size as a, and this matrix Q1, Q2, although it looks fat, because I've separated the columns into two parts, is actually square, this is actually a square matrix, and it's orthogonal; it's columns are orthonormal. Now, if I take ax - y norm squared, this is a, that's the full QR factorization x - y, and what I'm gonna do is take this expression here, that's a vector in RM, and I'm gonna multiply in the left by the M by M matrix Q1, Q2, that's orthogonal. Now, when you multiply by an orthogonal matrix, you do not change the norm. So I can shove a Q1, Q2 transpose on the left here, I can shove that into both sides, it doesn't affect the norm. So over here I get this. This is the identity, because Q1, Q2 is orthogonal, so you get this. So this is the identity, and I get R1x - Q1 transpose y, and the second block entry is zero x, it's minus Q2 transpose y. So this thing is equal to that norm squared of that vector. But the norm squared of a vector, if you've blocked it out, is the sum of the norm squares of the blocks. That's what it means. So this is equal to this. By the way, we haven't solved the problem yet, this is simply a formula for norm ax - y norm squared. That's what it is. And so the question is, if you look at this, now we can ask the question, how should you choose x to make this as small as possible? Well, if you look at the right-hand side, it has nothing to do with x; there's nothing you can do about it. So the right-hand side is irrelevant if you're minimizing. Now, you want to choose x to minimize this. Well, let's see, how small could this be? Well, it's a norm squared, it can't be any smaller than zero, but it can be made zero, and it can be made zero by taking x=R1 inverse Q1 transpose y; that's our same old formula before. So that's the picture. And the residual, with the optimal x, is Q2, Q2 transpose, which you can work out directly from this because that's the residual. So what that means is that Q1, Q1 transpose gives you projection on the range of a, and Q2, Q2 gives you projection of the orthogonal compliment of the range of a. This stuff is all very abstract for now, but when we put in the context of actual applications, I think it'll become clear, and possibly even interesting or something like that. So that's the connection to full QR factorization. Now, we can talk about applications. So, basically, we already had the conversation about y=ax. Frankly, it's a short conversation, we already had it, and it's good you have it. So the short conversation goes something like this: If you have inversion problem, you have y=ax, then if the columns of a are independent, which is to say that a is full rank, I'm assuming it's skinny here, then it says, yes, in that case you can reconstruct x exactly from y. What would do it? Well, a left inverse would do it, but you already know a left inverse now, which is, in fact, this a dagger, or a transpose a inverse a transpose. So you know a left inverse, that's this. Now, generally speaking, there's gonna be some, in all cases where you actually make measurements, there's gonna be some small noise. Now, v might be very, very small, v could, in fact, be the error in simply calculating a times x with double precision numbers. So v could be as small as round off error in a numerical calculation. But generally speaking, this actually refers to some inversion estimation or reconstruction problem, it involves received signals or things like that, and v is simply – well, if these a sensors, these are sensor noises, errors, things like that, they could be errors because a sensor is only 14 bits. All that is lumped into this y. So the real picture is y=ax + v. Now, you want to reconstruct x given y. If there were no v, you can reconstruct it exactly if a is full rank and skinny. But, in fact, now there's an error, and, in fact, now I'll tell you the sad news, you cannot reconstruct x exactly if there's noise. That's kind of obvious, right? So you have to make some prior assumptions about what v is, and then actually you're not gonna simply get an x that – well, you don't know what v is. If you knew what v is you could subtract it from y, and you're back in the other case again. So if you don't know what v is, then you have no choice but to do something like best effort here. And the best effort is gonna be, well, we'll see, is least squares estimation. So least squares estimation says this: You choose x hat, that's your guess as to what x is, by choosing x hat to minimize norm of ax - y. And ax has a specific meaning. It is what you would observe if x were really x hat and there was no noise; y is what you really observed. So you're basically getting the best match between what you would observe if the input were x hat and what you did observe. So that's the picture. So you minimize the mismatch between what you actually observed and what you predicted – oh, by the way, in a lot of fields a times x is sometimes referred to as the forward model, that's a good term. So it's the forward model, and basically, it's something that maps x into ideally what sensors you would see if the sensors were flawless and perfect. That's y=ax. Some people call that the forward model. By the way, it's a big deal. Somebody could work a couple months to come up with some code that computes the forward model. It simply maps x's into what you would observe. And it's a big task, depending on the particular problem. So it would be highly non-trivial, for example, to work out the forward model if the problem is a mechanical structure, and the x's are loads that you put on it, and the y is a vector of 1,000 displacements. It would involve all sorts of finding an element, modeling, you'd have to read in some standard description of the structure and all this sort of stuff. But for us it's just ax, and that's called a forward model. So what this is doing is, it says you get something which actually doesn't match anything in the forward model. And what you're doing is you're minimizing the mismatch between something that the forward model can give you and what you actually observed. So that's the picture. Now, what is it? It's nothing more than this. It says, you take a transpose a inverse a transpose, and that gives you your least squares estimate. Very simple. You'll see in a minute this works extremely well, does very, very clever stuff, all rather automatically. So let me mention here, the BLUE property. You also see it with BLU. It's the Best Linear Unbiased Estimator. So let me explain that. B are measurements of noise, so y=ax + b; a is full rank and skinny. And let's suppose you're gonna look at a linear estimator. A linear estimator takes y, multiplies it by, in this case, a fat matrix B, and returns x hat. That's a linear estimator. Well, you just multiply this out, this is x hat=B times ax + v here. And it's called unbiased if, without noise, it's infallible, it always reconstructs x. So an estimator that works perfectly in the absence of noise is called unbiased. Now, what that means is this: If v is zero, it says x hat is Bax, and the only way for x hat to equal x always is for Ba to be i. Well, that means it's a left inverse. So, by the way, when someone says left inverse to me, I think only of an unbiased estimator, that's what it is, it is nothing more. They are identical. If you see Ba=i, in that case a has to be skinny, if you see Ba=i, it means if a is sort of a measurement system, B is a perfect equalizer or a perfect reconstructor, that's what it is, or an unbiased estimator. Now, if you have an unbiased estimator then your error, that's x - x hat, that's the error in your estimate, is x - Bax + B. But Ba is i, so this x - x goes away and it's just this. So what happens, if you have an unbiased linear estimator, is that your error is now completely independent of what x is, and it depends only on B. And, in fact, it's minus B multiplied by this error. Now, what you want now is the following: Now you can see that not all left inverses are the same. Right. Because, in fact, when one person has one left inverse, and another has another, they both give you unbiased estimators. And, by the way, if there's no noise they will both reconstruct x perfectly. Now, you introduce noise. When you put in noise, the two estimators differ if the two left inverses are different. And they differ because the estimation error is actually that matrix multiplied by v. So now you can see what you'd like, you really want a small left inverse. Now, obviously, B can't be [inaudible] zero because then you could not possibly have Ba=i. So now you can see, if you want a small left inverse, and by the way, in this context small means it's error is not sensitive to the noise. Well, it turns out our friend, a transpose a inverse a transpose is the smallest left inverse of a, in the following sense. If you take simply the entries of a matrix, and think of them as a vector, then its norm squared would be sort of the sum of the squares of the entries. And so it says that our friend, the pseudo-inverse, is the smallest in the sense of sum of squares left inverse of a. And this is not too hard to show, but I'm not gonna do it. And so people say, then, that least squares provides the best linear unbiased estimator. I should mention one other thing. So far everything we're gonna talk about, when we talk about estimation we don't do it in a statistical framework because, for some reason, long ago, it was determined that probability was not a prerequisite for this class. I'm not sure why, but that's the way it is, and it seems okay. I should mention that, of course, estimation, obviously, should be looked at from a statistical point of view. In which case, you know, after weeks and weeks of discussion of probability and so on, in statistics you'll come up with things like least squares. Someone will say least squares, and you'll say, "What's that? Oh, I'm really doing maximum likelihood estimation with a Gaussian noise model," for example, that's what least squares would be. But that's always the things are, something that's a good idea can usually be very well justified by several completely different methods. So you'll also see this in statistics, but with completely different discussion all around it. And it's not bad to know that ideas like least squares, although they are deeply statistical, you can also understand them completely, and be absolutely effective in using them without knowing statistics. That was not a recommendation to not learn statistics, by the way. So let's look at a simple example, and let's just see what this does. So this is navigation from range measurements and it's distant beacon, so we're gonna use the linearized measurements. Now, what that means is, each measurement is an inner product along this normal vector, pointing from the unknown position to the beacon. So this guy simply measures not circles, but in fact planes, like this, lines, I guess in R2, that's what it measures, you get this distance. Now, the linearization is quiet good. And so what you're gonna get is this, you're gonna get four measurements of x, linear measurements. So that's a 4 by 2 matrix, and there'll be a noise here. Now, I chose a specific noise here, and the noise I chose from a random distribution. Again, this is totally irrelevant, but let's just give a rough idea of what it is. The noises are independent Gaussian; they have a standard deviation of 2. This is totally unimportant. What it means is that these v's, the entries, are off typically by ± 2. They can easily be off by ± 4, but it's very unlikely they're off by ± 6, and essentially impossible that they're off by ± 8 or something like that. Just to give you a rough idea, that's noise. And if you like to think of that, what it really means is that each of these measurements is something like a ± 4 meter accuracy measurement; if you want to get a rough idea of what these are. So what you'd say here is that what you're gonna get is you're gonna get four ranges, measurements of the range, but each range has an error of about ± 3, ± 4 meters, just roughly, that's the idea. So, for example, if the real range is 10,000-blah, blah, blah, you could easily have 10,000 + 2 or 10,000 - 3. Given those four noise corrupted measurements, what you want to do is estimate the position. Now, here you've got four measurements, you have two unknowns to do. So the quick arithmetic tells you right away, just a quick accounting, says you've got about a 2 to 1 measurement to parameter redundancy ratio. That's good. It's better than the other way around, which is a 1 to 2. So you have 2 to 1 measurement redundancy, just roughly. Now, the actual position happens to be this, we just generated it randomly, there it is. And the measurement you get is this. Now, by the way, there is no x that actually will give you, if you plug in two numbers here, multiply it by these four vectors, you will not get these numbers. These are inconsistent measurements. Why? Because there's noise. In fact, we can probably get a rough idea of how inconsistent they are. They're probably inconsistent in each entry by about two or three or something like that because that's how the noise was generated. Now, we're gonna do two things. The first one is this, we're gonna take someone who took a linear algebra class. They didn't talk about non square matrices, they only got to inverses and things like that, but they didn't talk about left inverses or least squares. And so they say, "Look, I don't know what you're talking about here, I never learned about tall matrices. We did square matrices. And we learned that if you have two things you want to find, two measurements does the trick, and you get a 2 by 2 matrix, and I even know the formula for the 2 by 2 inverse or something like that." So one method to do here is to simply ignore the third and fourth range estimates. So this is the just-enough-measurements method. You've got two things to measure, you take two measurements. And without noise, that certainly is gonna do the trick, assuming your two measurements are independent. Well, what that corresponds to is taking the top half of this matrix and inverting it, that's how you get x. And that would actually exactly reconstruct x if there's no noise. So, in fact, this method of sort of inverting the top half, and padding it out with zeros, this creates a left inverse you can check. Well, I didn't give you a. But had I given you a, you could multiply this thing by a and you would get the identity, it's a left inverse. And a left inverse means this, it will reconstruct perfectly if there's no noise. That's what this does. By the way, when you see a matrix like that, let's call that B just enough measurements, I should maybe call that B just enough measurements, like that. If we write this out, if you see this 2 by 4 matrix, and I say this matrix maps range measurements into position estimates, when you see this, and you see these zeros, you should immediately come up with an explanation. And in this case, what is the meaning of these four zeros?

Student:

[Inaudible.]

Instructor (Stephen Boyd):It means the last two measurements are irrelevant. This zero is sort of just an accident, and it just means to the precision I'm printing these, which is not exactly high precision, it says that, basically, our x1 hat doesn't depend on y1. Did I say that correctly? I think I did. And, by the way, you can probably look back at the geometry and figure out why that's the case.

So here the norm of the error is 3.07. We should give the rms error, so you should divide by square root 2. So 3 divided by 1.4, I don't know, what's that? So you get an error of about 2 in each – I'm getting the rms error here. So you get an error of about 2. And, by the way, if someone says, "Hey, you're off by two." They'd say, "Pardon me, your range measurements are off by about two." Does that sound reasonable? That would be the response if you use this method. They're off by about two in both x and y, but the measurements were off by about two, so who can blame the person.

Let's look at the least squares method. When you work out a transpose a inverse a transpose, you get this matrix here. And you see a lot of things about it. Now, let's talk about some of the things you see about it. Oh, by the way, the estimate it produces a 4.95 and 10.26, the norm of the error is .72, you divide by 1.4, and you have a norm of error, which is about .5. So the first thing you know, is at least for this one random sample I generated, this thing did a lot better, it did a lot better. So that's the first thing you notice.

Now, this is another left inverse. And, in fact, it's interesting to compare the two left inverses. Let's compare these two. So let's just look at these two left inverses. This one we already said this estimator doesn't use the second two measurements, it doesn't use the third and fourth range measurement. What you see here is if all these numbers are kind of the same – they're all there, they're smaller than these, you don't get numbers as high one and minus one, but you're using everything. By the way, there's a name for this, it's called sensor blending, that's what's actually happening, or fusion, that's a beautiful name. People talk about sensor fusion. And so if someone said what have you done here, and you don't want to say, "Oh, it's just least squares." If you want to impress the person, you say, "I've implemented sophisticated sensor fusion algorithm." So that would be the correct way to say this. And they'd say, "What does that mean?" And you'd go, "Well, as you can see, I'm blending all four measurements to get each estimate in just the right proportions." And you'd say, "Well, how did you get these?" And you'd go, "It was tough. It was really tough." But the idea remains, this is one of the ideas you see in – you laugh, but there's whole fields where people don't even have a clue about these ideas, where you get a lot of data and you want to make some estimate of something, and there's totally straightforward ways to do it, like this, but that's simple. There are three ASCII characters. This is like a backslash y or something like that. But the point is, it's actually doing something quiet sophisticated, it's kind of blending all these things the right way. So that's the first thing you say. I want to ask you a question, though. Obviously, for this one instance, this estimate was a whole lot better than this one. But I have a question for you, could you please tell me a situation in which this estimator will outperform this one. And, in fact, let's make it simple. I want it to outperform it by a lot. Having just made fun of this estimator, I would like you to explain a situation in which this estimator outperforms this one. What is it?

Student: Do you just randomly [inaudible] noise [inaudible] sensor and [inaudible]?

Instructor (Stephen Boyd):There you go, perfect description, I'll repeat it. It goes like this. Suppose, I mean, the one thing you can say about this one compared to this one is it doesn't use the last two measurements here. So if those last two measurements were way off, in fact, suppose that you were being spoofed, for example, or suppose the range sensors got totally whacky, dropped a bit somewhere and they were way off, way off. This one is completely insensitive to the errors in the last two measurements. This one, however, if you're way off, and v3 or v4 is really big, you're in big trouble because of this. So it's not that simple to say that this is always a better estimator than this one. And, indeed, there are cases where it might not be. But the point is, generally speaking, if what you can say about the v's is that they're all small or all about on the same order of magnitude, this is very likely to be a lot better than that. So that's the picture. Now, we can go back and we can actually do our example from the first lecture, which is this: You had an input that was piecewise constant for a period of a second or a nanosecond or microsecond, if you like to make a different time scale, but it doesn't matter. It goes into a filter, and that simply means that this signal here is a convolution of that input here, with a convolution, kernel or impulse response. Just sample that at 10 Hz, that means you sample it every 100 milliseconds, you get this. And, then, now comes the part that's tricky, we actually subject that to three-bit quantization. So the quantizer characteristic is this, and this basically selects the nearest of eight levels spread over the interval minus 1, 1. So that's what this formula does. And so here's the input signal that we don't know. It might have been, for example, a coded communication signal being sent to us. Here's the step response. This convolved with the derivative of this, that's the impulse response give you this. And you can see that it's kind of smooth, but it kind of tracks it. Here you can see sort of where this dips down, you see a dip here. It's delayed on the order of one second. And you can see sort of where this goes up, there's a bump. It smoothed out, though, like that, so it's delayed a half a second or something like that, and smooth. And then this is what it looks like when you severely quantize it, three bits. And the idea is, given this, estimate that. Now, by the way, looking at it by your eye, I think the answer would be forget it, it's not gonna work. It's already hard enough here, because this thing is this thing kind of all smeared around. In other words, this point here is produced not just by what's in this interval, but what's here, here, here, and so on. So you just say forget it, it's all mixed together, can't do it. And then when you quantize it, it gets even sillier. It seems sillier. So we'll just write it as y=ax + v, where a is an R 100 by 10, and aij is this, and you can just work out the details. This is how you map x into y. Then we'll take v here, in our model y=ax + v, is gonna be the quantization error. It's the difference between y tildy and the quantized version. And, by the way, it's actually very complicated, it's a deterministic function of y tildy, but I can tell you one thing about it, all of its entries are between ± .125. Because the width of an lsv here is .25, a least significant bit. And so the error's no more than .125. And the least squares estimate is this. And you can see that it pretty much nails it, or at least it gets very close. In fact, oddly, it gets twice as accurate as the quality of your a to d. I tried to shock you with this the first day, and I remember you were like, that's cool. So you should be impressed by it. So that's the idea. By the way, this example is a perfect example of how all of this actually is used and how it goes down in practice. No one will bump into you later tonight and say, "Quick, I need help." "What is it?" "I've got this matrix a, and I've got this vector y, and I need to find x so that norm ax minus y is minimized, can you help me?" No one is gonna do that. What will happen is, you will go back to some lab or something like that or go off at some company, and someone will give you, except it won't be clean like I made it, they'll throw some huge thing down, and if it's communications there'll be acronyms flying all over the place, and they'll say, "Yeah, were using four quam, and this," and it'll go on, and on, and on. "Oh, and you should use this such and such model, and there's this." And there'll be some finite elements, and PDE's thrown in. Horrible thing. Nowhere will they say least squares, they will certainly not say linear algebra, they probably don't even know linear algebra, they won't say it. They'll have all sorts of things, they'll talk about who knows what. There'll be spherical harmonics, and cosigns, and signs, and different coordinate systems. And your job will be to just sit there, very calmly, read the pile of stuff, go read a couple of books on the field, and sit there, and realize I will write all of this as y=ax + v. And, then, you will go to your computer and you will type a backslash y, very calmly. So the last part will be very fast. But the first part is all the work. So I'm just saying this example is a good one, this is exactly how it happens. There's block diagrams, and all sorts of transfer functions, and all sorts of nonsense all around. And it's your job to get rid of all the field specific details and abstract it away in something that fits this way. So that's it. Actually, I want to show you one thing here. In this case, I just selected a couple of rows of a transpose a inverse a transpose. Now, what this does is this thing maps y, that is this smeared quantized received signal. This maps the smeared, quantized received signal into our estimate for these ten real numbers, the ten real numbers in the original signal, x1 through x10. That is a 10 by 100 matrix. It maps quantized, smeared received signals into estimate of x1 through x 10. I have plotted here Row 2, Row 5 and Row 8; these have a meaning. Row 2 of that matrix actually shows you how those 100 received signals get multiplied to create your estimate of x2. So when you take an inner product of your 100 received signals, with this vector, that's forming this unusually good estimate of what x2 is. Look at it and you'll see some really cool stuff. If you look at this, this says that to get the estimate of x2 here – in fact, let me ask you. It says that the quantized signal from six seconds to ten is pretty much not used. Why am I saying that? Because these entries of b are small. Do you see what I'm saying? That makes perfect sense, if you think about it. Because you know that the smearing is kind of local in time. It says that to get a good estimate of x2, it says the most important thing is actually y right at time 2. By the way, x2, for the record, is actually not there, x2, this value, sort of the middle, is 1.5 seconds. So this thing was smart enough to realize, very roughly, that you should sample one-half second late. Why? Because that's roughly what that smearing did, it added a one-half second delay. So that's what this shows. This little negative bit here, that's cool, that's actually what, in communications, is called equalization. And it basically says, you should sort of subtract from what you see here, this thing, because this thing was kind of corrupting your measurement. And so if you subtract for it you're actually kind of compensating for it. This is equalization. By the way, this is the third part of how this goes down in practice. We already talked about the first part. The first part is decoding all these books and the silly field specific notation, which finishes when you put it in the form something like y=ax + v. The second part is a backslash y, that goes right away. Now, the third part is to go back and explain your estimate to people who have never heard of pseudo-inverses. If you say it's just least squares, and they're, ""Okay, that's fine." But sometimes they'll just look at you and they'll say, "What's that?" It just means you have to explain it. And then you'd plot these pictures, and you'd give this story until they go away. And you'd say, "This is how I estimate the second one. This is the how I estimate Row –" truly, what's really estimating, of course, is a transpose a inverse a transpose. But at least this make the story clear. By they way, you could even do things like now truncate this, and they'll have sort of a finite equalizer if they were uncomfortable about your estimating using stuff from the future and stuff like that. So that's the idea. Any questions about this? And you're still not impressed with this, I can tell. We'll look at some more applications. And the first one is data fitting. It's very famous. You're given some functions on a set s, s could be anything, it actually could be finite, it could be r, it could be an interval, it could multi-dimensional, like it could be r3 into r, and these are called regressors or basis functions. It's called basis functions even if they're not independent, but we'll leave that alone. So you have a bunch of functions here from s into r. And you have some measurements. Now, a measurement looks like this, by the way, some people call them samples. So the other one is you have measurements or samples. And a sample is a pair si and gi. So that's a sample. And so, for example, let's make this specific, let's let s be r2, let's make it simple. In fact, let's make s zero 1 cross zero 1, that's the unit square, it's something that looks like that. And so one of these functions, a basis function here, is a function from this unit square to r. And so, for example, I'll tell you f1 is just the constant 1, f2 is a, let's put a physical interpretation here, that's the temperature at a position on a square plate, that's what it is, that's the meaning of this. So each of these basis functions is now a temperature distribution. So the first one is just flat, uniform distribution 1. The next one is like, let's say, an x gradient. That means it's something where as you move across here, the temperature grows linearly; f3 is gonna be a horizontal gradient; f4 is some weird thing, it depends on the field. In fact, these things come from the field. So you go to the person, let's say it's Bob, who's been doing this for a long time. And Bob says, "No, you should use the spherical la Gare harmonics of the 12th order. We've been doing this for years, trust me." You're like, "Okay, whatever." So that's usually where these come from or they might come from some other place. For example, you might say, "Oh, no, no, it's heat equation, that's Poisson equation – "you know what, these should be the Igen function of the heat kernel, and so these should be the cosigns and signs, and this is actually a Fourier series. So it just depends. So you're given measurements. That means that you sample the temperature at some of these points. And what you want to do is you're gonna form a temperature distribution, which is a linear combination of these temperatures in such a way that it matches approximately your measurements. Now, here we'll talk about the other regime, but in this case we're gonna assume you have a lot of measurements and not that many bases. So I have six bases, in fact that's kind of the point of having a well chosen basis, is that if you have a well chosen basis, you can actually describe all the temperature distributions that you're likely to actually see in practice quite well as a linear combination of let's say eight basis elements. Now, you sample the measurement a 1,000 places. Well, you've only got six numbers to choose, and you have a 1,000 measurements. So a least squares fit does this, it says let's choose these coefficients, the x1 through xn. By the way, in this context these are often called, not that it makes any difference, something like a1 through an, because – well, I'm using x because it's gonna end up as x when we write norm y – ax, but, in fact, normally it would be called alpha. By the way, to make it really complicated for you, these things are actually very often indexed not simply from 1 to n, but by multiple complex indices. So, for example, if you're doing image processing, these might be the such and such wavelets or the edgelet, these might be the edgelet series, and they might have like three indices. This is just to mess you up. But we just enumerate them as 1 through n. And your job is to choose those coefficients like this to make this thing small, to get a good fit. Once again, I'm not gonna do it, but you look at this, and you just take a deep breath. And, remember, they won't do this, they'll index these horribly. They'll give you 40 papers on edgelets and the wavelet transforms. They'll say, "Oh, you don't know about wavelets? Wow, where were you?" And they'll throw a bunch of papers at you and walk away. And so these x's will be called alphas, and they'll have, let's say, have two indices in the superscript and two as subscripts, and it'll be hideous. You're job is to just convert this very calmly to this, to just ax - g. And aij here is nothing but fj of si. So that's it. And you just work that out because you just realize that this is axi, and that means that that's actually giving the ith row, and so, in fact, it's nothing but this. So it's extremely straightforward, it's just this thing. So the least squares fit here is a transpose a inverse a transpose g, you just work it out. And the corresponding fit is this. So this would give you your fit. And this has a lot of uses, a lot of uses already. And let me just say what some of them are. But mostly what it is is it's a very good way to develop a simple approximate model of data. And let me make up some examples and you'll get the idea. When you have a simple model, you can do things like interpolation. You can say I'm measuring the temperature on this core at 75 places, but I'm really interested in what the temperature is, very likely at these 22 other places, I couldn't get a sensor in there. What is it? You study the thermodynamics, and somebody tells you, "'Well, actually, here's a good basis, the prolate spherical something harmonics." And you go, "Whatever." So you get 30 of these or 20 of these or whatever the appropriate number is, and you now fit the data you have to those. Now, you have a model, that's your least squares fit. You can now make a guess as to what the temperature would be anywhere on that plate. Are you right? Who knows. It depends on whether, in fact, the real temperature distribution is in the range of the basis. It probably isn't. So as to whether you get a good estimate, a good interpolation or extrapolation, for that matter, here it depends on the quality of the basis chosen. But the point is, it give you a very good method to do it. By the way, these methods will work very well if they're done responsibly. You can also do smoothing of data. So that's one of the other things that happens, if if the data has lots of noise, but you have a lot of it, and you fit it, you're actually gonna get a nice smooth thing that will actually – so you can use it as a method to de-noise measurements that way. You can even go back, and, actually, if you think of, let's say, gi as an extremely noisy measurement of what some underlying f of si should be, you can actually first do the fit, then come back and look at fls fit of si. And you say, you know what this is? I believe that's really what this is, this is just the horrible noisy measurement. And now you've got a beautiful way to de-noise the data. I'm just making up applications because there's lots of them. I'll give you one more. So here's a real application. Actually, all of the ones I talked about were real, but here's one. You're developing an arrow model of some vehicle. Finally, you do all the computational fluid dynamics, all this stuff, finally you take the thing down to NASA, or NASA Aims, and you stick it in a wind tunnel. This is very, very expensive. You fire the whole thing up, and you take a bunch of data, and you tilt it up, and you do all sorts of things, and you change the speed of the thing, and you take thousands of measurements of the moment and the force put on that structure, on that vehicle, as a function of, for example, it's angle of attack, it's various other angles, so you've got three or four dimensions there, and the speed of the fluid it's moving in. Everybody see what I'm saying? It's unbelievably expensive. You take a million measurements because you just take it continuously, and so on, that's what you do. Now, you want to come back and you want to just make a simple simulator for this vehicle. Well, you need, at some point – basically, it looks like this, it looks like x [inaudible] f of x, there's some other stuff in there, but this is this function you just took samples of. So what you do now is this, you don't have f, what you have is one million samples of f from your wind tunnel tests. Here's what you do, you do have a very good idea of what f should look because otherwise you shouldn't have put the thing in the wind tunnel, so you clearly had some idea, and everyone knows roughly what this looks like. I mean, the first derivatives, you know what those are, those have well known names and arrow or the stability derivatives or whatever they call it, you know roughly what those are. So, actually, you can figure out a very good basis for these. You know, for example, about the phenomenon of stall, you know that if the angle of attack gets too high you're lift is gonna precipitously drop. So you know all these things because you've done those. So you can come up with a very appropriate basis. You now take that one million data points, and you fit an approximately f. By the way, that wouldn't take so long, if you're curious how long it takes to do a least squares with a million variables, it doesn't take that long. But in any case, let's say it takes 10 seconds or it doesn't really matter what it takes. You do that, you know what you have now, you now have a very small piece of code, like one screen of c or something like that, that will evaluate an approximation of f in well under a microsecond. So you just took 10 million pieces of data, and you scrunched it down, you fit some data, and you now have an approximation, it's an approximation of f but a good one, and it's just 50 lines of c. You can evaluate it way under a microsecond. You know what that means? It means you can simulate this thing like crazy now. Because in a simulator you're gonna call that every time step, and you're gonna call it tens of thousands of times. I get the feeling that no one has any idea what I'm talking about. Did any of that make any sense? A few people. So that's the basic idea. And we'll look at a quick example, and then we'll quit for today. So least squares polynomial fitting, this is very famous. You want to fit a polynomial of degree less than n. So your coefficients are a zero up to an - 1. The basis functions are, of course, the powers of t. By the way, that's a very poor choice of basis function, unless the data happens to be between, for example, remarkably the t's happen to be between minus 1 and 1. Even then it's a poor choice. And this matrix a has this form, which is a Vandermonde matrix. So the rows are increasing powers. And I won't go through this, but we'll just look at a quick example. Here's a function, which is by no means a polynomial, I just made it up, it's some rational function. And we take a 100 points and we'll do least squares fit for degrees one, two, three, and four, and we'll just take a look at what happens. Now, a degree 1 polynomial, that's a line. This thing has almost zero slope, but not quite, and this is the fit to this function of degree 1. And you can see it's trading off the underestimate here with the overestimate here. Because it's a line, it can't do very much. Here's the quadratic fit. And you can see it got a bunch of the curvature here a bit, the dashed one is the curvature, so it kind of got some curvature. Here's the cubic fit. And notice you're already getting a pretty good estimate here. The cubic allowed you to put this little tail over here, and here's your quartic. And these are all just done least squares, something like that. So we'll quit here.

[End of Audio]

Duration: 78 minutes

IntroToLinearDynamicalSystems-Lecture07

Instructor (Stephen Boyd):[Inaudible], if you go down to the pad, I wanna make a couple of announcements.

So the thing I wanted to announce – you're gonna have to pan up here a bit here – is that we have a couple of sets of notes on the course Web site. These are, you know, either two pages or barely three or something like that. One of them is a set of notes on least squares, or least-norm solutions. We haven't covered least-norm problems yet, so you can ignore that part.

But you must read these notes. They're very short and connect to these we've seen in the class, and is actually gonna be very important for you to do the problems and things like that. This is one of the sets of notes.

The other is, of note, that collects in one case. It's [inaudible] unambiguous and clear, unlike any [inaudible], which actually explains how you solve [inaudible] linear equations.

So the question there is given a and b, how do you find out [inaudible] x such that x equals b? How do you find – if there is such an x, how do you find one? And this will explain it. And it's connected to all the stuff we've been doing, so. So now that's a – that's official announced. A few parts here we haven't covered yet, but we will in the next week, or even we'll hit it today. I'm not sure.

So let's continue with least squares function fitting. And the specific case is polynomial fitting. This is very famous and something you should know about.

So here's the problem. I have a polynomial degree less than n – looks like this – that's characterized by n coefficients – a0 up to an minus 1. So these are these – these are essentially the variables for us. And we wanna fit this to data. So data are a little – are samples, basically, of this – of some function, or they're just data. So ti and yi. So they're little pairs which consist of a t value and a y value.

And we want to fit – wanna choose a0 through an minus 1 so that when we evaluate the ti's at these – when we evaluate this polynomial at the ti's, we get something close to the yi's.

So the obvious basis functions here – by the way, if you really wanna do polynomial fitting somewhere, these are among – this is about the poorest basis you could choose. But that's another story.

So the obvious base function are simply the powers. So the first function is simply the constant 1. The next is t, then t squared, and t cubed, and so on.

Here, the matrix a is gonna have this form. It's a very famous matrix. It's called a Vandermonde matrix. And it looks like this. So each row is actually a set of ascending powers of a number. So this is t1 to the 0, t1, t1 squared, and so on.

By the way, you should never – you should not look at this matrix and just say, "Yeah, so what?" It should be completely obvious what this matrix does. This matrix, when you multiply by a0, a1, an minus 1 – so this matrix maps [inaudible] polynomial coefficients into [inaudible] here, you get p of t1 [inaudible] to p of tm.

So if you had to have an English name for this matrix, or you wanted to tag it with a comment or a description, an extremely good description of this would be something like a polynomial evaluator matrix because that's exactly what this does. It maps coefficients into a vector of the polynomial evaluated a number of points.

So it's actually known as a Vandermonde matrix, I suppose, because Mr. Vandermonde had something to do with this or something like that.

So that's – and you have – you're almost done now because it's essentially a least-squares problem if your goal is to minimize the sum of the squares of the differences between the measured data or the data here – doesn't have to be measured – and the model data. So that's the – you get a least-squares problem.

Actually, this matrix here, we can deal with that right away. It turns out it is a full rank, provided, actually, that these points, the evaluator points are different. So as long as the evaluated points are different and the matrix is as skinny, this is full rank.

And to check this – [inaudible] it's a good exercise just to see how that worked. Assume that these tks are different from each other – the sample points. And let's suppose it's skinny. If you have a times a equals 0, that's very interesting. What it means is that this polynomial here vanishes at n points t1 through tm.

But if you have a polynomial of degree n minus 1, and it vanishes – in fact, at n points or more – distinct points, I should say – vanishes at n distinct points, then that polynomial is 0, period. So [inaudible] if you have a cubic or something like this, and it vanishes at four points, it is 0, period. It's identically 0.

But to say that p is identically 0 says that a is 0. And that's just a restatement of the fact that the matrix a, here, has 0 null space. In other words, you're saying that is the columns are independent, or a is full rank.

So that's just a quick thing to check. But let's look and see how polynomial fitting works and looks. It's a [inaudible] bullets meant to have any actual, particular use, but here it is. I have a function, which, by the way, is a perfectly simple function to evaluate, so [inaudible] coming up with [inaudible] polynomial model for it. But it doesn't matter.

So here's the function. It's 4t over 1 plus 10t squared. [Inaudible] that, most assuredly not a polynomial. And we're gonna fit it with a polynomial just to see how this works. We'll take 100 points between t equals 0 and t equals 1. And we're gonna fit this thing with a polynomial. And the RMS errors for the bits of degree of 1, 2, 3, and 4 drop as .135 and .076, .025, and then .005. And here's a picture showing this.

So the first – this is the first 1-degree polynomial is the dash line here. Actually, it looks like it's almost the same as a 0-degree polynomial, which is a constant. But in fact, it has a very slight slope. You can see that the dashed function starts off below .5 over here, and it's slightly above over here. So it's got a very small coefficient of t in it.

This is the quadratic fit, and you can see that it sort of picked up this kind of dominant curvature over here. The cubic allows you to sort of get this little – this skewness, actually, quite literally. It allows you to get the skewness here. And the [inaudible], as you can see, is now at least visually fitting extremely well.

There's a question?

Student: What's like the best way to deal with a polynomial [inaudible] same [inaudible] 0?

Instructor (Stephen Boyd):That's an excellent question. And that's exactly what I was gonna talk about next, which is not in the notes. I was thinking your question while I was talking. So let's talk about it.

So the question was, is this a good way to get a polynomial fit to a function? And the most obvious other time you've heard about polynomial fits is, of course, calculus. So that is, in fact, what calculus is, although it generally stops somewhere around the second-order fit. So here's my short description of calculus. It says, take a function, and it says, develop a first- and second-order polynomial fit of that function near a point t0. Everybody know what I'm talking about? So that's what it does.

How does it measure the fit? It actually doesn't measure it by a sum of squares of errors or anything like that. It measures it by how close it is as you get closer and closer to the point. This is not making any sense, so I'm gonna draw some pictures.

So here's my function, like this. Here's my point. And the first order of the series will give you this function here. And if you were to look at here, you'd see you're pretty close here.

The second-order fit is gonna be – is – get something like this. It's gonna be f of [inaudible] plus m of t bar times t minus t bar. That's your first order Taylor expansion. [Inaudible] have f prime, prime of t bar times t minus t bar squared, like that. And this is t bar here. So that's your [inaudible]-order fit.

Now, one is gonna capture some of the curvature, like this. And the second-order fit nearby is gonna be not just very good – sorry, I should say it's gonna be very, very good. That means that's an order, then it'll fit very well.

So if you wanna – and this is, by the way, not the same you'll get from least squares. It is absolutely not the same. [Inaudible] if you take a little, tiny interval and chop it up into – and sample it at a bunch of point [inaudible], you should get really close to the first and second derivative. But they won't be the first and second derivative. They absolutely will not be.

The reason is this: In calculus, the fit is local. That's what it is. You're finding a second-order – you're finding a quadratic function that fits the function extremely well locally. How local is local? Well, that depends. I mean, you have to ask – you have to – it depends on the function.

In contrast, something like this [inaudible] get an excellent fit over a big – that's a macroscopic interval here. Oh, by the way, for this function, we can look at the Taylor series. What would the – what's the first order of Taylor series look like? Well, it's not – I mean, it looks like this. You draw a line – that there. That's the – there's the first order of Taylor series. And you can see right away, these are very, very different.

Now, this first order of Taylor series, though, will have an – a very, very low error if you're near 0. I could do this in another place, too. I could try the first order of Taylor series here and get something else.

The second order of Taylor series is gonna look like this. And then oh, let's – I would say there's not much of a curvature there, so it's gonna kinda – it's like this, and it's gonna do [inaudible] that.

It's [inaudible] – what it – what will [inaudible] this second order of Taylor expansion here is that near 0, it will be very, very, very good.

So we come back to your question, which is better? And the answer is, it depends on what you wanna do? Are you looking for a [inaudible] fit over an interval? You shouldn't be using Taylor series.

Are you looking for something that gives you the best fit when you zoom in arbitrarily close? You should probably be using Taylor series in calculus.

So actually, what I'm telling you now has some – has serious implication. The typical mathematical indoctrination now just drills into people derivatives, calculus, and so on. And so any time the – you're sort of conditioned – we’re all conditioned – that includes me – whenever somebody says something like, "Give me a linear or an afine or a quadratic model of a function or a point," I just – it's in our – well, we've been totally indoctrinated. We just say, "Okay, it's the Taylor series or something like that."

And that's used in a lot of cases. A lot of – in a lot of applications now, people are getting away from the Taylor series, and they're actually looking at things like least-squares fits. In fact, some of the best estimation methods for nonlinear – this is, by the way, just for fun, now. This is not covered in the – this is not real material in the class.

Some of the absolute best estimation methods for nonlinear systems are now based on, literally, at each step, spring a bunch of points into – take a bunch of random points, plugging them into some horrible nonlinear function, taking those points, and doing a least-squares fit, not using calculus.

So – and those are – and those methods work unbelievably well. They are so much better than the ones based on calculus, it's not even funny, in practice.

Interested in these applications and finding a function that is close to the function you have when you're very, very close. You have a pretty good idea of the interval over which you wanna fit. And so these things work very well.

They're called particle filters. That's – if you care, – but I think that was a very long answer to your question. But thanks for bringing it up.

Couple of other topics we're gonna look at. One is this – I give growing sets of regressors. Actually, this comes up in [inaudible] problems. It's very interesting.

So it's this: What's gonna happen is, I'm gonna give you a list of vectors. These can be the columns of a matrix. And this is very important. They are ordered. So I give you a 1, a 2, a 3 in – they're ordered.

And this says, you have a family of least-squares problem, which fit a vector y as a linear combination – as a mixture of – well, first, just – and I'll vary p. So it's a sequence of questions. I say, here's my set of – fact, to this, we just did it with polynomials. [Inaudible]

[Inaudible] squared [inaudible], one of these 2, 3, 4, 5, and so on. Oh, and I don't mean 5, any 5, I mean the first 5. So let me say it right. I'll rewind and say it again.

You're given a sequence of vectors, and you're asked, what's the best fit with the first one, the first and second, the first, second, and third, first, second, third, and fourth, and so on? So it's in a specific order.

Now, it actually, in this context, these are called regressors. So in statistics, in a lot of other areas, you'll hear these vectors referred to as regressors. We might call them columns. And actually, in fact, what you're doing is you're solving a bunch of least-squares problems where you're actually taking a leading columns of the matrix.

So if we were to write this as ax minus – or a with the p up here, what a is, is it is [inaudible]. Well, I might have some master a here. That's my [inaudible] p, p [inaudible] of a. And that's what we're solving. That's the idea.

Geometric idea is basically, you're projecting y, a given vector, onto the span of a growing set of vectors. That's the idea. And I guess the verb you'll sometimes hear is to regress y on a1 through ap. That's a verbal – that's the syntax used in statistics. You would say that you would – you – we would say for doing least squares or something like that, or you're calculating a projection, or you could say lots of things. They would say something like you're regressing y on these regressors.

So obviously here, as p increases, you get a better fit. You can't fit – you can't find the best least-squares fit with seven columns and have it come out worse than the best least-squares fit with the first six columns because I mean, if – I mean, you could, by the way, have them the same. And that would only be the case in the optimal fit, with seven columns. That mixture calls for 0 usage of the seventh column. That can actually happen, of course.

But the point of this is, you get a better fit. So the optimal residual decreases. Now, of course, for us, this is quite straightforward. For us, you just do this. In fact, it's even more embarrassing than that. The way you might do this is you might do this. I'll just write the code for you right here if you don't mind. So right, that's how you would write it, say, in MATLAB systems as well. But it's pretty straightforward. But it's just [inaudible].

So you take – this says to take all rows, and take the first p columns. And then do a least-squares fit. So this would give you this.

And the formula's just this. Now, what I'm about to say was highly relevant in 1967. It's less relevant today. And let me actually have – I need to have a little digression, so little digression here about how all this stuff – where this all fits in, in terms of least squares, about actually doing it.

If you need to do least squares, let's say with 100 – let's take 100 variables – and let's say, I don't know, 1,000 equations. So that's the size [inaudible], and I'm gonna do least squares. How long does that take to do nowadays? I just wanna order of magnitudes. Are we talking minutes, hours, seconds? What are we talking?

Student: Seconds.

Instructor (Stephen Boyd):Seconds. Anybody who knows about scientific computing wanna refine that number?

Student: Milliseconds.

Instructor (Stephen Boyd):Thank you. I heard it. Milliseconds. That's measured in milliseconds, maybe a couple tens, maybe more. Do not be confused by the interpreted overhead in MATLAB. [Inaudible] longer [inaudible] for it to figure out, you just type [inaudible] in, and call it the right allay pack function, in most cases, then to solve it.

So let me repeat. I should've actually worked out the exact number and just tested it on my laptop. You can do this.

Student:[Inaudible]

Instructor (Stephen Boyd):What is it?

Student: Twenty-one milliseconds.

Instructor (Stephen Boyd):Twenty-one milliseconds. That was Jacob. Now, again, you're the class who couldn't care less when I had a 3 [inaudible] a to d. See, I'm still pissed off about that this – for the record. I had a 3 [inaudible] a to d, and subsequently was making predictions of the input to [inaudible] six bits. You're like, "That's cool. That's fine."

So this probably means nothing to you. Anyway, these are things that actually – well, you know, you can just back – you can just propagate Morse law backwards and figure out what it would've taken, what can your [inaudible] to do this.

So the point is, is it's all just a nonissue. So if you needed to do this for p equals 1 to 30, the whole thing would be over and done with in a loop [inaudible] for a second. Boom, half second, it's all done. This is not the case 30 years ago or something like that.

So by the way, let me say what I think the implication of this is. The fact that you can do this – by the way, this is absolutely an enormous amount of signal and data processing. Think about what this is. This is – you can match 1,000-long vector with a linear combination of 100 things in 21 milliseconds. That's on Jacob's laptop.

So this is absolutely amazing, and it's something you should fully appreciate. It means that a lot of the stuff we're looking at now actually can penetrate into areas where people have absolutely no idea can penetrate into. So I wanna make that point that a lot of people who think of least squares – somebody who's older, took – statistics, "Oh, we use least squares all the time. Oh, yeah, we used to regress like ten things on 15 measurements," or something idiotic like that.

This is a totally different era now, totally different era. You can solve – and that is actually a small problem – but think about what this would involve. But just think of the data – the data here is 100,000 – well, a 101,000 real numbers.

That's not a small amount of stuff. So I just wanted to press upon you, you now live in an era where the stuff we're talking about, it's not just that you can do it. You can do this unbelievably fast. And my opinion is that very few people in the world actually appreciate the significance of this. That means this could be done – there's lots of places where you could be doing all the methods in this class. They could – they're now down to things like frame rate for video. They're gonna hit – in many cases, if they're much smaller, they're already at audio rate. And a lot of people don't appreciate. Of course, some do, but sorry.

That was just an aside, but I wanted to say it because it's actually very important to say how all these things fit into the big picture. Now, having said that, there's actually something very cool, which, as I said, is no longer relevant, in my opinion, but is much relevant in the '60s when this would be a really big deal. So if you get a paper from 1960 or something like that, this would be a really, really big problem, about as big as things would – this would take a long, long time, so.

So it turns out that if you form the qr factorization of a, then it turns out whether you like it or not, you have actually, for free, calculated – you've actually calculated the qr factorization of all the leading columns of a, so these a's of ps, like this.

So – and what that means is absolutely amazing. It says that if you do least squares using qr, it says that you can actually calculate the least-square solution. You don't have to have a loop that says, for p equals 1 to n, carry this out. And in fact, you can do it much, much faster. In our case, as you can see by the numbers involved here, it really hardly matters.

By the way, it would matter if you're doing some kind of [inaudible] thing, doing this in microseconds or something, or you're solving your problem with 10 million variables. Then things start mattering. But [inaudible] – what this says is you get – then this used to be a big deal. It's not a big – it's not – I think it's not a big deal now, so.

So right now, they – really no shame in your writing code that looks like this, as long as you put a comment there to preserve – just as a safety factor. You put a comment that says yeah, yeah, I know this can be done much faster, but I'm lazy. Then put your initials after the comment. [Inaudible] then you're totally protected.

Now, much more important is actually to understand what you can do when you do least squares with varying numbers of residuals. Let's take a look at it. Well, the most obvious thing to do is this is you take a plot, and you plot the number of regressors you take. That's the number of columns of a. And then what you do is you simply plot the norm of the residual. By the way, 0 has meaning, and it says, please fit my vector y with a linear – with a optimal linear combination of no columns. Well, you're at – your fit is gonna be there is no fit. And so your error is simply the norm.

So this, by the way, sort of sets everything here. So right at p equals 0, you get the norm of y. Then this first point here gives you exactly the optimal fit, or if you like, the geometric distance. This is the geometric distance from the point y to the line spanned by a1. That's what this is. And it drops here.

Hey, by the way, could that point be – could this point be here? No, not if your least-squares software is working. Could it be here? And when would it be there?

Student:[Inaudible]

Instructor (Stephen Boyd):x when the optimal x1 is 0, which would occur when, geometrically? It would occur when –

Student:[Inaudible]

Instructor (Stephen Boyd):– y and a1 are orthogonal. I'm getting a weird – did I say that right? Is it right? I'm getting some weird looks. I'm gonna blame you if I said something really stupid, and you failed to correct me. I think it's right. So that – it could be here.

So this drop, by the way, in statistics, they have a beautiful name for this. You'd say – you would say something like, "In this case, it expresses a percentage." Say something like this. If this dropped 15 percent, you'd say that a1 explains 15 percent of y. That's what you'd say. Not an impressive number. That's not something you'd brag about, probably.

This number is the distance from y to the span – to the plane spanned by a1 and a2. And you can see it dropped a healthy amount. And then this is – and of course, this has to go down, and so on. And that's it. So you get pictures like this. These pictures are extremely important. In many applications, you need to look at these because usually, this thing has something to do with the complexity of your model. And so you're gonna look at figures like – pictures like this.

Certainly, you would not want to fit a model with something more complicated than it needs to be. So we'll look at this in a very, very practical context, which is least-square system identification. This comes up in tons of fields. This is in economics. It's in controls. It's in signal processing. It is in finance. It goes on and – I mean, the applications of this just go on and on and on.

So actually, I once taught an entire class on, well, that, of which two-thirds of the class was based on that, so entire classes on this.

So here it is. Let's just say I have a scale or input and a scale or output. And I have an unknown system here – a dynamic system. And the system identification problem is the following: I give you a record – some observe data – input and output. So I give you – some people call that IO data – input/output data. And what I wanna do is I wanna fit a dynamic system model to this. So that's the picture.

Well, here's an example with a scale of u and y because vector is easy – vector case is easy to do. So one very famous model is called a moving average model. I think you may have seen this on the first homework or something like that. And it basically says this. It says, I will – my model is – and the model is with a hat on top – the model here is that the output is a linear combination of the current input – the input lagged one time instance and the input lagged up to n time instance.

So here, you have a set of coefficients in the model. That's h0 through hn. There's n plus one of them. And they're real, in this case, because you just scale it. So that's a moving average model. And it's a – the h's parameterize the model, they give you the coefficients in this moving average. It's a – I mean, if you wanna be fancy here, you could say it's a moving weighted average. But what everyone says is moving average.

Now, how do you set this up for some kind of – and your job is gonna be to pick h0, h1, and hn. Of course, you know by now that the – what you have to do is you have to wrestle with the matrices in the equations to make it look something like this, where the things you want to estimate or choose appear here. There's a matrix of known stuff, and then there's a result over here. And so I've done that for you. I've written it this way, and it looks like that.

By the way, this is kind of bizarre because normally, when you think of u as an input to a system, usually we think of inputs as appearing here. And you can write this equation a totally different way with the u's over here and the h – so lots of ways to write it. For what we're doing now, you write it this way.

Now, the model prediction error is this. If I commit [inaudible] – we have a coefficient y, y hat here is actually what I predict the output is gonna be. y is what I actually observed it to be. So the error is called – is – that's the model prediction error. It's just the difference like this.

And in least-squares identification, you choose the model, that is, the parameters that minimize the norm of the model prediction error. And the answer is the way to get these h's is this thing\that. That's how it's done.

So I won't even go into how that's done. You should know how that's done.

Here's an [inaudible]. Here's an input trace, like this, and here's an output trace, like that. And you wanna fit a model, some kind of moving average model to this data. So there it is. And [inaudible] kind of look at this, and you can say a couple things. You can say, "Well, it looks like the output is sort of a smooth version of the input."

Oh, by the way, one of the things you might also wanna do here is to – one possibility is in fact that these are completely unrelated.

By the way, how do you imagine that would come out, I mean, if you have enough data? How would that come out? What do you think?

Student:[Inaudible]

Instructor (Stephen Boyd):What's that?

Student:[Inaudible] model [inaudible]?

Instructor (Stephen Boyd):No, no, no, no, no, no, not – it's not the [inaudible]. Let's – no, that could be a problem. But I'm talking about even a more fundamental problem. The more fundamental problem is that this is the price of some asset here. And this is the temperature in Palo Alto. Now, by the way, they could well be related. But let's just imagine that, in fact, they are not. And I gave you the data. And I said, come on, you took 263. Let's – how – what would happen? What would suggest to you that in fact these are unrelated? What result? What would happen when you started fitting models to it?

Student: Equal weights.

Student:[Inaudible]

Instructor (Stephen Boyd):You can get equal weights? I sort of agree with that, but can you be more specific about the weights you might get, the h's?

Student:[Inaudible]

Instructor (Stephen Boyd):0. Basically, the optimal h's would kind of be extremely small. And your optimal error would be terrible. You would get a huge pile of data, go to this thing, find tiny coefficients, and say, "Well, I'm done. I just explained 1 percent of the data," meaning that basically, your guess is just 0 each time, or something like that.

Everybody see what I'm saying here? So that would be how you do it.

Now you had a – there's another problem. And that is that your data is not rich enough. You could throw in a few more regressors, and then you could actually get a better fit.

Okay, so [inaudible] how this works. So here's our example. And but when we work this out, we calculate the eight coefficients, and we get something like this. And a relative prediction error is .37. Actually, that sounds like that's not that good. It says, basically, that when each time – before a new sample comes, I can predict it through within plus/minus 37 percent. I mean, roughly, that's what this was saying. That doesn't sound like it's gonna be a good model. It's actually a pretty good fit. I'll show you in a second here. And then I'm gonna come back to the coefficient.

So here's what a 37 percent fit looks like. Here, what – the black is the actual output. And the y hat is predicted from the model. I think it's kinda fuzz your eyes a little. If you squint, it's really not bad. It's kinda getting the gross characteristics of the movements. There are places where the error's large.

Actually, in places like this, because the error is [inaudible] – the errors are actually large here. And they're large here for a while and so on and so forth. So that's what sort of explains this.

That – there's your picture. By the way, it's actually not too bad. I mean, if – for some applications, if you wanted to get a rough idea, this would – this is clearly giving it to you.

I have a question for you. Here, this last coefficient is pretty big. Actually, it's the largest coefficient. That should make you uncomfortable because it basically says that to calculate the current output, this model, with a lag up to 7, is heavily relying on what the input was seven samples ago.

It really strongly suggests you should increase 7 to 10 or 15. And let's see what happens if you get bigger. And that's exactly this business of growing regressors for this thing.

So the question here, then, is something like this: When you do a moving average fitting, how large should the model be? Now, the larger you make n, obviously, the smaller the predict here on the data used to form the model. That has to be because you just – your – you have a richer set of regressors. You're projecting on a bigger subspace. The distance can only go down. The fit can only get better. That's clear.

And if you – if what you wanted to do was actually just fit the data for the model, then you'd use the largest model you can possibly handle. I mean, there'd be no reason because the error just keeps going down.

And for the particular data example I just showed you, here's what happens. So this is it. This shows you sort of the fractional error, as you increase the order of the model. I think we were here, maybe. Is that right, 5, 6, 7. I think we were here.

Very interesting plot, very good plot. You should produce it, essentially, always whenever you do things like this. And there's even a concept of variable number of regressors. You should do this plot.

So there it is. By the way, this is carefully constructed to show you what it should look like in the cleanest, simplest, and best case. Now, this thing goes down all the time. We've already discussed that. But here, your eyes should be drawn to something, which is kind of obvious. That's why we made the data this way. And that is this guy right here.

And when you look at this, that's the traditional knee of the curve. And it basically says, somewhere at up till order 10, the models predict more and more. And above 10, well, technically, they predict more and more. And this is on the data used to model. But they start at – and what they start adding is very, very small. So there's some tradeoff here.

Oh, by the way, I should say that in statistics, there are very complicated – and signal processing and economics – very complex theories on how to choose the model order, very complicated. But basically, it's very important to understand it from a simple point of view, which is the one I'm talking about here now first. Later, you can go off and find out about the [inaudible] information criterion and blah, blah, blah. That's fine.

But for now, the first thing you do is just get it – how this works in the simplest case. So this would say something like n equals 10 is already coming out as a possibly good choice.

Now, the problem with this is the following: If in fact all you want to do in your model is reproduce the data you've already seen, then no one can argue against this. It's got a better fit, period. But in fact, we are creating that model probably to use it on data you've never seen.

For example, tomorrow, you wanna make a prediction about tomorrow. Or you wanna make a prediction five nanoseconds in the future. That's what maybe – this is the kinda thing you wanna do. What this means is you shouldn't – I mean of course, this is important, but you really should be checking that model on other data not used to fit the model. And that's a very famous method. It's called cross-validation.

In statistics, but in lots of other things, you may wanna zoom out a little bit here because it seems like [inaudible], but – or not, I guess. So cross-validation's very simple, totally obvious idea. It says that when you're fitting a model, you should actually judge the victive performance, not on the data that you use to develop the model. That's kind of obvious. Actually, if you're not doing well predicting on the data used to form the model, you don't need to cross-validate because your model is terrible. So you don't need to do – you don't need to go to this step.

So this is sort of if you're getting good fits on the data you've seen, and I mean that in like the c sense, meaning that if that's not the case [inaudible] what comes next in the – in that.

So here's a model validation set. Oh, the first one could've been one week's of data. This is another week's data. And here's u and y bar. That's some other recent data. And then what you do is you take the data that you – that the coefficients you got from the other data set, and you apply it to this one. And you make a plot like this. And it's quite pretty, quite sensible, and you see something here. This shows you the error on the modeling data. This has to go down unless you made a mistake. Has to go down.

This one here does not have to go down. Obviously, it doesn't. And this is really interesting. You see this? This says that when you use a fifth-order model to make a prediction – to actually now try it on next week's data or something like that. Or I guess it wouldn't be next week's data, or you couldn't validate it. So some other historical week of data, you actually do worse than if you use the fourth-order model here. So this can go down.

And actually, now you – this is made – this is a thing that should be burned into your brain. This is sort of the beautiful – this is what you're hoping for is something like it because now it's totally obvious. It's right in your face that probably the best model here has order 10. So this is a fantasy. This is what it looks like. You might ask, "Will it ever look like this when you do it?" The answer's, "No, it'll never look like this."

Student:[Inaudible]

Instructor (Stephen Boyd):Well –

Student:[Inaudible]

Instructor (Stephen Boyd):Why here? Because in fact, the data here really is – the data here was generated [inaudible]. By the way, it's [inaudible] with order of 10. That would be unfair. So it's well fit by order of 10. So – but I'm not sure I understand what you're asking. Are you asking what – why does – why did it work here?

Student: Yes.

Instructor (Stephen Boyd):It worked because the –

Student:[Inaudible]

Instructor (Stephen Boyd):Oh, no, it could've – oh, no, absolutely. You – I mean, you could argue that 9 is fine here. So by no means – in this case, I mean, you could even argue that 3 or something is a good order if your noise level is high enough. And if the level of prediction you want is just what I showed on the graphs over there, that would be fine. So I just wanna show you the idea here.

Now, of course in practice, it never looks like this, although it can sometimes look like this. This is the fantasy.

Student:[Inaudible]

Instructor (Stephen Boyd):Oh, [inaudible]? They're the same size. They're the same size.

Student:[Inaudible]

Instructor (Stephen Boyd):Sorry?

Student: If I have a function [inaudible], [inaudible]?

Instructor (Stephen Boyd):Right. So there's lots of ways how you do cross-validation to [inaudible]. Then you get – it's a huge field, I might add. And there's lots of ways, so in fancy studies where things really matter, like medical studies, they'll actually do things where you're not – I mean, they will insure – you will not ever see the validation data if you're the one doing the modeling. You're not allowed to.

There's a nice – there's a protocol. You develop your coefficients. You send them to someone else who basically had signed something that he will not share the data with you, and you check it. [Inaudible] he'll check it on the other side of this opaque wall. And you'll simply say, "Here's your error." So you'll never see it.

So in some cases, this stuff is taken very, very seriously, I mean, for obvious reasons. In other cases, you might do this. You might take your data, chop it in half.

Or some people do this. They take two-thirds of it or something and fit a model. And you can do insane things. Of course, if you have these two data sets, you can develop a model from this data set and try it on the other one. Then you can do weird things. You can chop them up into different pieces. And you can develop a model on all the chunks except one and check it on one. So cross-validation, then, gets very, very complicated and so on. And you can think of all sorts of things to do.

And then the theorists come along, and it gets even more complicated. Then they'll be the information criterion and blah, blah, blah, and the Kullback-Leibler divergence. And it'll go on and on and on.

So very big topic. This is a basic idea. This is – I mean, this is the fantasy. This is the pure thing. This is the way it should look. Of course, it'll only look like this in this class, but still, it's not that – it's not impossible that you will sometime do some fitting, and it will look like this. It's not impossible. Actually, it's likely in this class, but I mean, real – some real fitting.

Was there another question? You also asked about the size of the two sets, right? If you actually give the errors as RMS errors, then the size of the two sets now – at least the first order – does not scale your error. And you can actually talk about it sensibly. So here, they're equal, so norms are fine.

But if you – if you had – if your model data was 100 points, and your validation was 37 points, you would definitely need to look at something like an RMS error to be sure you're comparing the right things.

So – but I should add that when you do this thing – this phenomenon – and in fact, this example's a little bit off. I can tell you why in a minute. This phenomenon here where you get – as you get a higher and higher order model, and your modeling error residual goes down, but actually, your prediction ability's getting worse, the – this is called overfit. That's the name for this. It means you overfit your model. Another way to say it is you're fitting noise or something like, so that – and that's essentially what's going on.

By the way, sometimes it's not graceful like this. This can go like this. This goes – that would actually – in fact, I should probably redo the example to make the crime associated with overfit have a higher penalty and have this thing go like that.

So we'll look at that. We'll – [inaudible]. And let me explain what I mean by the [inaudible]. First of all, let me say what we just did. We just looked at the [inaudible] sort of x minus y, like the sum [inaudible] like this. So it looks like – oh, there we go. So I did this. And what we were doing is we were looking at controlled bits of a. So this comes up in things like that when you add one more [inaudible]. So that would be – that's where this is at in adding regressors. Come up in [inaudible] things and things like that.

Now, we can also look at another [inaudible], and that is where the a in the least-squares setup is actually growing not by adding columns but by adding rows. So that's the – that's instead of growing sets of regressors – growing sets of [inaudible] that actually comes up in a bunch of cases. So let's see what that is.

Now what we're gonna do is we're gonna write least squares in row form. In row form, you write norm ax – well, we write in norm – we write norm ax as a vector, which is whose ith entry is the interproduct of the ith row of a with x. That gives you this form here. Hopefully, it's identical to the other formula. It's just written it out to [inaudible] those [inaudible].

Now, the rows in a, if you're doing estimation, they correspond to measurements or data. That's what they correspond. [Inaudible] other applications. They have other meanings and things like that. But here, they generally correspond to measurements in a sensing situation. And so each row of a is a measurement of some parameter x that's unknown.

Now it's interesting because you start asking questions like this. You say, "Given a set of measurements," you say, "please guess what y is." By the way, this kind of has to start with a full rank and skinny.

And by the way, later, we're gonna ask questions, like if someone walks up to you and says, "Oh, I have one measurement. Please guess these ten numbers." For the moment, that's off – we can't do that.

So in this case, it's gonna work like this. We will only consider the case when you've already got enough measurements that that matrix is full rank. That means that if you're estimating n parameters, you have at least n. If you have n, it means they're also independent. So that's how that works.

So here, you imagine x is a vector to be estimated, and these are measurements. And the solution of this, that's a – this is assuming this thing is invertible. That's, of course, this is a transpose a here if you write it out row-wise. I wrote it out – the rows of a are the ai transpose. That's why it looks like ai transpose here.

So once a becomes – as you grow a row by row, once it becomes full rank, rank n, it will stay rank n, of course, as you keep increasing it. And that's when this becomes invertible. And this is the least-squares solution written out. This is a very interesting and very valuable way to think of the least-squares solution. Looks like that.

It's actually kinda cool, and you can ask all sorts of things here. By the way, the same story – well, you can actually do things like, say, well, let's look at cases where this would come up. This would come up this way. Suppose we're estimating a parameter x, but we make measurements once every second. Oh, let's suppose your measurements actually are GPS pseudo ranges or something like that. So you're getting those once a second. You're getting a GPS pseudo range. But x is constant. x is, well, in that case, it's three numbers. Actually, it's four because it also includes the absolute time.

So x is four numbers, which is your exact position and location. And every second, you get a new measurement. Now, the satellites in GPS have moved slightly, so actually, the rows of the a are not the same. They're actually slightly different – more than slightly different, actually. They're a bit different. And so what this would do – the reason this would come up is because at each – at every time you get new information, you would wanna have a new estimate, in fact, a sharpened estimate, an improved estimate based on the new data.

You can't use the other 50 rows that are coming because that's in the [inaudible], and [inaudible] to estimate your position now. So that's the idea here.

Now recursive least squares is something for that. It's a simple version of something called a Kalman filter, which is for another class. Actually, once you understand this, that's not particularly complicated. So here it is.

Instead of calculating this, and then we're gonna calculate this for each m. And it's a beautiful thing. This is basically saying, "Here's my estimate of the parameter." And you can actually plot this in time and things like that. It'll shift. You'll say, "I think I'm here." Another measurement comes in, and it'll more a little bit more and more and more. It'll move around. And it'll just get a better and better estimate. Of course, except in the most extreme cases, you know now this can be calculated with very, very fast.

But these things are coming at you, let's say at a megahertz, so new examples are if m measures microseconds, that's maybe a different story. Actually, that's not even fast enough, but if it's suitably fast, you'd have to do this – well, that one should be enough to make this worthwhile.

Okay, so the way you do this recursively – actually, let me just explain what the idea is. This least squares now, I'm gonna be very, very crude here. This least-squares estimate is [inaudible] that's a vector. But I'm just gonna be crude and say it's one sum divided by another.

You're not allowed to do this, by the way, what I'm doing right now. Or you can do it, but just don't ever write it down or have it recorded. I'm allowed to do it because I said explicitly – end nonsense. And then I'll put the end xml tag and [inaudible] when I finish this discussion.

But for the moment, this is, again, nonsense. This is, roughly speaking, a ratio of two sums. Of course, that's a matrix inverse, so it's hardly a ratio. But it's a ratio of two sums. And the sum [inaudible].

Now, if I walked up to you, and I said, "I need to calculate something that's a ratio of two sums," that’s – or if I have a sum, and I say, "Here's a new sum and," how do you calculate the new sum? You don't go back and re-add everything. You simply add the new thing to it. Everybody see what I'm saying?

So basically, you have something that looks like this. And when a new measurement becomes available, you take this thing, and you plus equals ym, am. That's what you do. You add – you accumulate it. That's a matrix.

So what? You plus equals, and a diad, which is am, am transpose. And what this means is your storage is now bounded. It said that as you do this, you don't need to store all of these things. You basically keep track of a running sum. Your storage is absolutely bounded.

So this is this thing. That's what I was just talking about here, is you simply add these up. And already, I mean, already have an algorithm, which [inaudible] interesting because it says, basically, you can do this – you can run this out, [inaudible] equals 100 million or something like that. It's just completely linear in m. The storage is absolutely bounded.

You have to store a symmetric matrix of size n by n. And you have to store an n vector. And that's it, period. And whenever anybody asks you, "Hey, what is [inaudible] so far?" simply calculate this. So that's how that works.

Now, calculating this, it depends on the timescale. It depends how frequently you calculate it. So you can actually [inaudible] match them, or you're simply accumulating these all the time. And then only occasionally do you get a query as to what the estimate is. When you get the query, you calculate this. Or you can imagine a system where you actually have to calculate xls of m for all m, for m equals 1, 2, 3, and so on.

But here it is, a famous formula. This one may be really is actually still useful – [inaudible] from the '60s, very useful. This one may actually still be – is still useful. And it's this. It's – and it's a good thing to know about. It's very famous. In this [inaudible], you have to calculate the inverse – actually, you don't calculate the inverse, but you have to solve an equation with a matrix, which is the previous matrix plus a rank 1 term. That's a diad. It's rank 1.

So by the way, there's a name for this. This is called a rank 1 update. If there's a minus sign here, it's actually called a rank 1 downdate. So – but that's – [inaudible] I – it kinda makes sense. So this is called a rank 1 update. If you go to Google and type "rank 1 update," you'll get tons and tons of things. Or you'll actually read papers, and people will say things like, "Here's our method, blah, blah, blah." And they'll say, "Well, of course. By exploiting rank 1 updates, we can do this, and blah, blah, blah."

If the numbers here are tiny, like 10 or 100, the whole thing is silly unless it's gonna be built-in hardware and run at a gigahertz. So there's no reason to have this. But if these numbers are getting into the thousands and so on, and it's gonna run at kilohertz or something like that, these things make a huge difference.

So that – this is a rank 1 update. Well, there's actually a fast way to calculate the inverse of a matrix if you've already calculated the inverse of this matrix, and then you do a rank 1 update.

So this is the so-called rank 1 update formula. Actually, it's got a zillion other names, but this is a good enough one. And it's – the formula's this. It's p plus a, a transpose inverse is p inverse minus, and then it's 1 over 1 plus a transpose, p inverse a times this thing.

Now, notice that if you've already calculated p inverse – in fact, you don't even really have to calculate p inverse. Look, suppose you calculated p inverse, a is [inaudible] to calculate. And that's an outer product here. You've already calculated p inverse a, so in fact, you only have to calculate p inverse a once. And then you use it here, here, and here with the transpose there.

So this is very, very straightforward. Notice that, by the way, the update to the inverse is a rank 1 downdate. That's a scaler. That's a vector times a vector transpose. That's a rank 1 matrix. And there's a minus there. So that's – this is what people will call a rank 1 downdate in the inverse.

By the way, it kinda makes sense when you add something to a matrix, and you invert it – invert sorta when things get bigger, the inverse makes the inverse get smaller. So the idea of the rank 1 update should lead to a rank downdate in the inverse, so it sorta makes sense. You can include that in the nonsense, by the way.

Oh, sorry, I forgot to say this. I'll declare it now. Oops. It's the end of nonsense. There, okay, so it's over now. What I was just saying was not nonsense. The only last part was vague.

So what this does is this gives you an [inaudible] of [inaudible] n squared method for computing the inverse of the second one of from the first. And If you did – [inaudible] in fact, n cubed. Of course, this is – doesn't really make any difference unless n is 10 or 100 or 10,000. Then it starts to make serious difference. Equals 1,000, it makes a big difference.

Now, you might ask, [inaudible] or whatever. I'll show you the final artifact. To verify – we just have to verify that if I multiply this matrix by this matrix, we get i. And the way you do that is this – that's just a sum of two things. You multiply that, and then you get four terms. So the first one is p times p inverse. That's i. Then you get p times this thing over here. That's this term. Then you get a [inaudible] times p inverse. That should be here somewhere. There it is. It's right there. And we should have one more term, which is a, a transpose times this stuff over here. And I guess that's down here.

And now various things start canceling. There's a p and a p inverse that go away. Then you look at this, and you realize this is a times p inverse a. Let's see what – down here, let's see what's gonna – what is gonna cancel here. Well, this one goes there. That one goes down there. Ah, these are both a times p. These are both a. I transposed that a, a transpose, p inverse. That's this thing. And it's the same thing you're gonna get over here. You look at those two terms, and they actually cancel like this. Actually, all three of these go together because that's – well, the coefficient is 1 minus 1 over 1. I'm not gonna do it. I just leave it there. It works. And you get i.

So that tells you that this matrix is the inverse of that matrix, and of course, vice versa. Now, you might add, "I wonder if – did anyone come up with that?" Well, it wasn't this way. It – I mean, it's not this way. It's the way these things usually work. You work it backwards, and then you reverse it when you explain it to someone, which makes you look smarter, I guess. So that's standard procedure.

So we'll – there's a question.

Student:[Inaudible]

Instructor (Stephen Boyd):I'm sorry, does what play a role?

Student:[Inaudible]

Instructor (Stephen Boyd):I'm sorry, there was a noise. I didn't get it.

Student:[Inaudible]

Instructor (Stephen Boyd):Yeah.

Student:[Inaudible]

Instructor (Stephen Boyd):You're right, all of these. Oh, I'm sorry, no, no. It's okay. Let's be very careful about it. Thank you. m is irrelevant. m is just – when you're calculating the invert p, m is t, if you wanna think of this as time. So this is fine. Oh, no, no, it's all right, you just confused me. Yeah, I'm okay. You're not.

All right, let's – we'll both take a deep breath. Now we're gonna go over it. n here is a parameter – is a problem size. It's the number of things you're estimating. It's the size of x. m is something like a time index in most applications. It's a time index. So m is irrelevant. It doesn't have a value. It's – m could be 30 million. It could be minus 50 plus 200. It makes no difference whatsoever.

The question is how much work you do. Of course, it's interesting to say that it has nothing to do with m. We knew that because this method here – each increments, the amount of work is the same. You increment p. You [inaudible] q. So m being 10 to the eighth doesn't make any difference here.

Does this make any sense? No, I can tell I didn't convince you. We'll –

Student:[Inaudible]

Instructor (Stephen Boyd):Let me –

Student:[Inaudible]

Instructor (Stephen Boyd):Oh, oh, oh, here. You want here.

Student: Yes, [inaudible] p [inaudible].

Instructor (Stephen Boyd):If p is what?

Student: If p is 12 [inaudible].

Instructor (Stephen Boyd):Right. There's a similar formula that is valid when p is not symmetric. You can work it out if you like. No, you can, really. You can guess it. Did you have a question about that, or –

Student: No.

Instructor (Stephen Boyd):Okay, just – okay, yeah. So this is the formula when p is symmetric. Now we're gonna [inaudible] to [inaudible] and things like that. And the first topic involves a multiple objective [inaudible]. And let me just put – let me just tell you where we are and where we're going.

So least-squares methods, they're widely, widely used. They don't always have the name "least squares." It depends on the field. They might use some other name for it. They're widely used. But it's actually a couple of tricks. You learn just a couple of tricks, and you will be in very – you will be very, very effective at using least squares. Actually, just the basic least squares is an awfully good start. But there's a couple more things you need to learn about.

One is called regularization. We're gonna get to that today. Another is when you – if there was a special case of multi-objective least squares. So we're gonna cover just a couple of things, and you will be in very, very good shape for actually using least-squares methods. So let's look at multi-objective least squares.

So in a lot of problems, you have two – we'll focus on two, but in fact, you can very generally have more objectives. So it might be something like this, and in the case of two, you want one objective, let's say ax minus y, small. Now, so far, the only thing we've looked at is how to choose x to make j1 as small as it can possibly be. That's the only thing we've been looking at, just that.

But here we have a second objective, which is at the same time, we want another least-squares cost, norm fx minus g small. So we have two. Now, if they're the same thing, if a is f, and g is y, these are the same thing. And it's silly, by making one small, you make the other small. And there's really no competition between them. I mean, there's no issue. But in general, they're competing. And that means that basically, you – to make this smaller, you're gonna have to make this bigger, and vice versa. That's what competing means.

Now, very common example is this, and it's really common, and it just looks like that. So basically, in this case, what you're saying is, "I want ax minus y to be small." But at the same time, I want you to do it with a small x.

So basically, you're saying, "I wanna do polynomial fitting. I want the fit to be good." That's what this is. But you know what? I don't want insane polynomial coefficients with coefficients like 10 to the 8 in the coefficient. In fact, I don't even want 100 in the coefficients [inaudible] polynomials.

Why? Well, because you're using this thing, for example, to interpolate or extrapolate or something like that, and the larger these coefficients, the more trouble you're possibly gonna get into when you use it for that. So that's the basic idea. And that's the most classic case is where you wanna trade off these two.

Well, conceptually, the right way to think about this is as follows. You think of the plot of achievable objective pairs. And so what this is – and by the way, x here is r100, say. So x is an r100. y is an r200, and g is an r300. It – none of this matters. I'm just saying these are big dimension things. What this is a picture of is the following: Basically, for every x and r100, you simply evaluate j1 and j2, and you get two numbers. And you plot them in r2. So it's not a plot – that's [inaudible]. That is not x1. It corresponds to x1. So in fact, to really write that label correctly, I should write it this way.

That's j2 of x1, j1 of x1. That's really what that blob point is. This is – this just means that's the point that corresponds to x1, and that's x2. So that's the picture. Now, the picture looks like this. And actually, it's a real interesting picture, and it's one [inaudible] in a lot of context, it won't – the stuff I'm about to say is kinda obvious. Everybody knows it, but it's actually very worthwhile to have seen it once or twice. It works like this. You have two objectives, j1 and j2. And now we can actually talk about various things here. Look, in fact, let's talk [inaudible] all now. So I'll clean it up.

[Inaudible]. You might ask, "Why is j1 vertical and j2 horizontal?" And the answer is, I don't know because [inaudible] for some reason. So let's look at some points here. So suppose somebody proposes an x and gets this point. And someone else proposes another point, gets that. So let's call this x1, and this is x2. What can you say about x2 compared to x1 in the context of this problem? x2 is better, period, even though – and why is it? Because it beats – x2 beats x1 in both objectives. So if one is – if j1 is fit, and j2 is fit, then x2 beats x1 unambiguously. It gets a better fit, and it's smaller. Everybody got it?

And in fact, if I draw this region down here, so anything down and to the left of where a point comes up here is actually the set of points that are better, unambiguously. And it doesn't matter how much you care about j1 relative to j2. Every point in here is better than that one because basically, you beat it on both. Or if you're up here, you meet it, and let's be more precise. You meet the same objectives as the other one in both, and you beat it in at least one. So that's better.

So all these points are better. How about these points up here? What can you say about [inaudible] compared to x1? [Inaudible], so up in this quadrant compared to a point is unambiguously worse. And over here is unambiguously better. And now what about this guy? How would you compare that point to x1? What would you say? Say it's better? No. Can you say it's worse? No. They're not – it's not comparable.

So what you would say about x4 is you'd say, "Well, it's better in j2, but it's worse in j1." And someone says, "Yeah, but is it better?" And you'd say, "I don't know. Depends on how much you care about j1 and j2." The point is, you can't say. And the same is true here.

So the first and third – is that what they call it? Are those the third quadrants. Yeah, so the first and third quadrants are the unambiguous [inaudible] the other period. The second and fourth quadrants on a plot like this tell you that's ambiguous.

So – all right, so let's go back to this thing. So these are all points. This point here [inaudible] r3. x3 is a choice, but look at this. If I look – let's look at the things that are better. Unambiguously, but in x3, I'm seeing there's a lot of other points. In particular, x2 beats x3 on most things, and therefore, x3 would be an exceedingly poor choice if you cared about j1 and j2.

The thing that kind of should become obvious here is what you really want is a point. Oh, by the way, there's a name for this in economics. They say that x2 dominates x3. So it's just better. And you'll find other words for it in other fields, too, but it dominates.

So what you want is a nondominated point. So you want a point that's achievable so you can say there – the fact that it is clear says that there's no x that achieves that pair of j1 and j2. However, if you look at x2, and you say, "What's better than that?" It's everything over here, but there is no – there are no points over there. x2 is not dominated. And there's a name – it's got all sorts of names. One is it's called a pareto opal [inaudible] – it's pareto optimal. That's [inaudible] years old. [Inaudible].

This curve here is the set of points that are pareto optimal. They're nondominated. These are the ones where if you were only told that you should minimize j1 and j2, you cannot be fired for. The nonpareto points you can be, unless there's another j that you're not writing in here, like for example, a time crunch or something like that. But that's the way it works.

So I mean, these are sort of obvious things. Actually, it'll get not obvious pretty soon. So this is, by the way, called – some people call it the optimal – this is called the optimal tradeoff curve of j1 versus j2. That's one name for it. It's the optimal tradeoff curve. It's also the pareto optimal points, all sorts of other things.

Let's see. So I think we've talked about all of this. By the way, if you have three objectives, you have a pareto optimal surface, and now we're talking about octants. So you have pareto optimal surface, and you have three objectives. And it's – or people call it the optimal tradeoff surface. By the way, people do plot these. You plot it by slices, very typically.

I should – these ideas just – they go – I mean, you should actually incorporate these ideas. They're so obvious that you probably already had all these ideas. But it's nice to know that there's a language for it and that people have been thinking about this and talking about it for a couple hundred years.

So for example, if you do circuit design, you, generally speaking, have a tradeoff between, say, power and speed. And you think – you should think in these terms. Basically, a point up here is – the technical name for a point up there is a bad design. That's the technical name for it. These are good designs. This would be called – if this was – if this is power, and this is delay, then these – this would be the optimal tradeoff curve – optimal power delay tradeoff curve, for example.

And it's very interesting. I mean, it's very interesting. It's very useful. And in fact, you should probably do these – any time someone tells you to do one thing, you should probably do is do tradeoff analysis anyway. So we'll get to that.

Now, this brings us a question. So as we stand right now, by the way, you cannot argue that x1 is better or worse than x2 or vice versa. They are absolutely not comparable. All you can [inaudible] x1 is better on j2, and it's worse on j1. And if someone says, "Yes, but is this better?" You say, "Depends. Depends how much you care about j1 and j2."

So the most obvious thing to do is to form a weighted sum objective. So that's done other ways, but here's the most obvious one. You form a composite objective, which is j1 plus mu times j2. mu is a parameter we're gonna use. It's positive, and it gives the relative weight between j1 and j2. So that's what we're gonna use. It's a composite objective of this.

By the way, mu has a lot of meaning here. For example, it is often the case that these two measures have different units. They can have different units. This – well, obviously, in a circuit – I mean, circuit's not gonna be least squares. But anyway, let's imagine that it was. You'd have power and delay or something like that. These are in watts and seconds.

So you can – it's actually very useful, and maybe in this case, watts squared and second squared. It's very useful to think of mu here as actually containing the units that translates both of these into units of irritation because that is, after all, what an objective that you minimize is supposed to be. It's a unit of irritation. It could, for example, be units of dollars or euros.

That didn't come out right. It's the payment you're gonna make. Now it came out right. So it's a payment. Then it's units of irritation. And so here, mu maps the units of the second objective to – makes it comparable with the units here. I'm gonna give you an example.

Suppose we're gonna do a function fit with – of some data. So you have some basis. Everyone agrees that's the – we should use the prolayed spheroidal harmomics for estimate a [inaudible]. I have a ton of data acquired at considerable expense and trouble, and I wanna fit a small model to it. Now, of course, one thing is gonna be the fit. You wanna fit the data. But the other one might be the smoothness of the data, so – I mean, the smoothness of the model. Sorry.

So I don't wanna get – I could get an excellent fit with something that wiggles like crazy. It would wiggle to fit your data, which might have no noise in it. But no, I want it – so I wanna trade off smoothness here with fit. This is like classic example of multi-objective fitting.

Now, what – these have actually kind of – they actually have different dimensions because smoothness is related to a difference or a double difference divided by a spatial difference or something like that. So this might be – let's suppose we're actually – what we're trying to fit is a temperature distribution. This will be – if you take this value, that's gonna be in degrees centigrade squared. This is gonna be in degrees centigrade per meter quantity squared.

Therefore, mu has units of meters squared. Thank you. I was gonna say it, but thank you. [Inaudible]

So that's just – so you think of mu as translating. Now, there's one more thing I wanna do before we quit today, and that is to look at what j1 plus mu j2 does on a j1, j2 plot. Well, if I look at the points where this weighted objective is equal to [inaudible] alpha, these are actually just lines with a slope minus mu. That's what I'm doing.

And so in fact, if I ask you to minimize j1 plus j mu 2, the visual – it's very simple what you're doing. Again, you don't have this, but you conceptually, if someone gave you [inaudible] achieve, [inaudible] what it means, and they said, "Please minimize j1 plus j mu 2," what you do is you take things of [inaudible], and you move them down [inaudible].

This is lower – this is when you move in this direction, the line – actually, there's a name for this. It's – I just forgot the name. It's a beautiful name in – it's the equivocal set or something like this. It's the things you care about equally for in economics. What is it?

Student: Level set?

Instructor (Stephen Boyd):Level set. It is a level – it's a level set of j1 plus mu, j2. But there's an economics – there's a beautiful name. It means you're just – basically, it means –

Student: Indifference curve.

Instructor (Stephen Boyd):What is it?

Student: Indifference curve.

Instructor (Stephen Boyd):Thank you. Oh, thank you. [Inaudible], this is an indifference curve. It basically says if you [inaudible] that, that, or that, [inaudible], they look – they're different, but they're cool for cool by me. So this is an indifference curve, meaning you don't care which it is on there. But you do care the lower you go.

So the point here is, you take this thing, and you move it down like this until you just touch here. And that is the minimizer of j1 plus mu, j2. That's the picture. And we're out of time, so we'll continue on Thursday.

[End of Audio]

Duration: 77 minutes

IntroToLinearDynamicalSystems-Lecture08

Instructor (Stephen Boyd):I have a feeling that we're on. Confirmed. Can you go down to the pad and I'll make a couple of announcements. The first is that homework four is posted. I said that I wouldn't announce these types of things. In fact, I think we posted it yesterday. The next one – I shouldn't have to say this, but can you turn off all amplification in here? I shouldn't have to say this, but the midterm is actually the end of next week, so it's actually eight days from now. We're more panicked than you are, just for the record. We have a lot of work to do on that. It's coming along actually quite nicely, I have to say.

That's of course next Friday to Saturday or Saturday to Sunday. What we'll do, I think, is today post last year's midterm, if you just want to see what one looks like. You will also find out where homework problems come from. Many homework problems started life as midterm and final exam problems. We'll post one so that you know what it looks like and so on, and then maybe a bit later we'll post the solutions. In fact, as to whether or not you really have to go over last year's midterm, I think you actually don't really have to unless you want to.

If you've been following the homework and understanding all of it and the lectures, you're welcome to do last year's midterm. Let me not discourage you from that. I don't think you really need to. Let me say a couple of other things about the midterm. The midterm will cover through lecture eight, which is material we'll cover a little bit today and finish up next Tuesday. It will cover through homework four, so that's the coverage of the midterm. We'll probably put something to that effect on the website so you know. Any questions about the material from last time or the midterm?

We'll continue. What we're doing today is actually just looking at some extremely useful extensions of least squares, and many of them involve this idea of multi objectively squares, so in multi objectively squares, instead of having just one objective like AX-Y norm squared that you want small, there are actually two, and so you want them simultaneously small.

Now one of the problems there is the semantics of that is not clear. It doesn't really make any sense to say please minimize these two things. It makes absolutely no sense. The first thing you have to work out is what is the semantics? By the way, if they're not competing, it does make sense, but that's an extremely rare case. Otherwise, you have to figure out what even does it mean to minimize two objectives.

So we started looking at that last time. As a thought experiment, we did the following. We simply took ever XNRN and we evaluated J1 and J2, the two objectives. You want both small. For every X, we put a point. All the shaded region shows you pairs as we've written it, J2, J1, which are achievable, and then the clear area here are pairs that are not achievable. We talked about this last time. We talked about the following idea, that if a certain X corresponds to this point, then basically, all the Xs corresponding that map into what is lower and to the left – these are actually points that are unambiguously better than this one.

Everything up and to the right is unambiguously worse. Unambiguously better or worse means that on both objectives, it's better. The interesting parts are the second and fourth quadrants, because here, it's ambiguous. In fact, this is where we're going to have to really work out what the semantics is. Here, a point – if you want to compare a point here and here, in fact, one of the ways you'd say this is you'd say that in fact, these two points are not comparable. One has better J1 but worse J2. The other has better J2 but worse J1. They're incomparable is what you say.

The boundary here, which is called the Pareto optimal – these are Pareto optimal points. This is the Pareto boundary. It's also called the optimal tradeoff curve. These points are characterized in the following way. Any point on here, there's no other point that's better. That's what it means. The least you can say, and in fact, that's all you can say.

If someone says you have a multi-objective problem, you want J1 and J2 small, and they're not yet willing to commit to how you trade off one and the other. If they simply say no, I want both objectives small, you can already say something of substance. You can say the following. You can say that that point is a stupid choice. Why? All of these are better. You can say that's an infractive choice, but it can't be done.

If someone wants to minimize these two objectives, the only non-stupid choice – non-stupid and feasible choice – are gonna be the points on this boundary. You've already done a lot to say that you can focus your effort on these points on this optimal tradeoff curve. That's the basic idea in this. This extends to three objectives, four and so on and so forth. It's an idea that's completely obvious and that you probably already have in your head anyway.

There's a very common method to find points on that curve, and it works something like this. Before we do that, I want to talk about what the curve might look like qualitatively. Let's talk about that, and let me try to do this consistently. I'll draw my first objective there and my second here. This is what it looks like. That tradeoff curve can have lots of different looks. I'm gonna draw a couple of them. One would be something like this. It actually has to go down. These are all achievable. That's actually quite interesting. I'll make this three and I'll make this one.

This is very, very interesting. In fact, if you work out that this is the tradeoff curve, and you'll see how to do this very soon. This has huge meaning and implication for this problem. The way you would describe this sort of casually or informally is this. Basically, you would say there isn't much of a tradeoff because the lowest you could ever get J1 might be over here, and that might be 0.9. That'd be the lowest value you could get J1 while ignoring J2. It might be here. This might have an [inaudible] or something, and this might be 2.6. So here, you'd say the smallest you could ever make J2 ignoring J1 is 2.6.

On the other hand, look at this. There's these points right around here, which give up ten percent in each objective and yet get both. This is so obvious that I almost hate to go over this. This is the proverbial knee of the curve. It's an efficient point. These ideas are extremely important to have because I guarantee you will be working on problems. You will finish something or do something and the point you will find will be right here. That's what's gonna happen. My opinion is it's not good enough to simply return that point. It's not responsible. The correct thing to do is to say oh, yeah, I got J1 down to 0.9.

Let's say it's ride quality. It doesn't really matter what it is. Say I got the ride quality down to 0.9. They'd say that's great. You'd say but you know what? This is when you go back and there's a design review. You'd say you know what, though? It turns out if we accept a ride quality of 1.0, I can do it with one quarter of the fuel. I think if you don't point out that there's this point here, I think you're actually being irresponsible. The same goes for over here. If someone says find me a point and you find this point and you say – it'd be like if you're doing circuit design. You could say oh, I can make that thing clock at 2.6 gigahertz. But actually, if it clocks at 2.45, I'll use one-half the power.

As to which is the best choice, it depends. But the point is to me, it's irresponsible if you don't point this fact out. Basically, you don't ever even do – when you do least squares and things like that. Anything involving this, you should always just as a matter of responsible engineering – you will do studies like this just to check. You wiggle things around to see if things could dramatically change.

This is one where there's essentially no tradeoff. To really get no tradeoff, you do this. That's absolutely no tradeoff. This point – actually now, it's great. This is the one case where you can say that is the optimal point. That's the only time when a biobjective or multi-objective problem has a unique, well-defined answer where the semantics is clear. That point is good. It's the best one. No other point would be reasonable here. Any other point would be worse than that point. This is when there's absolutely no tradeoff.

Now, let's look at the other extreme. The other extreme happens and the other extreme looks like this. You have a point there and a point there, and this might look something like that. That might be the tradeoff curve. Now, there's a tradeoff. In fact, this is the opposite. Now the tradeoff is almost linear in the sense that when you give up one, you actually gain in the other by a fixed amount. This is what people call a strong tradeoff, and the slope actually – one of the names for the slope is the exchange rate.

You're actually exchanging J1 for J2 when you move here. When you go from this design to this design, what have you done? You're doing something like you're exchanging J2 for J1. You're doing better on J1 by giving up on J2. The slope literally is sometimes called on the streets the exchange rate.

This is conceptual models. We're gonna leave them up here and we're gonna come back to them. Let's look at the idea of the weighted sum objective, which comes up independent of discussion of tradeoff curves and things like that. It's also completely normal if you have two objectives to – if you want to come up with some answer to actually just add them with some weight in between them. So I add this function plus Mu, this objective plus Mu times that one, and the idea is that Mu is supposed to give a relative weight between J1 and J2. Question?

Great question. I was trying to go real fast and kind of avoid that question. I'll do it. Could the optimal tradeoff curve look like that? You want me to draw it in like that? It cannot. It actually has to be convex here. It has to curve up, and that's because for least square problems, the J1 and J2 are both convex functions. That's not part of this class. I was drawing them the way they must look in 263. If you have non-convex functions, they can absolutely look like that. I will show you one that they cannot look like. It cannot look like this ever. It can't look like this. That's not possible.

First of all, these points are not Pareto optimal because if that's a feasible design, everything above and to the right of this point has a technical – the technical name is it's a bad design. That means that this is not part of the tradeoff curve. In this case, the tradeoff curve for something that would look like that actually is discontinuous. It's got this point and then it's got this line segment. These things can happen in the general case with general objectives. They can't happen with quadratic objectives like you'll see in 263.

Before a weighted sum objective, it turns out that you can interpret this easily on a J1 J2 plot or J2 J1 plot, and that's this way. If I look at level curves of this composite objective – that's J1 plus Mu J2 – in the J2 J1 plane, these are nothing but lines with slope minus Mu. If you were to minimize J1 plus Mu J2, here's what you're doing. You are doing nothing but this. You're moving this line with a slope of minus Alpha, which is fixed, and you simply move it down until you last have contact with the set of achievable points.

That is always a point on the Pareto optimal curve, and actually, there's a lot more interesting stuff about that point. Another one is this – if you were to zoom in locally, these local slope here would be exactly Mu. Let me summarize that. By minimizing J1 plus Mu J2, you will actually find a point on this tradeoff curve. That's fact number one. Number two, you will find a point where the local exchange rate is exactly Mu or where the angle is Mu. This picture, though simple, explains everything. If I increase Mu, what happens?

If you increase Mu – let's think about what happens. If I increase Mu, what you're really saying is you know what? I care more about J2 than I said before. Presumably, we'll find a new point where J2 is smaller. You're gonna pay for that. J1 is gonna go up. That's the way these things work.

Let's just see if you can get that visually. It's very simple. If you crank Mu up, that's the weight. You simply change the slope like that and you do the same experiment. You take this thing and you move it until it just touches and sure enough, that's the new point for the new slope. Here I cranked up Mu by a factor of three or something. That's a new point. Sure enough, it's a new point on the optimal curve, and indeed, it has reduced J2 and to pay for that, it has increased J1.

If you have the ability to minimize the weighted sum of the objectives, you can actually now sweep out the optimal tradeoff curve by simply sweeping Mu over some range and minimizing this weighted sum objective and you will sweep out points. By the way, if the Mus you choose are not over a big enough range, you will sweep out just a little tiny thing.

In fact, in practice, a lot of people use Mu on a log scale because it has to go for usually a pretty big range. This is just a practical detail. Conceptually, you simply solve this problem for lots of values for Mus, store the design and the J2 J1 achieved, and plot that. You have the optimal tradeoff curve. That's exactly how these curves were created.

Not only that, but the picture gives you a lot of geometric intuition about what happens when you mess with Mu. Now, I want to go back to my two tradeoffs. Here's a problem where there is no tradeoff. Let's do the one where there's a slight but very small tradeoff. Let's do that one first. J2 J1 and I'm gonna put a slight but small tradeoff like that. Now, let's talk about minimizing J1 plus Mu J2. What will happen when I minimize J1 plus Mu J2? As I vary Mu, what happens?

Well, when you fix Mu, you get a slope like this, and you simply march down this thing until you first lose contact with it and you get a point there. Now, you change Mu a lot like that, and you go down here and you get a new point. What you should see here is that in fact the Xs are not changing much. You're always getting – over a huge range of Mu, you're getting points right around there. In other words, you're getting the knee of the curve over some huge range of Mus. Everybody see this? The two things – here's what you'll notice. Number one, the actual design you get is largely insensitive to Mu.

That's the first thing. If you crank Mu to ten to the eight, then you might start getting something up here. And if Mu is ten to the minus eight, you might start getting a point down here. But the point is that for this huge range of Mus in the middle, you have a lot of Mus and you're just tracing out this little tiny thing here. By the way, if you see that, it means you're seeing a problem where there's not that much tradeoff. The two objectives are not particularly competing in this case. That's kind of the idea.

Let's do the other one now. Let's do this one. Honestly, I don't know why I drew it with J1 vertical. Let's do the other one where there's a strong tradeoff. Here's the curve like that. Now, let's talk about minimizing a weighted sum. What happens now? How sensitive – what happens is you vary Mu and minimize the weighted sum objective. What happens? It's very sensitive. Basically, for Mu below some number, you kind of get points here. Let's say it flattens out over here. For Mu above some number, you start getting points over here.

Right when Mu is around this slope, right as you sweep Mu through that point, this thing jumps tremendously. Everybody see that? That's the point here. You will see this, and it has a meaning. This is the meaning. It means you've got a linear tradeoff. If the tradeoff in here were exactly linear, you'd actually get an amazing thing where the weighted sum objective would jump from this point all the way to that point with nothing in between. It would be absolutely discontinuous. For quadratic functions like we're looking at, that can't happen. That could happen.

When you get many dimensions, all of these things – you can get all of these phenomena. You can get parts where the surface is kind of angled. You can get other parts where it's very flat, and as you mess with weights, things will jump from one place to another. Other regions where you mess with the weights and in that case, it's a tangent hyper plane touching this optimal tradeoff surface. You mess with the normal and it kind of rolls around and doesn't do very much. You can get all of these kind of phenomena, but it's important to understand these ideas.

Now let's talk about how you would specifically do this for a biobjective least squares problem. How do you minimize this? The way we can take two quadratic objectives and reduce it to a problem we've already solved is all we have to do is say that the sum of this norm squared plus that norm squared is just this. It's absolutely nothing more. You can check.

The top part of this is AX minus Y. The bottom part is square root Mu FX minus square root Mu G. Now, the norm squared of a stacked vector is the norm squared of the top plus the norm squared of the bottom. Norm squared of the top is that term. Norm squared of the bottom is that. If you like, you can put the square of Mu in both of these places, but when you square it and pull it outside, it looks like that.

That means we're done, because this we know how to do. This is no problem. We'll call that A~ or something like that. It just means in [inaudible] it's even simpler. It's something like AN – if you really want to do this, something like that times F. I hope I'm doing this right and then backslash and then Y and square root Mu times G. There you go. There's the code for it. You shouldn't have to write that down.

The formula for it is this. It's gonna be A~ transpose A, one of the inverse, times A~ transpose. Well, A~ transpose A – you can work out analytically what that is. That's it. That's A~ transpose A~ like that. You see something actually quite beautiful here. You see A transpose a plus Mu of transpose F, and that's how this works. Over here, you get A transpose Y plus Mu F transpose G. It just sort of works out. It's a very pretty formula.

Let's look at some examples just to see how this works. These are going to be really simple examples, but just to see what happens. This is our friend the unit less mass on a frictionless able. We have a ten second period. We apply forces X1 through X10, each one for a second in turn. We're just gonna have one. We don't care about the velocity. We care about the position, FT equals ten, and so we have Y equals A transpose X where A is in [inaudible], and I think you may remember what A is. I think it goes down by one. It's large for the first one and then goes down a half or something.

By the way, when you see a vehicle or objection motion problem where the person specifying the problem tells you where this object has to be but doesn't seem to care how fast it's going, generally speaking, that corresponds to a non-positive social use. Usually, this would be something like a missile hitting something. When someone says no, I'd like you to be – if you say what about the velocity and they go it doesn't matter, that – you should be suspicious at that point.

This is one of those cases where there are no specifications about the velocity of this mass, so no specification on the velocity. Here, you just want to be near the point one. This is a stupid problem. We could solve it by hand. It's totally idiotic, but just to give you a rough idea, we could work out what this is. J2 is the sum of the squares of the forces used. This has units of Newtons squared.

You might ask why would you care about the sum of the squares of the forces applied? Let me ask that question. Many people take whole courses here where the entire course, everything is quadratic. You go take a controlled course in aero astro, everything – squared Newtons, integrated – that's all you see. Why do you care about it? It corresponds to energy.

I'm going to tell you something. That's not true. That's what they tell you in those classes. That's what I should be telling you now. That's the party line. That's what we're supposed to say, that it corresponds to energy. That's total nonsense. Generally speaking, I know of almost no case where it actually corresponds to energy. By the way, any case where it does correspond to energy like in a disk drive servo, there's actually no limit on energy and no one really cares. I said it. I'm being honest.

Now why do we really care? Why do we work with the sum of the squares? What do you think? Thank you. It's easy to analyze. Right. Because we can. That's why. I just wanted to be clear on this. There are lots of other things here. If this was a thruster, you could probably care. That's something you would really care about. That's the field use. You might care about this. That's the maximum force applied.

Why? Because this dictates how big a thruster you have to get. This has practical use. The sum of the squares, it might, but it's very unlikely. You never get an actuator, and this comes up in signal processing, too, where it says here it is. It needs 28 volts in, five amps, and it never says under no circumstances should the norm of the input exceed this. You just won't see that. That's not what it is.

We're gonna go back, now that I've said that we do this because we can. Let's go back. If anyone asks you and you don't want to get into this big argument, you just say it represents energy. If they buy it, move on quickly. That's my recommendation. It's kind of a stupid problem, so let's talk about some things here that we can just know immediately.

This says that basically you're gonna apply a force for ten seconds. You're gonna move this mass and you'll be charged – there are two things you care about – how much you miss being displaced one meter and the sum of the squares. Let me ask some questions. Does it make any sense to move the mass backwards before moving it forwards? Obviously not because you're running up a J2 bill and not for any particularly good reason in terms of J1.

Does it make sense to overshoot the target, which is one point, and to say here are my masses and say oh, look at that. My final position was 1.1. I overshot. No, that's totally idiotic because you'll run up a bill here for overshooting and it's stupid because for the same amount, you actually could have landed right at the point and run up zero bill here. You're gonna always undershoot. You can also figure out in this problem that you're always gonna push, and you're gonna push more at first because it's more efficient. You can figure out a lot of this before you ever even form a formula.

The optimal X is this. That's a function of Mu. That's that tradeoff parameter. As we vary Mu, we're gonna get different full trajectories. By the way, you can even work out an analytical formula for this. That's not the point and it doesn't really matter. Here's the optimal tradeoff curve. So there it is. It's very pretty. Here's the energy. Notice how I said that without making any apology. Here's the energy and here is the square of the missing distance. This curve actually hits zero at a point. It doesn't approach zero [inaudible]. It hits it.

This is a very interesting point that we're going to discuss either later today or maybe Tuesday. At this point, you are hitting the target exactly, and you're using the minimum energy possible. That's what it means to say that this curve hits this line. Zero means Y is one. At ten seconds, that mass is exactly at a position of one. That's the energy bill you run up. There are many, many force programs that will displace the mass one meter after ten seconds. They all lie along this line, and they're characterized by using more energy. This is gonna be the least energy thing that gets you right there.

What about here? Does this curve hit or is this [inaudible]? Does it hit it? Let's ask the question. Would it be possible to run up a bill J2 of zero? Could you? It doesn't move. You could do that. You could just have X equals zero. So you do nothing. You're doing very well in terms of J2. You couldn't do any better. You just take the hit, which is the cost on J1, and that's here.

By the way, this is a beautiful example. Take a look at what that curve looks like near zero. So basically, if someone comes to you and says I'm sorry, I'm just not gonna do anything, this curve – not only does it have a steep slope, it has infinite slope there. That says that with extremely small levels of force applied, you can reduce your miss-hit distance by a correspondingly very large amount. That's the picture.

This is a silly example. You could have done all of this analytically or figured it all out, and there's no surprises here. Trust me, if this was a vehicle with 12 states and 13 different inputs representing different thrusters and control surfaces you can actuate and things like that, this is not obvious. You already have four lines or five lines of code that will beat anything any person could ever come up with, and I'm talking about good pilots and things like that. Same code. Three lines. I think it's just the one I wrote before. It's not even three. Three with a lot of comments, actually.

This stuff looks simple. This example is stupid. Trust me, even if these matrices – if the dimensions of vectors get to be five, ten, let alone 500 or a thousand, you're doing stuff that is absolutely impossible for someone doing intuitive based stuff to even come close to.

Now there's a very famous special case of biobjective least squares. It's where the second objective is really simple. It's just the norm squared. Here, the way to understand the semantics of it is you have a problem where you say I want a good fit. I want AX minus Y norm squared to be small, but I don't want to do it if the only way to do that is to have a giant X. I want some tradeoff there. I will accept a poorer fit in return for a modest X.

Where you operate on that curve determines the tradeoff of the optimal size of X versus the fit. At least one end of the trade we actually know. When you only care about J1, that's just least squares. That's classic least squares. We know the solution. If you only cared about J2, let's get that out of the way right now.

If you only cared about J2, what's the best choice of X? Zero. And the objective on the other side? Norm Y squared. The two end points of this tradeoff curve are now known. By the way, that's an exercise you should always do is figure out what you can say about the endpoints, because all the action then goes down in between the two.

In this case, you get X is A transpose A plus Mu I inverse A transpose Y. This has got lots of names. Maybe the most common is tickenoff regularization. In statistics, you will hear the following phrase. There's probably many others. In statistics, this is called ridge regression, and Mu is called the ridge parameter. In tickenoff regularization, Mu is called the regularization parameter.

I'll show you something kind of cool about this. This formula makes sense for any A – skinny, fat, full rank or not full rank. I have to kind of justify that, so I'm going to. Here's my claim. My claim is that A transpose A – there's lots of ways to prove this, but I'll do one. I'm gonna claim that that's invertible provided only that Mu is positive. This formula even makes sense for A equals zero. It's a stupid formula, but it makes perfect sense. I'm saying provided here Mu is positive. If A is zero, no problem. It's X equals Mu I inverse. Mu I is perfectly invertible times zero, so X is zero in that case.

By the way, when Mu is zero, you recover least squares, and now this formula is one you better watch out for, because that formula only makes sense when A is skinny and full ranking. It parses, but it doesn't pass the semantics test if, for example, A is fat. The weird thing is you can do tickenoff regularization if A is fat and this makes perfect sense.

This is why some people use tickenoff regularization because they're lazy or they tried this and they got some error or something somewhere. Some software told them that they were trying to invert something that was singular into working precision, and they were like, well, whatever, and then they just put in plus 1E minus 6I, and they said now it's working. Trust me, you see a lot of that.

Let me justify why this matrix here is in fact invertible provided Mu is positive. Totally irrespective of A – I don't care about the size of A. I don't care about the values, rank – couldn't care less. Let's check. What we have to do is we have to do an [inaudible] experiment, and you'd say, well, look, that's a square matrix. Suppose it were singular. That means there's some vector Z, which gets mapped to zero, and Z is non-zero. That's what it means. It means this matrix here [inaudible] a square being singular means it's got an element in a null space. It's non-zero. That's the case.

Let's do something here. If this is the case, I'm gonna multiply that equation on the left by A transpose, and I'm gonna write this down. That's surely zero, and this is a zero vector. That's zero number. Why? Because Z transpose times the zero vector is zero, obviously. Now what I'm gonna do is I'm gonna take this and let's expand it. I'm gonna write it this way. I'm just going slow here.

Plus Mu and I've commuted the Mu and I get that. Now, I'm gonna write this as norm AZ squared plus Mu norm Z squared equals zero, and now I'm gonna use a very deep mathematical fact. If you have a sum of two non-negative numbers and the sum is zero, you can make an amazing conclusion. That is that both the numbers are zero. Everybody follow me? Norm AZ square to zero, so norm AZ is zero, and norm Z is zero.

If you want to know where does the Mu positive come in, it's right now, because if Mu is zero, all I can say is that. This says because this is not zero up here, we have our contradiction and we're done. That's why this works. That's one way to say it. Another way to say it is to look at the stacked matrix and just show that the stacked matrix [inaudible] is full rank. That's the other way to look at it – that if you take any matrix at all, any size, any shape, any values, and you stick below it, square root Mu a positive number times the identity, that matrix is full rank and skinny. That's the other way to think of it, which is also probably a better way.

This is called tickenoff regularization and has lots of applications. Here are the types of applications you would see. It's very common in estimation inversion, and it works this way. Typically something like X minus Y is a sensor residual. Here, if you choose the X that minimizes sensor residual and you like the X you see, great. No problem.

But in fact, you might have some prior information that X is small. If you have prior information that X is small or another application is where [inaudible] is actually your model Y is AX is actually only approximately valid, and it's certainly only valid for X small. We're gonna see an example of that momentarily.

So the regularization trades off sensor fit in the size of X. That's what you'd do. By the way, this comes up in control, estimation, communications, and it's always the same story. There'll be some parameter and some method or algorithm, and when you turn this knob, because that's really what Mu is. It's a knob that basically tunes your irritation is what it really does. It tells the algorithm what you're more irritated by.

As you turn Mu, what will happen is at first you'll get – if you turn it all the way one way, you'll get a control system that's kind of very slow and doesn't do such a great job but it doesn't use huge forces. You turn the knob all the way the other way and you'll get something that's very snappy and is using very big forces or things like that.

In communications, you get something that equalizes things out beautifully, but it's very sensitive to noise. You turn the parameter the other way around and you get something that is gonna be really – it's kind of very calm, doesn't overreact any noise or anything like that. It cleans it up a little bit but not much. These are the types of things you'll see all over the place.

Let me mention some examples in image processing. That's a very famous one. We should actually add that to the notes. A very famous one in image processing is this. It's called La Placean regularization. Let me say what that is. You've already done one, or you will at 5:00 p.m. today. Look at an image reconstruction problem. That image reconstruction problem, the sensor measurements are pretty good and there won't be any problem. It will just work. Why? Because we arranged it to.

However, in general, you'll have the same sort of thing, and actually what you want to do there is you want to add – let me just explain what we're doing. I want to estimate an image, so I have an image here and I have some pixels. What I'm estimating, my X is sort of the value in each of these things. If with your sensors you estimate X and you get some crazy numbers that vary pixel by pixel by huge amounts, this kind of hints trouble, because normally when a person writes down pixels, there's sort of an implicit assumption that you're sampling fine enough.

So for example, if this were plus ten and that was minus 30 and there were wild swings here, what you'd do when you looked at that is that you'd probably say you need to sample finer is probably what you'd say. So now my question is let's say that AX minus Y – X is a vectorized version – it's a rasterized version of the image. This is gonna be misfit from my sensor readings. I want the following. I want to trade off the fit – that's this thing – with the smoothness of the image. Now, I'm waiting for you to tell me what do I do.

Let's add a new objective. To do this, we have to add a new objective, and the new objective is gonna be not smoothness, actually. It's the roughness. We have to write down a roughness objective. Someone please suggest a roughness objective, which is kind of a norm. Perfect. We're gonna form a vector, which is the difference between neighboring pixels.

We could have vertical or horizontal. We could have diagonal. It doesn't matter. We could have all of them. I'll skip a few details here. They're not that complicated. When you do that, you can actually write that as a matrix D, because it's really a differentiation. I'll take DX and I could have things like D horizontal and D vertical X.

This is a new image whose value at a certain pixel is the horizontal difference here, and that's the vertical difference. By the way, if both of these matrices – describe the null space, actually, of these two matrices. What's the null space of these two? Constant. [Inaudible], but in fact this would be any image that is constant along horizontal lines but can vary in X. The null space in this one would be any image which is constant along vertical lines but can vary this way. I don't know if I got that right. I think I did. Something like that.

So what you do is simply this. We're done. There. That's a least squares problem. Now you've turned Mu. You turned Mu all the way to zero, and you get the problem you're doing now. You get no regularization. In other words, if you like what you see, in this case, there's no problem.

If, however, you do this – by the way, in many problems with noise and things like that, when you actually just minimize this, you'll get an image which is way too wiggly and stuff like that. Now, you turn Mu up. By the way, if you turn Mu all the way up, what does the solution of this look like? Don't do the math. I just want the intuition. Yeah, it's totally smeared out. It's just a big, gray mess, and it's just equal to the average value or something like that.

Somewhere with Mu in between, you're gonna see the right picture, and then you might ask how do people choose the regularization? Everyone see what I'm saying? This is how you use regularization. This is it. I don't know why we don't have an example of that in the notes. We'll add one, I guess. Then there's the question of how do you choose Mu? How do people choose Mu? We should distinguish two things – how do they really choose Mu and then when someone asks them, how do they choose Mu? What do they do?

How do they really do it? They try this for one Mu and they go no, it's too smeared out. Reduce Mu and they go nah, it's getting too much speckle in there. I can still see some weird artifacts and they go increase Mu. They iterate over Mu. They're just tweaking Mu. That's how they really do it. What happens if someone says in a formal design review how did you choose Mu?

Then they go, well, I calculated this optimal tradeoff curve and statistically, this corresponds to the posterior variance of this, that and the other thing, and I used the such and such model and that's how I came up with Mu equals two times ten to the minus three. That's what they would say. Whereas in fact, they tried Mu equals 0.1. It got too smeared out and they tried Mu equals 1E minus six, and they didn't get enough regularization. That's how they really did it. Any questions about this?

By the way, if you know about least squares and regularization and tricks like this, you're actually on your way to – this can be very effective in a lot of problems. By the way, one more point about this. If you go back down to the pad here, this penalizes smoothness, but suppose you also care about the size of X. Suppose you run this but X is huge – it's smooth, but huge. How would you reign in the size? I'll take my pen out. What do you do?

You got it. Less Lambda norm X squared. How would you choose Lambda? By messing around. How would you say you chose Lambda? You would talk about the optimal tradeoff surface and tangent and exchange rates and then if you've had some statistics, you could throw in some statistical justification. But you found it by fiddling with it. It's not just total fiddling with it. As you increase Lambda, you can say one thing about the image – what happens? It gets smaller. It's gonna be a little bit rougher and it's gonna have a little bit of a worse fit to the measurements, if that's what these are. That's it.

So now you know how regularization works. It works quite well. Next topic is related. It's also a huge topic – non-linear least squares, NLLS, and here it is. It says I have a bunch of functions. Now, for us, we have the residuals as AX minus Y. That's an affine function. It's linear plus a constant. What we do in least squares is we minimize the sum of the components of the residuals. Now the question is what if these RIs are non-affine? That's the general case. This is called the non-linear least squares problem. Actually, of course, this residual's not linear, either. It's affine. Linear sometimes means affine.

How do you solve a problem like that? We'll get to that in a minute, but let's just look at some examples. These come up all the time, non-linear least squares. A perfect example is GPS type problems where you have ranges so you have measured ranges and from that, you want to estimate where a point is. There, you don't have to linearize. You would just minimize.

This is actually now the exact range error squared. This comes up in tons of places. In estimation problems, it would come up because you have some kind of non-linear sensor. Instead of something like a line integral through something, you might have something that is non-linear. This comes up all the time. That's an example.

How do you solve these problems, and here, I have to tell you something. The first thing that has to be admitted, if we're being honest – there's nothing that great about being honest, but to be totally honest, no one can actually solve this problem in general. That's the truth is basically this problem cannot be solved. Instead, we have heuristics that solve this problem. They don't really solve it. That would be they solve it like that. You don't actually really solve. So non-linear least squares problems in general are not solved period.

If you go to Wikipedia, if you go to Google and type in non-linear least squares, whole books, everything all over the place. You will probably find nothing that admits this fact. That's a very big difference from linear least squares or least squares that we'd been looking at so far. We said that A transpose A inverse A transpose Y is the least square solution. We weren't lying. That vector minimizes the norm of R if R is AX minus Y absolutely. There's no fine print, nothing. That's the minimizer. All methods for [inaudible] a least squares problem don't have that property. They are all heuristics.

You probably won't find out certainly from people who have an algorithm for this. It gets kind of weird after awhile to say things like how'd you do that? After awhile, in [inaudible], you say I solved a non-linear least squares problem. Technically, that is false. You'll see that in papers. It's false. When someone says that in a paper, there's always the question because there's two options – either A, they know they haven't solved it and they're a liar because they're saying in a paper they solved it or B, they don't even know that they may not have solved the problem.

It's usually the latter. That's usually the problem. They're just totally clueless. They're like I don't know, I got the software. I downloaded it from the web. It was called non-linear least squares. It didn't complain. You have to know it's all a heuristic. You don't get the answer.

By the way, in practice, you very often do get the answer, but you don't know that you got the answer. I made enough of a point about that. Having said all that, and also having said that I will probably slip into the informal language where I'll say how do you solve non-linear least square problems, the answer is you don't because you can't because no one knows how to do it in general.

How do you approximately solve or something like that, so you have to put a qualifier like an asterisk, and then at the bottom of all these pages you just have a little note where it says solve, and then down here, I'll just write it in so we're all cool here, it says not really, but maybe. That would be the right thing. That's just in force until I say it's not.

So how do you solve least square problems? There's lots of methods that are heuristics and they often do a very good job for whatever application you're doing. The very, very famous one is Gauss Newton. By the way, the name of the method suggests that this was not developed five years ago. This is not exactly a new method. It's actually pretty straightforward. It goes like this. You have a starting guess for X, and what you do is you linearize R near the current guess.

Now, linearize means find an affine approximation of R near there. Then what you do is you replace that non-linear function with this affine one. Affinely squares, which on the streets is called linearly squares – we know how to do that. That's what we've been doing for three days. That's no problem. Then you update. That's the new X. Then you repeat so you get a different model. This is called the Gauss Newton method. Here's some more detail, and here's the way it works.

You take the current residual and you form – basically, what you're doing is you form an approximation that says if X were to be near XK, then I would have R of X is about equal to R of XK plus – that's the Jacobean – times the deviation. Here, if you wanted to, you could say something like provided something like X is near XK. I can put a little comment there like that.

Now what we do is we write down the linearized approximation. We rewrite this to make it look like an affine function, but we can put it like this. It doesn't really matter. Now what you do is you minimize the sum of the squares of these things over choice of X. That gives you your next iterate. The next iterate is that, and it's just a formula for that, which is this thing. You repeat until this converges, which, by the way, it need not. Question?

Absolutely. You're right, and I'm very glad you brought that up. The question was this, that last lecture, I believe I was ranting and raving about calculus versus least squares where calculus is getting a linear model of something that's super duper accurate right near the point, but it's vague about the range over which it's – did you notice that in your calculus class? They go this is a really good estimate near this thing, and you go, well, what does near mean? And they're like, you know, near. Don't you – that's the beauty of all this, right? I don't have to say how near it is. It's the limit.

The point was how about doing Gauss Newton where you don't use the derivative of the Jacobean but you get a least squares based model? Instead of this, use a least squares based model, and that was your question, and I can tell you this. That is an outstanding method, often far superior to Gauss Newton. My answer to your question is yes, that's a good idea. It often works really well. In fact, there's a name for this. In the context of filtering, when you're doing estimation of a dynamic system, that's called a particle filter. Let me just say a little bit about that.

By the way, I had some pseudo code. At this level, it's so vague that the linearized near current guess, you could use several methods there. One would be calculus, and the one I just described is calculus. In fact, linearized near current guess – that could be done by a least squares fit, just as you recommend. You wouldn't call this Gauss Newton anymore, but you might or something.

Anyway, in the context of filtering, it's called a particle filter. These work unbelievably well. Instead of this thing, which is sort of calculus, what you would do instead is you'd take XK and you'd add to XK little [inaudible], and you'd actually evaluate R of XK for a whole bunch of points right around there, and then you'd fit the best affine model to what you've got, and then all the method would work the same. That would work really well, by the way.

By the way, one little comment here. This says provided X is near XK, so you could well get into trouble here, because you linearize – in this case, we did it by calculus – you linearize and at least it says if X stays near XK, this is a good model. But you just solved this problem, and there's no reason to believe that this point is near XK. If it's not near XK, then the whole premise for how you got XK plus one is in doubt, because you're using this model which need not be valid. There's a very cool trick, which since we just did regularization, I can tell you what it is. The super cool trick is this. You add this. There you go.

This says minimize the sum of the squares of my model for the residual. That's what this says. This says but please don't go very far from where you are now, because if you do, there's no reason to believe that's a good model. By the way, this is called something.

This is sometimes called a trust region term because when you form this model – by the way, either with least squares or with calculus, you have the idea of what is the area around XK where you trust that model? That's the trust region. This basically is your trust region term. Everybody got this? Once you know about regularization, you start realizing you should be using it in a lot of places, and this would be a perfect example.

What I described here was the pure Gauss Newton. Let's look at an example and see how this works. Here's an example. We have ten beacons. No linearization here, at least not in the main problem. The true position, that's this point right here, and that's where we started off, and let me tell you what the measurements are. I have ten measurements. What I know is the distance of a point to each of these beacons. These are noisy.

By the way, if they were noise free, this problem would be trivial. If they were noise free – if I knew the range to this point, I would draw a circle with that radius around that point. I would do that for all the points and they would all intersect in one point and I'd say that's where you are. The problem is the range measurements have errors. In fact, they're considerable errors. They're plus/minus 0.5. So they're not really going to all come to one single point. We'll use Gauss Newton for that. Actually, here, there's other methods you could use, but this is just a simple example.

By the way, this is what the non-linear least squares objective looks like as you vary X. This is going to be – that happens to be the true global solution in this problem, but you see, this is not a quadratic function, which would be a nice, smooth bowl. It's got all these horrible little bumps and things like that.

In this problem, it doesn't have any local minima that are not global, but we could easily have done that by putting some point here, and then they would have had a little valley here that would have filled up with a lake or something. As soon as you have that, I guarantee you Gauss Newton method will, given the wrong initial point, land right there very happily, at which point you have not solved the problem and you don't know it.

Okay. If you run Gauss Newton on this starting from here and you're way, way far away. In this case, it actually works. In a sense, you actually do compute the solution, but you don't know it. I only know it because I plotted it. It's got two variables and I plotted it for everything. But the minute you've got five variables, you're not going to know it anyway. What happens now is the objective and the Gauss Newton iterate just keeps going down, and you can see it in about five steps. It hits what appears to be a minimum, and it takes maybe five, six steps or something like that, and that's it.

By the way, you don't really know – the final estimate, by the way, was quite good. The final estimate's minus 3.3 plus 3.3, so that's the actual true position. That's where you started. After one step, you were here, and two you were here, then there, then there, and now you can see you’re sort of in the region where you’re going to get the answer. It’s pretty good. You are getting from the least squares part of it.

You are actually getting the blending of ten sensor measurements, so you’re getting the power of blending. Some people call that a blending gain or something like that, and I forget what they call it in GPS. They have some beautiful, colorful word for blending lots of sensors and measurements and then actually ending up with an estimate that’s better than any of the individual sensors.

Here, you actually end up with an unusually good estimate – better, actually, than the accuracy of your individual sensors. That’s the picture. In this case, it actually worked. In other words, we got the global solution, but that’s only because I plotted it. I think I already mentioned this.

Let me ask – well, I can ask you. Suppose you have a problem. Real problems don’t have two variables, I might add, right? You have two variables, you plot it and you use your eyeball, okay? This is silly. You have three, you write three or four loops and you go to lunch. Let’s just bear that in mind. Real problems have ten variables, 100, 1,000. You cannot plot pictures like that.

So you have a non-linear least squares problem where you are estimating, let’s say, some non-linear estimate. Maybe it’s a topography thing. You have a non-linear sensor. It’s a variation on the problem you’re doing for homework right now, except instead of having a linear sense, you have a non-linear one. How big is that problem we gave you? Thirty by thirty? Tiny. That’s 900 variables. That’s pretty small.

Now you run a Gauss Newton in 900 variables and you get an image. By the way, if you’re imaging somebody’s head and it came out looking like a head, that’s good. That would be your feedback that something is approximate. If it came out not looking like a head, that wouldn’t be good.

What would you do as a practical matter there to check whether you got in fact or just to enhance your confidence that you may have actually minimized the non-linear least squares? What would you do? Exactly. You’d run it once and you’d see what you got. By the way, if you had a pretty good estimate ahead of time, and that’s actually likely to help. You start with that. But what you might do is exactly what was just suggested. You run it multiple times from different starting points and you just see what happens.

Here are some of the things that can happen. The first is that no matter where you started from, it always goes back to the same thing. What do you know if you do that 50 times? Let’s be very careful in our verbs. I used the word know, so what do you know when you run it 50 times and you keep getting the same point? Here’s what you know. You know nothing. I mean both in theory and in practice. You know absolutely nothing, because R900 is a huge place, and the fact that you just sampled 50 points out of it, that’s nothing. You know nothing.

Now as a practical matter, what can you say when someone says to you I did the imagining. There it is. I believe that's the image. It looks good. It's clearly a head and there's somebody's brain there. It looks good. Someone says do you know that's the global minimum and you go no. But I'll tell you what I did do. Last night when I went home, I started up a batch of doing 1,000 of these from different initial conditions, and I found that in something like – let's suppose you found in all of them, you say in every single case, I converged the exact same final estimate. They'd go cool.

You're getting the global minimum. If there's no lawyers present, you can say good bet. But other than that, if someone says what can you really say you know, you can't say anything. You can say I don't know. You would say if you ran 50 and got the same answer each time, you'd say that as a practical matter, you have enhanced your confidence in the design. Actually, there's a great phrase, which makes absolutely no sense and is wrong. It's very useful. It's this. You could say exhaustive simulation. Have you heard this phrase? That's great.

So this is what you say to someone when you open the door and you say hop it. You go are you sure this works? You go no problem. We used exhaustive simulation. Hop in. That's a very useful phrase. Of course, it makes absolutely no sense if there's more than three or four things varying, and is generally speaking just wrong. You say we checked a million values of bursts of wind and things like that, all sorts of terrible things. We've simulated them all. Of course, you cannot.

If you find yourself in a situation, you can always use that phrase. Then you mention the number, and as long as this person doesn't take the nth root of that number where N is the number of parameters, everything is cool, because a million simulations in R10 is like zero. It's like the tenth root of a million, which is a small number.

That's non-linear least squares. I hope we have a problem on that, but I have this weird feeling we didn't. It just didn't make it? That's horrible. Fortunately, we have some recourse, don't we? It seems that over the next week, you might not see non-linear least squares. It's an important topic, and really one that should be covered in the first half of the course, and I mean covered like I think you should do one. Let's look at the next topic. I'll just say a little bit about the beginning of the topic. It's actually the dual of least squares. You'll get used to this idea of duality.

I don't think we'll ever get very formal about it, but at least I'll give you the rough idea. Let me say a little bit about duality. It's going to involve ideas like this. It's going to involve transposes, so there are going to be transposes. Rows are going to become columns and things like that. Null spaces are going to become ranges and that's the kind of thing. By the way, there's a duality between control and design and things like that and estimation, because there you're switching the roles of X and Y, typically. These are all sort of the ideas.

We've done least squares so far. The dual of that or a dual of that is going to be least norm solutions [inaudible], because we've so far been looking at over determined equations, and we'll see what this is. This is actually pretty straightforward stuff. Now we're going to take Y equals AX, but A is fat now. That means M – you have fewer equations than you have variables. You have more variables than equations. Another way to say this is X is underspecified. So even if there is one solution here, there's going to be lots, because you can have anything in null space of A, which, by the way, has to be more than just the zero element now because it's got a dimension at least N minus M here.

We'll assume A is full rank, so that means that you have M equations and they're actually independent equations, so the rows of A are independent. Then all solutions look like this. The set of all Xs to satisfy AX equals Y is you find any particular solution here. You will very shortly see a particular solution. You take any particular solution and you add anything in the null space. By the way, if a person chooses a different XP here, you get the same set, because the difference of any two solutions is in the null space, and you get the same thing.

Here in this description, Z essentially parameterizes the available choices in the solution of Y equals AX, and you can say roughly that the dimension of the null space of A, which is N minus M, because A is full rank, that gives you the degrees of freedom. That says you can choose X to satisfy other specs or optimize among solutions. I guess as we talked about before, as to whether or not that's a good or bad thing, that depends on the problem. If this is an estimation problem, degrees of freedom are not good.

Basically, that's stuff you don't know and cannot know with the measurements you have. If this is a design problem, this is good, because it means you can do exactly what you want many ways, and therefore you can choose a way that's to your liking. That's the idea. I might as well just show this and then we'll continue next time. Here is a particular solution. It's A transpose times AA transpose inverse times Y. It should look very familiar but be a little bit off.

You are used to A transpose A inverse A transpose Y. You're used to that formula, and this looks very different. It's a rearrangement. You move a few things around. It looks perfectly fine. You have to be very, very careful here, because it's very easy to write down things like that and that. What you must do, and I'll show you my pneumonic. My pneumonic is this. You look at that and you look at that and you quickly do the syntax check. The syntax scan goes like this. If you saw A inverse, your syntax alarm would go off unless you know A is square. But here, A transpose A, that's square, and therefore at least by syntax can be passed to the inversion function.

That's cool. In fact, you can multiply it by A transpose on the right. You can also form AA transpose, and that's invertible, too, so syntactically, this one is cool, too. Both of these pass the syntax test regardless of the size of A, fat or skinny. Now let's get to the semantics test. Now, if A is fat, then this one is basically a fat times a tall matrix, and the result is you get a little one. Over here, if A is skinny, this is also a fat times a tall, and the result is a little one.

So here is the semantic pass. The semantic pass says if you propose to invert a fat times a tall matrix, unless there's something else going on like some rank condition, you're not in trouble yet. If these two reverse, you're in trouble independent of what the entries of A are, because if you reverse these – if A is fat and you go to this formula, that is a square matrix. That's fine by syntax. It fails on semantics because A transpose A is going to be square, but it is low rank, and you should not invert low rank matrices. We'll quit here and continue next week.

[End of Audio]

Duration: 78 minutes

IntroToLinearDynamicalSystems-Lecture09

Instructor (Stephen Boyd):Our main screen is not working today. I'll — for the first ten minutes while they're desperately trying to get our big screen up and running, I'll say some things about the mid-term to give them as much time as possible.

You can go down to the pad here. Let me say a couple things about it. You're welcome to move to a seat where the monitor is more visible or something like that. There's probably plenty back there. If you can go down to the pad, I could make a couple of announcements.

I mean, the first thing is I'm — well, I'm sure there is no one who doesn't know that the mid-term is coming up. In fact, there's — it's even possible we're gonna have an alpha tester take it tonight, which will be interesting. So it's coming along very well. It's of course this — end of this week. Let me say what it covers just to remind you. It covers group homework four, that's the one you're working on now, and that will be printed on Thursday, and it will include through lecture eight, that's today's lecture. In fact, we're gonna finish lecture eight probably before the class is over. So we'll finish lecture eight and that — and it'll cover all material in all lecture up to there including even materials that we accidentally forgot to exercise you on in the homework. So there were some glaring omissions. That was just our fault but we'll — we still — it's still valid material — it's fair game for the mid-term. Okay. Let's see. I'm gonna hold extra office hours from Thursday 1:00 to 3:00. I could do it also today from 1:00 to 3:00 if anyone was gonna come by or something like that. It's not an — oh, a hand went up. Okay. Sure, I'll do it today too. Why not? There we go. So I'll be around both today and on Thursday from 1:00 to 3:00. Watch, I probably have some meetings scheduled today but we'll see. If — maybe I'll be there. No, I'll probably be there. Let's see. I was — those who are taking the course remotely via S.C.P.D., we would strongly encourage you if you're local to come and pick up the exam like everyone else and drop it off. That's what we'd really prefer to do. If, however, that's inconvenient or something like that, we will send you a PDF of the exam, but please send email to the T.A.s to let them know — or, sorry, well, to the staff address to let us know when you would like to take the exam so that we can do that. Don't just sit there, wherever you are, waiting for it to arrive. So — and make sure you get a response from us saying, "Acknowledged. We're sending you the exam on this time — at this date at this time." Let's see. Homework 4 — we've posted homework three solutions last night. We'll post homework four solutions; those are the ones you're working on now. And what we'll do is this. We'll post those Thursday evening. So Thursday you'll hand in homework four. We'll post homework four solutions. Now in the past, we've always let a few people with generally speaking very, very good excuses, such as joining the class late or whatever, turn in a homework a bit late. Unfortunately, we won't be able to do that for homework four. So homework four, you hand them in, within hours we're gonna post the solutions. That's the — yeah, Thursday evening. Okay. I don't know if anyone is — we did post last years mid-term just so that you get to see what a mid-term looks like. I think as part of that you found out where homework problems come from or how homework problems are born. They're born general — often as mid-term and final exam problems. I also have a question for you. And the question is, when should we post the solutions for last year's midterm?

Student: Now.

Student: Now.

Instructor (Stephen Boyd):Now, okay. I bel — this — it includes, like, one or two problems on homework four, right? Something — or maybe one — is it just gonna have one on it? It — one overlaps? Okay, fine, no problem. We'll post it now. Great. So that's fine. So that means that — that suggests that people have actually looked at it.

Student:[Inaudible].

Instructor (Stephen Boyd):Good. Well, we know some people have looked at it because we got request to post the M files required to do it. So that's — well, that's not absolute proof that people looked at it but it's — we'll take it as a good sign.

Okay. Any more questions?

Student: What time will the exam start? Instructor:

I think we've posted that on the website. The question was, "when will the exam start?" It will start — I think it's bet — I think you pick it up between 5:00 and 5:30 or something like that; is that right? But, again, you should never trust me. You should trust the website.

Oh, I do want to say thank you. We got several — we rearranged the website a little bit last week. And I guess I was caught in the — I was in the middle of doing 50 things and didn't come back and messed a few things up. And actually, we're very happy that — people that — people caught my mistakes very quickly and fixed them. So thank you for those of you — so if you ever find anything that's off on the website like a missing link or something like that, please do let us know because often it's just because, well, we messed up. So thanks to those who corrected that last week.

Okay. Any other questions? If not, we'll continue our discussion of least norm solution. Now, there — come on, there's no way anybody can read that. Can you actually read that? No, okay. So you could — there's a couple things you could so. You could move close — every — if you can't read, you can move closer to a monitor or you can extract just enough information out of it — out of this little TV to get a rough idea of where I am actually in the notes. That's your other method, and do it sort of a correspondence. But, anyway, your choice. But you're free also to just move somewhere where you can read it. So — yeah, you can either crowd up here or in back at one of those. I guess they're working on trying to get the big screen routed. Okay.

So least norm solution. As I said last time, this is something like the dual of least squares approximate solution. So in least norm solution we're studying the equation AX=Y. But in this case, A is fat. And we're assuming it's full rank, so that means you have M equations that can strain a variable X. But you have fewer equations and unknowns, so it means you have extra degrees of freedom. What that means is that AX=Y actually has lots of solutions. There are lots of solutions. It means the null space of A is more than just a zero vector. In fact, it's exactly N minus M dimensional, the null space. So there's a lot of freedom in choosing X. So one particular X that satisfies AX=Y is the vector of least norm. So that's the least norm solution and that's XLN and it has the — it's just given by the following formula, A transpose AA transpose inverse Y. So that's the least norm solution. It's easy to see it's a solution because if you multiply this by A, you get AA transpose times A transpose inverse times Y, and the transpose and the other one, they annihilate each other and you get Y. So you get a solution that's clear.

This relies on the fact that if A is fat and full rank, AA transpose is invertible. That's a basic fact. And actually, what you can show now easily using QR factorization. And in fact, for all practical purposes, we're gonna do that ourselves in a few minutes.

Okay. So this is a least norm solution. It's a solution. Now, watch out because the least squares — I mean, the main thing you want to do with this material is make sure that — although it looks very similar to the least squares approximate solution. Formulas look the same. A lo — everything looks similar. But be careful to sort out in your mind, which is which just because they look so dangerously close. So this X least norm is actually a solution of AX=Y, whereas in general XLS, which is A transpose A quantity A inverse times A transpose Y, and that formula is only for a skinny full rank matrix A. In that case, that's generally not a solution of AX=Y. It is the X that minimized essentially the hit distance or the error or the residual so — and is generally not a solution of AX=Y, whereas here this one certainly is. Okay.

So this point, X least norm, essentially solves this optimization problem. It says among the vectors that satisfy AX=B — I don't know where the B came in but AX=Y. You should minim — among those, you should min — take the one of minimum norm and that's this optimization problem. The solution is unique and it is given by X least norm. Now, we can show this directly by direct argument — that's easy. Let's let X be any other solution of AX=Y. Well, then AX minus X least norm is zero because AX is Y and so is AXLN. They're both Y, so you subtract them and get zero. And now let's calculate the inner product of X minus X least norm and X least norm. Well, you just — simply just plug this in and do some matrix manipulations here. Here you have this thing transposed times A transpose. But the product of two transpose is the same of the product in reverse order quantity transposed. So I write it this way. Now, this is actually — this is gonna be zero because AX minus AXLN is zero. And so actually, the right-hand side doesn't even matter. This vector is zero, so that's zero. That says that the X minus X least norm and X least norm are perpendicular. Now, when two vectors are perpendicular, it means that you — if you want to calculate the norm squared of the sum, it's very simple. It's the sum of the norm squared of the individual components. So some people call that Pythagor — the generalized Pythagoras theorem or something. Anyway, it's nothing.

You write out the formula for the norm squared of a sum and the cross term goes away. So it says that — if we write out X as — in a strange way, X least norm plus X minus X least norm, no one could argue with that. But this thing and this are orthogonal, and therefore the norm squared of the sum is the sum of the squares of their norms squared separately. So you get this thing plus that. Well, that says this thing, of course, is going to be non-negative. And you can see immediately that the norm squared of X is bigger than the norm squared of X least norm. And that tells you this, since X was any solution of Y, that tells you that any solution of Y is gonna have a norm at least as big as X least norm. And this is the proof now that X least norm, in fact, minimized the norm among all solutions of AX=Y. So that's just sort of a direct argument. And the geometry is pretty easy to see.

The set — you consider a set of vectors that satisfy AX=Y. Now, I mean, this is silly because it's an R2 and here this is a one-dimensional set, it's an affine set. In general, it's just an affine set here. In fact, with a dimension which is N minus M in gen — in the general case here. And so you can imagine that as a plane or something if this is an R3 with a — actually just one equation. It's a plane. And then you're asked to find the one of least norm. That's the point on that plane or hyper plane or affine set which is closest to the origin. It's the one of least norm. And that's this one here. And you can see if you shift this, you get the null space of A. That's — that actually gives you the part that's sort of the — it's the parallel part of AX=Y. It's shifted to the origin. And you can see, in fact, just visually here that X least norm is actually gonna be orthogonal to the null space of A, and that's this orthogonally condition. And of course, you can have a projection interpretation. X least norm is the projection of the point zero on the solution set of AX=Y. So that's it. Okay.

Now, this is a — this formula, A transpose A transpose inverse that's the — that's also the pseudo-inverse. But this is the pseudo-inverse of a full rank fat A. So far the symbol, dagger, I guess has two overloadings. It's overloaded and it applies in two contexts. A dagger applies when the matrix A is skinny and full rank, in which case a dagger means A transpose A inverse A transpose and it's associated with least squares approximate solutions. You also have now an interpretation of a dagger or a definition of a dagger when A is fat and full rank, in which case it's A transpose times AA transpose inverse. And it's actually something that gives you the least norm salutation. So that's a dagger. By the way, in about three weeks we will complete the overloading of dagger. I think the machine just turned all the way — okay, gonna reboot it. Or some — or does that mean you're giving up? Okay. No, sounds like it's — yeah, it's reboot minus H, that's hard. Okay. Okay. So we — in a couple of weeks we're gonna complete our overloading of A dagger and we're actually gonna assign a meaning to A dagger, to any matrix except the zero matrix. So all non-zero matrices will actually have a pseudo-inverse. Only zero will not.

Hey, great. So — and, yeah, great. Thank you. Okay. Great, all right. Okay. So we'll get to that. But for the moment, the only contexts in which you know about the pseudo-inverse are full rank matrices. So all full rank matrices have a pseudo-inverse. They have different formulas that apply in different contexts. That's what overloading means. Okay.

Now, this matrix, A transpose A transpose inverse, that's a right inverse of A, we know that. I minus A transpose AA transpose inverse A gives a projection onto the null space of A. By the way, this matrix alone gives projection onto the null space of A for this thing, the orthogonal complement. Okay. So this is A transpose AA transpose inverse A. Okay. Now, the same formulas for a full rank skinny matrix are not the same. The analogous formulas are something like this, A dagger or the pseudo-inverse, or I guess in the U.K. the Moore-Penrose inverse, is A transpose A inverse A transpose. And that's a left inverse and interestingly in this case it's A times A transpose A inverse A is projection on range of A. So the anomaly you see is the 'I minus' here. That's the anomaly essentially, so that's it. Okay. So do watch out for these. I always check. My mnemonic is real simple. If you see this — let me see if I can do it right. I'll try to draw it right. If you see this, everything is cool. You know what I mean by that? So skinny times fat inverse — well, sorry, it's not cool but it's not obviously uncool. Okay. This is always trouble. See that? That is never cool ever. Okay. So just — oh, and by the way if I — if — of course if these multiplied out and became non-square that's super uncool because that's a syntax error. Okay. So my mnemonic is this — and you might ask, really? You mean I actually — when I'm working and doing stuff I actually — yes, I do. So I draw this picture. I don't let anyone see it, you know, because it's embarrassing a little bit. But this is what I do, okay. That's cool. That is totally uncool. Not totally. Totally uncool is this times — see if I can get it right. There we go. See that? That's uncoo — that's really uncool. Okay. By the way, I think now you should be able to read the little note on the web — on the course website that's called Crimes Against Matrices, so you should just read it. Should make sense. Okay.

Well, let's see how the solution connects to QR factorization. It does. A is skinny and full rank therefore A transpose is — sorry, A is fat and full rank, therefore A transpose is skinny and full rank. And that means that when you write out — when you do the QR factorization of a skinny full rank matrix, here is what it looks like. You're gonna have A is — you get Q and then you get R. But in — but R not is invertible. R is square and it's invertible in this case. Okay. So it's — well, it's invertible. Okay. So it's non-singular, R. And it turns out, you work out the formulas. You just plug in QR for A transpose, so A is R transpose Q transpose and you just plug in the formulas and let things — I mean, carefully. So you should do this yourself. I'm not gonna do it now. You should just do this carefully. Carefully let things cancel watching out for the usual things. Like Q transpose Q, that's I, but QQ transpose is not. So just ca — when you do this carefully, you find out not surprising that this A dagger works out to be nothing but QR minus transpose or R inverse transpose, like that. So that's what it works out to be. And I forget what the formula is for the least squares one, but it's very similar and it's just kinda got — maybe it's — I don't know. Does anyone remember? It's — maybe it's R inverse Q transpose. It's something like this. So this is the — from a few lectures ago. Is it this? That — this is in the context of least squares. Is that it? You have the notes there. Is that right? Yeah, so [inaudible] close. Okay. So — and, you know, after a while you're gonna get used to these things where these things look similar, but the order is different and some things are transposed and all that sort of stuff. So you're gonna — so it's why you have to be careful. Okay. Oh, and the norm of the least norm solution is in fact the norm of the inverse — it's simply the norm of R minus transpose Y. So that gives you, in fact, the norm. Okay. So that's the idea.

Okay. Now, I want to now talk about — essentially — actually, want we want to do is do the parent of all of these, is go up in abstraction to the parent of both least norm and least squares. Because it's actually quite — it's useful to know because they're both — they're obviously relat — deeply related. Let's see how they're related.

Well, the least norm — we'll start by handling the least norm problem and solving it in a more conventional way. If you want to minimize X transpose X, that's of course the norm squared subject to AX=Y, the standard method I guess in — I guess since the early 19th century, actually earlier than that is to do the following. You take the objective and to that you add a Lagrange multipliers times the constraint. So here is a vector of constraints and we take a vector multiplier lambda. By the way, I don't mean for this to be obvious about how all these Lagrange multiplies work. To tell you the truth, I never understood it myself. In fact, it's generally taught as a behave — a set of behaviors, right, that a monkey can do. I guess it's generally taught, like, in high school. No one has a clue what it means, what the pictures are or anything. Is that correct? Does anyone here actually — did anyone, like, draw pictures of this that anyone understood? Actually, how many people have seen, like, Lagrange multipliers for constrained optimization? So how many was it taught absolutely simply a set of behaviors, this is what you do. Wait, does that mean that the rest of you actually understand it? No, it's possible. Maybe things have changed since I was subjected to this. It's possible. Okay. All right. Anyway, I don't mind saying, I never understood it until, well, a while ago. But I certainly didn't understand it for a while. So here I'm not going to go into it. I'm not gonna go into it. I'm just — we're just gonna say, here's how Lagrange multiplies — here's what you do. So here's what you do. You form this Lagrangian like this, and then the optimality conditions are that the gradient of this with respect to both X and also with respect to lambda should vanish. If I take the gradient of this with respect to lambda, I get AX minus Y and I find that should vanish. Well, that was really super duper useful because it tells me that the optimal solution must satisfy AX=Y. Well, I knew that because that was a constraint. Okay. So this was not exactly informative. Over here, though, it's actually very interesting. If I take the gradient with respect to X, I find out that it's 2X, that's the gradient of this. And that's why, by the way, a lot of people will just put in a ½ here just to clear the twos out of formulas and things. That — so you'll see that. You get the gradient of that and the gradient of this thing with respect to X is actually A transpose lambda. So we get 2X + A transpose lambda is zero. Well, that's interesting. So you solve that. And it says that X is -1/2A transpose lambda. Let's take this and plug it into this, which was hardly a revelation, AX=Y and you get a formula for lambda. So lambda is -2AA transpose inverse Y. Now, I take this lambda and I plug it right in there and I have my final solution which is this. So we've re-derived by a mysterious method the same thing we derived by a direct algebra three pages ago. Okay.

So this is just to do this because we're gonna use Lagrange multipliers to look at the general case. So let's do some examples of least norm. This is a stupid and silly one but it's, you know, just — that's a good way to start. So we go back to our mass and we're gonna apply forces on it for ten one second periods consecutively. And we're interested in the position at the end of ten seconds and the velocity. So you have Y=AX where A is 2 by 10 and A is fat. And I think you even should remember some of the entries in A. I think the top row of A that — the entries are shrinking as you go along it and the bottom one, they're all ones or something like that. Okay. And we're gonna find the least norm force that transfers the mass unit distance with zero final velocity. So it's got to take the mass. It's got to accelerate it, and then it's got to decelerate it over here. Although, we leave open the possibility that the right thing to do would be to take the mass and move in the other direction and then — I mean, that doesn't sound too plausible. It's actually not the case but anyway we're leaving that open. We don't require it to simply move the — although it does. Okay. Now, when you work out the solution — in fact, this one has an analytic solution and it's really — it's — when you work it out, it turns out you should apply a force that's is aff — that's an affine function of the time or of the discrete time. So basically you should push it on the first instance and the first second you push it hard, less hard, less hard, right at T equals — right around T equals five, you switch — or sorry, right around T equals five you switch from pushing it very neatly, so this is basically up to — for five seconds you accelerate the mass, although you push — you would — you push less hard later. And we can make — I mean, we can anthropomorphize this easily. Why is the least norm solution doing this? Why would you push harder at first than later? What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):Just a vague — this is gonna be a hand waving answer but you just need a vague one. Why would you push harder at first? Why shouldn't it just be like this? Why shouldn't you just push hard and then ex — and then pull?

Student:[Inaudible].

Instructor (Stephen Boyd):That's it. That's it exactly. Okay. So it is more efficient in terms of meters per Newton to push early on. That's what it is. So this weights — this weights the force with the efficiency. So you're pushing harder at first because you get more meters per Newton of push at the beginning, okay. And then it's symmetrical so you — the — you accelerate and you decelerate like that and that's the picture. Okay.

Let me ask you a couple — as long as we're on this one topic, I'm gonna ask you a couple of other questions just for fun. I think once before I admitted publicly that least squares type objectives, and in particular the sum of the XI squared here — the sum of the forces squared here generally speaking, actually are of no particular practical relevance. It's generally not what you want to do, right? So thrusters don't come with a box on the label or a tag hanging off the side that says, "no matter what you do, do not apply a signal whose sum of squares is more than this." They don't come that way. Okay. So what they — the way they really come is they have things like this, there's a maximum force you can apply or there's an amount of fuel you use. Now, by the way, these have names. The — this is just for fun. All right. But just to give — just to let you know a little bit about this. The infinity norm — I think we encountered this once. This is

— it's the maximum of the absolute value. So in fact the way you would say this, for example, in electrical engineering is it's the peak of the vector. It's the peak of the — if that is a signal, that's the peak of the signal. And that's an absolute value. That's a norm and it's also the one norm, which is the sum of the absolute values. Now, this one here tells you how — essentially how big a thruster you actually need to apply the forces. This norm actually is a very good first order approximation. For example, if you really were using thrusters to position this mass, this would be something related to fuel use because that's generally how it works. Fuel use is generally proportional to the force that you apply. Okay. You can have more complicated things but for a thruster, that's a pretty good approximation. Okay. Now, these are both norms like — by the way, our good old friend the Euclidian norm, in this context inherits a two at the bottom so that you can distinguish it. These are norms. These are all three norms. They all three measure how big a force program is. This one measures it by the peak, this measurers it by the — essentially the sum of the apostolate values which you can think of as fuel usage. This measures it by the sum of the squares which we often say is energy and that's mostly to hide the fact that in fact we don't really care about this. It's just — this is what's easy to do mathematically. Okay. That's the real reason.

Now, I have a question for you. I would like to know the following. What do you think — supposed I asked — instead of minimizing this over moving a mass one meter, I'd like to know, what happens if you minimize the maximum, and I want you just to guess. What do you think is the optimal thing to do? What's the minimum? So you can — we can call this the gentlest transfer because I'm applying the smallest maximum force to the mass. So this you could call the minimum energy transfer. That's what we just worked out here. And I want to know, what's the gentlest transfer?

Student:[Inaudible].

Instructor (Stephen Boyd):Exactly. So the minimum — I don't know the level but it's whatever it has to be. It's gonna be this. You're gonna apply a force, a constant force up to five. You're gonna constantly accelerate until five seconds at which point you will decelerate like that, with the exact — with the same force. Okay. But there's a name for this. This is very famous. It's called bang-bang control for obvious reasons. It's always up at the limit each time. And let me ask you this. You all use disk drives constantly and those are — in a disk drive, what happens is the little thing is sitting there, track 23, and a signal — a command comes in to seek track 125 and you have to move it there. Okay. The — I got news for you. That's this problem, okay? And you have to do it, by the way, in a handful of milliseconds. Once you get there, you have to get rid of all the shaking and stuff like that. You have to be tracking something within microns or less. This is serious stuff. Okay. What do you think the current signal in a disk head drive positioning system looks like? Does it look like this or does it look more like that? I'm just — just guess. What's that? Yeah, the answer is, it looks much more like this. Actually, it's not sharp like that. It's actually got a little bit of a rounded thing there because it's a little bit more complicated, and it's taking into account all sorts of other vibration modes and stuff like that. But basically it looks like that. Why? Because the amplifier will source or sink a maximum amount of current and the goal is to seek as fast as possible. And — so you don’t — you’re not — your goal is not to minimize the sum of the squares of the currents in your thing. By the way, if you’re worried about power, the power is closer to this in a disk drive so — okay.

Now, let me ask you this. How about this one? What if I asked you — so we worked out what the gentlest — well, I don’t know if you’d call that gentle. But the gentlest in terms of the maximum force you ever apply on the mass transfer is this — what about the most fuel-efficient? Again, just go ahead and take a guess. People in aero-astro could probably guess this. If you’ve studied satellite — if you’ve actually studied how satellites are, for example, moved back on orbit then you might know — any other — any guesses? You have a guess. What’s your guess?

Student:[Inaudible].

Instructor (Stephen Boyd):What’s that?

Student:[Inaudible].

Instructor (Stephen Boyd):You got it. So the optimal here is a giant force there. And — oh, that’s not right. There. So the optimal — the X that minimizes the sum, which is — which would be something like the fuel use, is gonna be this. It’s an impulse. Well, I mean, this is silly. It’s not an impulse. It lasts for a second. You do a fuel burn at the beginning, and then what that does is it just accelerates the mass. And then this is actually called the ballistic phase in the middle. Ballistic means it’s just moving with no forces on it other than gravity of what — in this case there is no gravity. So it’s just floating along. And then right in the last second, you apply a counteracting braking force. And this minimizes the fuel. Okay. And by the way, you’ll see this if you actually look at a satellite or something like that positioning itself. You’ll see little puffs come out. You’ll see, like, little puff, puff, puffs come out one side and then a little bit on the other side and stuff like that. That’s exactly this so — I can tell you have absolutely no idea what I’m talking about but that’s fine. Okay.

So all of this was an aside, just to say — or if you want to learn about these things, then you’d learn about this stuff in 364, which is probably not exactly the top thing on your mind at this moment. But that’s where they — so it turns out that you can actually solve these things not with analytical formulas but it’s totally straightforward to actually work out these things. Okay. Any more questions about this? Okay.

So the next thing I want to do is connect — is make some connections between regularized least squares — actually connect least squares and least norm solutions. And the way they connect is this. Suppose we have a fat full rank matrix. Let’s imagine now a two objective problem and it looks like this. J1 is AX-Y norm squared and J2 is norm X squared. Well, the least norm solution basically requires that you be a solution so it requires AX=Y so it says, plea — it says minimize J1 absolutely to the limit and you get — and it minimizes J2. So in a tradeoff plot that’s one of — the least norm solution is one point on the tradeoff curve between these two. The other point, by the way, is X equals zero, which is not very interesting but still, it's the other point. Okay. Now, let's imagine doing this. Let's take a weighted sum objective which is J1 + ?J2 like this and let's minimize it. That's A — this is AX-Y norm squared + ? norm X and we're gonna let — the solution to that is A transpose A+BY. Now, what — by the way, when A is fat and someone writes A transpose A, your — first of all, your height — heart rate should increase slightly. You should start breathing sort of shallow breaths and things like that and why is that? If you have a fat matrix and someone writes, "A transpose A," your vocal cords should get ready to cry out in protest. Your autonomic response should be triggered. What am I talking about? Do you know what I'm talking about? Yeah, good. Okay. That's all. Okay. You should — because when someone takes — writes — has a fat matrix and writes — yeah, is that right? Yes. Then this is actually — this is the product that passes the syntax scan but is — you're just waiting. Especially if you see that left bracket there, that's when you should be tot — you're like — you should be like, [Makes Noise], like that. But everything is fine here because of this. Okay. So that's all I'm saying. Okay. So this is actually cool, although it's very close to something that's not cool. And it's only cool if ? is positive. It's really not cool if ? is zero here. I mean, really not. Okay. So now what happens is we're gonna let ? go to zero. That says I care less and less about the size of X. Now, when ? is zero, I actually know how to — when ? is zero — if someone just walks up to you and says, "please minimize J1," actually someone can hand you back legally any solution of AX=Y. So if someone hands you back two solutions of AX=Y and the specs actually only call for minimizing J1, that's absolutely valid. Because someone says, "Boy, that's crazy. Someone else gave me this solution of AX=Y where X is much smaller." And you go, "Sorry. I checked the specs. I didn't see any mention of the norm of X." So minimizing just J1, there are lots of solutions and, in fact, any solution of AX=Y does the trick, big, small or otherwise. The minute you put in ? here — for example, even if it's 10 to the -8, now there's a difference between the two. So if you now find a solution of AX=Y with a big norm X, you're gonna pay slightly more — and therefore, as long as ? is positive, it's gonna come up — it's gonna show up in the composite objective. So what that tells us is that as ? goes to zero, X? should go to X least norm. And, in fact, that's exactly what happens. Now, you want to be super careful here because as ? goes to zero, this matrix becomes singular. So you want — that's — you want to be very careful. That's essentially a denominator going to zero. That's what it is. So you're gonna have to be very, very careful here. And it turns out, it's not that hard to show. It turns out that for a full rank fat matrix A, it turns out that A transpose A + ? on inverse, A transpose goes to A transpose AA transpose inverse. So it actually converges to that. And it's not too hard but it's a little bit tricky in the sense that you don't simply plug in ? equals zero. Because if you plug in ? equals zero, the left-hand formula doesn't even make sense because you're inverting something which is not invertible, okay? Nevertheless, it's — this is the case, so okay. So that's the connection between those two. That explains one of the points on those trade off curves. And now we're gonna go to the parent of both least squares and least norm because it's not bad to know it. So here is the common parent. The common parent is minimize the norm AX-B subject to CX=B. So minimize a normal — a general norm of an affine function subject to a linear equality constraint. So that's the parent of both of them. And let's see. So in this problem how would I reconstruct, for example — well, least squares, it's just you forget the objectives. You just — sorry, you forget the constraint. How do I make this into least norm? What would I choose to make this a least norm problem? This thing. I'd take A=I and B=0. If I take A=I and B=0, that's a general least norm problem because I'm minimizing then just norm X subject to some linear equations. Okay. So how do we solve this? Well, as usual we square the norm because minimizing the norm is the same as minimizing the square. And when you minimize the square, it's nice because we have a nice formula for the square in terms of inner products. Then that ½ goes in front. Why? Because it makes all the formulas prettier because we're gonna differentiate, basically, a square and we didn't want the two polluting all our formulas so this is what we do. You form a Lagrangian now. That's the objective plus ? transpose times CX-D. That's this Lagrangian. And then we rewrite — we expand everything out and then it looks like that. So this term is from — that first term there, that cross term is from here. This term is the third term from here and then these are the two terms there. Now, one of these — that's the gradient with respect to ? being zero just recovers our equality constraints. It's not interesting. The other one says that the gradient with respect to X of the Lagrangian, that's A transpose AX minus A [inaudible] is zero. That's actually a real equation right there. Now, you can actually solve all of these equations. I'm gonna do it on the next page but it's not pretty. And it turns out there's a better way to do this. It's to write it as an equa — a joint equation in both X and ?. So we're gonna do that. This top equation is AA transpose times X plus C transpose times ?. That's this term and this term. Equals — and then this goes over to the right-hand side and you get A transpose B. This equation, CX-D=0, well, that's really just the constraint. I write that down here this way as C times X plus 0 times ? equals D. So you get this equation here. That's a square matrix, but it's a very famous matrix that comes up in lots and lots of contexts all over the place. It comes up in, like, economics and, oh, tons of areas. It — I mean, this form of matrix. Now, if this matrix is invertible, we get the solution immediately and that's this. It's X and ?. So both the optimal X and the optimal ? are — you get them at — simultaneously and it's given by simply — well, obviously it's the inverse of this matrix times that. Okay. And now, I actually strongly recommend this is — that this is the one you should keep in mind. It's the right one. By the way, some people call this a primal dual formulation, and I can say why. X is thought of as a primal variable here and this Lagrange multiplier is a dual variable. And so in this formulation, you're really jointly finding both the primal and the dual variables. I mean, that doesn't matter but I'm just saying that's what this is. Now, this will recover all of our forms. So this is the common parent of both least squares and least norm. And you can recover all of our formulas. So for example, if A transpose A is invertible, that means, of course, that A has to be skinny and full rank. Then you can get a — you can actually block solve these equations here or you can just block solve these equations. So what you do is if AA transpose is invertible, I multiply this equation by AA transpose and I get X equals, you know, AA transpose inverse A transpose B and so on. That's here. You get this formula for X in terms of ?. Now, this form — now, you take this X and you plug it back into CX=D and you get this equation. And now you can get ?. ? is this. It's CA transpose A inverse C transpose inverse times this thing. And now finally you go back to this formula. It gives you X in terms of ? and you get that. So actually, really, it's your choice. You can remember this one here or that. So it's really your choice. I mean, of course they're the same thing. This is just working out in detail what solving a block two by two system gives you. Okay. So this is the picture. You can check, by the way, if you go back to the original parent problem here.

You can check. It recovers everything, absolutely everything. So for example, if A is I and B is 0, you can go down here and plug this into the horrible formulas here — down here. If you have B is 0, a lot of things simplify, right? That goes away, that goes away. If A is I, all these things that say — they all go away. And I think — yeah, sure, it looks like — except I'm seeing a — no, I'm not seeing a minus sign. There's a minus here and a minus there that cancel each other and you're recovering it. So it does kinda recover all the equations. This is useful. I think we made a terrible mistake and didn't assign any homework problems that required this. Is that true? I think it's true that we failed to assign any homework problems that use this. But we just kept to least norm and least squares type things. But you should know this. Okay. So that finishes up all the material that will be on the mid-term. And it finishes up in fact the first, I don't know, 40 percent of the course or something like that. So that finishes up a whole block. I'm gonna start the next material because we're actually in a very good position. Sometimes we don't finish the material until, like, Thursday.

Student:[Inaudible] least squares problem?

Instructor (Stephen Boyd):Oh, how do you recover the least squares problem? Well, there's actually a couple of ways to do it. So the simplest way is to just not have C there. And I believe this will actually — it will actually work in that case. So you make C an empty matrix, whatever that is. So, yeah, it works. Look. If I just pretend C is — I actually can't pretend C is zero. That actually won't work because this matrix won't be invertible because it will have rows down here that are all zero. So what we have to do is C is null, so it's not even there. If C is not here, you do get this, right? You get this thing inverse times A transpose B and it looks good to me. I mean, it's not totally straightforward but that's the right thing to do when C is null as opposed to being zero. Are you buying that? No, you're not. What part of it are you not buying?

Student:[Inaudible].

Instructor (Stephen Boyd):Sorry. How does what? Oh, you mean up above? Oh, yeah, that's easy. Let's go back to that. Oh, how did — I've lost it. There it is, okay. So here if you want to make this least — the least squares problem all we do is we eliminate that. That's least squares. Okay. Now are you buying my other one? Okay. Good, great. Any other questions about this material?

Student: Just one more question.

Instructor (Stephen Boyd):Yep.

Student: Isn't there a diagram about [inaudible]?

Instructor (Stephen Boyd):Yep, that's the — back here somewhere. I'll find it. I've lost it. Here it is. There you go.

Student:[Inaudible]. Instructor:

It was what?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, sorry. Did you mean this for the mass? No. Oh, do you mean the geometric picture? Okay. I'll draw it again because it's gonna be faster than my finding it. Okay. So here is — you know, the pictures are somewhat unexciting, right, because they're generally in R2. So here's a set of X such that AX=BY, I guess we use here. Like that. Okay. That's all these points satisfy AX=Y. I mean, this is silly because A is actually — AA is A transpose and there is A. Okay. So that's what it looks like. The least norm solution is the — so any point on here satisfies AX=Y. This point right here is the point of closest approach to the origin. That point actually has least norm. And that would be — this point would be X least norm for this problem.

Student: And how did you get the null space of A?

Instructor (Stephen Boyd):Oh, and how did I get the null space of A? Well, the null space of A in this case is this. And I can do that several ways. A is A transpose. A is a row vector here. And A is a — is the normal of this hyper plane. So all the thing — if you look at all the points that are orthogonal to A, it's this line right here, okay. Now, there's another way to see it. This in — this is the solution set of AX=Y. And the point there is that the difference of any — if you ask — if someone comes up with — one person has an A, and another person has an X and they both satisfy X=Y, the one thing you can be absolutely sure of is that the difference is in the null space. And, in fact, that's if and only if. You know, so in other words if one person has a solution and it has an element of null space, you add it, you get a new solution.

So what that said, that sort of makes sense here because it says that when you're moving in this direction, you're really moving in the null space. And so that's another way to understand why this — why the null space would be the same thing but translated to the origin. Okay. So my claim is you know quite a lot now. And it's not that much math in it, but it's not trivial. You know a fair amount. And these methods — maybe you're convinced, maybe not. These — you can already do serious things. You can do all sorts of stuff that you could not do by some heuristic or hacking method. Just with the least norm, least squares, throw in a little regularization, a little multi objective, throw in a smoothing parameter, you'd be surprised what you could do. That's you, of course, and computers and high quality open source software, I might add. Because you can't do a whole lot — people did least squares before they had computers. It was not pretty. Okay. It was basically you would do these things with a calculator — I mean, with a mechanical calculator, and that's if you're really lucky if you had the mechanical calculator. So it was done. It's a lot easier now. You should be glad you weren't born 80 years go, something like that, longer, a hundred. Okay.

If there's no more questions about that, we'll move on and actually cover just kinda some of the boring stupid stuff for the next topic which is autonomous linear dynamical systems. So if you can go to — which is I guess what the class is nominally about so we got to it finally. Okay. So what we'll do is I'll just go over some of the nomenclature. I'll talk about some of the basic ideas and get that over with.

So autonomous means that it goes by itself and that means, in fact, that there's no input here. So what we're missing from the general formulization is this — that's just gone for a while. So we'll first understand just what happens if you have Xdot=AX. It looks very simple. It's a first order vector differential equation. And we should probably just as a warm-up, answer the following question. If A is one by one — would you say if X is scalar, let's get this out right now. What's the solution of Xdot=AX in that case? Well, it's an exponential, right? It's something like this. It's X of T equals E to the TA X of 0. Something like that. No, no, it's not something like — it is that. Okay. That's the solution when A is lower case, which is to say it's a number. Okay. So you can expect something like this to come up. By the way, the qualitative behaviors of the scalar differential equations are kind of boring. Let's talk about them now. If A is 0, Xdot is — X is a constant. It just says Xdot is 0 so X is a constant. If A is positive, this — you get a growing exponential. And if it's negative, you get a shrinking exponential, okay? So that's it. That's my discussion of Xdot=AX where A — where X is scalar, okay? There's basically three qualitative types of behavior. They're all kind of boring. You can't really have anything that interesting, okay? So just file that away. Because what we're gonna do now is you'd think if you overload this idea to vectors, how much more interesting can it be? And you'll find out very soon. Actually, it's pretty much as interesting as any dynamical system can get almost. There's another level but we'll get that later.

Okay. Now, here X of T is called the state. N is the state dimension or informally it's the number of states. So it is slang to refer to XI as the I state. However, it's widely used slang. Basically, you wouldn't say that, I think. You wouldn't write that but you would say it. Now, of course in a lot of applications like in dynamics of structures or aircraft or something like that, the Xs actually have names like, you know, X1 or they have meanings, in which case you would actually talk about that. You know, what is the YA and what is the YA rate and what is your angle of attack and all this kind of stuff, your altitude. So in that case, of course, you would — it's ok — well, it's still slang but you would talk about that — those as individual states. Okay. So N is the state dimension or the number of states. A is called the Dynamics Matrix. By the way, in lots of different fields it's got a different name. Let's see. I was just talking about aeronautics. So what is A called in aero — there's somebody — there's a bunch of people here in aeoro-astro. What is A called when this is a predictational model of a flight — some steady state flight? You know, I mean, the entries of A have — A has a name and the entries have names.

Student:[Inaudible].

Instructor (Stephen Boyd):That's it. So in that case the entries of A — in that case A is called the matrix of stability derivatives. I am not sure where that came from except that indeed it will depend — the entries in that matrix will determine whether the flight is — that flight mode is stable or not. So they're called the stability derivatives. And I guess it's obtained from linearization of a non-linear system, so that would explain the derivatives, so okay. And other fields have other names for it. In circuit design it's called the small signal dynamics matrix or I don't — who knows. But anyway, lots of fields have different names for it. Okay.

So here's a pictures. It's very stupid and extremely useful. It's this. So here's your state at X of T. And it's very useful to do the following. Of course AX of T, it's just a linear and basically A maps X — basically where you are into where you're going because X is where — essentially where you are in state space. Xdot is where you're going. So A maps X into Xdot. Oh, by the way, what are the physical units of A? Assuming let's say all the Xs are in, you know, some common units. Let's just leave it that way. So all the Xs have some units which are irrelevant. What are the units of A?

Student:[Inaudible].

Instructor (Stephen Boyd):It's inverse seconds, exactly. It's a frequency. It's a rate. That's what it is. But, I mean, this is kind of obvious but that — so A is a rate. A is an inverse seconds. I mean, depends on the units in X, but generally it's an inverse seconds. By the way, that means that big A and this is a fast system and small A is a slow system. I guess this is kind of obvious, so I'm gonna move on. Let's go back over here.

So Xdot which is AX is where you're going. And it's extremely useful to take that vector and to glue its base to X. And so you have a picture like that. So if you're over here, AX might point in that direction, okay? And if you're over here, AX might point in that direction, okay? And what it says — it does not mean, of course, that X is gonna be traveling along this line. What it means is that along the solution of X at this point whatever that curve is, it's tangent to this line. And the length of that line gives you the actual speed at that point, okay? This is kind of obvious. All right.

Now, if you draw a picture of Xdot for a whole bunch of points X in a plane, you get a picture called one — oh, this — so there's a name for this. Actually, it's a vector field, okay? So that's both, by the way, a mathematical description. That describes something which on some set at each point gives you a derivative on the set. That's formerly a vector field. So, in fact, Xdot=AX, you would actually call in mathematics a vector field, okay? But it's also used informally to mean something like this where you have a field of points and it — and sort of at each point conceptually — of course you don't draw it at each point. You draw a little arrow that gives you a rough idea of where you're going and how fast. Okay. So this is the example for Xdot=-1021X. You — we can check things. But the cool thing about this is when you see this vector field, you can actually start visualizing the trajectories. That's actually very important to understand really what's going on. So let's see what it says. It says if you're here, you're moving up and to the left and you're moving at a pretty good clip at least compared to over here. So although you're not gonna end up here, you know, you don't know where you're gonna end up but it might be like here. And you can see now that you'll actually keep moving up. You might even — it looks to me like it's even accelerating. So you can imagine a point starting here as actually kind of moving up like that, okay? On the other hand, if you're sort of over here, if you start here, you can sort of imagine now various things. You know, you might slow down. You're not gonna actually hit zero. You'd slow down a lot, and then it looks like you might actually start accelerating as you go along there, okay? So these two are just the kinds of things you would get. And by the way, if you ever have a system and you want to quickly figure out what it does, you need to look at pictures like this. It only works in two dimensions. Actually, it depends on your visualization skills. You could probably do this in three, but it would be tricky, I guess. Okay.

Here's another example. Another little baby two by two matrix, and in this case it's this. You will later come to understand that you'll look at that matrix so — the same way but so far you look at just a matrix and you know what it means in terms of its input, output entries, right? If I write a matrix down, there's zeros, you know what it means. If there is large entries, you know what it means. If there's negative numbers, you know what it means. In terms of just how the input affects the output. So that much you have. That should be wired into you by now. You will actually develop something like that for dynamics matrices. So certainly for two by twos and three by threes you'll start getting a real — very good idea. You'll look at that and get a rough idea. There's gonna have to be some complication to really know what happens but that's the idea. So here's the vector field here and you can kind of get a pretty good idea for it. Here it looks like the trajectories are kind of elliptical. Now, I'll tell you what you can't tell by your eyeball here is — unless you were super duper careful. You can't tell if the trajectories are actually — are they winding in or are they winding out? You'd have to really kind of trace this very carefully and figure out if — when you kinda come around one cycle, you're bigger or smaller than you were before. Okay. So that's, I think, not obvious from here. It will be very obvious to you in a week as to how to do that. But that's the idea. Okay.

Now, another very useful thing is a block diagram. So you can write Xdot=AX this way. By the way, it's done not with differentiators but with integrators. So that's — and there's historical reasons for it. Well, I'll tell you what the historical reason — actually, does anyone know the historical reasons for it? It's entirely likely that you're all too young to have any — this is in the deep — this is — we're talking slide — we're talking before slide rules here. Anyone here ever use a slide rule? Cool, zero, you did. That is so cool. Did you do it as a joke or, no, you really used it?

Student: Well, it was my dad's.

Instructor (Stephen Boyd):It was your dad's, well, there you go. So all right. So but still it's cool, though. Do you actually know how to use it?

Student: It's probably [inaudible].

Instructor (Stephen Boyd):Cool. That's about the right — that's about how it should stay. Okay. So I can tell you the — I'll tell you the historical reason for this. So first let me just say what this is. This is a vector. These are vector signals and it's sometimes common — I guess this is from digital circuit design — to take in a signal flow graph to put a little note with a line through it. I don't know why this tradition came up. And this tells you the dimensions. So that's a vector signal with N components, X of T. It goes into A so what comes out here is AX. And that goes into — 1/S is actually — you really should write that as I/S because this is — so you would interpret this because it's a vector signal in and vector signal out as a — you would actually — the slang for this on the street should be a bank of integrators. That would be the slang for this. Because it I exploded this out and showed the individual components, it would really look like this. It would look like that. Let's say if it's two by two. So that's if I clicked on that box and asked for the detail, I would get this, okay? So it looks like that. So it would be a bank of integrators. These are now scalar integrators here. Okay.

And now, let me get to Y integrators. So nowadays you will soon see how to actually solve the equation Xdot=A of X. It won't be surprising to you that you can work out the whole trajectory for X1000, 2000. I mean, these are just enormous systems. Just immediately on a laptop. I mean, 2000 is not immediately, all right. But a thousand, even 500 is extraordinary, okay? So a 500 state model will model a lot of things. I mean, that's actually a fairly detailed structural model of a lot of things. You can actually just solve Xdot=AX. It's nothing. It's gonna be two lines of code, something like that. If that — it'll run on a laptop, not yet a phone but that's coming. It'll — and just get it so it's like — it's sort of like Lee squares. For you it's nothing, it's a backslash, right? For your parents, it was much more complicated. It was a half day of coding Fortran. Don't even ask what your grandparents had to do to do Lee squares. They — maybe not your exact grandparents but somebody's grandparents did it and it wasn't that cool. It was mechanical calculators or sheets or slide rules. Lots of people in rooms. So it was done. So all right. Back to this. In the, I don't know, in the 20s, 30s — actually even earlier than that. I think this — anyone know? You want to do a Wikipedia on differential engine, Vannevar Bush, differential engine, differential analyzer, there you go. So that actually — I believe it might even be late 19th century. So in the late 19th century it was already recognized that Xdot=AX was an — was that if you understood what the solutions of that did, you could actually say a lot about how a machine or something like that was gonna work. That was all or how — 19 what?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, I was off as usual. It's a good thing I'm not in the history department. But I'm allowed to — I got it vaguely right. It was a long time ago, so 1927. So in 1927 — oh, but maybe that's the — is that the mechanical one? Okay. So this guy built a mechanical system that will actually give you the approximate solution of Xdot=AX, okay? Not long after that people built vacuum tube computers like analog computers. This means nothing, thank God, actually, to anyone. Nothing, no one has even heard of this. That is so good. Usually — you've heard of it? That's so good. Did you actually see one? No, okay. That's too — I should bring in some pictures just so you know how lucky you are now. Yeah, what's that?

Student:[Inaudible.]

Instructor (Stephen Boyd):Yeah, they're typically in basements now or storage closets. Yes, that's right. You saw —

Student:[Inaudible.]

Instructor (Stephen Boyd):Yeah, sure, they really used them. Okay. So what it was was this. It was an electric — you had a big patch panel and you had electronic integrators. I guess anyone here in electrical engineering knows how to do that with an op-amp and a capacitor in the feedback loop, you get an integrator. And you had a big — you had a whole bunch of integrators and then you had a little like banana plug things and you could plug these up and you could wire them up. They had little gain units that you would dial in. They're really quite beautiful. I — actually, I never touched one so — just so you know. So — and you dial in little gains and things like that and you'd have a whole — so how would you actually program this analog computer? You'd do it by actually physically hooking wires up between these things, okay? And then there'd be a big button and you — a big button and you'd press start. And also the red lights would come on meaning that things just overflowed their ranges, right? And that either means you messed up the programming which in this case literally means plugging wires in or it means you probably shouldn't build that aircraft. It means one or the other. You'd have to figure out which it was. Oh, and the way you would — the way if you had like a class like a homework exercise on the analog computer, the way it would work was actually kind of cool. You'd have gra — your program would be this big thing like this with all your wires on it. And you would detach the whole thing and then walk around with it. And then the other — another student would come in and plug theirs into the analog computers.

Are you at least a little bit grateful now about when you were born and stuff? I mean, I hope so. Yeah, I'll take that as a small sign of gratitude. Watch out because provoke me and homework eight, analog computer. And if you don't think I can find one on E-Bay, you are wrong. Okay. So all right. What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, take a bite. I'm gonna take that as an open challenge. You —

Student: It is.

Instructor (Stephen Boyd):Okay, cool. No, you know what we'll do. What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, okay. Yeah, actually the T.A.s have to take this course on, you know, ethical actions and this, so Jacob is exercising his right to not be involved in such an escapade. But that would be great. No, maybe we'll do it with like white protoboards and op-amps and capacitors. That would be great. Okay, all right. But it's noted that there's been a challenge. All right. Back to this.

So there — the reason was that you find integrators here is because of this. Oh, and by the way, I guess you mentioned — someone, you knew what the story was. So this was used in 1939 through 1945 at M.I.T. They put these things together. They weren't quite linear. They actually had other terms in. In fact, it was just basically, you'd fire a shell and wanted to work out firing tables. And a firing table was for a certain shell, if you fired at this angle and there's a wind in a certain direction, the question is where does it land? And you would solve a difference equation like this. It wasn't quite linear. It had one non-linear term in it. And this was done, in fact, in secret in a basement in M.I.T. with analog computers. And then results, you just tabulate — just ran all the time and they'd worked out the things and when you wanted to use it, you checked the wind, figure out, you know, go in the table, find the range and find out you should elevate it 22.63 degrees. Okay. So that's what this was. Okay. All right. So that's a — just a historical comment about why you see integrators and why this block diagram would strike fear into the hearts of your parents and grandparents if they did this kind of thing. But not you. So — and why — that's why you should be grateful. Okay. All right. Okay.

So if you draw a block diagram out, if you explode the block diagram of A, you can actually get interesting information. Here's an example. Suppose you have Xdot is AX where A is block upper triangular. Well, by now if you just see this — if you see Y=AX and A looks like that, you know exactly what it means. Without even thinking you would say, hey, how interesting. The bottom half of — let's suppose that's Y. You'd say that the bottom half of Y doesn't depend on the first half — it doesn't have to be half, of course — but the first part of X. That's what you'd say when you see that. But now, that's the derivative which is actually more interesting. So you read this equation this way. You'd say something like where the bottom half of X is going, that's English for X2dot, doesn't depend on X1, that's the zero. Okay. And when you draw the block diagram it's super — it's totally obvious because you draw it this way. Here's X1. X1dot is A11X1+A12X. Oh, I didn't say something here. The rule here is this. You want to know how do you get X1dot if on — if all you have are integrators. You look at the output of an integrator and you ask, well, what went — if that's an integrator and what came out is X, what had to go in was Xdot. So that's how you do. So you simply — you go backwards through the integrator if you want to get Xdot. So the inputs to integrators are derivatives. So this is X1dot and this is X2dot. And this says X1dot is A — it's a sum of two things, that's what the summing junction does. It's A11X1+A12X2. Now, when you stare through this block diagram, something exceedingly obvious comes up and that's this. If I draw a dashed line like that, you see something really interesting and that is that information flows from the bottom to the top but not vice versa. So nothing that happens up top ever has an effect on what goes on down here. Okay. And well, and basically it says X2 affects X1 but X1 has no affect whatsoever on X2. That means all sorts of interesting things. We've concluded things like this. It says that X2 — you can actually calculate the solution of X2 separately because it has no — it is in way affected by X1. That's what this says and that's what you get out of looking at that equation. It's kind of — well, we'll see lots of other ways to do it but this is kind of the idea of the way to get the intuition for how this works. Everybody see this? So that's the picture here. So let's see.

Let's look at a couple of examples. I think I'll just look at just one, which is a linear circuit. So here I have a cir — a linear static circuit. Now, that means it's a circuit that can contain things like resistors, transformers. It can have, oh, let's see. Well, it depends on your model of a transformer. If it's an inductive model you have to put it out here. So we'll skip transformers. But it can have things like dependant sources and things like that. So that's what's in here. And I pulled the capacitors out to the left and the inductors off to the right. And the equations here are very simple. It doesn't matter if you're not in E.E. and don't know these equations. So that doesn't really matter. It's just an example. So here I'm gonna — the equations here for each capacitor are this. It's CDDBT is the charging current and I've drawn the charging current to go into the capacitors like that. For the inductors, it's the same thing. It's LDIDT is the charging voltage. So for an inductor — again, I'm addressing people who do E.E., right? For an inductor, you think about voltage as charging it. When you apply voltage to an inductor, it ramps up the current. When you apply a current to a capacitor it ramps up its voltage. Okay. So you get these equations. And then this thing is some horrible complicated thing. But the point is, it's linear. So it's a set of linear equations that relate — these are the port variables, the voltage and the current and the voltage and the current at these ports. That's called a port when you hang two wires out of a circuit. It's a port. Okay. And there's a linear relation that covers these — the voltage and currents at the port, the port variables. And we're gonna write that in this way. We're gonna say that the inductor — sorry, the capacitor's current and the inductor voltage — actually, these are the charging — these are basically the charging variables is some matrix times VCNIL. So we're gonna write it that way. All right. And we'll let C be a diagonal matrix with these capacitors and L, this thing, so that I can write these out as matrix equations. And if you have state, VCNIL. So the state is the voltage on the capacitor and the inductor current. Then you can write out everything here as — it's very simple. It's — this is CVdot here is IC and this is LIdot — uh-oh, that's hard to make a dot and make it clear — equals VL. And you simply put those equations into here, take C inverse on the left-hand side and you get a set of equations like this. Okay? So this tells you that you can write out and this is, of course, an autonomous linear system. That's AXA is this matrix here. Okay. So — by the way, this is already of huge interest. It says that, for example, again, this is addressed to people in E.E. It says, for example, if you have an interconnect circuit and some leading edge, you know, 45 nanometer design it says that if you want to analyze the interconnect in some digital circuit, in some high performance circuit, which you can model as — certainly with some inductance, capacitance and resistance. It says you write that as Xdot equals AX period. That's what it does. So that means it's already of extreme interest in practice to know what the solutions of this do. Okay. So we'll quit here.

[End of Audio]

Duration: 74 minutes

IntroToLinearDynamicalSystems-Lecture10

Instructor (Stephen Boyd):Great. It looks like we're on. Let me make a couple of announcements to start. You can turn off all amplification in here. So if you go down to the pad, I can make a couple of announcements here.

First, let me remind you – here we go – that today I'm gonna have extra office hours from 1:00 p.m. to 3:00 p.m. today, and the other – this is important, announcement, is that next Monday's section is gonna be cancelled, so we'll let you rest from – I'm sure I don't have to mention this, but of course, the midterm is tomorrow, or – it's either tomorrow or tomorrow after Saturday or Saturday to Sunday. Your choice. And of course, all the information is on the web. Any questions about the midterm or anything like that before we continue to material?

Student: Where do we go to pick it up?

Instructor (Stephen Boyd):That's on the website announcement. It'll be – it's actually Bytes Café. It's just the ground floor of Packard building, is where you'll go. I hope that's there. It should be there. Yeah.

Okay, for the – the only thing I would maybe make an announcement – I should report we – an alpha tester took the exam. He survived, and he's recovering steadily, so no big deal. Just some extra fluids and some light antibiotics. He'll be fine. He's gonna be just fine. And we got a beta tester taking it now, and as of last night around midnight, he was okay too. So it looks like – and they did, by the way, you can thank them, because they got a couple of bugs and we – a couple of bugs and typos, and we rephrased a few things to make them clearer. So they're suffering for you, just so you know. All right. Any questions? Any other questions? Are you ready? Okay.

Just remember, we put a lot more time into it than you will. Okay, we'll continue.

So we're looking at examples of autonomous linear dynamical systems. This is not – basically systems of the form x dot is a x. Maybe zoom out a bit here so we can get the whole page in there. There you go.

So we're looking at examples of autonomous linear dynamical systems, the systems that look like this, and it basically just says that the derivative of each component of the state is a linear function of the state itself.

So here's a very standard model. It looks like this. It's a reaction where some species, a, converts or decays to species b which, in turn, decays into species c, and this looks something like this. Its x dot is minus k one zero zero, and we can actually work out what that means.

This is basically x one dot equals minus k one x one. So here x one is the amount of species a present, and this says that, basically, the amount by which – the rate at which that decreases is proportional to the amount. Something like that. So k one is this reaction constant here.

The second one is actually very interesting, the second row. So as with looking at y equals a x, you should never look at x dot equals a x and actually just kind of say, yeah, okay, that's fine. You need to actually understand exactly what everything is here.

So the second row says x two dot equals k one x one minus k two x two, and how would you explain it? What is – what is that term? It's – what is it?

Student:[Inaudible].

Instructor (Stephen Boyd):This one is minus x one dot, but how do you – how would you explain, just in words, what is the meaning of this term?

It's – this is a buildup of x two coming from the decaying of x one or – actually, I said it wrong. It's a buildup of species b, because that's a byproduct of the decay of species one, and this is actually then – is actually the decay of x two, of – well, whatever it is, it's the decay of x two because some of species b is turning into species c. And the final one is this, which is x three dot equals k two times x two, meaning that species c, here, only comes from the decay of species two. That's this, and this – that's this bottom row. Okay?

And what if I were to put something like this? Minus k three. What would that mean? Where k three is another positive constant. It would have a meaning. What would it mean?

Student:[Inaudible].

Student: Yeah, x three decays, and where does it go? Right. To vacuum, to somewhere not on our – where we don't account for it, to the environment. Somewhere else. Everybody see what I'm saying here? So we put a zero here and that's fine.

Let's see, there's a couple of interesting things about this matrix. One is that the row – the column sums are zero here. So that has a meaning, actually. I'll just mention that briefly. We'll go into this later in much more detail, but let's see what it means that the column sums are zero – we should restore that zero there. That the column sums are zero.

Let's try one thing. Let's – what is the interpretation of one transpose x? X of t. One is the vector of all ones, so what is one transpose x of t? It's a sum. It's the – so what – how would you say this? It's the total amount of all species in your system, right? All right.

What is this? What's DDT of one transpose x of t? Well it's one transpose times x dot of t. X dot of t is d x d t. Okay? Well wait a minute. X dot of t is a x of t. Now for this a here, the column sums are zero. What is this? In fact, more specifically, what is that? It's zero. Exactly because the column sums are zero, that's zero. What does all this mean?

This says that the time derivative of the total amount of material in the system is zero. That – you know what that says? That says that one transpose x of t is actually equal to one transpose x of zero. It's constant. So you know what the column sum zero means in x dot equals a x? It's actually conservation of the sum. Conservation of the total. Okay, so there's a name for systems like this. I'm just mentioning this, because it's very important, as when you see y equals a x, to not – to look at it, understand what every entry means, what every feature means. They all have meanings.

In this case, the column sums are zero corresponds precisely to conservation of mass or material, whatever you wanna call it. Okay? That's just on the side.

Well let's see what happens here. Here's an example, where we start with one zero zero. So we start with one unit of species a, and you can actually – what happens here, I think, is kind of – is obvious. By the way, you will very shortly know how to actually work out what the solution of x dot equals a x is, but here what happens is kind of obvious. X one immediately starts decaying. In fact, what is the initial slope, right here? If this is in t, at point zero one, what is the amount of x one present?

What is it? Well what's – what is x one dot of zero?

Student:[Inaudible].

Instructor (Stephen Boyd):It's minus k one, okay? K one is one, so what is this initial slope here? It's minus one. So if I ask you, after, for example, ten milliseconds, how much of – how much of x one has decayed, the answer is point zero one. That's how much.

So at point zero one, this thing is very, very close to point nine nine, is the amount of – is x one. It's not actually point nine nine, but it's very close. Okay? That's the initial slope.

Now what happens is, this decays – by the way, as this one decays, x two builds up and, in fact, if you were to add x one and x two together, you would get something that, at first, is almost constant, but it's not quite constant, because x two is further decaying into x three. So as x one decays, that corresponds to x two building up.

Now, by the way, as x two builds up here, here, once x two is high, actually, x one – you can see that x one slows down here – no, sorry, it doesn't. What I just said was wrong, so scratch that. When we get the ability to edit the videos, that'll be fine, so I can just like cut those things out, and I'll claim I never said them. Okay.

All right, so x two builds up, and x three doesn't really start appearing, because the only way you can get x – species c – the only way x three can go up, is for x two to first build up, and then a significant amount of x two to today. So if you kind of look at this, everything looks the same. And by the way, as you sweep along here and add x one, x two, and x three, what you'd find, of course, is that the sum is constant and it's one. That's what we just worked out with conservation of mass here. Okay?

I mean – this is a stupid little problem. You could do this with undergraduate methods. It's not a big deal. The important part – actually, in all of the things I'm looking at here, the important part is to understand the following. The same way least squares is not used to combine four arranged measurements to get two positions. Least wares is used, and least wares methods are used, to do things like to blend 1,000 sensor measurements, to make an estimate of 100 parameters. That's the real thing. So these are just little vignettes.

This is called a compartmental model, so this whole book is written on models like this. It's used in – obviously, it's used in chemistry. It's actually used in economics, with the flow of – to describe flow of materials and goods. It's used also in, I guess, pharmacokinetics, is where you see this used a lot, where you trace things moving from one place to another.

Some of those models are not linear, but linear ones are sort of the basic ones. What's nice about our abstraction is that, if you look at the source code I wrote to simulate – to draw this plot, it defines a, and a is this thing, but it would just – the same code would work just as well if a were 1,000 by 1,000.

Now, I promise you, if I gave you a compartmental system with 1,000 species and different things leaking into other things, and making multiple byproducts, and things like that, what happens is not obvious at all. Same as least squares. Least – you know, if you have, give me four range measurements, anybody can kind of make a good guess as to what your position is. If I give you 1,000 samples of a signal or tomographic projections, and ask you what the image is, there's no way. So same as here. I just mentioned that.

Next example is a – is actually a discreet time linear dynamical system. So a discreet time linear dynamical system. Of course, that's just an iteration. It's just this. It's x of t is a x of t, right? For which, by the way, the solution I can write out right now. It's no problem, I'll just write out the general solution. It's a to the t times x of zero. There. That – by the way, this formula even works because one convention is that a matrix to the zeroth power is the identity, so this will even be valid for t equals zero, but it doesn't matter. So, there's the full solution of that, but let's look at an example.

The example here is a discreet time Marvok chain. By the way, you don't – I mean, probability is not a prerequisite for this course, so – but in any case, these are just examples. So these are just examples to show you that these things do come up. Actually, you really should know it. I can't imagine not knowing about Marvok chains. It would be a mistake in pretty much any field I can think of, but okay.

So if you have the background to understand what I'm – this example, great. If you don't, don't sweat it, but maybe you can get the idea. So here's what happens. You have a random sequence, but the values in the sequence – it's just a finite number of values. It can only be one, two, three, up to n. Okay? And, generally, these are called states or something like that, and we'll look at an example shortly, but – so the states could be thing – this could either be states, like a system is up, down, it's in standby mode – it's also called modes of a system. These could – it could be in standby mode. Something – you could have admission block in a network or something like that. This can also give a q length, so this could – this state could be the number of jobs waiting to be processed. This could be the number of packets queued in a communications system to go out on some outgoing link, for example. So that would be the type of place where you'd see this.

Now what we're given is this. If you know what z of t is – that's one of these numbers. It's one of these states or modes. You actually don't know what z of t plus one is, but you have a conditional probability, so here's what you know. You know that, given that you're in state j at time t, the probability that you're in state i at the next step is given by some numbers p i j. I should warn you here that people who do Marvok chains use the transpose, and they use row vectors. Okay?

So what you see here is not what you'd see in – this whole course is on Marvok chains, but you will not – they'll use a different notation. Actually, it's just transpose. Everything is transpose. Probability vectors are row vectors, and the i and the j are switched. So, for us, p i j is the transition probability from state j to state i. Why? Well, because that's kind of – for us, if someone walks up to you and says y equals a x, and you say what is a i j, you say, well that's the gain from input j to output i. So we'll keep our standard. They do it the other way around. For them, p i j is the transition probability from i to j, but this is just to warn you.

Okay. So this is a given matrix of transition probabilities, and that means, for example, that the column – that the – for us, not in probability, but for us – or statistics, a column is a – the first column, actually, is gonna be a probability vector. It's actually – it's a bunch of numbers that add up to one, and it tells you, if you're in state one, these are the probabilities of where you'll be at the next step.

So, for example, if it was this, right? If that was the first column of p, you would say, if you're in state one, then with 100 percent probability, at the next time, you will be in state three. Okay? If it looked like this, point five and point five, and zero, zero and so on, this says, if you are in state one at time t, then at the next step there's a 50 percent chance you'll still be in state one and a 50 percent chance you will move up to state two.

By the way, if the state represented some q length, it would be something like this. It would say that, with 50 percent chance, the q length will remain one, and with a 50 percent chance it'll go up to two, which means maybe a new packet or a new job arriving, for example. Okay.

So the way you analyze these is, you represent a probability distribution of z of t as an end vector. As I said, in probability, in statistics, and in OR, people represent these as row vectors. Some. Some people also do this.

So you make it a row vector, like this, and this is a vector that adds up to one and all the entries are positive, and it basically tells you the probability. Okay. Now if you wanna calculate something like the probability that something is one, two or three, you would simply – that's a linear combination. It's simply the sum of the first three entries, and so you'd have a row vector here times your probability vector.

Now if you simply work out – if you write out what this is in matrix form, it is – it's just nothing more than this. It basically says that the next probability distribution on states is capital p times the current probability distribution on states, so that's what it was, and so this is a discreet time linear dynamical system. So you'd say, for a Marvok chain, the probability distribution propagates according to a discreet time linear dynamical system.

Now the p here is often, but not always, sparse, and a Marvok chain is often depicted graphically, and I'll give you an example. I'll show – here's – I'll just jump to an example. So here's a baby, baby example, but here it is.

I have three states, and let's say they even have – I've been – we've added a meaning or a comment to each one. So state one is something like, the system is working. State two is the system is down, and state three is the system is being repaired.

This is the Marvok chain. Now, of course, you could go through it and look at rows and columns, and figure out exactly what each one means. These zeros have various meanings, for example. That point nine has a meaning. But let's look up here. This is the way you would draw it. So you imagine a particle sort of sitting at one of these states. You flip a coin. Actually, a coin that has a 90 percent probability – anyway, you flip a coin or whatever. You get a random variable that has a 90 percent chance of coming up heads or whatever. So what this is, is if you're in state one, which means that the system is okay, it is – there's a 90 percent chance that at the next step it will also be okay. There's a 10 percent chance that it will go into state two, which is the system down. If the system is down, there's a 70 percent chance that, at the next step, it will up again and running. There's a 20 percent chance it will remain down, and there's a 20 percent chance it will – it'll go into – there's a 20 percent chance you'll actually have to call in somebody and repair it, or something like that.

However, if it's repaired, this one point zero says that the repairs are infallible. They always work, okay? So this is – that's a – and this is a silly little – this is even not so silly, because there's some very interesting things you can do. You can take this linear dynamical system and you can do all sorts of interesting things. You can run it for a long time, to see what happens.

Now, in general, of course, you don't know what's gonna happen here. Does anyone have a rough idea of what would happen here, if you run this for a long time? You can just guess. By the way, if you know about Marvok chains, don't answer. So I want someone's who's just intuition. What – let me ask you this. If I run this like 10 steps, do you think you can tell me intelligently what the state was 10 steps ago? 100. Suppose it had been running 100 times. What would you say? What would you say about where do you think the state is?

Let me – let's just answer that question. We'll start it off in this state, so it started off in the state one zero zero, and I run this 100 times. What do you think the probability distribution looks like? This is pure intuition. I just want intuition, but I claim you have the intuition.

Well what do you think is the probability that you're in this state? Obviously, I don't want a number to four significant figures here, right? I want a number between zero and one, and, in fact, all you have to do is say, small, large, zero, one?

Student: It's small.

Instructor (Stephen Boyd):It's small. Okay. So, good, fine, small. And for this one – it's what?

Student:[Inaudible].

Instructor (Stephen Boyd):It's larger, but maybe – but still relatively small, and for this one it should be, I don't know. In fact, you wanna guess some numbers? Go ahead and guess the number on that. What is it?

Student: Point nine five.

Instructor (Stephen Boyd):Point nine five? I don't – really?

Student: It's large.

Instructor (Stephen Boyd):No, no, no, because this basically says that one in 10 times it goes here, okay? If it – actually, if it always went back, if this was a one point zero here, it would be point nine exactly, wouldn't it? Because one in 10 times it would be here, but the point is sometimes it actually goes a couple of times here, and a couple of times it goes over here, so it's gotta be less than point nine, I think. Are you buying this? Yeah.

So it's – I don't know what it is. I'll make up a number. It's point eight five or – hey, Jacob. You wanna calculate the steady state probability distribution on this guy? Only if your laptop is on. Is it? Okay. We'll – don't worry, we'll – take your time, and we'll come back to it.

It's the – I guess, in this case, it's not the left, it's the right eigenvector. Okay, so – all right, so what'll – what happens is, actually, you'll be able to see, you'll be able to do this immediately. You'll – within two or three weeks, you'll be able to say very cool things like this.

Someone can take a – this system, and say things like, by the way, when something – when the system goes down, what's the average number of time periods it takes before it comes back up again? And that has an – I mean, it's some number. Well we could – you maybe – you could calculate it. These are incredibly useful. You could ask, what fraction of the time is this machine up and running? That’s very important. That will tell you your throughput or whatever, and if the machine cost $14 million, that’s very important to know what the throughput is, because, well, the throughput is gonna be the – it’s gonna be the denominator.

Okay. All right, so this is the type of thing. By the way, once again, this is a baby problem. We can get it. If I put 10 states here, if I put a little queuing system with 10 states and, maybe 50, 80 transition probabilities, I guarantee you, you have no idea what the probabilities are gonna converge to and all that. What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):And point what?

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. Great. There you go. So here it is. If you run this for a long time – not even a long time, probably 10, 15, 20 steps will do the trick, the probabilities will go to this. Okay? These are extremely important numbers in practice. They tell you that, actually, after a little while, this thing, 88 percent of the time, the system is up and running. .1 percent of the time, it’s actually down, and .2 percent of the time – what’s that? Thank you. It’s exactly 10%. What did I say?

Student:[Inaudible].

Instructor (Stephen Boyd):I said .1 percent. Well, you know what I meant. When I say .1 percent, I mean 10 percent. Sometimes. Right. No, you know, look, come on. This is a – that’s a graduate class, that’s allowed, so that’s fine. Undergraduate class, you have to distinguish between .1 percent and 10 percent. Not here. That’s – you can check, that’s in the official rules. Okay.

So these are the numbers, and that’s what it looks like. So – I mean you’ll be able to look at these. These are interesting things. So now we know the answer. The system is under repair – it’s under repair 2% of the time. Now, by the way, you can estimate your repair bill. Yeah, by the way, you could do things like say, I don’t know, maybe I can get – I can upgrade to a fancier machine or whatever, and these probabilities change, and you might – someone can put a price tag on it and ask, how long would it take to pay for itself?

By the way, if that’s like – if it’s infinity or a negative number, it probably means you shouldn’t buy it or something like that. I guess not a negative number. If it’s infinity, you shouldn’t buy it. So – I mean these would be the types of questions you should answer, so – you would be able to answer. Okay. All right.

Next topic is – or next example. These are just examples. Let’s look at a numerical method for calculating x dot equals a x. Hey, I was just involved in something like just this two weeks – three weeks ago. Okay. You have x of zero is – we're gonna – we have x dot equals a x, x of zero equals x zero – by the way, within about a week you're gonna have the exact solution of this. I mean, exact means sort of in an analytical sense, and you can – you'll be able to compute it and all that sort of stuff very, very, soon. You'll be able to work out the solution of this, and you'll know a lot about it.

But here's a method for approximately solving x dot equals a x, and the idea is this, is you take a small time step h, right, and the hope is that x – the state should not change much in h seconds, and this is the simplest possible way to approximate – to get an approximate solution of a differential equation is this. It's really simple.

Basically it says this. You're at x of t at time t, that's your current state, and someone says, where will you be h seconds in the future where h is this small number? Well, a very good approximation is this. Where you will be is where you are plus where you're going, roughly, your speed, multiplied by the timing from an h. Now this is, of course, this is an approximation. It is not actually equal except in certain special – very special cases, but it's not equal. The reason is, the minute, the instant you move away from x of t, your velocity vector changes, and this sort of assumes, instead, that your velocity vector is gonna be the same for all h seconds or something like that.

So this is that of x of t plus h is about equal to x of t plus h x dot of t, but that's a x of t, and so this is, basically, a plus h a, i plus h a – that's – I could say that too. If I point to i and say a, you should interpret it as i, unless what I've written is wrong, in which case, you should interpret it as a. That's clear, isn't it? Good, okay.

So x t plus h is i plus h a x of t, and this, if you – if we change the indexing, right, this gives you a discreet time linear dynamical system, and it means you can actually work out an approximate solution. That's an approximate solution. That's what it looks like, so that's the idea. That's x of k h. By the way, we can actually have some – well no, I won't do it. I'll keep with – I'll keep at this thing for now.

Now, actually, this method is never used in practice, because – for a couple of reasons. The first is this. When you're doing this, you sort of – you start at one place, you make an approximation as to where you'll be next step. Now you make an approximation based on that approximation, and so on and so forth, and although this can actually work, the error can and does build up in lots of systems.

Actually, you'll later be able to do a perfect analysis of when the error – what types of system the error would build up, and also where it doesn't, and there are also much better methods, but this is just the simplest method. Okay.

So those are just a bunch of examples, just to show you that these things do in fact come up. So let's look at – let's get one thing out of the way. We are looking at x dot equals a x. That's a first order vector, linear, differential equation. It's nothing more. Depending on how you count it, it's – let's see, one, two, three, four, five – it's five ASCII characters or, I don't know, however you want to count that. Very compact. But what about second order, and third order, and fourth order, and all those sorts of things?

By the way, second order systems come up constantly in mechanics and dynamics. Basically everything is second order. Lots of other systems, lots of things are second order. Actually, for discreet time systems, you have the same thing. A lot of things, your next state depends not just on your current – sorry, your next value, I shouldn't call it the state. The next value of something depends not just on the current value, but actually on the previous one as well. So that would be higher order recursion.

Okay, so let's take a look at this. This is x differentiated k time, so this is a kth order linear dynamical system, like that. Lucky for you, we can reduce these to first order, otherwise there'd have to be another class after this one on – this would just be the first order one, then you'd have second and third, and it would get very boring anyway.

So here it is. There's a standard trick – by the way, this doesn't work in the scalar case. So this is actually a payoff of taking the abstraction of matrices. So when you're an undergraduate and study that, hopefully for not too long, and then someone comes along and says, yeah, but you know, I have to satisfy this equation or something like that, unfortunately you gotta dig in and solve this separately or something like that, right? Because there's no way you can reduce this to that, because these are scalars.

Now the cool part is, once you have passed that boundary and become – and grown in sophistication, and overloaded this equation so that these are vectors and a is a matrix, then it turns out you don't have to worry about higher order stuff ever again, because it comes for free in this higher level of abstraction. So this is a very standard method. It works like this.

We're gonna take the new variable. We're gonna stack the derivatives here. So we stack x of what – by the way, you cannot call x the state here. So I might accidentally say it, but if I do, it's wrong. So you take the variable, the vector variable x, and you stack x, its derivative, second derivative, all the way up to the penultimate derivative, so that's this guy here. The k minus one derivative. And now I wanna work out what is z dot. I'm gonna call that z, and that's the state of this system.

So I take z dot. Well, if I differentiate all of these, that's easy. The first one, differentiated, is just x dot, but that means – that's the second block of elements in the vector, all the way down to the bottom one, and when I differentiate this bottom one, I get x differentiated k times, and now I use my formula here. So if you look at this matrix, you will see that it faithfully reproduces this thing here.

This first row, for example, says that x one – no, sorry. It says that x dot, that's x differentiated one time, is equal to – you're gonna multiply this by – over here, by x, x dot, x dot dot, and so on, it – we just across here and down here. These are block multiplication, and it's just x dot, so it's correct, and the bottom one tells you what x differentiated k times is. That's this thing, okay? By the way, you're gonna see matrices like this coming up a lot. You should already have some ideas about what this matrix is.

This is called a block companion matrix. You don't need to know this, but its pattern, you should already have a feel for what it does, and let me ask you this, just for fun. Let's just make sure. If I showed you a matrix that looks like this – yeah, that's good. There we go. Okay. And I asked you – everything I haven't shown is zero, okay? Tell me what it does. Just describe it. If I said – if we looked at this, can you please explain to me what y is, as a function? Just in words, what does it do? It's an up – is it an up shift? Or is it a downshift? No, it's an up shift, sorry. You take the vector x, like this, and you simply shift it up. And what do you do at the bottom? Yeah, okay.

So here's how you would say – the slang for this on the streets is, you up shift x and you zero pad. So up shift is kind of obvious. The zero pad means that when you go up, you have some spaces where there was nothing – you can't – well unless you're the kind of person who likes indexing out of bounds of arrays, which we frown on here, you don't do that, so you zero pad. That's just what to do when your formula for shifting indexes is outside the array bounds. Everybody see what I'm saying?

So it basically – I shouldn't even – I mean you should – when you see that matrix, you should say, that's an up shift and zero pad. Actually, when I fill this in with a bunch of matrices here, it's actually really cool. It's actually an up shift, an up shift, and then, down here, you pad with a linear combination of all the things, or fill in. It's not – I guess padding is usually used with a zero pattern. Okay. Yeah.

Student:[Inaudible].

Instructor (Stephen Boyd):Why – yeah, it would work. I'll show you. Yeah, no problem.

Student:[Inaudible].

Instructor (Stephen Boyd):So – no, it does work, and that's why I think the whole thing is silly. So let's go back to that and say, I really wanna solve this like that, okay? So we can write this out this way. It's no problem. We write it this way, that's the state, dot equals, and then this is easy, that's zero one, and then let's see if I can get this right. Maybe that's b and a times x x dot. Did I do it right? Bear in mind, this reflects on you, collectively, as a class. If I write something down and it's like totally and completely wrong, and you just passively sit there and go, yeah, sure, it's fine, because, basically, I don't care, so – I – it doesn't bother me. I don't mind, but it will reflect on you if I say something really stupid and wrong, and you don't collect – correct it.

Actually, I once yelled at a class – well, by email, after it was over, because it turns out those class email lists, they persist, because I found some horrible area in a homework problem or something like that or whatever, or in a lecture, and I collectively held the class responsible. Well I got some responses, actually. I said, you know, you took this class last year. This was completely wrong, and you – I remember seeing that, and I remember all of you looking at me like, yeah. Yeah, it's totally clear. I got various responses for that. What's that? What's not right?

Student:[Inaudible].

Instructor (Stephen Boyd):Here?

Student:[Inaudible].

Instructor (Stephen Boyd):Down here? Good.

Student:[Inaudible].

Instructor (Stephen Boyd):This is x dot – let's do it together, and we'll see, but no, no –

Student:[Inaudible].

Instructor (Stephen Boyd):You think – no, no, this is what I want.

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, it's the – dot is the whole thing. Here, watch. There we go, there we go. No, this – listen, this is what we want. That's what we want, see. That's good. In fact, better – I mean for your collective – I mean what you really want, is you want that, is you wanna raise some things and say, is that right? And then you want me to go on and give a long – and say, yeah, at first it looks like this should be a minus, but in fact, it should be a plus. I should give you the story about it and every – and then it can turn out wrong. Then I'm the one that looks bad and not you, but your honor is preserved then, because you protested. Okay. No, good, so let's – keep being honest. All right.

Back to the question, why can't we do this? Well we can do it, it's fine. Here's the problem, is that when you first learned about this, no one had told you about matrices yet, or at least, that was the case. So that's the problem. It's basically why people should be taught linear algebra a lot earlier than they are now, because it just short-circuits a lot of really stupid and painful and idiotic material such as, for example, multiple, multiple weeks studying second order equations.

So – but I'll stop. So a block diagram – by the way, when you see x dot – no, sorry, z dot equals a z, and a was big – or z dot equals big a z, and it's got – that it like structure crying out at you. You should have an overwhelming, uncontrollable urge to draw a block diagram. So here's the block diagram, it's this, and you can check that this is right.

This – by the way, this shift business, you can see immediately here, because actually when you – when we're actually calculating a z, and we're shifting z, we're actually getting the derivative. So that's right, that corresponds exactly. By the way, people, there's a phrase for this. It's called a chain of integrators. So you'll find that very frequently. It comes up in lots of things, and it's quite beautiful.

It says something like this. It says – by the way, the arrows, if you can look here, you'll see that all the arrows go down, so you'll see that, in fact, that's just a chain of integrators. They don't do – each of these things simply – so, in fact, you can even say, if that's x k – if that's the input signal, this is – it integrated once. That's double integrated, triple integrated, and so on. That's what these are, okay?

It says take these things, which are integrated, form a linear combination, and feed that back into the input, so that's what this picture is. I mean, not that this matters, but – although, this should maybe look like – if these were z minus ones, this might look like some horrible filter you might have encountered in some stupid class on signal processing, right? No? Yes? I don't know, hopefully – are you still tortured with these signals? This has some stupid – what name do – what do they call this? It's like an IIR direct form blah blah blah? Is that it? It's what?

Student:[Inaudible].

Instructor (Stephen Boyd):No, no, no. That would be FIR – no, I'm sorry. If I did this, and then pulled this off, that would be FIR, but with this guy here, that's an IIR. By the way, if some of you don't know what we're talking about, you're very lucky. Okay. All right. You should aim to keep it that way. All right.

Mechanical systems, there's a huge number of – I mean, this is a perfect example of a higher order system, so mechanical systems. So a lot of mechanical systems. Again, by the way, this is a beautiful example where, once you have matrix notation, a lot of stuff works out very, very cleanly and beautifully.

You have m. So q is a vector of generalized displacements in some system. That means, basically, a displacement in a certain direct – each q three, for example, might be the horizontal displacement of some point on a system, but it can also be like an angular displacement or something like that.

So you get m, q double dot, that's the – this is the acceleration vector, m is like the mass vector, plus d q dot, that's a damping matrix, here, plus q k. And by the way, this is nothing more than – here, I'll draw it. It's the matrix analogue of this, so there's a spring with a stiffness constant k, a mass, here, of – with mass m, and then, of course, we have the famous dashpot. This is one of my favorite things. It's damping, but it's drawn like that. I guess – and it has been since the early, like, 19th century.

I don't know, maybe – have people seen this? Like a figure like this? You have. Did you ever actually – except for maybe a shock absorber in a car, have you ever – I mean, so you don't see these? It's sort of – they actually – they look like this because they – you know, in the early 19th century, when they built all these machines and things like that –

actually, even earlier, they actually added damping and there would be things like this. They would add a little piston, a little nozzle, and some oil or something like that that would be circulating, and it looked just like that, and it's called a dashpot. Don't ask me why, but that's the history of it, so there it is, and that's d.

And what this is, is – you know, this is basically m a. That's the force, and the force is equal to minus d q dot minus k q. Now k – we'll go back to this one, is simply the displacement, so the units of k would be in newtons per meter, so it's a stiffness, and d is in newtons per meter per second, okay? Or newtens per leftbren meters per second rightbren, right? That's what the unit of d r – and mass is kilograms, say. Okay.

Now what's actually kind of cool about this is it's the same thing. This is sort of the – there's your high school version, here, and all you do is you capitalize things and everything, and you – this describes a lot. I mean, a whole lot. So same number of ASCII characters as you saw this thing in some physics class for children, okay? This thing describes a whole lot here. I mean, like, this could be hundreds – in fact, typically are hundreds or thousands of – the dimension of q can be thousands, and then this would be, for example, a model of a bridge, for example, undergoing small motions, or a building or something like that, and that would be described by this.

So m is called the mass matrix, k is the stiffness matrix, d is the damping matrix, and you do the same – this so-called phase variable trick that we just did, and you take a state is x is q and q dot, so – and you get – so you get positions and velocities, and you have x dot is q dot and q double dot, but q double dot I can get from here with – by putting m inverse through here, and I get this. And I think this is maybe on Homework 1 or something anyway, except you didn't know about it then.

So there's some actually rather interesting things here, and just for fun, I would like you to explain to me, what is the meaning of k one two? The units, newtons per meter, and I want you to tell me, what is k one two in a mechanical system? What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):It has a very specific, physical meaning. Wanna help us? No. You gotta –

Student:[Inaudible].

Instructor (Stephen Boyd):That is it exactly. It's something like a cross stiffness or – there probably is some name for it, like a trans-stiffness or something. I mean I know what you'd call it in circuits, but I don't know what you'd call it mechanically, but something like a trans-stiffness. It says, basically, it's the force – this tells you the – it says that, when you displace the thing, a node two in some structure, you feel a force at node one, and it's proportional to this, so this is the trans-stiffness. It's the number of newtons of force you feel at node one when node two moves one meter. That's what it is.

Whereas, in general, k one one, for example, is basically what you think of as a stiffness. It's basically, if you grab node one, push it a meter – that might not be the right unit, but nevertheless, I guess you can grab the top of a building and push it a meter, a big tall building, but you grabbed it – you grab it, and you move it a meter, and it pushes back with some number of newtons. That's k one one. Okay.

So let's look at – we should also mention linearization. That's as a general source of autonomous linear dynamical systems. So some systems actually really are – really look like – actually have – the modeling of x dot equals a x is actually pretty good. There are a number of cases where that's true. There's a bunch of others where it's less true.

So let's – in general, you get things that look like this. X dot is f of x. That's an autonomous, time invariant, differential equation, and this could – this comes up all the time. This could describe an economy that's propagating, it could describe the dynamics of a vehicle or something like that, or anything else, or a circuit. This would describe – essentially all circuits would have a description like that.

And, here, f is simply a function from r n to r n, and, in fact, the meaning of f is simple. It basically maps where you are to where you're going. It's a vector field. Okay.

Now, an equilibrium point of a general, time invariant, differential equation is simply a point in state space where f is zero, and what that means is interesting. It says, if you're at one of these points, your derivative is zero, and that means you stay there. So it says that the constant solution x e satisfied the differential equation, because x e dot is zero, x e is a constant, and, on the right hand side, you plug in f of x d. That's zero too. So an equilibrium point corresponds to a constant solution of a differential equation. That's an equilibrium point.

Now suppose you're near an equilibrium point, but not exactly at it. Then you write, x dot, well that's f of x, but you're near this equilibrium point, so we'll use a first order Taylor expansion. By the way, to connect to a discussion we had last week, it would – you'd probably do better if you knew the approximate range of x wiggling around. You might, instead of using this matrix, use one that you got from sort of a particle filter method, which would be to say the – do a bunch of – evaluate a bunch of points and then fit a least squares model, but let's just move on.

So we'll just take the derivative, the Jacobian here, and this says that your f is about equal to the f where you are plus the Jacobian where you are multiplied by the deviation from where you are, like that. Okay?

Now we'll put this – this is zero, right here, like that. This is x dot, but I could just as well write this – I can subtract, if I like, x dot minus x equilibrium dot, because x equilibrium dot is zero, so I can do that. Very – a traditional term for that is delta x. That's, by the way, to be interpreted a single token. The question?

Student:[Inaudible].

Instructor (Stephen Boyd):I didn't say that.

Student:[Inaudible].

Instructor (Stephen Boyd):What?

Student:[Inaudible].

Instructor (Stephen Boyd):That was true, so I – but it's true. What I said was true this time, really. What I said was, if you start an equilibrium point, you will stay there, period. That is a true statement. You're getting onto our next topic. The question you're asking is exactly what we're gonna start looking at very soon, is what happens if you're not exactly at the equilibrium? What if you're a little bit off? And there's is actually, roughly speaking, two dramatic things that could happen. Actually, there's three, qualitatively.

One is that you – if you're a little bit off, you could actually sort of start moving back towards the equilibrium point. That would be stability.

Another one is that, if you're a little bit off, you actually start veering away the wrong direction. You move farther away. That would be an unstable equilibrium point.

And then there's weird stuff in the middle like, for example, it could just sit there happily and stay there. That's stability. Now what you would say then is, had I not made a mathematical statement, but had I made a practical statement, in general, you don't see – you will not see, if I have a mechanical system in front of me here or something like that, you won't actually physically observe a system in an unstable – sitting in an unstable equilibrium for obvious reasons, because the formula – the actual – the derivative – it's not really this, it's this. Right? Plus something like w of t, where w of t is tiny little noises acting on it. It really doesn't matter what they are. They could be even just from thermal noise or minor things.

If it's an unstable equilibrium and there's a little bit of extra noise here, the – you move off immediately. Once you're off, you start diverging, and you don't stay there. That was a very long and weird answer, but I go back and I say that I – what I said was correct. If you stay in a – start in an equilibrium position, you stay there.

Now we're gonna talk about what happens if you start very near an equilibrium position, let's see what happens – equilibrium point. So, here, delta x is this deviation, and you can write it this way. Delta x dot is d f, that's the Jacobian times that, and that's a – that is exactly what we have been calling a autonomous, linear, dynamical system, right?

By the way, a lot of people get bored with the deltas, and they drop it. So, for example, if you ask somebody who is studying aircraft dynamics, they'll just say, here's the system, and you'll say, what's x, and they go, well that's the roll angle, that's the roll rate, that's the angle of attack, the angle of attack rate, and all that kind of stuff, and you'll say, really? And they'll say, well no, no, no, it's phi. These are the deviations around level flight of a 747 at 40,000 feet and 580 miles an hour, something like that. That would be what the – so a lot of people just drop these.

I guess in electrical engineering, what do you call these little delta xs? What are the delta xs? It's called small signal, right? What do you call an equilibrium point?

Student:[Inaudible].

Instructor (Stephen Boyd):Bias. So you call it a bias, I think, right? Is there any – do they call it equilibrium point ever? I don't think so. No, I think you have a – well you have a little transistor circuit, and you figure out – they call it the DC? Something – DC operating point? That's a really old one, that there. So you can call it DC operating point or bias condition, and – let's see, and what do you call it in Aero Astro? I know there's people in here who are in that department. Isn't it called like the trim condition? I think it's called the trim condition, but anyway. You can – someone will correct me if I'm wrong.

So you'd be – I mean, all sorts of fields have their own names for an equilibrium point and things like that, but it's just that equilibrium point. That's sort of the high language, to describe this as an equilibrium point. These other ones are all just dialects. Okay.

So when you approximate a differential – when you approximate the right hand side of a differential equation, this is like forward oiler approximation. It's very – if you just approximate a function, you can say intelligent things about it like, well, you know, if you don't – if you – if delta x is smaller than such and such a thing, your error is no more than 3 percent. You can make specific claims.

When you approximate the right hand side of a differential equation, you might be in trouble, because in a – when you approximate the right hand side of a differential equation, you're really sort of – I mean, when you look at the trajectory, you're really building approximations on approximations, because you're really saying, where am I going right now? And I go, well you're going about in that direction, that's what d f of x e times delta x is. And then you go, great, so you step over here, and you say, where am I going now? And I go, well you're about in that direction. So the point is you're building up approximations on approximations on approximations, and so you might be lucky and you might not be.

So the best verb to describe how delta x dot equals d x of x e delta x, in what way that approximates the actual trajectory is – the best verb would be hope, so – and that's – which – actually that expresses it later, and we'll talk about that a bit – actually a bit later today.

Okay. So let's look at an example, is a pendulum, so we have a pendulum here and the angle it's hang – it's at an angle of theta, and that's gonna put a torque on it of minus l m g sine theta, so that's the torque. G is whatever, nine point eight meters per second squared or whatever gravitational acceleration, and that's the rotational inertia. So this is basically rotational inertia times the angular acceleration is equal to the torque on it. And the minus is that – it means it's a restoring torque, so it means that if you're this way, that appears to be how I drew theta positive, it says that the toque actually acts – it twists this – you can't see that. It twists that way, okay? So that's a restoring toque here.

Now we can write that as a first order differential equation, non-linear. This way its x dot is x two. X one dot is x two, x two dot is minus g over l sine x one, so it looks like that, okay?

Now the first thing you do when you see something like this is you need to do – the first thing you do is you analyze the equilibrium points. I mean, unless it's obvious, unless someone gives it to you and says, we're interested around this point. But here, so let's figure out what the equilibrium points are.

Equilibrium point says that this is f of x, here, and the question is, when does that vanish? Well if this vanishes, x two is zero. By the way, x two is the angular acceleration, so that says, at an equilibrium point, you're not moving. The pendulum's not moving. The second one says g over l sine x one is zero. That means that x one is a multiple of pi, and so that means, in fact, there's an infinite number of equilibrium positions, okay?

So you could have zero, and that's actually pendulum down, that's like this, like that. You could have pi, and that's pendulum up. You could also have two pi, but that's basically the same as pendulum down again, so it's kind of silly, but it's an equilibrium position. Okay.

Now, actually, we're gonna get to your question of like stability and things like that. You probably have a very good idea of how that – of how this works.

If a pendulum is hanging down, and you poke it a little bit and let go, it will just oscillate. There's no damping in this, so it'll just oscillate forever. On the other hand, what happens if you a pendulum straight up, and then maybe just give it a little – just knock it a little bit? What would happen, with no damping? I want you to integrate the differential equation by intuition, so – anyway, you know what happens. What happens? What?

Well it goes down here. It has a high velocity at the bottom. In fact, at the bottom, its potential energy is as low as it can get, so its kinetic energy – all of the potential energy up here has now been converted to kinetic, and it's moving fast, and then it goes all the way up to the other side and then – it depends how I hit it and all that kind of stuff, but if I hit it at a velocity like that, it will arrive with just enough to keep going and do it again, and it'll just oscillate like that. Right?

Had I done something like released it stationary one degree at 89 degrees from – 89 degrees, so – well from vertical – from horizontal. What would have happened is it would've gone like this, would've gone all the way around, slowed way, way down, got to one degree in the other direction, stopped for an instant, and then gone back, and it would just keep making a long oscillation like that, okay? So that's what would happen.

So let's look at the linearized approximation near these two equilibrium points. If you linearize in your equilibrium point, all I have to do is go back over here and take the Jacobian of this, so I take the partial derivative of this first row was vector x one, that's zero. The partial derivative of this is vector x two, that's one, here, and I fill in this matrix, and at the bottom, I take the partial derivative of this thing, with respect to x one, that turns this into a cosine, and I plug in x one equals zero, and I get minus g over l over here, like that, and then the partial derivative of this vector x two, that's zero, and I get this.

Now, actually, you'll know soon enough that this, in fact, defines an oscillation, and this one would be a pretty good approximation of what actually happens.

By the way, let's calculate the linearized system near x e equals pi. So let's do the pendulum up. In this case, delta x is – nothing happens up here, it's the same. That's a zero. And what'll happen here is, I take the derivative again, it's cosine, but I plug in pi now, and I'm gonna get the following. I'm gonna get plus g over l, and that's it. That's the difference between the two linearized versions of a pendulum in up and down position.

Now, actually, this is kind of interesting, because this one corresponds to a restoring torque. That's what this is. If you just simply work out what it means, it means that the second derivative is as proportional to your displacement, but with a negative sign. So if you're displaced two degrees – however you displaced, the torque is a restoring torque.

Do you see that? That is not a restoring torque, it's a – I'll just say it in English. What's the opposite of a restoring torque? That is a not restoring torque. That's what this is. It's a not restoring torque, meaning that once – if you move off, it's like, no problem. It puts a torque on it, but it puts a torque that exacerbates your deviation, okay?

And so you'd actually see it kind of all actually all makes sense. This predicts exactly what you pointed out, what happens if you're near there. It tells you if you're in the bottom, if you're in this mode, and you different – and you move a little bit, it's actually gonna have a restoring torque on it. Up here, you're gonna have, actually, a torque that pushes you away.

Okay, so this brings up the question, which we will look at later, I think, actually – no, I can't remember. Might be in a different class. Does linearization work? And the basic answer is, yes, but with some – there's some footnotes, and there is some – there are some legal – you have to kind of sign a release when you do a linearization, so there are some conditions.

So here it is. The answer is this. A linearized usually, but not always, gives a good idea of the system behavior near an equilibrium point, and to give an example where it fails, here's one. Let's take x dot equals minus x cubed. Well, it's a scalar differential equation. By the way, that's sort of a restoring – it's restoring. If x is positive, it says your derivative is negative. It pushes you down.

But it's quite interesting what this looks like, and forget the solution with – this is one of the 13 differential equations you can actually solve analytically. That's not relevant. It's much more important to understand what this says.

So x dot equals minus x cubed says this. If x is big and positive, what is x dot? If x is big and positive.

Student:[Inaudible].

Instructor (Stephen Boyd):Okay, really big and negative. I like your answer very, very much, except for one thing. Let's be a little more precise about it. Ready? Instead of really big, I would say it's really, really big and negative. You know why that? Because it was big cubed. So big is big, really big is big squared, and really, really big is – so I – your answer was right, but I – a slightly more correct answer would be, if x is big and positive, what is x dot? The answer is really, really big and negative. So it means if you're big, you are shooting towards the origin very quickly.

What happens if x is – you will – as you approach the origin, right, so x gets small, what is x dot?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. If x is small and positive, x dot is really, really small. So we can already predict what this differential equation does. Compared to x dot equals minus x, which gives you a solution which is e to the minus x, we can say, this thing, when x is big, this gets smaller way faster than an exponential. It shoots towards the origin, when – as x sort of passes through one. Once x gets small, this thing decays way slower than an exponential. Okay? So we got the qualitative behavior a very simple way.

If x is negative, it repeats, but you're – the point is, you're always moving towards the right place. Okay. And indeed, the solution looks like this. It grows – it falls like what – you know, one over a square root or something like that. One over square root t, which is very slow.

Okay. Now let's flip the sign, and study z dot equals z cubed. Now, here what happens is it's the same story, except, if you're big, your velocity – your actually derivative is positive and it's really, really big, so once you're big, you start accelerating upward – sorry, you really, really accelerate upwards. Okay?

So this one is actually sort of gonna be wildly – you can predict, just by looking at it, it's gonna be wildly unstable. Actually, this is so unstable it has a phenomenon called a finite escape time. What actually happen is the solution goes like this. It actually goes like this. And actually, at a finite time, it just goes to infinity. Okay? So – which is a fairly dramatic form of instability, which we, by the way, haven't formally defined. We will later, but that's – okay.

Now we know how the system really behaves, let's look at the linearization. If you linearize x dot equals minus x cubed near a zero, here's what you get. You get delta x dot equals zero. Why? Because you say, what is – you say, what's f of x when x is near zero? And the answer is, it's really, really small, but someone says, yeah, but first order what is it? Really, really small is zero, so it's this.

So it basically says, the linearization would predict that x is sort of constant. Actually, for this one and for this one, so, in a sense, neither is right. Actually, they're – neither of these – neither linearization predicts the correct long-term behavior, but actually, if we were to zoom way in on this, way down here, you'd actually see that both of them are correct in the following sense. For short times, they both give excellent predictions, because, actually, both the wildly unstable and the stable system once – when x is small, they, in fact, both are moving really, really slow. I said it right. Really, really slow near the origin. One is moving really, really slow, and increasing, and that's the one that, at some point, is gonna go through – is gonna get big and then go through a finite escape time. The other one is moving really, really slow towards the origin, and it's just gonna keep going and, very slowly, move to the origin.

However, most of the time, it makes good predictions. Actually, later, we'll find out exactly when linearization makes a good prediction.

Another version of this is linearization along a trajectory. So linearization along a trajectory – something like this, a linearization around an equilibrium point. This would come up if you're designing a circuit, if you are looking at something like vehicle stability or something, or vehicle dynamics, you wanna find out how does a vehicle do, what happens if there's a wind burst under an airplane or a wind shear or something like that, it's off a little bit.

By the way, it would also come up in bigger things like a – something like a big healing system or something like that. So you'd say you'd have a big network, and you could say, what if I – or a big – just take a big network, and you'd say, what happens if, all of a sudden, 10,000 packets arrived at that node destined for this one? I mean, that's supposed to be a small number, right? Compared to – but whatever. It would – so you could actually work out the changes in the queues and everything. It would be just – linearization would work quite well.

But now we're gonna talk about linearization around a trajectory. So linearization around a trajectory goes like this. I have a trajectory, and now I'm actually gonna consider a time varying differential equation here. So I have x dot equals f of x and t, and I have a trajectory. I have something that actually satisfied that differential equation.

So this could be, for example, the – give you the dynamics of a rocket, or something like that. It doesn't matter. Something like that, okay? So that's what it does.

Now suppose – and, in fact, this could be, here, some sort of – I – proposed trajectory – I – it doesn't matter. Some calculated trajectory. And what you want is you wanna take another trajectory, which is nearer the original one. I don't wanna be vague about what that means. It means, basically, at all times, you're never too far away. And we wanna work out what happens now.

So here you have DDT of the difference, that's x minus x trajectory. That's of x t minus f of x trajectory t, and that's about equal to the derivative – the Jacobian of f, or the derivative of f with respect to x of this times x minus x trajectory, and this gives you a time varying, linear, dynamical system. It looks like that. And that's called a linearized or variational system along a trajectory, and this is used constantly, always, constantly used to, for example, evaluate trajectories.

Here you have an idea like stability for an equilibrium point. So you would ask – you'd say, I just calculated a trajectory for a vehicle, and you'd say – then you'd ask a question like this. What if you just sort of get slightly off? What if there's a little wind gust, and you're slightly off? What will happen? And the question is, there's the trajectory you want, like this, and then the things that have been blown off, and so now it's small, and the question is, will the trajectories diverge? That would be one possible behavior. If they diverge, by how much will they diverge before this goes where it's supposed to go, or will, for example, the trajectories converge, for example? That would be something like a stability, and we'll see things like that.

So here's just a classic example is a linearized oscillator. So an oscillator is a differential system, a differential equation, a nonlinear dynamical system, which has a t periodic solution, so that's a – that's an oscillator with frequency one over t, and there's a solution which is t periodic, like this.

Well the linearized system is delta x is a of t delta x, and a of t here is actually periodic, because you – it's the Jacobian of this thing plugged in along the trajectory. By the way, there's a whole name – I mean there's a whole field for studying perturbations of periodic systems. It comes up all the time. It's called flow k theory. You don't have to know this, this is just for fun. It's called flow k theory.

So here a of t is t periodic, and so you have a t periodic linear system, and you would use this to study all sorts of things, and I gave you an example. The one that's actually quite important would be in circuits. So you might design, for example, an oscillator. It could be an LC oscillator, or it could a ring oscillator, or something like that. If you don't know what I'm talking about, it really doesn't matter for this – for the purpose of this example. You might built a ring oscillator, or something like that, and then you'd be – you'd ask a question like this. You'd say, what if – what happens in this thing if there's like thermal noise? Or what if some other circuit on the chip draws a lot of power and the voltage – the supply voltage to the oscillator drops 30 mV, for example.

That actually is gonna give you – it's gonna knock you off this trajectory, and now the trajectory – let's imagine the trajectory sort of looks like this, right? You're going around like that. You get knocked off, and one of two things happen. Well several things could happen. You might converge back in to this trajectory, or you might go – you might diverge. If you diverge, it's what we call a bad – it's a nonfunctional oscillator, right? Although you could ask interesting questions like this. How big a hit can you take here and actually reconverge to the solution? Okay? Actually, generally speaking, real oscillators will recapture from a huge range, but you can ask all sorts of cool questions like, when you – as you go back here, when you come back, you've actually – the time it took you to go around changed a little bit, that's called timing jitter, and so you might ask, how much does a 30 mV step in VDD change – affect the timing jitter. And in fact, that would be exactly analyzed by a linear dynamical system like this.

But is there anyone who knows what I'm talking about? Because, if not, I'll just stop talking about these things. That's no. For those of you watching this on TV, that's like, no. Okay, fine. So you don't care about circuits, right? No. There we got an actual, explicit – there we got a response, which is people shaking their heads no. Okay, fine. No problem. I'm not wedded to them either.

Okay, so this finishes up – maybe this is a good time to quit. If anybody has any last questions about the midterm? I'll take that as a no. So have fun. We'll see you tomorrow, or we'll see some of you tomorrow.

[End of Audio]

Duration: 74 minutes

IntroToLinearDynamicalSystems-Lecture11

Instructor (Stephen Boyd):Let me make a couple of announcements. I guess the first announcement is that the midterms are actually graded. So those are done and they'll be available for pickup in Packard after class today. And Denise, if she's in, you'll get them from Denise's office; her daughter was sick yesterday so I'm not sure if she'll actually be in today. If not, you can get them from my office or possibly, if I'm not there, from the TA offices – or TA's office or something like that, so we'll figure that out. But these are available in Packard, the midterms.

We posted the solutions, literally minutes ago. This is in Packard for pickup. We posted the solutions a few minutes before the lecture. And I'll tell you, let me just say a few things about it. Oh, I should say that usually the way that everything gets schedules out and everything, it had to do with a shift in schedule. By tradition, we return the midterms the day after the grade change option is – deadline has finished, that's the tradition. This time however, because the new schedule, we're actually able to give you graded midterms beforehand. That doesn't apply to the vast majority of people but there might be a handful of people who, I don't know, whatever, decide they want to change their grading option.

So I'll say a little bit about the midterms. I guess we can all speak freely about the midterms now. They were good. I do have to apologize formally on air because it seemed like it was – I think we undershot a little bit. Normally when people turn them in, we should see like every third person should kinda look like they just finished a marathon or something like that. And just by the people – two people were too happy and too civilized and all that, so we really feel that we undershot.

And I mention this because, you know, it's a very small world and I'm sure the rumors of this, this undershoot, have already made it to – they've – it's already at MIT and in Chennai and all these other places where, you know, people are gonna be – they – just people are gonna be disappointed. In fact, I already had one comment from current people who saw it – I mean, sorry, current grad students who saw it and they said, "They had that? That's not a midterm. Ours was a real midterm." So that's what they said, so anyway just wanted to apologize.

Nevertheless, it seemed like it provided some amusement for you, at least, so that's good. And you could hardly have done it and then say you didn't do anything, so I think that's not possible to say.

Let me make a couple of other comments about it. Oh, if you – and by the way, we do make mistakes grading, so sometimes we add things wrong, that's entirely possible; sometimes we've missed whole things or something like that. So do feel free to come forward and ask us about things, but only do that after you have looked at our solutions very, very carefully. And you better be prepared to defend yourself. We saw some serious nonsense, stuff that looked kinda right but was basically wrong in some cases, and so just be prepared to defend yourself if you feel something's not there.

By the way, we do make mistakes and so I would encourage you to look at it and in the – you should definitely do a postmortem, look at our solutions, look at yours, trying to figure out what happened, if anything. So that's – any questions about the midterm? How'd you like it? It's okay? Yeah, see that's – see, we really should get a much more visceral response, you see. That's how we know we didn't hit it just right. I mean it wasn't too far off but it wasn't, you know, it should produce like, at this point, it should be like a major traumo or something from the weekend. But all right, I'll move on.

There's another announcement. It has to do with numbering of exercises. I guess all of you know we went ahead and assigned the Homework 5. It's very short. Covers some of the material we're doing now. That's just to keep us in the loop on the homework. There is one problem.

In the printed readers, the – you know they're sort of done by lecture, they're numbered by lectured, so there's like Problem, you know, 2.6, 2.7. Anyway, they went up to like 9.25 or something like that, and then in the next lecture, 9.1 again. So just a mistake, it means that the numbering, but only the numbering, in the printed readers is wrong. So you're – just be aware of that.

We've updated the – the PDF file on the website we updated. Yeah not – I think that's it. But anyway, nothing else is wrong. The problems are right or whatever. So if you want to just look at it and please do Exercise 10.5, which Exercise labeled 9.5 in your reader it is, you're welcome to do that. Circle the correct 9.5 and then do it later or something. So that's it, okay.

That homework is gonna due Friday – is gonna be due Friday. So and in fact, I know this is big midterm week. Yes? This is big midterm week so there's an option, which would be to make it due next Tuesday.

Student:[Inaudible].

Instructor (Stephen Boyd):You'll exercise the option?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay and then that's done. Okay, so the Homework 5 will be due next Tuesday. But we're gonna pipeline here, so. Oh, now look, this is modern times, okay? You don't – you know you can't sit – that's how processes work. We're not gonna wait till you turn in Homework 6 before you – I mean 5 before you start Homework 6. That's silly, that doesn't work that way. So you're gonna do – in fact, you should've been doing speculative execution the whole time. You should've been guessing what exercise we might assign and do them ahead of time, just in case we might assign them. I mean for speed, that is.

Okay so we'll – you don't have to speculative execution, but we will assign a Homework 6 on Thursday. So and we might even back off on our natural tendencies on Homework

6, just a little bit, because of the – it's pipelined, so. So make the – the Homework 5 is due next Tuesday, and then Homework 6 comes out Thursday and then we're back on a Thursday-Thursday schedule. So how's that sound? Okay.

Let's see, I'm trying to think what – oh, the email I sent out about our progress in grading the midterms yesterday, no Sunday, I can't remember when I did it, Sunday. I sent it to last year's 263 class first. Didn't know it for two hours until somebody actually came – found us in Packard and said, "Oh, you know, thanks for letting us know about the grading but we took it last year." So I got a bunch of good responses from that, including some people who said that did they really have to do Homework 5.

So I sent a new email out to the entire class saying, "No, you don't have to do Homework 5." I mean I told them it will be on the final so they can choose to do it or not; it's their choice but they don't have to do it, so. Anyway, so if you don't – if you know other people who are asking what – like if, I don't know, if they ask what's wrong with your professor, I don't know, you can just say he's lost it. That's all. I guess that's the best thing to say.

Okay, any more questions about any of these things. Okay. We'll move on.

And we're gonna do one thing today but it's pretty cool and it's this. We're gonna look at autonomous linear dynamical system X dot equals AX, and we are gonna overload. So far, we've actually overloaded – that's a scalar equation. Everyone here knows the solution of that, that's X of T equals E to the TA times X of zero, like that. So we know – everyone knows this. We are gonna overload actually all of these things to the vector matrix case.

So we've already overloaded this scalar, simple scalar differential equation by capitalizing A and making A an n-by-n matrix and X a vector. So we've already overloaded the differential equation itself. Later today, we're gonna overload the exponential to apply to matrices. So that's our goal today.

And the cool thing – I mean the nice thing about – I mean what you want in overloading and extending notion, is you want it to suggest – you want it to connect to things you already know, so it should remind you of things you know, it should make you guess a bunch of things, only some of which are true. So that's how you – that's what real overloading should do. Right? That's how it should work.

If everything were true, then it's kinda stupid. You should've defined it more generally in the first place, and I don't even really call it a real generalization. So that's – if you really want to do it, you want to extend it in such a way that it suggests many things, some of which are true. So. Okay, so the first thing we'll do is we'll just solve this Laplace transform and I'll just review this very quickly, even though is a prerequisite, so here it is.

Suppose you have a function that maps R plus into P by Q matrices. So we're gonna go straight to matrices from scalars. And so, Z itself is a function that maps non-negative scalars into P by Q matrices. So it's a P by Q matrix valued function on R plus; that's what Z is.

Now the Laplace transform, that's written several ways. One is to actually have a calligraphic or script L, which is an operator. And it takes as argument a function of this form and returns the Laplace transform which is another function. It's a function from some subset of the complex plane into complex P by Q matrices. So that's what it is. Now it turns out for us we're not gonna worry too much about what this domain is. I'll say a little bit about that but not much.

So the Laplace transform is actually quite a complicated object. It's actually very useful, maybe just once to sit down and think about what it is. For example, how would you declare it in a computer language, right? So for example, C or something like that, just so you understand. It's very easy to casually write down, you know, little things with A-ASCII characters, which pack a lot of meaning. So L is itself a – it is a function. It is a function that accepts as argument something which is itself a function. It is a function, which accepts as argument a non-negative real number and returns a P by Q matrix. Okay?

So L returns, the data type it returns is it returns another function, this is a function, which accepts as arguments some complex numbers and returns P by Q complex matrix. Okay? So it's important to sort of think about this at least once. After a while, of course, you'd go insane if you thought about this every time somebody wrote down Laplace transform. And so, it's not advised that you should think of it all the time but you should definitely think of it once.

I should also add something here. And that is that the value of things like the Laplace transform, or at least it's shifting if not decreasing. Because a generation ago or two generations ago, this was actually one of the main tools for actually figuring out how things work, for actually simulating things and all that sort of stuff. It's not now, basically is not. So it's mostly to give you the conceptual ideas to understand how things work and all that sort of stuff. So things are shifting and it's not as important, I think, as it used to be.

By the way, there are those who scream and turn red when I say that. So. Okay.

Now the integral – here you have the integral of a matrix and of course, that's extended or overloaded to be term-by-term or entry-by-entry. And the convention is that the upper – an uppercase letter denotes the Laplace transform of the signal. This would be called maybe a signal; some people call that a time domain signal, something like that. Obviously, T does not have to even represent time here. Makes no difference whatsoever what this means. It often means time but it doesn't have to be.

Now D is called the domain or region of convergence of Z. This probably – I mean there's long discussions in books that are actually mostly, in my opinion, completely idiotic. I mean there's absolutely no reason for this discussion; it makes no sense. It actually also has no particular use these days, other than confusing students. So. So I'll say a little about this later, but.

It includes at least the – it's a strip; it's a right half-plane to the right of sum value A. And that value A is any number for which this signal Z grows slower than an exponential with A here, E to the AT, something like that. So that's what the domain is. It's at least that.

Now you might ask, you know, "Why do you even care about signals that diverge?" That's a good question. Actually, you need to care about signals that diverge for a couple of reasons. First of all, that might be a pathology in something you're making. So if you want the error in something to go to zero, tracking error or something like a decoding error to go to zero, and you design the thing wrong, then instead your tracking error will diverge. So it's a pathology and you need to have the language to describe divergence.

Also, by the way, there's lots of cases where, although it's often bad if a signal diverges, that's by no means universally the case. If you're working out the dynamics of an economy, then divergence is probably a good thing in that case. So.

Okay so let's look at the derivative property. There's only a few things you use in the Laplace transform. It says the Laplace transform of the time derivative of a signal is S times the Laplace transform of the signal minus the initial value. Now this is – it's the basic property. You know this is what Laplace – this is the whole point of Laplace transforms, essentially.

It's actually reasonably easy to just work out why this is the case. You look at the Laplace transform of Z dot, evaluated a function S, so that's a P by Q complex number. And it's the – by definition it's the integral, E to minus ST Z dot of TDT. Now integrate by parts and we say that this is E to the minus ST Z of T, so this is – I guess this is UDV. That's UV evaluated over the interval. Then minus integral VDU, and that's what this is here. Okay?

Now here, we're gonna use the fact that the real part of S is large because that's the domain that we're looking at. And that means that this goes to zero very rapidly, no swamp even if Z is expanding this is will swamp that up. By the way, if Z is growing at some – if you don't pick the real part of S large enough here, this integral, actually this integral has no meaning whatsoever. It does not exist. Okay? So this is not sort of a convenience here; it's because this has no meaning unless the integrand here is integralable. And if this is diverging, this – the only thing you can say is that simply has no meaning; it's like one over zero.

Okay, so this thing here, of course, goes – for infinity, it goes away and this becomes minus Z of zero because I plug in T equals zero here and it doesn't matter what S is. And this gives me S Z of S, so that's your derivative property. And now we can very quickly solve X dot equals AX. That's an autonomous linear dynamical system.

So what we're gonna do is this. We'll take the Laplace transform of both sides. And on the left-hand side, and these are all vectors, I get S capital X of S minus X of zero, and that's A capital X of S. X of S is the Laplace transform of X here. And what I'll do is I'll collect – I'll move this over to the other side, and I'll write this as SA minus – SI minus A capital X of S equals X of zero. Now I've isolated stuff I know, that's this, from what I want, which is right here, and it's appearing in the right way. And therefore, at least formally, X of S is the inverse of this matrix times X of zero.

Now we're actually gonna talk a lot about that but this matrix here, of course, you can't just casually write the inverse of a matrix. If you write the inverse of a non-square matrix that's just terrible. Actually as far as I know, no one did that for the midterm, which makes us very happy. So the matrix police actually didn't actually file any complaints, I think. Actually, that's true. I don't know that but I think that's true. Okay.

However, SI minus A can fail to be invertible. We're gonna get to that later. It turns out SI minus A is invertible for almost all complex numbers, except a handful. And we'll get to those – we'll get to the meaning of those soon, but for the moment let's just say this is for S minus A invertible, for the moment. And now we take the inverse Laplace transform and we have the answer. So X of T equals the inverse Laplace transform of SI minus A inverse times X of zero.

Notice here that X of zero comes out. There's a question?

Student: Does it make any [inaudible]?

Instructor (Stephen Boyd):We are saying that, yes. That's exactly right. Right. So I'm using linearity is the other thing I'm using here which I didn't mention but probably should have. So I'm using linearity of a Laplace transform, and now this is the matrix vector case.

The way you can check it is very simple. The Laplace transform is an integral, entry by entry. So and then if you work out what a matrix vector multiply is, just write out with all the horrible indices, then stick the – the integral appears outside the sums. Put the integral inside, and then recognize it for what it and then put it back outside and so on and so forth, and you'll get that.

I don't know if that made any sense, but anyway. I'm using linearity of the Laplace transform. Okay.

So you get this. Okay, now actually a bunch of things appearing here are very famous; they come up in zillions of problems. This matrix SI minus A inverse comes up all the time. Lots and lots of fields, and it's called – so the mathematical – in mathematics it's called the resolvent of A. Notice it's a function of S, this complex number. So it's a function, it's a complex square matrix valued function. It's the resolvent, SI minus A inverse.

Now it's defined, of course, whenever this is invertible. If it's not invertible, of course, this inverse has no meaning here. So the places where it has no meaning are actually called the eigenvalues of A. And these are complex numbers for which det SI minus A is zero, so these are the eigenvalues of A. We're gonna say an enormous amount about this. There's only N of those or fewer, and we'll talk about that later.

So when you see SI minus A, you really have to – you have to understand when you write this – SI minus A inverse, sorry. When you see SI minus A, there's just no problem at all. When you see SI minus A, you have to have the understanding here that this – what you mean is, this expression, you don't want to put a little star or something like this and have a little footnote down here that says provided, "S is not an eigenvalue of A," or something like that. It gets silly. It's basically like if you write down a function like this: S plus two squared over S minus one. Okay?

And you don't want to write – you know every time you write this, you don't want to have a footnote that says, "Define for all S except S equals one." So after a while you get used to it and you just write this. By the way, you can get into trouble by forgetting that, in fact, there is in fact, a footnote there for this one. There's a footnote and the footnote says: whatever form, you know whatever you're doing, you plus in S equals one here and all bets are off. Okay? So there is that footnote.

But as long as you just remember that that footnote is in place, everything is okay. And the same thing is true here. So we'll write SI minus A inverse, that's the resolvent of A, and it should just be understood that there are up to N complex numbers for which this is not invertible and you shouldn't be writing the inverse. Okay.

Now when you take this inverse Laplace transform here, this thing, that is a matrix valued function of time. It's gonna have a name real soon, but it's got a name – first of all, we're gonna give it a general name. And it's the state transition matrix, and it's denoted as phi of T. Okay? That's just – it's called the state transition matrix and it looks like this. It says – it's actually already – we already have an interesting conclusion. We see that the state at any given time is a linear function of the initial state. So not surprising, it's a linear differential equation but there it is.

And if it's a linear function's initial state, it's given by matrix multiplication. There's some matrix – in effect, the matrix is the state transition matrix. Okay? So we get that. We'll be able to – I mean you can actually work this out. This is nothing but this – you know everything here in principle. You can take a matrix A, you can calculate SI minus A inverse, at least in principle, you can take the Laplace transform of that if the entry's a rational functions, you can go get some Laplace transform table, take the inverse.

So in some sense, it's done. You now know the dynamics of linear dynamical systems – I mean of autonomous linear dynamical systems. You know everything now, in some weird theoretical sense. Okay.

So this is called the state transition matrix, so let look at some examples real quick. First one is this one. It's harmonic oscillator, is the name of the system. And it looks like this. It says X one dot is X two, X to dot is minus X one. Okay? Now if you plot the vector field, it looks like this. And here it's certainly plausible that the trajectories are circular, plausible but it's not quite – I think it's not quite circular. Sorry. Scratch that.

Actually, I just had a discussion this morning with the people doing the video production. And so I said, "I'd like you to just remove – when I say things like that, just remove, so it never happened." And then they said, "Oh no, no. There's huge, huge expenses associated with that, so I can't remove these now." But that's the kind of thing, by the way, I'd like to remove. All right, so let's just pretend – let's just rewind, pretend I didn't say that and go back.

When you look at this, you can imagine, with your eyeball I guess, that the trajectories are circular or nearly circular, something like that. Now it turns out they're actually circular. We'll get to that. Let's see how this works.

So we form SI minus A, well the inverse, this is the one inverse you should've kinda know by heart, certainly. Well there's a few special cases but two by two inverse everyone should know. It's one over the determinate and then you – I guess you switch these and negate these, so that's one thing – that's reasonable to know.

And so SI minus A inverse, that's the resolvent, is this. And notice that this matrix makes perfect sense for all complex numbers except plus minus J. But J is just because this – in this course is officially listed in electrical engineering. This should be I. So you really – the truth is, in mixed company, you shouldn't use J because it's very – outside electrical engineering, it's very – it's a dialect so you shouldn't really use J.

And my feeling is you shouldn't use J in mixed company. Okay? So that's – but because the course is in EE, I'm gonna use J, so. But I'm making it explicit. That's this is not the high BBC mathematical phrase that would be used. That is absolutely universal in all fields, except electrical engineering where you have this. Okay?

And the reason, I think it goes back 100 years, an I apparently represented current. Now how I got connected to current, I do not know but nevertheless, it got – the two got stuck together in the late 19th century and then here we are 120 years later with J. So. Sad but – okay. I might change that someday because it's a bit weird but not this quarter, so. Okay.

Now the state transition matrix is you simply take the inverse Laplace transform of this. No problem. You go look up in some table or something like that, and you'll find that the inverse Laplace transform of entry-by-entry is cosine T sine T minus sine T cosine T.

And you saw this matrix before. That is a rotation matrix of minus T radiants; that's what it is. Okay? So that – it simply rotates.

What that means is this state transition matrix, and let's remember what it does, it maps initial states into the state T seconds later. This matrix here, it simply – it takes a vector, the initial vector, and rotates it negative T radiants. Okay? So that's what it does.

So we've actually now verified that the motion in this system is a – is perfect periodic motion. You simply take a state vector, the initial vector, and you just rotate it at a constant angular velocity, in fact, of one radiant per second here. So that's our complete time domain solution.

Now by the way, I want to point something out right off the bat. We are generalizing this. That's a scalar differential equation. Now the solutions of this look like E to the AT – oops TA. You'll know why in a minute, I keep writing TA. So that's the solution, okay? Now qualitatively the solutions of a first order scalar linear differential equation are pretty boring. Basically, there's only three qualitative possibilities.

No. 1, if A is positive you get exponential growth. If A is negative, you get exponential decay. If A is zero, you get a constant. Okay? There is no way out of this thing you can get an oscillation or any other kind of qualitative behavior, other than growth, exponential growth, exponential decay, or constant. Well this is X dot equals AX. A is two by two. And we just got something out of a first-order linear differential equation that you are not gonna get out of a scalar one. We got oscillation. Okay?

So when you generalize, when you go to vectors and you look at a first-order, when you look at the vector version, X dot equals AX, you get solutions that don't just have exponentials in them, they can have cosines and sines. Okay? So. I mean this is kind of obvious but I just want to point it out that our generalization here has already – you've already seen behavior you could not possibly see in the scalar case.

Okay, next case is double integrator. So for double integrator, you have X dot is zero one zero zero X, like that, so X one dot is X two, and X two dot is zero. And the reason it's called a double integrator is the block diagram would look something like this. And this is gonna be maybe X one and maybe that's X two. Everyone agree with that? Because X two dot – this is zero going in. X two dot, is over here, is zero. And X one dot, that's what went into the integrator, is X two. And I think that's this. So that's the block diagram.

In fact, when you saw this matrix, you should have had an overwhelming urge to cause a block diagram. Come to think of it, that should've happened here too. So let's just do this one for fun. Here's X one and, oh, I'm gonna try to do this right, that's the output, that's X two. Okay? And that's one over S.

So let's read this one. It says X one dot is X two, so I'll just connect up this wire here like that. Okay? And this one says X one dot is minus, so I'd put a minus one like that. There you go. So that's our block diagram for this thing, so that's the picture. So it's basically two integrators hooked into a feedback loop with a minus one in the feedback loop. That's what it is.

Okay. Double integrator, that looks like this. And the solution, you know, is totally obvious. You don't need to know anything. I mean you certainly don't need to know anything about matrices and things like that to solve this. If X two dot is zero, X two's a constant, but if X one dot is a constant, X one grows linearly. It's a constant plus the second constant times T. So the solution, we could just work out immediately, but let's just see if all this Laplace and other stuff works.

So, oh, here's the vector field, which shows you what it is. If you start here, depending on your height, that tells you how fast you're moving to the right, or if you're down here you're moving to the left, and so on. Let's work it out.

SI minus A is equal to S minus one zero S. That's a two by two invertible – sorry, upward triangular matrix; you should be able to invert that. SI minus A is this, one over S, one over S squared, zero, and one over S. Now this is defined for all S except for one complex number, which is zero. You can list either zero or a pair of zeros, that's – we'll see why that is in a minute, but I should really say something like the only eigenvalue is zero. At this point, I should say that.

Okay. So that's that. Inverse Laplace transform is this: phi of T, that's the state transition matrix, it's one T zero one. Okay? By the way, you've now seen something else that is just absolutely impossible in the scalar case. In the scalar case, if the solution of X dot equals AX grows, it grows exponentially; it cannot grow linearly in time. That's for a scalar.

And yet, in the matrix case, you can have the solution of X dot equals AX grow linearly in time. Look at that. Okay? So I just – it's very important to point out that you're seeing qualitatively different behavior than you could possibly see in a scalar differential equation. This is not a big deal. If you worked out what this is, this says X two of T is X two of zero. We knew that because it was constant. And then it says X one of T is equal to X one of zero plus T times X two of zero. That's obvious because that's the derivative of X one. So it all works out and makes sense.

Okay, so let's – let me just have some quick questions here about this matrix phi of T. We'll get to this. Let me ask the following. What does the first column of phi of T mean? What does it mean? What's the first column of phi of T?

Student:[Inaudible].

Instructor (Stephen Boyd):It says what? It's what? X one –

Student:[Inaudible].

Instructor (Stephen Boyd):Correct.

Student:[Inaudible].

Instructor (Stephen Boyd):Okay, so what does the first column of phi mean? It has a meaning.

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, it says the following. The first column of this matrix tells you what the state trajectory is if the initial condition was E one. That's what it tells you. Okay? What does the first row of phi of T tell you?

All right, let's write down this. Phi of ten equals zero, zero minus one, 30, 5, and, of course that's got to be a square matrix, there you go. Strange placement but anyway let's live with it. What does that mean? That's phi of ten. There's a very specific meaning. What does it tell you?

Student:[Inaudible].

Instructor (Stephen Boyd):Which state at ten?

Student:[Inaudible].

Instructor (Stephen Boyd):All of them?

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. This tells you this row is what maps the entire vector X of zero into X sub one of ten. That's what it does. So otherwise, I agree with your interpretation. So now, give me your interpretation again.

Student:[Inaudible].

Instructor (Stephen Boyd):Exactly. So these two tell you, well at least to the precision I've written them down. It says that X one of ten doesn't depend on the first two components of the initial state. Okay? This says it depends a whole lot and positively on the fourth component of the initial state. Everybody got this? So. Okay. And this says the first component of the third state actually has sort of an inhibitory affect on X one of ten.

By the way we're gonna see interesting things where when you plug in ten you get one thing, and when you plug in 100 or 0.1, you get something totally different. So now, you can actually say things talk about when something has an affect, when an initial condition has an affect. Okay? So all right. Okay.

So let’s talk about the characteristic polynomial. This is also very, very famous. The determinate of SI minus A comes up all over the place and it’s called the characteristic polynomial of the matrix A. Sometimes you put a subscript here to determine this. This is absolutely standard language, so this is not some weird strange dialect from electrical engineering or something.

So it’s called the characteristic polynomial. And it’s a polynomial of degree N and it’s got a leading coefficient of one. So by the way, some people call that it’s a monic polynomial. Not some people actually, actually just people; it’s called a monic polynomial, which means its leading coefficient is one.

And you can check that, I don't know, over here for example. Det SI minus A, it’s the determinate of this thing, and it’s just S squared. Well that’s about as simple as characteristic polynomials go. Let’s see let’s do this one. This is gonna be – that is the same one. We’ll do this one. So the characteristic polynomial of the matrix, which is zero one minus one zero, the characteristic poly is det of this thing, which is of course S squared plus one. So that’s the characteristic polynomial of this thing.

Okay, so that’s the characteristic polynomial. And the roots of this polynomial, basically by definition, these are the eigenvalues of a matrix A. Okay? So how many people have seen this somewhere else? Okay. So this is the – yeah, this should be review. So the roots are simply defined to be the eigenvalues of the matrix A.

Now this matrix has real coefficients here; I mean assuming A is. By the way, sometimes you look at complex linear dynamical systems, they do come up, they come up in communications, they come up in, for example, in physics, and they come up in all sorts of places. But generally speaking, we look at the real case, and then on an exceptional basis we’ll look at what happens in the complex case. All right? So A is, if I don’t say anything else it’s real.

So this has real coefficients. Now, the polynomial – a polynomial with real coefficients has a root symmetry property. It says that the roots are either real or they occur in conjugate pairs. In other words, if lambda is a root and it’s complex of this characteristic polynomial, so is lambda bar, it’s conjugate. Okay. So now, you can see why people talk about N eigenvalues.

When you have a polynomial of degree N, maybe the correct way to say it is something like this. You can have anywhere between one and N roots of an Nth order polynomial. Okay? It could be a full N and it could be just – it could actually just be one root. And a good example would be S to the N. The only zero of this polynomial is S equals zero. Okay?

Now what people do is they actually – in order to make the aesthetics of the fundamental theorem of algebra that says that a polynomial degree N has N roots, to make that, you know, that statement have no footnote, you have to agree to count the multiplicity of the roots here. And so this, here would count N of them, and then you can actually make the beautiful statement that an Nth order polynomial has N roots. Of course, they might all be the same but that's dealt with elsewhere. Okay.

Now if we resolve it, which is SI minus A inverse, it is likely I guess, that you were at some point tortured with something called Cramer's rule. Is that correct? This was his method for inverting matrices where you crossed out rows and – you cross out a row and a column, took the determinate of what was left, and then you divided by something else and sometimes you put a minus one in front of it. Is this – yeah, how many people actually saw that? Okay. How many people know how useful that is?

Student:[Inaudible].

Instructor (Stephen Boyd):Do you know how useful it is?

Student: Yeah, it solves equations. [Inaudible].

Instructor (Stephen Boyd):It solves equations. Yeah. It was useful only for you to take that class. It has no use of any kind. Well, other right now, briefly, we're gonna use it but not really. No, it has absolutely no practical use whatsoever. No, under no circumstances are linear equations solved using this method, at least after the late – mid to late 1820s. So no, no, no, that's not true. People maybe all the way up into the '40s or something, but only because they didn't know what was going on. And yet, there it is in the curriculum. There it is; might as well teach people how to do long division with Roman numerals. That would actually be more useful, come to think of it. So anyway, all right.

Sorry, pardon me. Okay.

So this rule basically said this that it said take SI – to calculate this matrix, you cross out a row and a column or something, like the Jth row in the Ith column, of this matrix. Calculate the determinate, that's this thing. Divide by the term of the whole thing. Well at least we have a name for that, that's the characteristic polynomial here. And then you put a minus one to the I plus J in front, and that gives you the –

Now I don't actually care to do this. This is computationally completely intractable in any case because the number of turns in this is growing hugely and the whole thing is silly. There's one thing I want out of this, and that's this: that every entry in the resolvent is rational function and they have all the same denominator, which is the characteristic polynomial. The numerator is another polynomial. It's the determinate of SI minus A when you cross out the, I, whatever, one row and one column, and take the determinate of what's there. That's what that is. Okay?

Now when you do that, the degree of this numerator polynomial is less than N. So what that says is that every entry of the resolvent looks like this. It looks like a polynomial of degree less than N, divided by this polynomial whose degree is definitely N. Because this thing, the coefficient of S to the N is one, in chi of S here, that's one. Okay, so they all look like that.

Let's see. There's a name for that. If you have a rational function, which is a ratio of two polynomials, and the denominator has a bigger degree than the numerator, it's called strictly proper. So again, don't need to know this but that's just what it's called, so every entry of the resolvent is strictly proper.

One way to say that is as S goes to infinity, the entries of SI minus A all go to zero. Okay? Which is kind of easy to see – well is it? I don't know. It sort of makes sense. As S goes to infinity, you get sort of like, you know, huge numbers times I minus A inverse. And it's plausible, at least, that that should be a matrix that's small. Okay.

Now comes the tricky part. It turns out that not all eigenvalues of A are gonna show up as poles of each entry. Because although each entry looks like this, here's what's gonna happen. In some cases, the numerator polynomial will also have some of the eigenvalues, the roots of chi, and those actually will cancel. Okay? So you'll actually not get the – I think this will be clearer with examples and things like that. Let me see if I have one here. Oh, I did have one; aha, yes, we have one. A perfect example, if I can find it. Here's our perfect example. Great. Okay.

Eigenvalues are zero and zero. Here is the resolvent, okay? That's the resolvent right there. Now I'll ask you about the poles of each entry of the resolvent. What are the poles of the one-one entry? Zero. Well sure, they're the eigenvalues. Okay. You could say the poles here is – you could say zero and zero for this. Right? You can say zero and zero and those are the eigenvalues, no surprise here. But now I ask you about this entry, the two one entry. What are the poles of the two one entry of the resolvent? There are none. Okay? So the two one entry is a case, where there's an entry in the resolvent that does inherit a pole from the set of eigenvalues.

Now what if this had looked like this, like that? What would you have said? Well if I'd asked what the poles of the two one entry now, what would you say? One. And then what would you say? You'd say it's impossible because the entries here, the poles have to be among the eigenvalues but it doesn't have to include all of them, as this zero entry shows. Okay. The significance of that, I think just take only examples and fiddling around is gonna make it – with these things, is gonna make it clear. Okay.

Next topic is this. We are now going to overload. Oh, by the way, we have overloaded this. If you didn't remember how to solve that, but that's the scalar case, but for some reason you didn't know how to solve this but you did remember all about Laplace transforms. I've always found that a little bit implausible but anyway let's just go with the story; let's go with that story. You would've said oh, you know, S capital X of S minus X of zero equals A, you know, X of S. And you would've gotten a formula that looks like this. X of S is X of zero divided by S minus A. Did I do that right? Something like that, you would've got that. Right?

And I'm allowed to write this because these are scalars. Okay? I mean you now know you would write this – you know that's – that is the scalar version of that. Okay? But this is what it looked like when you took an undergraduate class. And then someone would say, “Well what – so what X of T?” And you’d say, “Well it’s the inverse Laplace transform of this.” Okay? So we’ve just worked out all of that. We’ve overloaded it now to the matrix case. And the only thing is that what had been a fraction like this, this SI minus A, I mean pretty – look, it couldn’t really – really couldn’t have worked out many other ways. It came out in front as SI minus A and has its own name, which is the resolvent.

Okay but now we are going to overload the exponential. All right? So we’ll start with a series expansion of I minus C inverse. This is actually the matrix version of the scalar series you’ve seen. So I minus C inverse is I plus C plus C squared plus C cubed. And that’s if the series converges, actually quite soon we’ll know exactly when it converges. But it certainly converges when C is small enough. It’s small enough that the powers of the Cs are getting smaller fast enough. Then this for sure converges.

Student:[Inaudible].

Instructor (Stephen Boyd):I said we’ll get to it later.

Student: Oh.

Instructor (Stephen Boyd):So it’s in a sense actually that in one lecture I’ll be able – you’ll know exactly what it is. I don't mind. I’ll tell you. The absolute values of the eigenvalues of C, which are – have magnitude less than one. That’s the exact condition. Okay? So.

Okay so let’s look at this. And, you know, how would you show this. You’d show this by terminating the series at some point and then multiplying, you know, telescoping the series. You’d multiply this by that and finding out that what would be left over would be C to the N, where N is where you truncated it. And then if C is going to – if C to the N then gets bigger goes to zero then you’d get this. So you have that, that’s your series thing. And we could just take this as formal.

Now let’s look at SI minus A inverse and let’s do this. Let’s first pull out S out of this and we get I minus A over S inside. And we pull the S out which becomes a one over S outside, looks like that. It’s a scalar. And you get I minus A over S, now that’s this formula here. And I’m gonna use this power series expansion, here, of I minus C inverse. And if anyone bugs me about convergence, I’ll wave my hands and say, “Oh, yeah. Right. This is only valid for S large.” Okay? That’s how this is gonna – if anyone bugs me about it, that’s what I’m gonna say.

Okay? Because if S is large A over S is small, and then in the way in which I didn’t say if C is small enough this will converge. Okay, so we get this. And this is simply I plus A over S plus A over S squared plus A over S – oh by the way, of course that’s slang. Right? Everybody recognizes that? That’s considerable slang but a lot of people write it. Maybe the correct way to write that is this, but then you get too many parentheses and it starts looking really unattractive and stuff like that. So but I figure now, post-midterm, I can be little bit more informal, so that's slang, just wanted to mention it. So.

I still, I don't know that I can actually still take things like this. That just looks weird for some reason. You know maybe I'll get used to it or whatever, but. And this looks kind of sick and I just like why would you do that. I don't know it just seems odd, anyway. So. But for some reason this just – the S, this seems to flow, so. And it sure beats that because it'd be a lot of parentheses otherwise.

Okay so I write it this way. Oh, that's slang too. There we go. See? Right there. That's a lot of slang but that's okay. You know what's meant by it.

So you take this series expansion, and now let's take the inverse Laplace transform term by term. Well if I do that, the inverse Laplace transform I over S, that's easy, that's I. A over S squared, that's easy, that's TA. Then A squared over S cubed, that's TA squared over two factorial, and so on. So I get a power series that looks like that, okay?

Well that's interesting because that looks just like this, E to the AT, like that. Except I'm gonna start writing these as the E to the TA. You'll see why in a minute. E to the TA; looks just like that. So here's what we're gonna do. We're simply going to define. All of that was just sort of little background. We're simply gonna define the matrix exponential this way. E to the M is I plus M plus M squared over two factorial plus M cubed over three factorial, and so on. Okay?

Now just the way the series for – the power series for the ordinary exponential for a scalar real or complex number converges for any number. Right? Any number, even a big number, what happens is these terms get way big. It will – they will converge though, okay? Same way. It's true for all – so this series converges for any matrix M. How does the – how well does the series do for non-square matrices? X of a two by three matrix, what is it?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, it just – it makes no sense. And, in fact, where would be the – where in the syntax pass would you halt?

Student:[Inaudible].

Instructor (Stephen Boyd):Here? You'd stop already right here. I'd stop by the minute I parsed, when I got to – when I pulled the token M off and then asked somebody somewhere to add a two by three matrix with an identity, that'd be the problem. But you're right, I could say, "You know what? I'm gonna let one go. Just keep going." And then the M squared –

Yeah, it's like, "No problem." Right? No, that's actually what compilers do, right? They try to get through as much as possible because the more they can get through, maybe the more informative their description of the exact kind of idiocy you suffer from they can describe. So, you know, you'd say, "Okay, fine. This person is adding an identity with a two by three matrix. No problem. Let's just keep going." And then, indeed, you'd get the M squared and you'd say, "All right, I know what we're dealing with here." And then you return with a nice message. Okay.

So yeah, so matrix exponentials don't – they don't exist, but for any square matrix they exist. By the way, when you do an overloading, we've now just overloaded the exponential. It takes as argument a square matrix. Okay? Whenever you do an overloading, you want to check that in any context where the two different contexts overlap, they better agree. So for example, if someone walks up – you know if someone says XA, and that's a scalar, I mean there's this weird thing where you could say, "No, no, no. It's a one by one matrix." And you have to make sure it's the same thing. But of course, it is the same thing, so everything's cool here. Okay.

All right, so that's the matrix exponential just defined for any matrix, and now it turns out that's just what – that's what the state transition matrix is. It's E to the TA. And so what we've done is we've come around and we've figured out the following. The solution of X dot equals AX is this: it's X of T equals E to the TA X of zero. And I'm gonna try to do this. You know the problem is I guess if you learn, or teach in my case, you know, the undergraduate classes, they are all – they always looks like this. It's always E to the AT. Did people see that? Is that what you saw?

You know it's kind of like cosine omega T. Right? There's nothing wrong with a person writing this but it's just weird and kinda – it goes against convention and I don't know what. Does everyone know what I'm talking about? Okay, so for some reason, I have no idea why, you put the T like this. So that was so ingrained in me from teaching undergraduate classes that for a long time I wrote E to the AT. And actually, a lot of people will do that. But that's kind of – you know that's weird. That's that the post – the scalar post multiplication of a matrix.

It's cool in some – you know depending on the social situation it can be okay to post multiply a matrix by a scalar. Certainly among friends, on weekends, I don't see any problem with it. But it just somehow it's not right, so I'm now retraining myself to write this as E to the TA. I don't know, just so that when I then teach this class I have that. So that's the – so anyway, I'll slip up a few times and that's fine. Okay.

So there you go. Now we have a name and we know that the solution of X dot equals AX is E to the TA – is X of T equals E to the TA X of zero. Feel free to have – when A is a scalar, this goes back to your undergraduate days, there's nothing here you didn't know about, when A is a matrix, that's the matrix exponential. Okay? So it's not and it's the solution. So okay, there you go.

So the solution of that is this. Now that – as I just said, that generalizes the scalar case; note written here is TA. Now a couple of warnings here, and in fact this is what this makes this fun. If in fact everything just worked out, it wouldn't really be fun. And it wouldn't – if it didn't really require like outer cortical activity, I mean if it was just notate, it's not interesting. So here's the idea behind this.

The matrix exponential, it's meant to – of course, it's meant to look like the scalar exponential. That's absolutely by design it's supposed to look like it. Okay? Now what that means is that things you would guess, some things you will guess from your knowledge of the scalar exponential, hold. Okay? I'll show you one right now.

So for example, E to the minus A is, in fact, E to the A inverse. That's true, okay? But there's lots of things from your undergraduate scalar exponential knowledge base which doesn't go – doesn't actually extend to – it absolutely does not extend to the matrix case. So here's an example. You might guess that E to the A plus B is E to the A E to the B. That is absolutely the case if A and B are scalars it is false. In general, in fact for almost if you randomly pick and A and B, it will be false.

By the way, you will know soon why, when you understand the dynamic interpretation of what E to the A means and you thought about it carefully, other than as a – as opposed to notationally, you would not even imagine that this would be the case because it's making a very strong statement.

Anyway, this is false. Quick, we've actually worked out explicitly two matrix exponentials so we'll use that work. If A is this thing, E to the A is whatever that – it's a one radian – that's a one radian – negative one radian rotation matrix. E to the B is this thing. That's just straight from our formula. You work out what E to the A plus B is. We did not work that out but I worked it out to a couple of significant figures, and it's not equal to the other way around. Okay? So it's just – they're just way different animals. Okay? So be very, very careful with the matrix exponential, and with actually a bunch of the other stuff that we've overloaded.

By the way, you know this is not – it's not like you haven't seen this before. And I show you an example. You know, for example, that if these are scalars and I say something like AB equals zero, you know that either A or B is zero. That's true. But if A and B are matrices, this is – it is false that either A or B is zero, just false. Now it becomes true with some assumptions about A and B, and their size and rank and all that stuff. But the point is, it's just not true that that implies A equals zero or B equals zero. And you kinda – you know after a while you get used to it and that's kind of – same thing for the matrix exponential, so it's not like you haven't seen stuff like this before. Okay.

However, if A and B commute, so if AB is BA, if matrices commute, then in fact this formula holds, okay? And that's easy to do. You just simply work out the power series. You take the powers and then you're free to rearrange the As and the Bs and you can make this power series look like that. Okay? So, and that tells you immediately the following. If you have two numbers, T and S, then E to the TA plus SA is actually E to the TA times E to the SA, like that. Okay? And if S is minus T, you get E to the TA times E to the minus TA that's zero. Okay?

So that says that the exponential of TA is non-singular always and it has inverse E to the TA, inverse it just says E to the minus TA. This'll make a lot of sense in just a – momentarily. All right.

So how do you find the matrix exponential? Well let's take zero one zero zero. There's lots of ways to find it. You can start by – we already worked out E to the TA so that's kinda silly; we just plug in T equals one and we get this. But we can also do it my power series. So by power series, we just take I plus A plus A squared over two. What is A squared for this A? It's zero because this matrix is – oh okay, all right. Someone give me the English for what that does, give a name for that matrix, what it does. What does it do to a two vector?

Student:[Inaudible].

Instructor (Stephen Boyd):What does it do? I think I heard it.

Student:[Inaudible].

Instructor (Stephen Boyd):Shift up. Okay, let's call is the upshift matrix. So that's the upshift matrix. It takes a two vector, pushes the bottom entry up to the top, and fills in with – and zero pads, so fills in a zero for the bottom entry. That this – that's the – so if you do that twice to a vector, there's nothing left. So A squared is zero, any vector; A cubed is zero. And actually, now this something you don't see this in the scalar case.

In the scalar case, when you work out the infinite series for the exponential, it's infinite, oh, except for one case, when the argument is zero. But other than E to the zero, that series is infinite. Here, for a non-zero matrix, the series was finite. It only looked like an infinite series. It was finite. So that's one way to get this matrix exponential. Okay.

Now the interpretation – how many people have seen the matrix exponential, by the way? I'm just sort of curious as to how many. Some were. Okay, so. All right.

Now here – oh and I should say – let me say a little bit – let me just give you one warning about this and that's this. If you type XA in MATLAB, for example, but actually in many systems, what you'll get actually is it's not what you think. What you'll get here is actually a matrix that looks like this. It's E to the A one one, E to the one two. It's basically exponentiating all the entries.

Now let's forget the fact that there's probably one out of 100 million possible cases where you'd ever want to do such a thing. Okay? But nevertheless, that's what happens. Just to warn you. So in fact, the way this is – it's actually XM of A. And that means the matrix exponential. So that's how – that's what people call this.

So just be aware of this when you start – you will be fiddling with this, so just be aware of it. And you'll make this mistake. There'll be many ways to check what you're doing. By the way, the two would agree in, I think, in almost no cases or something like that, so.

But the worst part is it could – you might get something that's like plausible or – that's the worst part, so you just have to check and be aware of this. Okay.

By the way, the way to compute the matrix exponential, it is not done by any of the methods. Nothing is computing a Laplace transform, I assure you. You'll know soon a little bit how it's done.

It turns out it's actually not that easy to calculate the matrix exponential. And there's some – there's a wonderful paper about computing the matrix exponential. And the title is 19 Dubious Methods for Computing the Matrix Exponential. And they go through, it talks about 19 methods that people have used, shows how each one can, in the wrong circumstances with the wrong A, give you like totally wrong results and things like that. So that's it, so. But for paper titles, I thought that was – that's right up there, I think.

Okay so we'll be able to finish today. And actually, it's very important to actually, I mean know what is the meaning of the matrix exponential, and this is extremely important. It's this. So far, it has a very specific meaning. E to the TA is an n-by-n matrix. It maps the initial state of X dot equals AX into the state at time T. So I think of it as a time propagator; it propagates from initial time, to time T. Okay?

Now it turns out, actually – it – you can actually work out the following. That X of T tau plus T is equal to the E to the TA times X of tau for any tau here. So in fact, the matrix E to the TA is a – it propagates a state, forward in time T seconds. It propagates X of zero into X of T. But for example, it will propagate X of 17.3 into X of 17.3 plus T. Okay? This times E to the TA is gonna equal that because this propagates a state of linear system forward T seconds.

By the way, with a minus sign it works just as well here. You can check that. It works just as well for a minus sign. So E to the minus A is a matrix that propagates the state backwards in time one second. That's what it means. Okay? So these are kind of basic facts. That's what the matrix exponential means, right? So it's gonna mean all sorts of interesting things. And from that, you can derive all sorts of interesting facts about linear dynamical systems, how they propagate forward, backward in time, and things like that.

Okay, so now the interesting thing here is you have – if you know the state at any time, any time, you actually – fixed one time, you know it for all times because you can now propagate it forward in time with this exponential and you can propagate it backward in time. So for example, I can go to some chemical reaction or some bioreactor described by X dot equals AX. I can take a measurement of X at time 12, and then from that I can infer what X of zero was even if I didn't measure it.

Why didn't I measure it? Maybe because it was too – the numbers were so small the colonies hadn't grown yet, and I could only measure them when they got to the billions or trillions, or something like that. Everybody see what I'm saying here? So in fact, how do you get X of zero if I tell you what X of 12 is. What do you write here?

Student:[Inaudible].

Instructor (Stephen Boyd):E minus 12 A, that'll do it. Okay? So E to the minus 12 A is the matrix that actually goes back – it's a – goes backwards in time 12 seconds, okay? So that's what it is.

Now we can actually connect a few things up now that's kind of cool. We looked earlier at a forward Euler approximate state update. Now the forward Euler approximate state update said if you want to know what is X of T at time tau plus T. What am I doing?

If you want to know what X of T tau plus T is, you'd say, "Well that's about equal to X of tau plus," and this requires T small and it's an approximation, so I squigglize these. There we go. It's a new verb. X of tau plus T times X dot of tau, like that, okay? Now that's an approximation and it's based on – basically this is, some people call it – by the way, this is called dead reckoning in a lot of – because basically say you're going in that direction, you check your watch, check the elapsed time and say, "Where are you now?"

We're like that bearing times the time, that's where we are. So that's the approximation. Now this thing is AX of tau and so this is I plus TA times X of tau, like that. So this is an approximate. It's an approximate T second forward propagator. It's the forward Euler propagator, is what people would call it. But now we know the exact T second forward propagator.

The exact T second forward operator is the exponential. And look at this, this thing is merely the first two terms in the Taylor series. Okay? So now you can see forward Euler is basically just one term in the exponential series. You could take two and three, and all that kind of stuff. So that's the idea. Okay.

So let's take a look at this and let's talk about the idea of sampling. There's a lot of – actually already there's a lot of applications of what you see, just simple ones immediately. So if someone says, "I've got some measurements of X of T, you know, at different times, but I didn't know what it was in between," how would you do that? What if you – how would you do that? In fact, let's talk about that. Let's talk about that.

You have X dot is AX. Let's make it a bioreactor; we talked about that before. And suppose you make an assay, you measure the thing. And like X of 13.1, X of 15, you know, X of 22, like that, and someone comes along and says, "What is – what was the state?" And the state might be, by the way, the volume of different colonies or concentrations or whatever. Okay? And they want to know what's that.

And the first answer is sorry we didn't do an assay at T equals ten hours. What do you do? Let's say you measured at eight, too. What do you do? Give me some methods. Give me a method. You know A. You've measured X of 8, X of 13.1, X of 15, X of 22, and I want to get X of 10. Don't worry; so far, the measurements have been perfect. They're absolutely perfect. A is not a lie. What do you do?

Student: You measure [inaudible].

Instructor (Stephen Boyd):Perfect. So here's one. Ready? Reconstruction Formula No. 1, tell me what to write please. What do I write here?

Student:[Inaudible].

Instructor (Stephen Boyd):E to the two A. And the comment is propagate forward two seconds, oh hours, or whatever we said, whatever the unit is. Right? How about this, you said you could – we could take this one, X of 13.1, E to the what?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay. And this is propagate backwards – no, no, no, come on. That's not right. This is E to the minus 3.1. Okay great. I said that before, that reflects on you, you know, not me. So it's the length between I write something idiotic and you correct it.

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. I knew that, I was just testing you. Okay. Fine, so we have that. All right. Oh by the way, which of these is better?

Student:[Inaudible].

Instructor (Stephen Boyd):Hmm?

Student:[Inaudible].

Instructor (Stephen Boyd):They're what?

Student:[Inaudible].

Instructor (Stephen Boyd):This one. You like that one. Why?

Student:[Inaudible].

Instructor (Stephen Boyd):You think the – so we got two – two people over here say the former. They like propagating forward. But you – oh, because you propagated forward two hours, is that it?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh you have the – okay.

Student:[Inaudible].

Instructor (Stephen Boyd):Ooh, okay. All right. So –

Student:[Inaudible].

Instructor (Stephen Boyd):All right. Could you have calculated it from X 15? Sure, no problem. E to the minus five A times X of 15. Okay, so which of these is better? Well if there's no noise and A is exactly what you think it is, they're all exactly the same. So this could actually be an assertion here. And if it's not – by the way, if these are not – if you calculate these and you get two different answers, it means you're gonna have to do something more sophisticated. Okay?

And just for fun, just given this state in the course, what would you do, if someone gave you all this data? Just a quick thing, quick what would you do?

Student:[Inaudible].

Instructor (Stephen Boyd):You might do some Lee squares, exactly. You might – I mean first of all, you might propagate all of these two time ten. Okay? If they're like all over the map, you would say – you'd go back to the person and you'd say, "Can we talk?" Okay, that's what you'd do here.

Now if they're not all over the map but just sort of you know, one is estimating one thing, one's a – they're a little bit different and they're not like, you know, weird numbers, you know, varying by factors of ten, if that's the case – that's gonna come out really, really nicely by the way on the tape. That was me talking while inserting this thing back into its – okay.

What you might do is take all those things back and then do some kind of Lee squares fit. That's what you might do. Right? And by the way, that'd be a very, very good method. That would be a perfectly practical method. Actually, methods like that are used plenty. So. Okay.

So we will – we'll quit here and continue next time. And let me, for those of you who came in late, the midterms actually are graded. Solutions are posted. They'll be available, I guess if you follow me up to Packard.

[End of Audio]

Duration: 77 minutes

IntroToLinearDynamicalSystems-Lecture12

Instructor (Stephen Boyd):A couple of announcements – I don't know – it depends on how frequently you read email. If you read it very frequently, you'll know that I sent out a panicked message.

Actually you can turn off all amplification in here, if you don't – which was that – the CC3 AFS Directory disappeared this morning. So it's not unusual to have like AFS episodes where full swabs of AFS disappear for a little bit. And I've learned in the past if you kind of be calm, and just wait a little while then they come back. And I probably should've done that in this case, but it was very strange to see – to list EE class slash EE star, and to see EEQ61, EEQ62, nothing, EEQ64 – little bit strange. But anyway, so I put in multiple panicked calls, and I did finally – right before class I got a hold – some of you said, in fact, "Yes, there was an incident." And as – he said, "As we speak file systems are rebuilding themselves." So – and apparently they just came right back up online. So apparently our website's down, but oh well.

Okay. Any questions about last time or anything else? Midterm? No. Okay.

So I do want to say for the midterm make sure if we – I don't know – if we – if for example, we took a bunch of points off – let's say for example, problem 2 on the band limiting thing, please make sure – read our solution, look at yours, and it shouldn't be one of these things where you say, "I don't know. I mean theirs is okay, but mine's pretty good too." It shouldn't be one of those. So if you're anything remotely near that cognitive state, look at it again, and come to talk to one of us. By the way, we also do make mistakes, so that's fine, I mean if you actually find something where you – if you're cognitive state converges to, "Hey, mine is fine," then definitely come, and talk to us.,

Okay. Let's continue then. You can go down to the pad. Last time right at the end, we looked at – well, this is really what – this is what an exponential is. This is – in fact, this is really one of the – a [inaudible] exponential is nothing but a time propagator in a continuous autonomous linear dynamical system. That's what it is. It is nothing else.

So here's what it tells you. It says that to get this state at time Tau plus T from the state at time Tau – first of all they're linearly related. That alone is – well, it's not unexpected, but they're linearly related. They're related by an end-by-end matrix that maps one to the other. And that matrix is simply E to the TA. So E to the TA is a time propagator. It propagates X dot equals AX forward T seconds in time. If T is negative it actually runs time backwards, and reconstructs what the state was some seconds ago.

So we looked that – actually that alone is enough for you to actually do a lot of very interesting things, and we'll look at a couple of examples here. One is sampling a continuous time system. Suppose we have X dot as AX. I have a bunch of times. We'll call these the sample times. And I'll let Z of K be X of TK. So Z of K is the sampled version of X at the sample times TI. The fact is they just don't even have to be monotone increasing. It's totally irrelevant. Everything we're gonna work out now has nothing to do with that. Times can actually be [inaudible] go backwards, and it makes no difference.

Here's the important part. This says that to get Z of K plus 1, which is X of TK plus 1, from X of – Z of K, which is X of TK, you simply multiply by E to the time difference times A. Okay. That's matrix exponential of force here. So this is what propagates you from TK to TK plus 1. If uniform sampling, then that's says that difference between these is some number H, which is the sample period, and basically you get this – you get Z of K plus 1 is E to the HA Z of K. And that's actually just – this is sometimes called the discretized version of this continuous time system. But this one is exact. There are no approximations of any kind here. This basically says if you want to know how the state propagates forward H seconds into the future, you multiply by this constant matrix E to the HA, and that'll probably get you forward H seconds. So that's what this is.

Okay. So there's a lot of things you can do that are interesting now. Here's one. These things really come up actually in lots of cases. So if you have time varying linear dynamical system, that's X dot equals A of TX. These do come up. In some cases it's much more common to find the time in variant case, but these do come up. And there's a bunch of things you need to understand about this.

For example, the simple solution, which is a generalization of the solution of X dot equals little A of TX when these are scalers, is wrong, and I believe that's a problem on your current homework – or current homework. Oh, it is current. It's Thursday, I can say that. I forgot to announce that. We did post homework 6, and not only that, there's an M file which you can actually get to now that AFS has graced itself – graced us with its presence yet again. So we've assigned homework 6. Homework 5 is still pending or something like that. I think originally we said it was due Monday or something, and we decided that was weird. So it's just due Tuesday. Is that how it all converged?

Student: That's due the [inaudible].

Instructor (Stephen Boyd):It's due Tuesday.

Student:[Inaudible].

Instructor (Stephen Boyd):It was never due Monday. Well, okay. Fine. Like I said, "Don't trust me." I used to say, "Trust the website." That's why it was so upsetting when it was gone this morning. But it didn't bother any of you, right? Did – actually how many people here actually even noticed it?

Student:[Inaudible].

Instructor (Stephen Boyd):When?

Student: This morning.

Instructor (Stephen Boyd):This morning. Okay. Because we were mucking around at around 11:00 or midnight or something, and it was fine. So did it bother you?

Student: No. [Inaudible].

Instructor (Stephen Boyd):You just – you're like totally cool about it? You're like, "No problem. It's just not there."

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, you must have done it right before class. Okay. Jake was doing the follow up. He did the same thing rather. He was fiddling around. It wasn't there. He did something. It was there. And he said, "I fixed it." I think – he may have. We don't know. Anyway, it's back. So – okay.

Back to piecewise constant systems. Okay. So here if you have a piecewise constant system, it means that the dynamics matrix here is constant for certain periods of time. These come up all the time. Oh, by the way, there's a wonderful name for this. This is called sometimes a jump linear system is one name for it. And before we get into how you analyze them, I'll just give you an example.

So a very good example that I know about or heard about is people working on, for exam – well, power systems would be one. So a very good model for – at least for small perturbations in a power distribution network is linear. Actually that's not a good model for huge perturbations, but for small ones, very good model. And what happens is people will analyze something like this: A0 will sort of be the nominal dynamics, the dynamics of the system when everything's working. So this state is kind of – presumably going to zero. That's what – zero means everything's the way it should be. Zero means ever – all the phases are in line or whatever they're supposed to be, like this.

Then what happens is at some time – and that might be called something like TF for fault time. So at TF basically some – the lightening strikes or something happens, and some circuit breaker, for example, shorts or something like – some of the line shorts. There's an over current situation. And then they have another name, I remember this as TC, which is the clearing time. The clearing time means that a breaker opens now.

Now, by the way, then you would have things like A nom would be the original dynamics matrix, that's when everything's cool. Then you'd have AF, that's the dynamics matrix when this short is there. Then you'd have AC, is a different matrix still, which is the dynamics matrix when the circuit breaker's opened – something like that.

And so you want to fi – you want to ask questions now about what happens to the state in this time varying linear system. You might ask, for example, "How far – how big does the state get?" One hopes that it then gets smaller again – maybe we can answer it this way. So these are the types of questions you can answer, and you can do studies where you could vary – you could say, "Well, if the circuit breaker is opened in five cycles maximum, what happens? What if that were three? What if it were seven? How much trouble can the Western U.S. grid get into?" And these would be the kind of – the questions – the types of things you could answer.

Okay. That's just an example. Let's see how this works. So to get the state at a given time here's what you do from the initial state. You start with the initial state, and you propagate it forward T1 seconds in time. What is the stuff between my fingers there? What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):That's X of T1 because we just propagated it forward. Now, by the way, you can – if someone says, "Oh, wait a minute. You're using a matrix exponential. This is a time varying system. Everyone knows you can't use matrix exponentials for a time varying system." What would you say?

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. You'd say, "That's true, but over the interval zero T1 it's a time invariant system. The matrix A does not change." Okay.

Now, you take this matrix, which is X of T1, and you propagate it forward T2 minus T1 seconds. And what does this give you? That's X of T2. And you keep going until the very – until the last instant before T occurred, and then you propagate forward the remaining time. That's this. Okay. So this is the picture. And you should read that exactly that way. When you see it you should know what every single piece is, and so on. Okay.

So this – by the way, this matrix here is called – that is actually called the state transition matrix for the time varying system. And in fact some people write it this way. Phi of T and zero, meaning it is the matrix that propagates X of zero into X of T. So that's the name for that.

Okay. So we can also analyze things like the qualitative behavior of X of T. This will actually make a connection to things you know from undergraduate, and scaler stuff. So let's take X dot equals AX like this. Then X of T is E to the TA times X of zero. The Laplace transform of X is this resolvent SI minus A inverse times X of zero. Now, every component of the state Laplace transform is an entry of this, but that each one of these looks like this. Every entry of this matrix is a rational function with a numerator of degree less than N, a dominator which is exactly N, multiplied by then some numbers here. So the result is each of these entries in this Laplace transform of the state vector is a polynomial of degree less than N divided by the characteristic polynomial. Okay.

Now, this is just a scaler, and we know a lot about the qualitative behavior of a function like little xI of T now – based on it's Laplace transform here. So what you see here is that the only pulls that capital XI of S can have are among – are the eigenvalues of A. So that tells you that the only terms you're gonna see when you work out the solution – right?

Well, I guess we'll first assume the eigenvalues are distinct. That's the simple case. Then the solution looks like this of course. It's simply a sum of exponentials. Now, these can be complex here. Okay. So in this case you get a sum of exponentials, and the exponentials here determine of course the qualitative behavior. But these exponentials here are the eigenvalues of A.

So what that means is the following. The eigenvalues of the matrix A give you the exponents that can – and I'm gonna emphasize that's not can – it is not do, it's can occur in exponentials. Okay. And the reason it's not do is that some of these betas can be zero. Okay. A real eigenvalue of course, it corresponds either in exponentially decaying or growing term, and if it's – it's either the exponential in decaying, if it's negative or it's growing.

Now, a complex eigenvalue corresponds to decaying or growing sinusoidal term. So now, at this point you can say that you know qualitatively what all solutions of X dot equals AX look like. And here's the cool part. It's no different – what's interesting about it is this: you know already that you get much different behavior in a matrix of vector linear differential equation than a scaler one. In a scaler one the only thing you can have is exponential growth or decay. That's it.

Here, we've already seen weird stuff. We've seen oscillations. We've seen things growing like T, not exponentially. So we've seen some pretty weird stuff. And you ask the question, "How weird can it be?" Now, we know the answer. The answer is, "We'll get to the T's later." But for the moment it says that basically the solutions of X dot equals AX – if the eigenvalues are distinct basically are damped or growing sinusoids. They're exponentials multiplied by sinusoids.

And you know what that is, by the way? That's this. It's this. It's a – well, I'll put it all together. It doesn't matter. So what it basically says is that the types of behavior you see in second order linear differential equations or scaler – sorry – scaler second order of differential equations – because these equations of course will give you exponentials multiplied by sinusoids. And it basically says, "That's it." So you're not gonna see anything different from second order systems like that. That's not quite true, but for the moment it is as long as the Lambda I are distinct. [Inaudible]. Okay. So that's what that says. Okay.

So you get a famous picture, which you've probably seen. I don't mind showing it again. It's something like this. If you plot the eigenvalues of the matrix in the complex plane they occur in complex pairs. In other words whenever something is here its conjugate is here as well. Like that. And this tells you something immediately.

If the eigenvalues – oh, by the way, what's the size of the matrix here in this case? What's the size of A? Six by six. So it's six by six matrix. Here are the six eigenvalues. And it says that if this what – if these are the eigenvalues of A, it's say that when we look at the solutions we should expect a growing exponential here with – actually that grows at a rate given by that. This is sort of an oscillating – this one gives you an oscillating solution that's oscillating, but also decaying. And these things are just rapidly decaying things. That's rapidly decaying.

By the way, this one is a decaying exponential. Would you see the – sorry – a decaying exponential multiplied by a sinusoid. By the way, for this one, if I showed it to you, would the oscillations scream out at you? No. You wouldn't even see it. You'd be hard pressed to say it oscillates. And the reason is, although the solution looks like E to the minus Sigma T time cosine Omega T – something like that, in terms of E to the minus T has already – E to the minus Sigma T is already gone down by some huge factor by the time one cycle has gone over.

So technically the term associated with this goes through many infinitely number of zero crossings, you won't see it. So for all practical purposes this thing would just look like a bump – sort of something that goes like that, and down. So – okay. All right. So that's the picture. All right.

Now, let's suppose that A has repeated eigenvalues. Now, we'll do the complicated case. In this case the Laplace transform could have repeated pulls – by the way, it might not, and soon we'll see exactly what that means. So if you express the – if you write the eigenvalues as Lambda 1 through Lambda R these are distinct with multiplicities N1 through NR. So these add up to N. Then X looks like this: the solution's the inverse Laplace transform of a rational function with repeated pulls. And those look like this. It's an exponential that's complex. So that's both got a sinusoidal term, and a decayer growth rate factor – exponential multiplied by a polynomial.

Now, actually you've seen everything. So you get – and now you remember one of our examples we saw before was a system where the solution actually grew like T, right? And that's exactly this. That's where Lambda was zero, and you had a T in here. So that's all that is. Okay. That's basically it. That's the only kind of solutions you can see for X dot equals AX. That's the whole story qualitatively. Okay.

So we can answer a very basic question, which is stability. So that – it's a very old term. Actually in some ways it even started a lot of the study of this. I'll say why in a minute. So you say that X dot equals AX is stable if the exponential – if this matrix exponential or state transition matrix goes to zero as T goes to infinity. Let me just quick syntax scan check. What's the size of zero here?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):It's N by N. That's the zero matrix. That say's if every entry of this exponential goes to zero – now that has a meaning that basically says this – it says that since X of T is this matrix [inaudible] every entry of this matrix goes to zero as T goes to infinity. It says that no matter what X of zero is, X of T goes to zero period. That's what it says.

Another way to say that – I guess it's the same story, but it's in a more flowery language or something. It says that basically any solution of X dot equals AX – there's lots of them – all of them have the property that as T goes to zero, X goes to zero, so all trajectories converge to zero. Okay. So that's stable.

Now, they are – you will hear other definitions of stable often – hopefully with some qualifier in front. Just ignore them. So that's my advice because they're just silly, and you'll hear things like neutrally stable, and – anyway, just not only ignore the terms, but avoid those people. They just make things complicated when they don't need to be. So they're – it's not good enough just to have one idea of like stability, so they have to have 14 different ones. Okay. So they can hold forth about when – one implies to the other. All right.

So we can now say when this occurs. It's very simple. You can say X dot equals A stable if, and only if all the eigenvalues may have negative real part. We'll get the only if part actually later today, but the if part we can get right now. Actually how do you know that? I mean it's simple. Every solution looks like this – it looks like a polynomial times E to the Lambda T. If the real part of Lambda is negative, E to the Lambda T here is complex number, it sure goes to zero as T goes to infinity. Actually doesn't matter what the order of T is because an exponential will eventually swamp out any polynomial, and so you get zero.

The only if we'll get very soon, but in fact this is very, very unsophisticated this – it's very – this is very qualitative to say stability. So for example, already much more interesting, and much more useful is the following observation. If you take the maximum of the real part of the eigenvalues of a matrix, it actually tells you the maximum asymptotic logarithmic growth rate of X of T if it's positive – if it's negative it tells you that. So merely saying stability is pretty much totally of no interest. It's qualitative. It has no use.

So if you're designing something – if you're designing some kind of protocol that – where a filter or a control system or something like that, and you come back, and say, "That's it. I'm done." They say, "Yeah?" Then go, "Well, yeah, how does the aerodynamics work?" And the aerodynamics might be – X might describe some error that you're trying to drive to zero, and you say, "Oh, it's stable. Yeah, it's stable." But that's of no practical – there's not practical situation where mere stability is of interest. It's of no interest whatsoever.

You have to ask the time scale in which it's stable. If you make an altitude hold controller, and the aerodynamics is stable, but the smallest eigenvalue is just barely, barely, barely less than zero, so that basically on an entire flight you won't have actually converged to your altitude, it's not relevant.

So – I might add that goes the other way around. So I remember talking to some people once, and I remember asking about stability, and they're like, "No. We don't." But I had to – I have to tell you what they did. They made missiles. And they – so I said, "Well, what – " I said, "I remember some of [inaudible] stability." And they said, "Stability? What's that?" They're like, "Well, surely, surely the error as this thing moves can't diverge."

And he goes, "Why not?" He said, "These things have only set – they ten seconds worth of fuel. We don't – all we care about is E to the TA, which would magnify an initial error – E to the TA times initial error should not be too big when T equals ten." So they said, "We couldn't care less." These are the cases actually where you have – no, I wasn't working them. Is that what you're thinking? I can tell. I wouldn't do that. Okay.

Okay. Now, we're gonna actually tie this all to linear algebra, which is kind of the cool part. This should be review, I guess, for – you've probably seen all this. I guess – how many people have actually seen – heard about eigenvalues and eigenvectors or – so. And in how many cases was the discussion comprehensible? Cool. Great. Where was that?

Student:[Inaudible].

Instructor (Stephen Boyd):With – oh, that part. You mean just compared to the rest of the class, which was abysmal? I mean did you actually – did it actually say what eigenvalues and eigenvectors meant?

Student:[Inaudible].

Instructor (Stephen Boyd):No. Anybody can say that. I mean that's – here debt Lambda I minus A is zero. Okay. There I just said it. No. No. I mean like do you have a feel for it?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, okay. So that's the classical one. And for the record, that was the good experience. No one else raised their hand. So – okay. All right. That's fine. That's – okay. We'll fix that.

So a complex number's an eigenvalue if debt Lambda I minus A is zero. I mean that's the same as saying, "Lambda I minus A, which is a square matrix of course, is singular." Lots of ways to say that – of course, an end by – a matrix is singular if it has non-zero element, and a null space. That means Lambda I minus A times V is zero. If you multi – work this out, that's Lambda I times V equals A times V. That's AV equals Lambda V. Like that. Oh, by the way, later we're gonna see – there's a really good reason to write this as AV equals V Lambda. Even though that's slang, but there's a way in which this is slightly cooler. But we'll get to that later. But it's traditional to write AV equals Lambda V. Okay.

Now, any V that satisfies AV equals Lambda V, and is non-zero, is called an eigenvector of A associated with eigenvalue Lambda. So that's what you call that. And obviously if you have an eigenvector you can multiply it by three or minus seven, and it's still an eigenvector. And in fact, these are complex, so I can multi – they may be complex. I can multiply V by minus three plus J, which is I. So – okay.

Now, another way to say that this matrix is singular, Lambda I minus A, is to say the following – it's to say that it has a left eigenvector. This is essentially saying that the rows are dependent – sorry. This is the columns. This says that the rows of Lambda I minus A. Now, this – if you expand this out says, "W transpose A equals Lambda W transpose." And – oh, by the way, what is the dimension – what equals is that? It's equals between what?

Student:[Inaudible].

Instructor (Stephen Boyd):Row vectors. Exactly. One by end matrices are row vectors here.

Some people write this around the other way. They write it A transpose W equals Lambda W. There – they write it that way. So in fact you can see now that this is – in this case this is the right eigenvector equation for the transpose. So if you have something that satisfies this, it's a left eigenvector of A.

And actually how many people have heard about left eigenvectors? Cool. No one. Fine. You didn't hear about it, and you're just like holding back or something? Okay. All right. Fine. That's a left eigenvector. These will all get clear hopefully soon. All right. So these are eigenvectors.

Now, if V is an eigenvector of A with [inaudible] Lambda then so is – obviously you can multiply it. That's fine. Now, even when A is real, the eigenvalues can be complex. They have to occur in conjugate pairs in this case, and the eigenvector can also, of course, be complex. And in fact, if the eigenvector is real then Lambda has to be real because A times V is real, and that's got to be Lambda V, so it doesn't work unless Lambda's real. Now, when A, and Lambda are real, you can always find a real eigenvector associated with Lambda. So you don't need to – of course you can also find complex eigenvectors associated with a certain eigenvalue. And we've already talked about that.

There's conjugate symmetry, but there's more you can say about that. You can say the following. If A is real, and V is an eigenvector associated with Lambda, then it's conjugate is an eigenvector associated with Lambda-bar. That means roughly if you find one eigenvector – one that's complex, the truth is you really just got two in a sense – you just got two because you could conjugate that eigenvector, and it would also be an eigenvector of A – the conjugate would be an eigenvector of A, and it would be associated with the eigenvalue of Lambda-bar. Okay.

Now, we're gonna assume A is real, although, it's not hard to change the things we talked – to modify this to handle a case when A is complex. All right. So – there's a question.

Student:[Inaudible]?

Instructor (Stephen Boyd):Sorry. How do I define what?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, the conjugate of a vector? I'm sorry. Let me explain that. So a conjugate, of course, of a complex number – what do think? Should I switch to I like right now? Yeah, let's do it. That J step is – that's weird. Okay.

There's a complex number, and its conjugate is actually this. Okay. Looks like that. So vector is just the conjugate applied term by term. But actually I'm glad you brought that up because of something called the Hermitian conjugate of a vector, and that does two things. It transposes, and it conjugates. And that's often written this way: V star. That's one way. You will also say VH. Like that.

And you will also see this, and I disapprove, but anyway – that's what – this is the notation in MATLAB. In MATLAB, watch out – prime, which until now has meant transpose. In the case of a complex vector or matrix is actually gonna also take the conjugate, and that's called the Hermitian conjugate. So just a little warning there. Right. So you can – that's what this is. Okay. So that's the conjugate. Okay.

So let's look at the scaling interpretation. I think that was the one interpretation I heard. It's something like this. We're gonna take Lambda to be real. So if V is an eigenvector, the effective A is very simple – just scaling. Okay. I mean well, sure, AX – AV equals Lambda V, and so the picture would be something like this. Here's a vector X, and then here's AX unrelated. But here's an eigenvector, and AV comes out in the same – on the same line. By the way, what's Lambda for this picture?

Student:[Inaudible].

Instructor (Stephen Boyd):Negative what? Let's see. One – negative what? Two point something. I don't know – I know what it is. It's negative 2.3. I don't – it's something like that, right? Okay. So – by the way, this is the interpretation given normally when – if you take a math class, and the only one.

Now, of course it sort of begs the question like basically who cares. I mean so what if you multiple a matrix by a vector? Well, you can imagine cases where it would have some interesting applications. But anyway – okay. So that's what it says. It says, "You have an eigenvector. The effect – it's a special vector for which the effect of A on it is very simple, it just scales." So actually I used to say something like all the components get magnified by the same amount – something like that. Okay.

And you can say some obvious things here. For Lambda – if Lambda's real – if it's positive, it says the V and AV point in the same direction. If Lambda is real, and it's negative, they point in opposite directions. That's like our example here. If the eigenvalue has magnitudes less than one, AV is smaller than V, and so on. And we'll see later how this is related to stability of discreet time systems – actually all of these things.

But now here's the real interpretation of an eigenvector. This is it. It's this. Let's take an eigenvector – so AV is Lambda V, V is non-zero, and let's take X dot equals A of X, and let's say the initial condition is V. So let's start a linear dynamical system autonomous at an eigenvector. Okay. Then the claim is this. The solution is embarrassingly simple. It is simply – basically it's – that's a constant vector V. You stay on V, and all that happens is you either grow or you shrink with time, and each one exponentially – very simple solution.

Lots of ways to see it. Here's one. We could say, "Well, look X of T is E to the TAV." E to the TA I write as the power series. This is slang, of course. Matrix divided by scaler. And then we multiply that out. We say, "Well, I times V, that's V." TAV – now, AV is Lambda V, so I write it this way. If AV is Lambda V, then what is A squared V? What am I doing? My God. A squared is A times AV, which is this. AV is A times Lambda V. The Lambda pops outside. You get Lambda times AV. I reassociate, and I get Lambda squared AV. There we go. Okay. So –

Student:[Inaudible].

Instructor (Stephen Boyd):You don't like what I did? Oh, A goes away. Thank you. You're credibility is slowly growing actually. That's good. That's fine. [Inaudible]. Okay. Good. That's what you should be doing. All right. Of course it'd help if I wrote it legibly, but I think this – I don't know. This just makes it more of a challenge. That's good. It means you really have to be watching. Okay.

So if you look at this thing, these are now all scalers multiplied by V, and if you pull the V out on the right you get a scaler here, but that scaler is the power series for E to the Lambda V. Okay. So that's it.

So now this is really in fact what an eigenvector is. An eigenvector is an initial condition for X dot equals AX for which the resulting trajectory is amazingly simple. If it's a real eigenvector, it just – it stays on a line. It just grows if it's positive, shrinks if it's negative. It stays where it is if it's zero. So that's it.

Now, if Lambda is complex, right, we'll interpret that later. We'll get to that later because it's a separate case. So what it says is if you have – if you start in eigenvector, the resulting motion is incredibly simple. It's always on the line spanned by V. And that's actually called a mode of the system. So that's what you call that is a mode. And it's a word that's used kind of vaguely, and by – in lots of different areas, but it's what it is. It means it's a really simple solution. Okay.

So this brings us to a topic that's related to this, which is very interesting. It's the idea of an invariant set. So if you have a subset in our end, you say it's invariant under X dot equals AX. If – whenever you're in it at time T, you will be in it forever after. Okay. So that's what it says.

And the picture is something like this. You have a set like that, and it basically says, "If these are the trajectories of X dot equals AX, what you're allowed to do is you can cut into the set, but you can't exit." That's what it means to be invariant. Okay.

So that's the picture. It may not look as clean because often S is something simple. Let me ask you a question. What is – could you ever have an invariant set that is a single point called X zero? What would it mean to have a single point that is an invariant set?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):That's exactly what it is. That's a very pedantic way of saying, "You're in the null space." Is that – let's check if that's right. It's an equilibrium point. It's a point for which – if you're in there, well, you can't go anywhere. If you can only be one point, it means you're constant therefore you're derivative is zero, and therefore it just means AX equals zero. Right? So basically this is sort of if, and only if, X zero is in the null space of A. Those are the equilibrium points. That's just a – that's a silly example, but – so this is the idea.

By the way, the idea of an invariant set unlike the idea of stability, which is actually really only of conceptual help – whatever that means – conceptual use, the idea of invariant sets is extremely useful in practice. It's amazingly useful. It's real, and actually very – not many people know this or appreciate it.

That's real because if I say, "I just designed the altitude hold control system on a 747." And if your initial altitude is between plus – if I give you a box that's real that says, "From 29,500 feet to 30,500 feet," I give you another – I give you a box on other things involving things like your pitch rate, and all sorts of other stuff, and I say, "That box is invariant." That is extremely useful, and very practical. It says, "Basically if you start in that box – you might start in that box because there's a big down burst of wind." It says, "If you start in that box, you're gonna go settle back up to your equilibrium point without leaving it." That's really useful. It's quantitative, and everything else. So I just comment on that. Okay. That's the idea of an invariant set.

Well, we just saw one. If you have an eigenvector, what we just found is that the line is invariant, and that's kind of obvious. It basically says if you're here, and X dot – I mean if Lambda's positive, it means if you start here, you actually go along this – along the line.

And in fact, you can even say the ray is invariant. So if you start in the positive part it's invariant.

And if it's – if Lambda's negative, it basically says you're gonna be on this line segment between here, and here because you start at this point, and all that happens is you decay. You can't get out. By the way, can you get – can you enter this set? Actually we haven't got there yet, but you can't. You can neither enter nor leave it – a set like this. So – but anyway, it's invariant. Okay.

Now we can talk about complex eigenvectors because the best way to understand it is with invariant sets. So let's suppose you have AV equals Lambda V, V's non-zero, and Lambda is a com – so you have complex eigenvector. Well, then we have A E to the Lambda T. For any complex number A that is a complex solution of X dot equals AX.

Now, typically when someone says, "We're looking at X dot equals AX. A is real. We're only interested in real solutions of X dot equals AX." So – typically you're only interested in that. Nevertheless, for what it's worth, that thing – A E to the Lambda TV is a complex solution of X dot equals AX. But that means that actually both its real, and its imaginary parts separately are solutions. Right?

So let's look at the real part. X of T is the real part AE to the Lambda T times V. And now we're gonna do this very carefully here. E to the Lambda T, I'm gonna pull out the E to the real part, E to the Sigma T. Sigma is the real part. Here Sigma is the real part of Lambda. I'm gonna pull that out in front, and I'm gonna write the rest of this thing out this way. There's – of course there's a cosine T, and sine T, and so on in there. And you get this expression in terms of Alpha, and Beta, where Alpha and Beta are the components – the real, and imaginary components of A.

So – and you could just multiply this out, and check me, but the point is now you should actually see some things here that you recognize. That's just some vector that depends on this A. That's some – any old vector there. This matrix should be a friend of yours by now. That's a rotation matrix with angular velocity Omega. So it take this vector, and R2, and it rotates it. Okay. I'm not sure which way it rotates, but it doesn't matter. It's one way or the other it rotates with an angular velocity of Omega radiance per second.

So this produces – this takes your initial vector, and it rotates it. This says you then – you get two numbers, and you use – those are the two numbers that you mix the real, and the imaginary parts of that eigenvector with. So you form a real, and imaginary part, and then you can add the growth factor later. You could put the growth factor anywhere you like. It's a scaler. So that's it

And we see the most amazing thing. We see the following. It says that if you have AV equals Lambda V where V is – where Lambda's complex – V is complex, it says the following – it's says – basically says you stay in the plane, and I'm talking about real solutions. The real solutions stay in the plane spanned by the real, and imaginary parts of that eigenvector. That's what it says. Okay.

So when you get an – a complex eigenvalue, you get a complex eigenvector, you look at the real, and the imaginary parts. Those are two vectors in our N. They span a plane.

Oh, let me just ask one question here. What if they were dependent? How do I know it spans a plane? That's a good question. Let's answer it. I'm just saying these two vectors are – one is not a multiple of the other, right? That's what it means for – [inaudible] – what?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, I don't know. Let's see. What – let's just say – suppose VM were the same as V real. So the real, and imaginary parts of the eigenvector are the same. What does that mean? Is there a law against that?

Student:[Inaudible].

Instructor (Stephen Boyd):Precisely. So let's suppose the imaginary, and the real parts are just – they're equal, right? And you say, "Ha! There's no plane here. You got two vectors, but they're in the same line. They span a line." Okay. Or as you might say now, pedantically, "Huh, that N by two matrix there, it's not full, right?" Okay.

So what happens in the – if this is equal to that, you would say, "Well, wait a minute then. The whole – the eigenvector is really – it's V re times one plus I." There. See I made the switch. Well, on the page I have J, so I'll switch to J. There. It looks like that. If that's the eigenvector, so is that. And it says that V – the real eigenvector alone is eigenvector. And that says that Lambda is real – oh, by the way, which is not contradicted here, so I should say something like not real. Like that. So – to be technically right. But it's okay. It's fine. Okay.

So what happens – so now here Sigma gives you the logarithmic decay factor, and Omega gives the angular velocity of the rotation in the plane. So now you should be getting actually a – actually you want a picture of this. So basically you have X dot equals AX, a complex eigenvector, a real eigenvector that's easy. It's a direction where if you start the system anywhere along that line, it will stay along that line.

A complex eigenvector, you look at the real, and imaginary parts that spans a plane in our N. And it says if you start anywhere on that plane, you will stay – number one, you'll stay on the plane, but the motion of the plane is very – roughly kind of simple. Basically it rotates on that plane with an angular velocity of Omega. That's the imaginary part of the eigenvalue. As well, it decre – it grows or shrinks depending on the real part of the eigenvalue. That's what it means. Okay. So we're gonna look at some examples, and stuff, and so will you separately. So – all right.

Now, let's look at the dynamic interpretation of left eigenvectors. So here suppose W transpose A is Lambda W transpose, and W's non-zero. Then let's look at DD – let's let any solution of – well, let's look at this. DDT of W transpose X – let's let X dot equal AX, but now X is any solution. It's not started from an eigenvalue – eigenvector – nothing – just any solution. If I look at DDT of W transpose X, I get W transpose X dot. But X dot is AX. I get W transpose AX, but W transpose A is Lambda. So I get this. If you look at the beginning – if you just get rid of this, you see that is W transpose X satisfies a scaler differential equation, and therefore – linear differential equation, therefore the solution is this. Okay.

Now, what this says is really cool. It says that a left eigenvector gives you the coefficients in a recipe or a mixture. Okay. Because that's what a row – that's what W transpose is. W transpose is a linear function – a scaler value of linear function of X called linear functional of X. So it's W transpose X. And it says if you get those coefficients just right this thing will undergo a very, very simple motion. Even if X is going crazy, if X is in our 100 by 100 there's hud – there's a hundred eigenvalues – all sorts of crazy motions, and modes, and growths, and oscillations going on. W transpose X will have only the pure Lambda – E to the Lambda T component in it. Okay.

So – and you can say lots of cool things now. Now, here's something of actual use. It's this. If Lambda isn't R, and Lambda's negative, then the half space – the set of Z such that W transpose Z is less than A is invariant. Okay. The reason is this. Because W transpose Z does – it just decreases. That's all it does with time. Now, if you decrease, it says basically something – so let's draw a picture. It basically says that if W points this direction – it says that if you – N Lambda is negative, and W transpose A equals Lambda W transpose – like that.

It says basically the following. It says if you start here – oh, by the way, in the general case not – I mean here this region is a strip. You call this a strip in R2. In RN you would call it a slab. Okay. So that's a – basically it's a – this is a half space. It's everything in between two parallel planes – hyperplanes is a slab. So what it says is that in the general case this slab is invariant. What it says is if you start in that slab – and by the way, now you can enter a slab. If you start in the slab, you will stay there forever. That's what it says.

In fact, what it says is actually quite interesting. It says that in the – in this art – in this case in two dimensions, it says that when a – if you start here, and I draw whatever the trajectory is, I can tell you this. If you only look at a sort of component in this direction, that is just nicely exponentially decaying to zero. Now, by the way, this thing can be shooting off to the left or right or going to zero. All sorts of crazy stuff can happen in that slab. But sort of – if you look at the slab coordinate – that's this thing – it's just very happily decaying. So that's the picture. And this is already something very useful. This is already useful. It tells you there's an invariant – there's – you get an invariant perhaps [inaudible] tells you you're in there, you keep going.

Okay. Hey, Jacob, make a note. We should – let's construct a problem out of that. We'll edit that out of the – actually looks like I'm not gonna be able to edit the – these videos. It's too bad. I could get rid of things like that. Maybe I'll figure out a way. Okay. Okay.

Now, if you have a left eigenvector corresponding to a complex eigenvalue, then it says that both real, and imaginary parts of W transpose X, it gives you two planes, and they both have these – a sinusoid. Okay. So now we can – let me give a summary. Oh, by the way, it's examples that will kind of make all this clear.

So here's the summary. A right eigenvector – that's at a very special initial condition in a linear dynamical system. That's what it means. It's a point for which if you start the system there, you will get a very simple motion, either a decaying exponential, a growing exponential or a decaying – exponentially growing or decaying sinusoid. Period. That's it. Well – yeah, we'll leave it that way. That's what a right eigenvector is.

And by the way, if you just stick a random point, and then look at the solution of X dot equals AX, you're gonna see everything in it. You'll see all the modes, all the frequencies, and everything. You'll get a very complex behavior. Okay.

A left eigenvector has nothing to do with initial conditions. It basically – it's a recipe. It says if – in a left eigenvector, it says that if you form a certain – if you take any trajectory of the system – now, in general a trajectory system is gonna have all the modes, all the frequencies in it, all sorts of oscillations, growth, decays all added together. This says if you make very special linear combinations of the states, and those combinations are given by left eigenvectors, then what is – what looks like a fantastically complicated trace or trajectory will just simplify, and you'll be looking right at something real simple – a decaying exponential or a growing exponential or a sinusoid or something like that. That's what a left eigenvector is. Okay.

So the different – it's actually very important to understand which is – what they are, and that'll also come. Let's see. Okay. Let's do this. All right.

So let's look at an example. It's a stupid example. It's simple. It's X dot equals minus one minus ten ten one zero zero – I don't know – whatever. Now, all these ones, and zeros you should have an overwhelming urge to draw a block diagram. So this is the block diagram. Uncontrollable urge to draw – that's it. So you see that's like a – that's a change of integrators. By the way, the chain of integrators in here corresponds to this – the ones on the lower triangle, which corresponds to a up or down shift – one or the other or something like that. I think it's up shift. Yeah, something – whatever it is. Down shift – it's a down shift in this case. Okay. It's a down shift. So you get the chain of integrators then these numbers here go in here like that. Okay.

Well, let's look at the eigenvalues. Well, it's easy – by the way, matrices like this that have kind of a lowered – ones on the lower diagonal, and then a row that's non-zero, they're called companion matrices. There's a bunch of different kinds. You could put this row here. You could put it at the bottom. That's called a top companion matrix. And you have a bottom companion matrix. You could have a left or a right. It goes on, and on. And, I guess, in a classical course like this, you would be tortured possibly for up to a week on just companion matrices. I have no idea why, but this would be the case. Just to let you know. Okay.

So that's a companion matrix, and the cool thing about them is it's extremely easy to work out the characteristic polynomial. Well, it makes sense. A lot of zeros, and ones there, that's on your side, and so on. So actually it usually turns out the coefficients are these guys up in the top. But it's better since I never remember what it is, I just work it out myself. So basically you have to work out the determinant of this. Ten ten – that's minus one zero zero zero minus one zero. And you have to calculate that determinant – except I should put some S's here. There we go.

And I don't know – you can start where ever you like. You can start with the – it's S plus one times the determinant of this thing. There's a zero there which saves you from at least one of the terms. So – and you just get an S squared. And then you say minus ten times the determinant of, I guess, what you crossed – how does it work? You cross this, and this out. The determinant of that, again, there's a couple of zeros there saving you at least one multiply or something, and you get the S. And then you get plus ten times the determinate of that. And, once again, everything's – the determinant is one or something like that.

So these are not bad. I mean that's about my speed. That's about three or four floating point operations. That's about the number I can do without making a mistake. So I don't recommend doing more than that by your self. So – okay.

So that's the characteristic polynomial, and we just factor that, and that's S plus one times S squared plus ten. So the eigenvalues are minus one, and plus minus J times square root ten. Okay. That – these correspond to these. Now, by the way, immediately you should know what to expect. You should know that when we look at the solutions of this third order linear autonomous dynamical system you should see things that look like E to the minus T, and you should see pure sinusoids, which look like cosine squared ten T, and sine cosine ten T – that kind of thing. If you see anything else – if you see a growing exponential, you see anything with like a growing – like T or something, it's wrong. It can't be. So immediately when you see this you should be primed for what you're gonna see when you run a solution of this.

Let me say a couple more things. There are not – there's no one as far as I know who can look at a matrix, and give – and kind of look at the eigenvalues. I'm gonna tell you what the – I have a feel for the eigenvalues. Two by two – no problem, that can be learned – maybe three by three. Actually that's not true. No. Not even – this one I could fake it because I know it's a companion one. I could quietly factor that in my head, and if I distracted other people, it could like I looked at that, and go, "Yeah, well, you're gonna get some oscillations or I think some oscillations. Yeah, they're gonna be around – I'd say on the order of about three is my feeling. It's gonna be the three radiance per seconds. The periods gonna be around one – two I mean – something like that." You could do that.

And [inaudible] someone will say, "That's amazing. How'd you do that?" And you go, "Well, you know, it's just you work with these enough, and see the minus ten here? That kind of – see when X2 is going up, and X3 is going down, these kind of cancel, and then the minus one here, that kind of goes around." This is complete nonsense. I –actually what I'm saying is that while it's entirely reasonable if some one walks up to you on the street, and says, "You have Y equals AX." You look at Y for ten seconds. You should have a – I mean there's some details you won't know. But things like if there's zeros in that matrix, you have a rough idea of what doesn't depend on what. If you see huge entries, negatives, positives, you should know. No excuses. Okay.

X dot equals AX. This is wrong. There's no one who can look at A, and say, "That's good." I mean there's special cases, but you can't – that in general. And that's actually an interesting thing because it means it's not obvious. There's no one who can look at one of these things – and all this is, if you think about what X dot equals AX is, it's not much. It's – basically it's a bank of integrators. The outputs come out, and they run through a matrix, which kind of mix – blends everything around, and they get plugged back into the integrators. How hard can it be to understand what that does? Anyway, the answer is, "It's really hard."

Well, I mean it's not. Once you know about eigenvectors, eigenvalues, matrix exponentials, it's not hard. But the point is you don't know about those things, there's absolutely no way you will have a clue what happens to a system here. So you can't look at something, and say, "Here's how it work – here's how my – here's my model of my economy." And you have a five by five matrix. I mean with five sectors in the economy. That is tiny. You can not look at that, and say, "Wow. We're gonna have some business cycles. I'm also seeing some pretty – I'd say we're gonna get an overall growth rate about 5 – 6 percent." Nobody can do that. Nobody can look at 25 numbers, and tell you that.

I just want to emphasize this, right? It's like lead squares. You can't do lead squares in your head. Good thing you don't have to, but still – I'm not sure why I'm telling you this, but it's probably important to say. I'm not – I don't know why it's important, but it seems like it's important. Okay. All right.

Back to this thing. So here we are. We're expecting decaying exponentials, and we're expecting oscillations with a period of two. That is to say if you followed my intuitive argument about how all the interactions came in, and then the signal – you could – for oscilla – if you want to explain an oscillation, you could say something, "Oh, yeah. No, I can see it because it kind of goes like – but then it comes around again, and again, and again. Takes about two seconds to go around about, so that's how you do this." Okay. All right.

So here's just some initial condition we selected. Here's the solution. And maybe I'll actually say something real quick. You'll be doing this, and there's some things I should mention. I think I mentioned last time – I'll mention it again. It's very important. Please, please, please don't be the pru – don't be a victim of this. In MATLAB, and indeed in several other higher level languages, if you write X of a matrix, you're gonna get an element wise. Okay. This is – it's – I don't know what that is, but it is nothing that has anything to do with the matrix exponential.

And actually for that matter X of A works if A is non-square. Okay. And that would be true in lots, and lots of languages – higher level constructs. Okay. You have to say XM of A – and I guess that's the matrix exponential to actually get the matrix exponential. You try XM of A with a non-square, and you will prob – I pres – I hope – pray that – no, I know that you will get a stern warning about what can be matrix exponentiated. Okay.

So to make a plot like this, you just – I mean it's the easiest thing in world. You could actually even just do this. There's absolutely no harm in writing out stuff like this times X0 or something. There's just no harm in that. And then putting a four loop around this or whatever you like. None. No dishonor. Nothing. Okay. There's faster ways to do it, and so on.

By the way, there are some built in ways to actually find the solution of a linear differential equation. I actually do not recommend that you use them. Okay. One of them – and they've got stupid names like initial. That's supposed to be like an initial condition problem or something like that, and I – God only knows. It's like X, and A0 – unless you know about these, don't use them because this one you look at it, you know exactly what it does, and there's no mysteries, and this isn't just some stupid – some guy sitting in Walva – whatever – wherever they are – in Massachusetts decided some intact – what to do in some cases here.

This – in this case it's transparent. You know exactly what's going on. So that's what I recommend. So that was just a weird to the side. I don't use these. I think some – actually some of our solutions, maybe even some of the exercises may have some old references to this. Ignore them. So we'll clean those out. All right. I mention this just in case you're curious how I created this block. That's how I did it. All right.

Back to the story. So this shows you X1, X2, and X3 versus time. Everything is consistent. We don't see any growing exponentials. In the first one you see mostly just a sinusoid, and sure enough – let's see what the period is. Hey, wow, my intuition was excellent. So my intuition was that the period would be around two, and indeed it is around two. You don't see a decaying exponential in here too much. This one – there is a decaying exponential, but I guess it's kind of going up towards zero. That's fine. It's just got a negative term in front of it. And here in the third component of X, you see less – you see more of the decaying exponential, and less of these things.

By the way, just for fun, there is something you can see here. X1 is quite wiggly. X2 is less wiggly. And X3 is less wiggly still. Want to see my – see if you're gonna for this. It's kind of cool. So X1 – what happens is these things repeatedly go through integrators. That's what happens. And so when you go – push something through an integrator, it gets less wiggly. So that's why you'd expect – let's see if I can get this right. X3 should be the least wiggly because it came through three integrators. X2 should be more wiggly, and X1 most wiggly. You buying it? No. Why not?

Now, did I make the like opposite conclusion? Sometimes you make an excellent interpretation that supports the opposite conclusion. What you should do in a – how come you're not buying it?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, didn't you understand my explanation before about how X2, and X3 come out, and then they interact? They kind of cancel, and then they slosh around like every two – I thought that was pretty clear. Okay. I don't know – I thought it was clear. Now, come on. If you integrate a signal, it gets less wiggly.

Student: Why is [inaudible]?

Instructor (Stephen Boyd):Oh, between integrating a signal, and it gets less wiggly? Oh. Well, I don't know. Let's just do it. That's fine. No problem there. Are you in electrical engineering?

Student: Yes.

Instructor (Stephen Boyd):Knew it – cool. Okay. So it's – okay. Let's – well, let's just do it. Here's a signal. Ready? Here's this signal. What's the running integral of that guy look like? Well, it's – if I – why'd I make it so complicated? Anyway, it doesn't matter. I don't know.

So look. The point is these things, they kind of go up, and down. As you integrate, you're sort of averaging everything up to that point. And so the integral of – the running integral of this thing is something much smoo – it's much smoother. It kind – I don't know what it does, but it kind of goes down a little bit, and then – I'm making this up. Totally making this up – just for the record. There you go. Okay. It looks like that. So – buying it now? Yeah, and that is exactly the integral at that, by the way, but I'll just [inaudible].

That – by the way, that's not easy to do. You have to train for a long time to be able to do hand integrations like that in real time under pressure. And you'll see it's dead on if you check. Okay. All right. All right.

Let's look at a couple of things here. Let's do the following. Let's take the left eigenvector associated with the eigenvalue minus one. Well, that's this. It's G – it turns out to be point one zero one. That's the left eigenvector. And before we do anything, you have to interpret what this means. This eigenvector wha – this is telling you a recipe. That's what it's saying. This says if you were to mix point one of X of T – point one of X1 of T with one of X3 of T – if you were to add those two together, what will come out will be pure exponential – okay – which is to say it's a filter – it gets rid of the oscillatory component. Just gets rid of it. And indeed it looks like that.

And really – I swear the source code that generates figure really does do this. I could've just drawn an exponential, but – and hey, I didn't. I mean probably the source code really does this. And there it is. You see if you take 10 percent of the first component added to the last, all oscillation goes away, and you get a pure decaying sinusoid. Let's see if we can eyeball that. Ten percent of this – all right. So do that in your head. That gets scrunched – this gets scrunched way down to kind of a small oscillation like that. And then it says apparently you added to this, and – oh, yeah, I'm seeing it. I'm – no, I am sort of seeing it, right? Because look, see they're almost like anti-phase, right? That's kind of up here when that's down there.

So – anyway, never argue with a left eigenvector. It's right. You can see that's what it does. So if you scale this down, add it to that, oscillation goes away, and you're left – in fact, you would say that – the English description of this is you'd say by multiplying by G or forming these weights, you have – let's see. What would you say? Forgot the word now. It was a good one. It was really a nice one. You – I guess you have filtered out, or you have brought out the exponential or something like that. That's what you've done. You filtered out the sinusoid. You have revealed the exponential. I don't know. Something like that. Okay. So that's a left eigenvector.

Now, let's look at the eigenvalue assoc – the eigenvector associated with this eigenvalue. Now, that eigenvector says it's complex – has to be because that's pure complex. Now, what that says is the following. There's an invariant plane that spanned by these two vectors, like that. And it basically says if you start on that plane – now, this is an R3. So you're actually gonna – in fact, I'm gonna ask you to visualize this. In fact let's do that. Let's – take R3 the – you should be looking at R3. Make a plane going through it. That's the – spanned by these two. You don't have to get the angle right. Just put some stupid plane through it in your mind. Okay -- in R3.

Basically, it says if you start in that plane, the motion is really simple. You – first of all, you stay on the plane. So imagine this tilted plane, and initial condition, you just – actually, in this case because it's pure imaginary, it actually – you just oscillate. Everybody got that? And every – roughly two seconds, you come around. You stay on that plane.

Now, if you're off the plane, what happens? Now, you have that decaying exponential. So, the decaying exponential, it – you – in fact – well, we'll get to that in a minute, but that's the point. Oh, by the way, this G – that happens to be the outward normal of this plane, and you can check me. If you multiply G transpose times that you get zero. G transpose times that you get zero. And you know what that means? It means G is the normal to this plane. Okay.

So now, I think we can do the visualization exercise. Yeah, let's do it. Here we go. You ready? So you start it like this. Take R3, picture a plane. Okay. Now, in that plane – if you start in that plane, your trajectory is beautiful. It just os – it just undergoes a perfect – it's not a circle, it's an ellipse in that plane, so it just oscillates. So everybody got this? Okay.

Now, we're gonna do the following. The normal to that plane actually is the left eigenvector associated with minus one. Okay. So that means take that plane – if you're off that plane, and if you actually just study your height above or below that plane, that's G transpose Z. That simply goes down as an exponential. Okay. So I can see this is – no one gets anything. Okay. So we're gonna try it – so now, we're gonna throw it all together, and I think we have a perfect visualization of the whole thing. Here we go.

You have your plane. If you're on that plane, you oscillate. Okay. If you're off the plane, then your height above or below the plane simply goes like a negative exponential, E to the minus T, and you head towards that plane. So if you start – if this is the plane, and you start up here, you're – by the way, you're rotating the entire time like this. So you're rotating, same angular speed, but what's happening is your height above that plane is exponentially decreasing. So the solutions look like this. Is that – that's never gonna come out on the camera or any – it doesn't – in fact, I don't even understand – no. That's – this – is this making any sense?

If you start below the plane, you wind in the same direction, but you wind up to the plane. And so in fact, what would you call this linear dynamical system? It's got a name. It's an oscillator. And basically when it starts up after five seconds, if that's what the units of T are – is, you're on that plane, and oscillating. It's an oscillator. And in fact, you'd refer to that first bit – the first part when you're zooming into that plane or – that's called the startup dynamics of your oscillator. That's the way it works. Everybody got the picture? So – and that's it. And the normal is the left eigenvector, and so on, and so forth. So that's the picture.

Well, here's an example of it showing that it actually works. If you take X of zero – if you start in this real thing, and then propagate the solution, you indeed get an oscillation. I mean we knew that. So that's the picture. Okay. So that's the idea. Okay.

We'll look at one more example, and then we'll move on. By the way, this mystery about the left, and the right eigenvectors, and how they're connected, and weirdly one is orthogonal to – we're gonna get to all of that later. By the way, it's related to this mystery involving – which isn't a mystery. It's a mystery involving the – it's the inver – the rows, and columns of inverses or something like that. So in fact, let me go back to that because we're gonna need that, and we'll skip on – I want to talk about this.

If you have a matrix – if I take V1 up to VN – actually I won't take these because that's – that always – in the context of eigenvalues, those sound like eigenvectors, so I won't use those. Instead I'll call them T1 through TN. And I'm gonna make these – these are independent vectors. Okay. And if you take the inverse of this, and I call the rows S1 transpose down to SN transpose like that – okay. Then – I don't know if you remember, but we called these S's – that's the rows of the inverse matrix – right – which was formed by putting – concatenating a bunch of columns. We called this the dual basis here.

And actually – you remember what the interpretation of the S's are. Is says that when you express a vector in the T expansion, S3, for example, is the thing that tells you the recipe.

It tells you how much of T3 to show – to put into your recipe to recover your vector X. That's what it does. That's what these S's are.

And then we found out this. SI transpose TJ is equal to what? It's not complicated. I mean it looks mysterious, and complicated, but it's not. Basically this thing – if I take this matrix – if I plug in on the left here T1 up to TN – I'll insert it there, and I'll put T1 up to TN here. What's the right-hand side? That's I. And now, if I take this matrix written out by rows, and another matrix written out by columns, and I multiply them, I get I, but there's a very simple way to interpret a row – a col – a matrix viewed as rows multiplied by a matrix viewed by columns. It's – basically it calculates all – the one one entry, for example, of this mat – of this product is S1 transpose T1. So what's S1 transpose T1?

Student: One.

Instructor (Stephen Boyd):It's one. What's S1 transpose T2?

Student: Zero.

Instructor (Stephen Boyd):Precisely. This is Delta IJ. Okay. So that's the idea. So – and these are called dual basis, and things like that. And it looks – I mean when you first encounter it, it looks like it's kind of mystical, this strange thing that like – and you have to be very careful, right, because it's – I am not absolutely not saying that SI's are orthogonal. What do you know about this or this? Absolutely nothing whatsoever. Okay. So it's this weird thing where the S's are orthogonal to the other T's. The T's are orthogonal to the other S's. So there's this weird thing where one group is orthog – anyway, it's com – it's this. It's just that. Okay.

So we're actually gonna find out that the left, and the right eigenvectors satisfy exactly the same relation. So that's – but we'll get to that next time, and we'll quit here.

[End of Audio]

Duration: 76 minutes

IntroToLinearDynamicalSystems-Lecture13

Instructor (Stephen Boyd):Let me first answer your – there was a question about one of the homework exercises due today. That's Homework 5? Right. Okay. And what was the question?

Student:[Inaudible].

Instructor (Stephen Boyd):10.2? I haven't memorized them, so you'll have to tell me what it is.

Student:[Inaudible].

Instructor (Stephen Boyd):What did we ask [inaudible] – I'd have to see the context.

Student:[Inaudible].

Instructor (Stephen Boyd):Oh. It's your choice, but what's it for. Is it a simple one?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah. Okay. Do you know – I believe that to sketch a – not sketch, I guess to get a – there's a command. This could be completely wrong. I believe it's "quiver." Is that right? So some people are shaking their heads affirming that this function in MATLAB will actually plot – I guess quiver is for a group of arrows or something. Don't ask me. Apparently that's what it does. That would sketch it, but you're also absolutely welcome to sketch it by hand, just to draw – take a couple of points, see where they go. That's the important part. Okay. Any questions about last time? If not, we'll continue. We're looking at the idea of left and right eigenvectors. In our next example, very important, you go down [inaudible] pat is a Markov chain. So in a Markov chain, the way we do it, the state is a vector. It's a column vector, and it actually is a probability distribution. It's the probability that you're in each of N states. So PFT is a column vector. Its entries are all positive or not negative, and they add up to one, and they represent a probability distribution. That probability distribution evolves in time by being multiplied by the state transition makers, capital P. I warned you last time that if you see this material in – I guess, by the way – is that coming out. That is straight there. My monitor has this kinda twisted at a five-degree angle or something like that. It's okay. So I warned you last time that if you take a course in statistics or in other areas that this is the transpose of what you'll see there. Their matrix P will be the transpose, and they actually don't propagate column vectors. They propagate row vectors. Yes? Yes, what?

Student:[Inaudible].

Instructor (Stephen Boyd):It is tilted. Ah ha. Well, then. I wonder if you guys can rotate your overhead camera a little bit to make this straight. There we go. Thank you. Okay. So these matrices have the sum over columns equal to one, so that's what this is. Every column sum is one. And a matrix like that is called stochastic. Now you can rewrite that this way. If I put a vector in front of all ones – that's a row vector multiplied by P – I get this row vector here. This is the matrix way of saying that the column sums of P are all one. So this also if you look at it – if you like, I could put a lambda in there and say lambda is one. This basically says that P – that the vector of all ones is a left eigenvector of P, associated with eigenvalue lambda equals one. It tells you in particular P has an eigenvalue of one. But if it has eigenvalue of one, it also has a right eigenvector associated with lambda equals one. And that means there some nonzero vector with PV equals V. That's what it means to have a right eigenvector with eigenvalue one, so PV is V. Some people would say that V is invariant under P. It's invariant when multiplied by P. Now it turns out that because the entries in P are positive, this is not something we're gonna look at now. This eigenvector here can actually always be chosen so that its entries are nonnegative, and that means we can normalize it so that the sum of them is one. Now the interpretation then is if you have PV equals V, the interpretation is just beautiful. It's an invariant distribution. It basically says if you're in the distribution given by V, that probability distribution, and you update one step in time, you will remain in the same distribution, so it's an invariant distribution. By the way, that obviously does not mean the state is invariant or anything like that. The state is jumping around. This describes the propagation of the probabilities of where the state is. The state is stochastically invariant. That's what PV equals V means. Okay. Now we'll see soon in fact that in most cases, no matter how you start off the Markov chain – and this depends on some properties of P – you will in fact converge this equilibrium distribution. So that's something we'll see actually very soon. Okay. So let's look at this idea of diagonalization. Let's suppose that you can actually choose a linearly independent set of eigenvectors for N by N matrix A. So we're gonna call those V1 through VN. By the way, we're soon gonna see this is not always possible. It is not always possible to choose an independent set of eigenvectors of A, but assuming right now it is possible, you write this as AVI equals lambda IVI, and will express this set of matrix vector multiplies as – we'll just concatenate everything and make it in matrix language. And it says basically AV – A times a matrix formed – that's a square matrix formed by just concatenating the columns – is equal to V and to multiply each of these Vs, that is multiplication on the right by a diagonal matrix, so we have this equation. So you can write this this way. AT equals T lambda, so there you go: five ASCII characters which expresses the fact that the columns of T are eigenvectors associated with the eigenvalues of lambda I. So a very, very snappy notation to write this. So actually, now I can explain this business. I said mysteriously earlier that you can write an eigenvector this way. And in fact, that's the way most people write it, but in fact, if you wanna sort of follow this, and make this work for T being a single column or something like that, you can actually write it as this, and that's – if you see somebody write that, it's for one of two reasons. Either they're just kind of like weird, or being perverse by writing the scalar on the right of the vector not the left. Also of course, this requires – to interpret, this requires loose parsing. Or they're actually quite sophisticated, and they're simply writing down the scalar version of this eigenvector eigenvector. I should call this the eigenvectors equation. That's AT equals T lambda. Okay. Now T, because these are independent – T, which we haven't used yet – this matrix T is invertible because it's a bunch of – it's N independent vectors. That's nonsingular. It can be inverted. I multiply this equation on the left by T inverse, and I get T inverse AT is lambda. Okay? So that's the big – This you've seen before. That is a similarity transformation, and that's a diagonal matrix. It's the eigenvalues, and in fact some people would say the matrix A is diagonalized by a similarity transformation. And in fact, the diagonal matrix is the matrix of the diagonal. Diagonals are the eigenvalues, and the matrix that diagonalizes A is actually the eigenvector matrix, so it looks like that. Okay. Now this is just – it's really in some sense just a change of notation, but it's okay. Suppose you had any matrix T that diagonalized A by similarity. So T inverse AT is diagonal. Well, let's see. I'll call these diagonal ones – why don't I just call them lambda 1 through lambda N? And why don't I call the columns of V – of T V1 through VN? If I do that, and I take this equation, I rewrite this as AT is T lambda. Now I examine these column by column, and column by column that says this. So basically, if you see an equation like T inverse AT equals lambda, or AT equals T lambda – if you see either of these equations or these like that, it means the same thing. They're just different ways of writing out in different matrix form what this means. Okay? So actually technically, these two make sense. These two make sense even if the VI are not independent. But for this last one to make sense, obviously you have to have a VI independent. Okay? So that's the idea. Okay. So we'll say that a matrix is diagonalizable if there is a T for which T inverse AT is diagonal, and that's exactly the same as saying A has a set of linearly independent eigenvectors. This is identical. Now sometimes you're gonna hear – I think it's an old word. I hope it's going away. It says that if A is not diagonalizable, it's sometimes called defective. So I don't know where or how that came about, but that's simply what it's called. You still do hear that occasionally. Otherwise, it's called diagonalizable. And it's very important to quickly point out not all matrices are diagonalizable. Here's the simplest example, zero, one, zero, zero, characteristic polynomials S squared, eigenvalue – only eigenvalue is lambda equals zero. There is no other eigenvalue here. Let's try to find now two independent eigenvectors, each with an associated eigenvalue zero. That's the only eigenvalue there is. Well, to say that you're an eigenvector of A with eigenvalue zero, that's just a longwinded way to say you're in the vel space. So let's try to find vectors that satisfy zero, one, zero, zero, V1, V2 equals zero if you like times V1, V2, like that, same thing. Well, the first equation, if you look at the first row, it says V2 is zero. So V2 has to be zero so V1 is the only – that's the only thing we have to mess with here. The second row is always satisfied. And there's no way you can pick two vectors of this form that are gonna be independent. It's impossible. Okay? So here's the canonical example of an A which is not diagonalizable. Now we're gonna see actually later today that this issue – it's an interesting one. It's a bit delicate. We'll get to it. It's gonna turn out nondiagonalizability – hey, that came out right. I was about halfway through that and wondering whether it was gonna come out right, but I think it did. I won't attempt it again. Anyway, that property is not – it's something that does – it actually does come up. It actually has real implications, but in many cases matrices that you get are diagonalizable. Not all sadly because that would mean a whole chunk of – a bit of the class we could cut out, which I would not only cut out, but cut out with pleasure. Sadly, can't do that. So there are a couple – we're gonna see the general picture very soon, but for now we can say the following. If a matrix has distinct eigenvalues, so if the matrix – if they're all different, all N eigenvalues are different, A is diagonalizable. Okay? And we'll show this later, but that's just something to know. By the way, the converse is false. You can have repeated eigenvalues but still be diagonalizing. Actually, somebody give me an example of that. What's the simplest – just give a simple matrix with repeated eigenvalues and yet – identity, there we go. So the identity in R – what's your favorite number? Seven. Thank you. So that's an R seven by seven, and very nice choice by the way. I don't know if you chose that, but good taste. R seven by seven, so it's got – well, it depends if you count multiplicities – well, I can tell you right now the characteristic polynomial is S minus one to the seventh. That is the characteristic polynomial. It has you would say seven eigenvalues at S equals one. So that's about as repeated as you can get. All the eigenvalues are the same. Does I have a set? Can it be – well, I could ask stupid questions like this. Can I be diagonalizable? Can it be diagonalized? And the answer is it already is. So you'd say well what would be T? You could T equals five. So you could take as eigenvectors E1 through EN. By the way, is it true that any set of N eigenvectors of I will diagonalize I or as independent? Let me try – I'll try the logic on you one more time. You can choose a set of seven independent eigenvectors for I. For example, E1, E2, E3, up to E7. Now the other question is this. Suppose you pick seven eigenvectors of I. Are they independent? No. Obviously not, because I can pick E1, 2E1, 3E1 and so on, up to 7E1. They're all eigenvectors. By no means are they independent, so okay. All right. Now we're gonna connect diagonalization and left eigenvectors, and it's actually a very cool connection. So we're gonna write T inverse AT equals lambda. Before, we had multiplied by T and put it over here, AT is T lambda. But in this case, we're gonna write it as T inverse A equals lambda T inverse. Now we're gonna write this out row by row. I'm gonna call the rows of T inverse, W1 transpose through WN transpose. I'm gonna call those the rows. And then in this way, you can right matrix matrix multiply as a batch operation in which you multiply the rows here by all the rows of the first thing by this matrix here. In other words, this is nothing but this. This one is this. And now, if you simply multiply out on the left, the rows of the left hand side are simply this. And the rows of the right hand side are this because I'm multiplying a diagonal matrix by a matrix. If you multiply a diagonal matrix on the left, it means you're actually scaling the rows of the matrix. If you multiply a diagonal matrix on the right, you're actually scaling the columns. You might ask how do I remember that? Well, I remember that when I'm teaching 263 mostly, but then I forget. And then what I usually do is if I have to express row multiplication, I will secretly – because this is not the kind of thing you wanna do in public, let anyone see you doing. I secretly sneak off to the side and I write down the two by two example to see if row multiplication is on the left or the right. And then it's on the left, I'll find out, and I'll say I knew that, and that's how I do it sometimes. Just letting you know that's how I know. But at least for the purpose of – while we are doing 263, I will likely remember. All right, so this says this. If you look at that equation, that's very simple. This says nothing more than the row – than these WIs are left eigenvectors, period. Now in this case, the rows are independent. Why? Because T inverse is invertible. Its inverse is T, so the rows are independent, and this means that you have an independent set of left eigenvectors. And they're chosen – in this case, their scaling is very important. Actually, let me just review what this says. It says that if you take a linearly independent set of eigenvectors for a matrix and concatenate them column by column into an N by N matrix, you invert that matrix and you look at the rows of the result. Those are left eigenvectors, so that's what that tells you. Those are left eigenvectors. But it's not as simple as just saying that it's any set of left eigenvectors. They're actually normalized in a very specific way. They're normalized so that WI transpose VJ is delta IJ. Right? Which is – basically this is saying T inverse times T equals I. Okay? So what you would say is in this case, if you choose the left – if you scale the left eigenvectors this way, they're dual bases. Now I should make a warning here, and that is that if a vector is an eigenvector, so is three times it, and so is seven times it. In fact, any scalar multiple, so when you actually ask for something like left eigenvectors or something like that in some numerical computation, they have to be normalized somehow. Generally speaking, I believe this is the case, most codes – which by the way all go back to something called LAPACK – will return an eigenvector normalized in norm. Notice it'll simply be – have Norm 1. If you just walk up to someone in the street and say "Please normalize my eigenvector," or "I'm thinking of a direction. Where's the direction?" people will just normalize it by the two norm. There's other cases where they don't. I wanna point out that that will not produce this result. These are not normalized here, so be very careful. These are not normalized. Okay. We'll take a look and see how this comes out. This'll come up soon. Okay. Now we can talk about the idea of modal form. So let's suppose A is diagonalizable by T. From now on, you are to simply recognize this statement as saying this is exactly the same as suppose A has an independent set of eigenvectors V1 through VN. If you shove them together into a matrix, I'm gonna call that T. That's what this means. Well, if you take new coordinates to be X equals TX tilde, what this means is you are – X tilde are the coordinates of X in the T expansion. Or if the columns of T are the Vs, it's in the Vs. So you would actually say X tilde gives the coordinates of X in what people would call in this case the modal expansion, or the eigenvector expansion. In other words, instead of writing X, which is in the standard bases or the coefficients, X tilde are the mixture of V1 through VN that you need to reconstruct – to construct X. That's what X tilde is. Well, this thing is X dot, and that's A, X, X is TX tilde, so X tilde dot is T inverse AT. That's diagonal, and you get this. So in that new coordinate system, you get X tilde dot equals lambda X tilde. And that means that by this change of coordinate that the autonomous linear system, X dot equals AX is decoupled. So in X dot equals AX, the way you should imagine that is a bank of integrators with a matrix A in a feedback loop. But a matrix A basically takes N inputs and produces N outputs, but it's got all these horrible cross gains. I mean if A has all entries non-zero, it means every output depends on every input, and so if you're mixing the dynamics [inaudible] basically what's happening. When you diagonalize like this, you have completely decoupled all the dynamics and it looks like this. Okay? And that says that at least in this coordinate system, it's very simple. The trajectories give you N independent modes, and the modes are just simply – well, obviously they have nothing to do with each other. They're totally decoupled. And this is called modal form. That's a very common form. Now these can become complex, in which case it's a bit weird, and you have to explain a little bit, and make a story about how the real and the imaginary parts are separately solutions, and all that kind of stuff. Another very common form you'll see is real modal form, and you'll see this for example in mechanical engineering a lot as real modal form for example for a structure. That's how they would describe the dynamics of a structure by giving real modal form. Now in this case, you can – there's actually a way to construct a real matrix S so that S inverse AS is not diagonal, but it's diagonal with one by one or two by two blocks like this. Okay? So every time in T inverse AT, every entry in this which is complex – that means for every complex eigenvalue, you'll actually collect that and its conjugate, and then actually you can take the real and imaginary apart and so on, and you'll actually get a form like that. I'm not sure – I don't remember if we've actually – hey, we don't tell you how to construct S. That would make an excellent homework exercise. Yeah, okay. So that's the [inaudible], so find S. It's a good exercise to see if any of this makes sense, mess with matrices and things like that. Okay. So you get all these little blocks – by the way, these little blocks like this, you should by now start recognizing. So a little block that looks like sigma N, sigma N, omega N, omega N, the characteristic polynomial of this is S minus sigma N – sorry – squared, plus omega N squared like that. Now assuming here that omega is less than sigma – I believe that's the condition here. Oh sorry. Absolute – no, sorry – let me keep this out of here. The roots of this thing are gonna be minus – no, they're gonna be sigma N plus minus square root of – let's see – I guess it's omega N squared minus sigma N squared, something like that. Okay? So that's what it's gonna look like. Now these things are the complex eigenvalues, so that is actually negative. Otherwise, this would be two real parts and can be split. And it should kinda make sense because this is the self-exciting component between X – if this were a two-dimensional system with X1 and X2, and these are the cross components which actually give you the rotation. So this condition basically says you get enough rotation. Otherwise, it splits into two. I can tell by facial – just a quick gross look at facial expressions, I've now confused almost everyone, except [inaudible]. Yes?

Student:[Inaudible].

Instructor (Stephen Boyd):How did I look at it and immediately say it? Well, it wasn't totally immediate, but let's – I – two by two [inaudible] two by two matrices, that's where there are basically no excuses, right? So for two by two matrices you should be able to work out one of the eigenvalues with the inverse and things like that. Above that, no one could possibly hold you responsible for it. But two by two, let's just two it. It's det SI minus ABCD. I mean these are the kind of thing I guess you should know, like that. And so I did this in my head, but I sort of knew what the answer was, so you get this times – minus BC. And now if I like, I could write my quadratic formula which I don't dare do. That would be some horrible expression. This was easier because after all, B was equal to minus C, and A was equal to D, so fewer things to keep track of, but this is what I did in principle. Now one of these two by two blocks, which is a complex mode, they look like this, and they're really quite pretty. They are essentially cross-coupled. It's a pair of integrators cross-coupled. And by the way, if sigma is zero, you get the pure oscillation, and this is something we've seen before. In fact, that matrix is zero omega minus omega zero. That's when sigma's zero, so you get this. You get that. And this you should recognize by now as a rotation matrix, I mean maybe – sorry, this is not a rotation matrix. Well, it is, but in this case – this you should recognize as a harmonic oscillator. By the way, you don't have to recognize this as a harmonic oscillator, but we talked about it, and the little bit of playing around with this would convince you that it is. Yes?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah. Sorry. What's that? Oh, this is not it. Thank you. This is this? Like that? Sorry? With a square.

Student:[Inaudible].

Instructor (Stephen Boyd):Really? Good. Great. Fine. Sorry. Actually, you're right. Of course, this is – fine, so it's sigma N plus minus J omega N. There we go. It's that? Thank you for fixing it. How come no one else caught that? Hm. Well, that's one more demerit for this class. That's bad. All right. So diagonalization, it simplifies a lot of matrix expressions. So diagonalization, I'll say a few things about it. It's mostly a conceptual tool. There are a few places where it actually has some actual use. It's widely used for example in mechanical engineering. There in fact is a very famous – there's very famous codes that will take a description of a mechanical thing, and then spit out the modal form, so it'll list eigenvalues, and it'll actual give you the eigenvectors, which in fact are these modal ones, not the complex ones, the two by two blocks. But in fact, mostly it's a conceptual tool, and let's see why. It's gonna simplify a lot of things. So if you have SI minus A inverse, that's the resolvent. Generally, if A is two by two, no problem, you can work out what this is. If A is three by three, this is already not a picture – it's nothing you – you don't wanna see the expression for SI minus A inverse. Trust me. You can compute it easily for some particular A, but that's another story. However, we can do the following. Wherever you see A, you replace it with T lambda T inverse like that. So we do that here, and now I'll show you a trick. This identity matrix you write as T T inverse. Okay? Now I pull a T out on the left, and a T inverse out on the right in this inner matrix. I invert a triple product that's the same as the inverse the other way around, so I get T SI minus lambda inverse T inverse. By the way, inverting a diagonal matrix, that's fine. That's easy to do. You invert the diagonals, and you get this. That's the resolvent. Okay? By the way, this is sometimes called the spectral resolution of the identity or some – there's some name for it. There's a name for this way to represent the resolvent. Actually, let me say a little bit about that. Some of you might know about the idea of residues in complex analysis. Then again, maybe none of you know about residues or partial fraction expansions. Partial fraction expansions? Come on. Somebody's heard of that. Did you learn about that in some stupid signals and systems course? Is that – yes. Is that where you heard about it? Okay, great. So this says the partial fraction expansion of the resolvent is this. It's really quite cool. Let me try to get it right. Oh, I think I can get it right. It's this, and I'm using informal syntax here. So that's the partial fraction expansion of this. Partial fraction expansion of a rational function is to write it out as a sum of terms each of which is one over one minus – S minus a pol, so that's the partial fraction expansion. Okay? The other way to do is say that these Rank 1 matrices, VI WI transpose are the residues of this function at – that's the residue of this function at the pol lambda I. Okay. So the diagonalization simplifies tremendously this, the resolvent. It's also true for the powers. For example, if you raise a matrix to power, if you know it's eigenvectors and eigenvalues, it's very straightforward because you simply write T lambda T inverse to the K. Now when you do this, you just line these K times, and the T inverse here annihilates the T to the left of it. And this happens in all of these, and you get T lambda to the K T inverse. Okay? So that means it's very straightforward in fact to calculate powers of a matrix. And in fact, this already is a method perhaps not – maybe not a good one, but at least conceptually this gives you a very good way for example of simulating a dynamical system. Or if someone walks up to you on the street, and asks what's A to the one million, probably the worst thing you could do would be to write a loop that keeps multiplying an N by N matrix by A and let it run a million times. That would be one way to do it. And the cost of that would be ten to the six times N cubed if you did that loop. Okay? Now you can also do the following. You could also calculate the eigenvectors and eigenvalues, and although I'm not gonna get into the details, that could also be done in order N cubed, like five N cubed or something like it. This doesn't matter, but I just wanna make a point here. Once you calculate this and this, which costs N cubed, let's say five, this is nothing but a bunch of calls. It costs basically zero – it costs N, which is basically dominated by the N cubed. So this can be done in five N cubed. Okay? And the point is that is a whole lot smaller than that. So diagonalization in this case actually gives you a very serious advantage in how to compute something. Okay. Let's look at exponential. You wanna look at E to the A in general. Calculating E to the A is a pain. It's not fun, unless A is two by two or has some other special structure like diagonal or something like that. It's actually quite difficult. Let's write out the power series, and if you write out the power series here, not surprisingly when you make a – we've already seen when you have a power, it's the same as simply putting the T and the T inverse here, and having these things – these are all diagonal matrices, and that's nothing but this. Exponential of the diagonal matrix is ex and diag commute for a matrix exponential. You get this like that, and that gives you a very simple way to compute the exponential. That's not quite how it's done, but it's one of the methods that can be used is this. Okay. Now in fact, this idea extends to analytic function, so if you have a function which is given by a power series, that's an analytic function – you don't need to know this, but it's just kind of cool, so it doesn't hurt. So if you have an analytic function that's a function given by a power series – it could be a rational function. It could be exponential, anything else – then it's absolutely standard to overload an analytic function to be called on N by N matrices. So F of A is given by beta zero I where the power series expansion of F is this. Okay? You've seen one specific example. You've seen so far the exponential. This allows you to work out this thing. So here for example, we would have the following. We would have F of A is T times – well, let me write it this way: diag of F of lambda I times T inverse. There's your formula like that. Okay? So it gives you a very quick way. This is actually something good to know about because there are a lot of times when you do see things that are polynomials of matrices, rational functions of matrices, and things like that. Those do come up a lot, and it's good to know that if you diagonalize, they will simplify. They'll simplify actually just to this single analytic function applied to the eigenvalues. Okay? Actually, I'm not quite sure why this part didn't get into the notes. That's very strange. Okay. Let's look at the solution of X dot equals AX. Well, we already know what it is. It's simply this. It's X of T is the X of TA times X of zero. That's what it is. That's the solution. And we already have a rough idea of what this looks like and things like that, or we have some idea of it anyway. Although, this is not the kind of thing a person can look at right of the bat – just take a look at a four by four matrix or for that matter forty by forty matrix – look at it and say, "Oh, yeah. That's gonna be a rough ride in that vehicle," or something like that, or "Yeah, that's gonna be some business cycles. I can see them." I mean there's just no way anybody can do that, unless A is diagonal or something like that. Okay. Well, let's start with X dot equals AX. T inverse AT is lambda. X of T is E to the TA X of zero. That's the solution, but this thing is this, and then this has really the most beautiful interpretation because T inverse X of zero, I write it this way: WI transpose X of zero. That's a number.

Then it's multiplied by this thing, which actually tells you for that eigenvalue eigenvector whether it's – this tells you whether it grows, shrinks, if it's complex, oscillates, and all that kind of stuff. And then this gives you the VI, reconstructs it. So the interpretation of this formula is really quite beautiful, and every single can be interpreted. It's this. What happens is you take the initial state X of zero, you multiply by WI transpose, and you get something very specific. That is the component of X of zero in the VI expansion. So for example, if W3 transpose X of zero is zero, that has a meaning. It says if you expand X of zero in a V expansion, you will not have any V3. That's what this says. Okay? Because that's what this does. It decomposes it this way. In fact, let me write that down right now. X of zero is sum WI transpose X of zero times VI. That's the expansion. And that's true for any X zero. Okay? This looks fancy. This comes from the fact that – I mean this is actually quite straightforward. This basically is a restatement of this. There's nothing [inaudible]. Okay? So WI transpose, which are the left eigenvectors, they decompose a state into the modal components, if you wanna call the V1 and VN the modal components. That's what this does. All right, that's fine, so that you decompose it. Then this thing, time propagates that mode. That's what E to the lambda T does. It propagates the Ith mode in time, very simple formula. Why? That's the point of a mode. A mode propagates in a very simple way. It grows. It shrinks. It oscillates. That's about it so far, by the way. And then it comes out along a direction VI, so all the parts of this are supposed to make sense, so that's the interpretation of this. Okay, but now we can actually ask some really interesting questions and answer them. You might ask this. You have X dot equals AX. Now remember, we say by definition A – the system is stable. By the way, you would normally say the system X dot equals AX is stable. Sometimes, as a matter of slang, you'll hear people talk about A being stable, but that's – it should be understood that's slang. So this system is stable provided all solutions of X dot equals AX converge to zero, they all decay. But now we're asking this. For a general A, for what initial conditions do you have X of T goes to zero as T goes to infinity? By the way, this one answer, you can always give, no matter what A is. What's that answer? Zero. If X of zero is zero, then it stays zero, period. And therefore, it goes to zero. So the initial state zero, no matter what A is, gives you at least one trajectory that converges to zero. I mean converge is a little bit technical there. It is always zero, but that means it converges to zero. Okay. Now the way to answer this is you dived the eigenvalues into those with negative real part, so let's say that's the first S of them, and the others, so these have nonnegative real part. Now we can answer the question lots of different ways. One is this: you say now that's just a formula for X of T. This thing will go to zero, provided the following holds. The first S terms in here shrink. The remaining S plus one through N do not shrink. Therefore, this will go to zero provided these numbers are zero for S plus one, S plus two, and so on. That's one way to say. So one way to say it is that these numbers should be zero for S plus one, S plus two, up to S equals N. By the way, that is identical to saying that you're in the span of V1 through VS. Why? Because X of zero is equal to sum WI transpose X of zero times VI like that. You have this. Therefore, to say that these are zero from S plus one to N means that X of zero is a linear combination of V1 through VS. Okay? So these are two different ways to say this. And there'd be all sorts of names people would call this. They would say that – they would refer by the way to this span. They would call that the stable eigenspace or something like that. That would be one – or some people would just call it the stable subspace, and the idea would be this. If you start in this subspace, the trajectory will go to zero. It might oscillate. It'll go to zero. If you're not in this subspace, you will not. So that's how that works. Okay, so that's the kind of question you can answer now. And finally we'll handle this issue of stability of discrete time systems. So suppose the matrix is diagonalizable, and you have the linear dynamic [inaudible] XT plus one is AX of T, then the solution is trivial. It's just powers of A. But if you write A as T lambda T inverse, then A to the K is this. Now I understand – I know how powers of complex – I know what powers of complex numbers do. That I can actually handle, and so you get this. Powers of complex numbers go to zero only if their absolute value is less than one. Their imaginary part tells you about how much of a rotation in degrees you get at each step, but their magnitude tells you how the magnitude scales, and you realize that XT plus one is AX of T is stable if and only if the eigenvalues are less than one in absolute value. Okay? And it turns out this is gonna be true even when A is not diagonalizable, so I don't mind stating it as a fact right now. XT plus one is AX of T is stable if and only if all the eigenvalues have magnitudes less than one, so that's the condition. Actually, as in the continuous time case, there's a much more subtle statement. The spectral radius of a matrix is defined to be the maximum of the absolute values of the eigenvalues. Okay? And so one way – this is called a spectral radius. It's denoted row. This is relatively standard notation. What this says is X – the discrete time autonomous system XT plus one is AX of T is stable if and only if the spectral radius of the dynamics matrix A, or update matrix, or whatever you wanna call it – the spectral radius is less than one. That's the condition here. Now more generally, row of A gives you the growth or decay magnitude, asymptotic. So for example, if row is 1.05 – in other words, there is at least one eigenvalue with a magnitude of 1.05, it says that X of T will grow asymptotically. It depends on the initial condition, but it can grow as fast as 1.05 to the T. If the spectral radius is 0.7, it says that after ten steps roughly, the state has decayed roughly by 0.7 to the ten. That's a small number. Okay? So this is the continuous time analog of the maximum of the real part of the eigenvalues of a matrix. That's what this gives you. Okay, so enough on all that. We'll now do a bit on – maybe I'll even cover it, the Jordan canonical form. So here I actually have to ask, how many people have seen – or perhaps right verb is been subjected to the Jordan canonical form? So a handful of people. Did it make any sense at the time? How should we interpret this?

Student:[Inaudible].

Instructor (Stephen Boyd):It did? Okay. Good. Let's look at it. So the Jordan canonical form is essentially – it's as close as you can get to a diagonalization when you can't diagonalize the matrix. So let me explain that. It's this. Any N by N matrix – any, no exceptions – can be put in something called Jordan canonical form by a similarity transformation.

So there's a matrix T, obviously invertible because I'm about to refer to T inverse, for which T inverse AT is J. J – that's the Jordan form – is a block matrix. Each of these blocks, which is called a Jordan block, looks like this. It's a bunch of lambdas with ones on the superdiagonal. So that's a so-called Jordan block. By the way, a one by one Jordan block is a little matrix that looks like this. Okay? So a diagonal matrix is a special case of a Jordan form. It's the special case when there are N Jordan blocks and each block is one by one. So basically, you have these little ones in the superdiagonal. We'll see what that means soon, what the ones mean. So that's a Jordan block. Now a couple of comments about Jordan blocks. First of all, the Jordan form is upper bidiagonal. That's a name meaning it's got a diagonal, and it's got – one diagonal above it is nonzero. It's upper – it's much more than that because in fact there's ones in the upper – it's in the zero one, the upper diagonal, and not only that, it can only be one if the lambdas are repeated there. So diagonal is a special case of N Jordan blocks. And it's gonna turn out the Jordan form is unique. Now you have to interpret that very carefully. It is not of course on the details of the mathematics of linear algebra, so it's not like we're gonna get crazy with all this, but it's important to understand what it means to say it's unique. It says that basically if two people calculate a Jordan form for a matrix, they actually can be different. One difference is simply this. They might order the blocks in a different way. However, the following is true. If two people work out a Jordan form, they have different Ts here possibly, then there's a permutation – a block permutation which will change one Jordan form into the other. So the way you would say this is you would say the Jordan form is unique up to permutations of the blocks. So the things people can – the types of things people cannot agree on is what is J1. No one can agree on that because it depends on what you chose to put as the first Jordan block. However, you can't – no one can possibly disagree about the numbers of Jordan blocks for example, and the sizes of them, and the sizes associated with a certain eigenvalue. So for example, if you say this has three eigenvalues, this eigenvalue is associated with one Jordan block of size two by two, everyone computing a Jordan decomposition will actually – will agree with that. Okay. Now I should also mention Jordan canonical form is – it's an interesting thing. It is almost strictly a conceptual tool. So it's used to show things, to illuminate ideas, and things like that. It is actually not used in almost any numerical computations. Okay? So if you go to the web or something like that – if you go to Google and type let's say – if you type something like "source code Jordan canonical form," you will get – actually, what you'll mostly get is you'll get a bunch of harangues about how terrible it is, and no one should ever compute the Jordan canonical form by – numerically, and so on and so forth. That's probably what you'll get, but you'll find some strange things there, but it's basically not done. Even when you do find algorithms for it, every paper will start this way. It will say, "It's well-known that you basically – it doesn't make any sense numerically to compute the Jordan form." It goes on, and it says, "But let's suppose you did. You really had to. Then this paper's about how you might do it, if you were to do it, but we don't recommend it." So that would be the kind of abstract you'd find. Not that this matters. I'm just mentioning it. Okay. Maybe it's not never, but it's awfully close, and boy do you have to justify yourself if you actually do anything like this in any numerical application. All right. Now the characteristic polynomial of A is – of course, if J is block diagonal – so the characteristic polynomial of – actually, under a similarity transformation is the same. Wasn't that a homework problem? It wasn't? That's terrible. Well, similarity – wait a minute. Oh well. That's – maybe it shouldn't have to be. Was it one? Well, I'm glad to see though that everyone thought about that problem a long time and really – in fact, that's great because it's actually below even your consciousness now. It's so ingrained –

Student:I think it's the current homework.

Instructor (Stephen Boyd):It's what? Oh, the current homework. Oh well, that would explain it because that would've been terrible. I mean it's a quick calculation, but the characteristic polynomial under a similarity transformation, it doesn't change the eigenvalues. So the eigenvalues of A are the eigenvalues of this thing. That's a block matrix. Eigenvalues of a block matrix are the eigenvalues of all the blocks stuck together. Eigenvalues of that matrix, that's upper triangular. Eigenvalues of this matrix are lambda I with a multiplicity NI. The characteristic polynomial in fact of this is S minus lambda I to the NI here. That's just – SI minus JI. Okay. So basically, this tells you the following. If you wanna get the characteristic polynomial of the matrix, you take – it's the eigenvalues associated with the blocks raised to the block size. And now we immediately see the following. Once you believe in the Jordan canonical form, which I will not show how – I will not go through the week long proof that any matrix has a Jordan canonical form, especially because the computational algorithmic payoff is – to say dubious is putting it very nicely, so I won't go through that, but assuming you believe it, and you should – that is after all what we have mathematicians for, and they assure us that it's true. Then it says immediately that if a matrix is diagonalizable, its Jordan form must be – you can only have block sizes one by one. To say that a Jordan form has block sizes one by one says it's diagonalizable. That's basically what it says. Okay. Now this – when you see repeated eigenvalues now – so in fact, let me explain how this works. If you see repeated eigenvalues, it means maybe you have a nontrivial Jordan form. Oh, I should mention something here. If the Jordan blocks are all one by one, this is diagonal. People would call that a – if any block is two by two or bigger, people call that a nontrivial Jordan form, meaning diagonal is just diagonalizable. So if you see – what this says is the following. If the eigenvalues are distinct, your matrix is diagonalizable. And if someone says, "Yeah? What's the Jordan form?" You'd say, "I just said it's diagonalizable." Okay. Jordan form is N Jordan blocks, each one by one. That's the trivial Jordan form. If you see repeated eigenvalues, it does not guarantee that the Jordan – you're gonna have nontrivial Jordan form. In fact, somebody quickly give me an example of a matrix that has repeated eigenvalues and yet has a trivial Jordan form.

Student:[Inaudible].

Instructor (Stephen Boyd):I. What size? This is very important. We talked about this earlier. Seven. Thank you. So seven by seven – the seven by seven identity matrix has seven absolutely equal eigenvalues. Its Jordan form is trivial, which is a pedantic way of saying it's diagonalizable, and so – on the other hand, you can have more – in fact, let's talk about seven by seven. That's kinda big. Let's talk about like four by four. So here, that's the identity. Eigenvalues all are one. It's got four Jordan blocks of size one by one. Okay? How about this one? Eigenvalues are all the same. The eigenvalues of that matrix are one, one, one, one, period. What's the Jordan block structure? Well, there's one block here, and it looks like that. So you can have a block of two, and then two separate blocks of one. Are there others for this one? Well, I could do this. I could have a block of – let's see if I can get it right. No. Yes. If I'd given myself enough room, it would've been right. How about that? That matrix – what's the block – what am I doing? My god. Okay, let me just get rid of that. I didn't do that. There we go. Okay, here. Since I can't even get it straight, I'll show you the blocks. Okay? What about this one? What's the Jordan – I mean the block size here you would describe as it's got two Jordan blocks, one three by three, one one by one. By the way, eigenvalues, this matrix, this matrix identity, all the same. Any others? One more one. So we could have a single Jordan block – I don't know what I'm doing. Here we go. One, one, one – there we go. It's a single block of size four by four. And that would be the – and any others? Let's list all possible Jordan forms of a four by four matrix with four eigenvalues one. There's one we missed. What is it?

Student: Two two by two.

Instructor (Stephen Boyd):Two two by twos, so that's – exactly. So you can have this like that, and that's it. These along with I are the five possible Jordan forms for a matrix with four eigenvalues of one. Okay? Of course, a natural question in your mind would be – well, let me list some. The first might be who cares. So we'll get to that. And the second might be – and it's related – is what's the implications. What would it mean? How would you know? How would this show up for example in dynamics or something like that of a system? How would you know you had one of these, and what would be any consequences of it? Okay. And we'll get to that. I promise. So the connection between the eigenvalues and the Jordan blocks and sizes is a bit complicated, but it all comes from this. It says that basically if you have – the characteristic polynomial is a product of S minus lambda I to the block size I. And the null space for example of lambda I minus A is the number of Jordan blocks with eigenvalue lambda. And we can check that because what happens is if you look at the null space, if you look at lambda I minus A – I will multiply by T inverse in T like this, and T inverse in T goes in there and annihilates itself. It's basically – that's lambda I minus J, and that is equal to a block matrix that looks like this. It's lambda minus lambda one. And then there's some minus ones on the superdiagonal like that. And I won't draw the other blocks. Okay? Now if you wanna know what's the null space of this matrix, you have columns – at the leading edge of each Jordan block, you have a column whose only nonzero entry – possibly nonzero entry is lambda minus lambda I. So if lambda is equal to lambda I, you get a zero column, and that means that matrix is gonna drop rank. It is not gonna be invertible. So that's what – this happens. So in fact, this will happen. Every match at the beginning of a Jordan block, you will get a zero column, and that says in fact the dimension of the null space of lambda I minus A is exactly equal to the number of Jordan blocks associated with lambda I. So over here, let's look at that. What is the – let's look at the null space of lambda, which is one, minus the matrix A. And let's look in different – this is one times I minus A. Let's look at the different cases. If you take I, and I ask you what's the null space of one times I minus A, that's the null space of the four by four matrix zero. What's the null space of the four by four matrix zero?

Student:[Inaudible].

Instructor (Stephen Boyd):It's what? It's R4. It's all four vectors. So it's four-dimensional in this case. What is the null space of I minus A for this matrix? What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, I wouldn't say R1 because that means – that's just the set of real numbers. It's the set – it's one-dimensional, which is I think what you meant. It's one-dimensional, and it's all vectors of the form something zero zero zero – yes, [inaudible] something zero zero zero. It's one-dimensional in this case. Why? There's one Jordan block. It makes perfect sense.

This case, if you take I minus this matrix, these become minus ones and these become zeroes, and then you ask what is the dimension of the null space of that matrix, and the answer is two. That's two Jordan blocks. Same here, and in here it's actually – the dimension of the null space is three. Okay? So basically, the amount by which – the amount of rank that lambda I drops – lambda I minus A drops when lambda is an eigenvalue tells you something – well, actually it tells you exactly the number of Jordan blocks. That's not enough by the way to give you the full block structure. That comes out of lambda I minus A raised to various powers. And I'm not gonna go into this, but we can – in fact, I'm not gonna go into that, except that we can actually just – let me just ask a couple of questions. Suppose a matrix has eigenvalues minus one – with multiplicities, minus one, three, and five – five by five matrix. Let's enumerate all possible Jordan forms for that matrix. Let's start. What are the possible Jordan forms? What's the simplest one?

Student: Trivial.

Instructor (Stephen Boyd):The trivial one which is just diagonal, minus one, minus one, minus one, three, five – and if someone asked you how many Jordan blocks, what are the sizes, what would you say here? How would you describe the trivial – it's diagonalizable here.

Student: Five one by one.

Instructor (Stephen Boyd):Yeah, so you'd say it's five one by one. But you'd also say by the way, it's pedantic to talk about Jordan blocks when a matrix is diagonalizable. That should also – that should be the second part of your reply when someone asks you about this. Okay. Are there any others? Any other possible Jordan forms? For example, could I have – could the eigenvalue three correspond to a Jordan block of size two? No. Out of the question because its multiplicity is one. Same for five. So these two – no matter what happens, this matrix has two Jordan blocks of size one by one, one associated with eigenvalue three, one with five, period. And the only one where there's any ambiguity would be this little block of repeated ones, and what are all possible Jordan forms for three repeated eigenvalues of minus one? We've got one that's diagonal. What else? Two in one and what?

Student:[Inaudible]

Instructor (Stephen Boyd):And three. Okay. In this case, if I asked you – if I told you what the dimension of the null space of minus one minus I minus A is – if I told you that number, could you then uniquely determine the Jordan form of A? I'm getting this out and ready. What do you think? You could. And the reason is the dimension of the null space of minus I minus A can be either one, two, or three. If it is three, it means A is diagonalizable, end of story. If it is two, it means this – there is one block of size two by two and one of size one by one. If it is one, it says there's a single Jordan block, period. And therefore, you have determined it. Warning! That is not always the case. If I told you that a matrix is four by four, has four eigenvalues of one, and the dimension of the null space of one I minus A is two, that does not tell you the Jordan form of A. Why? Because you don't know if it is this one or this one. Each of these has two Jordan blocks, and you don't know which it is. Okay? So that's the idea. Okay. Let's look at – well naturally, the columns of T in T inverse AT equals J are called generalized eigenvectors. These – you group these according to the blocks, and these are the columns that you associate to the Jordan blocks. If you call these – if you split these out as columns, then you'll get something like this. The first one comes out just the way you think it does. Or sorry, not the – yeah, the first one comes out just the way you think it does. It's AV1 is lambda times V1 here. That's the first one. But the next ones because of that upper triangular – sorry, that upper diagonal, you inherit this one. Each AVIJ is actually the previous one plus lambda times this, and so these are called generalized eigenvectors. You actually won't need to know this. They don't come up that often, but you will see this every now and then. You'll see people refer to generalized eigenvectors. Now for a – if you have X dot is AX, if you put a change of coordinates, if you put it in Jordan block form, basically that splits the dynamics into – it splits the dynamics into I guess K separate – independent blocks. Each one is a Jordan block. Now you should have an overwhelming urge to draw a block diagram of a Jordan block system, and this is it. It's a chain of integrators. The chain of integrators by the way corresponds – that's what that upper block of – that upper diagonal of ones is this – gives you a chain. So you should start thinking of an upper block of ones as giving you things like shift. It's a shift or it's a chain in this case like that, and so on. And then the lambdas are simply wrapped around this way. So interestingly, people who do engineering and mathematicians both refer to sometimes Jordan blocks as Jordan chains for totally different reasons. People in engineering refer to it as a chain because it's got this chain of – it's dynamics built around a chain of integrators. And in math, it's a Jordan chain because it's a chain of subspaces. So this only shows why if you're in engineering, so that's the dynamics you see. By the way, when you see this, if you remember things like – so actually, let me actually explain a little bit now because the main thing is to get a rough idea of what on earth does it mean if you have X dot equals AX and A has a Jordan block. This says that some of the dynamics is sort of connected in – would you call that series, or cascade, or something like that? That's what it means. It means that some of the dynamics feeds into the other. Remember what the diagonal system looked like. It's N boxes that look like this. So the Jordan blocks are actually gonna – it's gonna turn out it's gonna have to do with dynamics blocks that cannot be decoupled. That's what it's gonna be. It's gonna be dynamics blocks that cannot be decoupled because they're connected in cascade, not in parallel. Okay. And we can look at things like the resolvent and the exponential of a Jordan block. If you look at the resolvent, you see you have this upper thing here, but if you take the inverse of that, the inverse of an upper triangular matrix is not that bad to work out, and it looks like this. Actually, it's quite interesting because now you can see something else. You can see that when you take the resolvent of a Jordan block, you're gonna get powers – you're gonna get S minus lambdas to negative higher powers. Didn't have that before in the resolvent. So it turns out it's gonna correspond to – repeated pols in the resolvent are gonna correspond to Jordan blocks. Could work out the Laplace transform, and this will actually at least give you part of the idea of what the meaning of these things is. When you work out the exponential of a Jordan block, it turns out sure enough you get this E to the T lambda part. We're hardly surprised to see that, but now you can see what a Jordan block does. It gives you polynomials. So I think what I'll do is – let me say a little bit here. This tells you what you needed to know. When you see X dot equals AX, and let's make it simple – let's say all the eigenvalues are lambda, period. Okay? Now I can tell you what it means for this – what the Jordan blocks in A – if A is diagonalizable, the only thing you will see in the solution will be things that look like that, period. If there's a Jordan block of size two by two, you will not only see exponentials, but you will see terms of this form like that. That's only if there's a Jordan block of size two by two or larger. If there's a Jordan block of size three by three, you will see not only TE to the lambda T, but T squared E to the lambda T. Another way – you can turn it around and say that if you see a solution that looks like that here, that says that there is a Jordan block of size K plus one there. Did I say that right? Yes, K plus one. That's what it says. So Jordan blocks are basically gonna be the matrix attribute which you're gonna associate with T to the K – these terms which are a polynomial times an exponential. Okay? And let's actually just look at one example just for fun, and then we're gonna quit. Let's look at X dot – I believe this might have come up yesterday in your section, so there. I allocated on the page enough for a four by four. There you go. Thank you. It's fixed. Let's look at that. What are the eigenvalues? What on earth have I done? My god. That was a terrible crime. There we go. Okay. But you didn't violate the eight-second rule or something like that. When you write something that stupid down, something should say something within some amount of time. Okay, fine. What are the eigenvalues?

Student:[Inaudible].

Instructor (Stephen Boyd):All zero. Okay. So just based on that, what do you expect to see on the solution when you look at X? Constants, right? And someone says, "Anything else?" Now in this case, what do the solutions look like? The solutions here – that's a Jordan block of size – a single one. You are gonna see solutions that look like this. E to the zero T, that's one. You're also gonna see T, T squared, and T cubed. The solutions of this X dot equals AX for this thing are gonna be polynomials. Everybody cool on that? And they're polynomials of up to degree three. Now let's do one more thing. Let's change that so that it looks like this. Here's the block structure. What do you expect to see? Not expect. What would you see?

Student:[Inaudible].

Instructor (Stephen Boyd):You don't have this, and you don't have that, but you do have this. And finally, if it was just this, if it's all zero, you don't even expect – you just see constants. And of course, that's correct because X dot equals zero – the solution is that everything is just constant. Okay? So the neural – I mean you really wanna kinda understand all of this, but then the real neural connection you needed to make is dynamically Jordan blocks are these annoying terms of the form – they correspond to these. That's what they – that tells you there's a Jordan block somewhere in the neighborhood. Okay. So we'll quit here.

[End of Audio]

Duration: 74 minutes

IntroToLinearDynamicalSystems-Lecture14

Instructor (Stephen Boyd):Finish up Jordan Canonical Form. I'll probably say a little bit at the end of the section and then we'll start, in fact, the next section of the course.

So if you go down to the pad here, we can talk about Jordan Forms. You're going to want to move, I think, this way. I don't know, though. I think you got it. Okay. So last time we looked at the Jordan Canonical Form, which has many ways to describe it. One is to say that it's the closet you can get to diagonalization, if the matrix is not diagonalized. So you find a matrix T. So the T inverse AT is block diagonal. It's more than block diagonal.

Each block is, itself, upper by diagonal. It's got a bunch of repeated eigenvalues on the diagonal. On the super diagonal, it's got one, so that's the Jordan Form. It's a generalization of a diagonal matrix, is a Jordan Block. That corresponds to Jordan Blocks, which are all one by one.

Okay. Now, we looked at various things involving Jordan Form, and the real question was, what does it mean? I think we saw some of that from a dynamic point of view. We can get some of that by looking at the exponential of a Jordan Block. So the exponential of T times a Jordan Block looks like this. You get the familiar, E to the T lambda. So that you'd expect. That's the eigenvalue.

But now you can see that you're getting these powers of T. So Jordan Blocks should be associated, in your mind, with – when I say Jordan Blocks, I mean nontrivial Jordan Blocks. Jordan Blocks bigger than one by one – should be associating [inaudible] with these polynomials X E to the [inaudible].

So that's the same thing in an undergraduate course. [Inaudible] forms is associated with repeated polls. So in an undergraduate class, you see S plus 1 squared in the denominator. You're going to see T – E to the minus T in some solution or in the inverse of the plus transform.

So here, Jordan Block is going to take the role of repeated polls in an undergraduate case. And by the way, that's exactly correct. You can see here that if you look at the resolvant of a Jordan Block, you actually get these higher powers like this. And if it's a one by one block, you just get that. There's an interesting way to think of this. If you see that a matrix, for example, has four eigenvalues at minus one. Then you look at the resolvant. That's minus one, I, minus A, inverse. And you look at the poll and polls of the entries of the resolvant at the eigenvalue minus one.

In fact, if all of those have degree one, it means that it's four one-by-one Jordan Blocks. If you get degrees as high as three or something like that, that tells you all of them are in one block together. So degrees in the dominators in resolvents, powers of T multiplied by exponentials, these are all the symptoms of nontrivial Jordan Blocks. Okay.

Now, the same way we talk about a solution of the form E to the lambda T times V where V is an eigenvector as a mode of a system, you can talk about generalized modes. Now, unfortunately, the nomenclature for generalized modes is horrible. You look at 15 books, you'll get 15 slightly different definitions of what a generalized mode is, so it's not something that you need to worry about. But you should also know that the nomenclature is not consistent.

So here, if you take anything that's in the span – so TI is the sub matrix of T. It's the vectors associated – it's the columns of T associated with the I Jordan Block. If you take any linear combination of those, you'll find that that's an invariant subspace, that you stay in that subspace. If you start in that subspace, you stay there. So you stay in the subspace, and what you get is this. All the solutions have an exponential, and then they have polynomials multiplied by them, so that's what they look like.

These are sometimes called generalized modes of the system, something like that. But again, this nomenclature is not uniform. So things of the form, you know, T, E to the lambda T, T squared, those are generalized modes.

Now the inverse of T, if we call is S, then if we take the inverse and then we break up the inverse, S, row by row, conformably with the Ts, we get matrixes SI. That will extract out from an initial condition, the coefficients of X of zero, for example, in a generalized eigenvector expansion. So that's what this is. This pulls out the coefficients of the block in a generalized eigenvector modal expansion of X zero.

Then you propagate them forward in time. That's this, and then you reconstitute them like that. So these are the analogues of the formulas that we looked at before when A was diagonalized. So this is the picture. That's how that works.

I mean, all of it really happens a lot of times because these things just get more complicated. They look quite similar, but the indexing gets horrible because these things have different sizes and all that sort of stuff. Okay.

We're going to do one application of Jordan Canonical Forms. It's a generic application. This is really what it's used for. It's used for showing something about matrixes. So if you want to show something about square matrixes, many things are simplified by using Jordan Canonical Form. The way that works is this. You should always warm up by assuming the matrix is diagonalizable.

So if someone says, "Show that – anything. Just make up something about square matrixes." Then you want to start by showing it for the diagonalizable case, okay? Because if you can't do the diagonalizable case, there's no point even worrying about the horrible headache of the Jordan Canonical Form.

So you first warm up by showing it for the diagonal matrixes. Then you step back and you say – now you've done that, you say, "Now, let's do the Jordan Canonical Form." Now let's do the real case, and you'd do Jordan Canonical Form. So that would be the way to do it. We'll just see an example. It's a classic one. It's also important in its own rights. It's called the Cayley-Hamilton Theorem.

Actually, before we start this, we're just going to have a discussion about matrixes, about N by N matrixes. So if you have an N by N matrix, like this, I can have the set of N by N matrixes. That is, itself, a vector space. Its dimension is N squared. Okay?

If you have a basis with lots of bases, here's one. EIJ, which is EI, EJ transpose, that is the matrix that has a one in the IJ entry and nowhere else. Okay? Then you have IJ equals one to N. So these are independent matrixes. I mean, this is sort of the analogue of just EI, in the general case.

So the set of, say, ten by ten matrixes is a 100 dimensional vector space. This is one basis for it. Now let me ask you this. If I take any matrixes. Let's take any matrixes. So if I write down A1 up to A101, and there in R ten by ten. What can you tell me about 101 vectors in a 100 dimensional space?

Student:[Inaudible].

Instructor (Stephen Boyd):They are not linearly independent. Right, which is to say, "They are linearly dependent." So many ways to write it, but one, you can say, one of these matrixes is a linear combination of the other 100. Okay? So in particular, if I write down the matrixes IA up to A to the 101, one of those matrixes is a linear combination of the previous ones. You agree? That just comes from the fact it's 100-dimensional.

Now to say that the powers are dependent basically says there's a polynomial because a linear combination of powers is a polynomial. It says there's a polynomial of that matrix that is zero. One polynomial that's always zero is the zero polynomial, but I mean a nonzero polynomial.

So there exists a polynomial, P, for which P of A is zero, and P has a degree – well, in this case, it could have a degree of 101. I could argue in a fancy way that it has to have a degree of 100. Okay? So there's a polynomial – a nonzero polynomial of a matrix makes it zero. Now comes the interesting part. It turns out that in this case, there's actually one of degree ten. In other words, you don't have to go up to 100.

When you take the powers of a matrix, you go IA, A-squared, and you think of these as vectors. They're independent. But by the time you hit N, they're dependent, period, and they're dependent, of course, from then on. So it says that the dimension of the subspace spanned – that's the Cayley-Hamilton Theorem. Cayley-Hamilton Theorem says this. If you have any end-by-end matrix and you plug in its characteristic polynomial, if you evaluate the matrix – evaluate the characteristic polynomial with the matrix, that's an overloading here, to talk about a polynomial evaluated at a square matrix.

You get zero. This says something very interesting. It says lots of things. Basically, it says the following. If you take I, A, A-squared, A-cubed, and the powers of a matrix, it says by the time you get to N, that A to the N is a linear combination of the previous As. So you're in an N-squared dimensional space, but you actually only sweep out by powers, at most, a subspace of dimension N. So that's what this is saying.

It's going to have important implication. You can actually already think of some of the implications. You've already seen a lot of cases where powers of A come up. They come up in discreet time-dynamical systems, for example. We've seen that.

Actually, let's look at a wrong proof of this. I'll show you a proof of this that's wrong, but it sounds really good. If I did it quickly, you'd probably go for it. Actually, I could probably get away with it. I don't know, but that's only because the class is this early. Let me just try it.

So the wrong proof goes like this. It says, "What's the big deal?" Tie of A is debt – well, how do you evaluate a polynomial? You just plug – wherever you see S, you put in A. So you go A times I minus A, and I know what that is. That's zero, and that's zero. That would seem to be the end of the story. Like it? Oh, for sure, if I'd done that quickly, you absolutely would've bought that.

In fact, especially if I waved my hands wildly and said, "Obviously," and "Clearly," and things like that, and then quickly moved on, you would've gone for it for sure. I can just tell. No? Okay.

So it's a completely wrong proof here. You can't just plug a matrix and have A – actually, the correct thing to do here would be to say, "This original matrix over here, this SI, actually looked like this." So if you really propose to plug A in there, you should've made a matrix that looks like this. Like that. You can see now, things are going downhill. It's just not working. Okay? So this is a non proof.

Let's poke through and see a real one. Let's just do an example first. It's a stupid one. Here's a matrix, two by two. One, two, three, four. You work out the characteristic polynomial. That's S-squared minus five S minus two. The Cayley-Hamilton Theorem says that if I plug in – if I evaluate this polynomial with A, then I'll get the zero matrix. So I work out A-squared and minus five A and minus two I. These may or may not be correct. I guess this term is correct. That's probably A-squared.

The point is, if you actually worked out what all this was, you'd actually get zero. We can audit it. We'll do an audit. We'll check the one-one entry. So seven minus five minus two, it worked. Okay? So that's the Cayley-Hamilton Theorem.

Very interesting, it says that this matrix, it's square, is actually a linear combination of the identity and itself. That's – in fact, you can read that right off of here. If you have this as zero, it basically says A-squared is five A plus two I. So it says that power of A is a linear combination of I and A.

Okay. Now, if you think carefully about what that implies, it's actually really quite interesting. It says that if you take a matrix to any power, we're talking about positive powers, now. If you take a matrix to any power at all, it is always a linear combination of I, A, A-squared up to A and minus one. Period. Okay? So that makes perfect sense.

How do you know that? Well, we can show that very quickly. What we do is if you take a power of A, what I'll do is I'll do polynomial division, and I'll divide S to the P by the characteristic polynomial, and I can write it this way. It's a quotient polynomial times this plus – and then a remainder. This thing has to have degree less than N. Okay? Which is the degree of this, my devisor.

So I get a remainder polynomial. In fact, the remainder polynomial is exactly what the coefficients are in this expansion. Let me actually say a little bit about this because it's actually kind of interesting. It says if you have any analytic function of a matrix – oh, we just put a new problem on this. But you've already seen it for exponential. So if I have any function which is defined by a power services, so this is alpha zero plus alpha one U plus alpha two U-squared and so on. Then we overload this analytic function to apply to end-by-end matrixes, and it's just this way. Alpha zero plus alpha one A plus alpha two A-squared and so on. Like that.

You've already seen this once. You've seen the exponential. In fact, you can do this for, actually, any analytic function, even ones that – we'll just – just any analytic function like this. This is an analytic, add zero. So any analytic function, you can define this.

Okay. Now, let's see what Cayley-Hamilton Theorem has to see about this because it says something very, very interesting. It says the following. It says that when I go up to A to the N, there's a term, alpha N, A to the N, and then I get alpha N plus one, A to the N plus one. Like that. These keep going, but every term from N and on is actually a linear combination of I up to A N minus one.

You know what that means? That means any analytic function of a matrix actually must have the following form. It must look like, you know, gamma zero plus gamma one A – and you can stop at gamma N minus one, A to the N minus one. Period. So the exponential of a matrix is a linear combination of I, A, A-squared, to a A minus one. By the way, so is the inverse. We'll get to that in a minute, and that's the basis of a huge family of methods for solving large problems. Okay. This is all a consequence of the Cayley-Hamilton Theorem.

All right. Let's look at the inverse. That's even more interesting. It has huge practical implications. Absolutely huge. Here it is. Let's take the inverse of a matrix. Now, of course, not all matrixes are invertible, so somewhere in here, we're going to have to encounter something that tells us that the matrix is invertible. So I'm sort of on – I'm saying, "Please suspend –" assume it's invertible. Somewhere in here, we better encounter something about that.

What we do is we write this. This is a characteristic polynomial. It's A to the N plus A and minus one, A to the N minus one, plus all the way up to A zero I equals zero, here. Okay. And what I do is simply this. I put this one term on the other side, and I divide by A zero, and I get this.

IN fact, this says it's more than that. I then factor out A. This is the inverse there. That's A inverse. It says that the A inverse is an explicit linear combination of I A up to A N minus one. Okay? In fact, the coefficients, we can say exactly what they are. They have to do with the characteristic polynomial. They're A one over – minus A one over A zero, minus A two over A zero, minus one over A zero. The catch is this. A zero, the constant term in the characteristic polynomial, is something like – does anyone – do we have a homework problem on that?

We may have just assigned it just now. It's either debt A, or maybe there's a plus/minus one in front. I think there might be a plus/minus one in front of it, or is it just debt A. Anybody remember this? What is it? Debt minus A? That's fine. So I'll just write it this way. It's plus/minus one debt A. It depends on – what is it?

Student:[Inaudible].

Instructor (Stephen Boyd):Here? I am. You mean like right there? Wow, that's cool. And now, through the miracles of SSH, it'll be fixed in a few minutes. I lost Jacob though. Okay. Thank you.

Of course, I could argue that's – when you get certain levels of development, you get to a level where you're so cool that you can write a scaler plus a matrix. It's understood. The I is understood. I actually haven't achieved that level of development, but had a reached it, I could write stuff like that. Right. Okay. All right.

So this is actually just an interesting fact. I'm just going to say a little bit about it because it's fun, and it's just a little two minute aside, and it's just for – this is just for cultural background. This is cool stuff everyone should know about. Just make a short story about this.

So let's talk about, for fun – this is just a little aside, and it's just for fun. Let's talk about solving A X equals Y where A is square. Okay? Rule of it's two by two. You can do that by hand. The three by three, I don't know. Somebody might force you to do it by hand. Maybe you should do it once. Who knows? If you do it more than once, it's a complete waste of time, especially because the method is not used for anything. However, I suspect that many of you have been subjected to that more than once, which I think is the right number. Although, one could make a very good argument for zero, too.

All right. Then you grow a bit older, and you realize that the way you solve this – and we're just talking about calculating this. It's not a big deal. The way you do this, actually, is you use a computer. You might use MATLAB, but you must remember that, in fact, MATLAB is doing absolutely nothing but parsing this. It's passing it to a high quality, open-source software called LAPACK. I'm just telling you this because these are things you should know, okay?

The computational complexity is on the order of N-cubed. So it grows rapidly with N, and let's see. On a current machine, 1,000 variables, 1,000 equations can be solved in, easily, under a second. That makes some predictions. How about 100 variables, 100 equations. If I can do under a second and it's in cubed, how about 100 variables, 100 equations? You should know these numbers, actually, because they have serious significance. How fast can you solve 100 equations and 100 unknowns?

First of all, if you use your stupid grade-school method, otherwise known as the method taught in most linear algebra classes, by hand, that would take you a long time. You can laugh at that, but, you know, people actually did solve linear equations by hand. So for example, as part of the Manhattan project and things like that.

Okay. So how fast is this for 100 by 100? It's ten times smaller. Scale's like N-cubed. What's the factor? It's 1000 to one. So how fast can you solve 100 equations an 100 unknowns? Under a millisecond. Now, I know you're not impressed by anything, but you should sit and just think about that for a second. You want to be impressed by that, you can imagine how long it would take you to do – well, that's silly. That's stupid. That would be, like, have you ever done JPEG encoding by hand?

It's really hard. So it's kind of stupid, but there's very few people who have really thought through and realized the implications that you can solve 100 linear equations and 100 unknowns with total reliability in under a millisecond. So these are just amazing numbers, right?

I'll leave it there. No problem. So you go up to 1,000. Everything's fine. 2,000, 3,000. It's growing like N-cubed, so these things become macroscopic times. So for example, 10,000 by 10,000, assuming the memory is there, is going to be 1,000 seconds, so we're talking ten minutes or something like that. Right? Something – you know, 15, 20 minutes, okay? Something like that.

So these methods kind of – unless you're going to do this on some exotic cluster of machines and all that kind of stuff, these methods, for just a normal person, they kind of lose – they become inconvenient at around 2,000, 3,000. Okay? Then you move into another regime where you take into account the sparsity pattern of A. So the fact that A's got a lot of zeros in it. There's methods for that, where you avoid all the zeros and things like that.

These methods will get you to 50,000, 100,000 variables, things like that. I mean, quite reliably. Then you move into the really big problems, and that's problems where X has a dimension of a million or 10 million. The people who did this – these come up all the time, for example, if you do real medical imaging, as an example. Then X represents some density, or something like that, in voxels. So the numbers are just huge. They can be very, very big. Even just a little slice that's – if you have, whatever, 256 by 256, these are big numbers.

If you solve PDEs, you're basically solving equations like that. But for X, maybe 10 million or something like that. These base methods, where you basically factor A and all that, they're not going to work. In fact, you can work out what 10 million cubed is, multiply that in seconds, and you'll find out that this is not going to work. It doesn't matter because you don't have the storage anyway.

So how to you solve huge equations like this? It turns out for many of those huge equations, the one thing you do have is you have a fast method, given Z, to multiply A by Z. That might be through some kind of Fourier transform, Radon transform, something special in your problem, specialized code. It can be all sort of things, but you have a fast multiply. In other words, you don't – first of all, you can't even store a 10 million by 10 million matrix. So in fact, what you really have is you have a method or a function that evaluates matrix vector multiply.

Now, if I have a method, that's all I have is a method for matrix vector multiply. By the way, this is often associated with an inversion problem. This is a simulator. For example, let's do medical imaging. X is 10 million variables. That's some density and a whole bunch of – 10 million voxels. If I ask you, "What is A Z?" I'm actually saying, "Suppose the density were Z. What would my MRI or whatever it is, my PET or whatever it is, what would I measure?" That's what multiplying by A does. That's the forward simulator.

So basically what I'm saying is this. Imagine a situation where you have access to a forward simulator that's fast. It's not just stupid. It doesn't – you can't even store a 10 million by 10 million matrix. So it does it fast. If you do this, then given Z, I can actually calculate – if I can calculate A Z fast, I can call it again, and I can get A-squared Z. I can call my simulator three times, and I can get A-cubed Z.

That means, actually, I can make N calls to my simulator, N minus one, and I can calculate Z, A Z, up to A and minus one Z. By calling N minus one times my fast simulator and multiplying each time. Everybody agree? Okay. We're doing this for 10 million variables.

Now, you just calculated these. Let me show you something super cool. Look at that. If you knew or estimated the spectrum, you've actually just solved the equation. If you plugged in these coefficients here, you would actually have gotten – you would've ended up with A inverse Z. Everybody see what I'm saying? The point is, it's not a minor fact that the inverse of a matrix is a linear combination of I A up to A N minus one. It's not just – I mean, this has serious implications, one of which is kind of at the root of the ability to solve absolutely huge equations.

Of course, I haven't told you how you can calculate these. It turns out, you can actually calculate these using something called conjugant gradients, but that's beside the point. The big picture is this. Inverse of a matrix is a linear combination I A up to A N minus one. Those, you can actually – although you can store those matrixes if their 10 million by 10 million, you can actually get A to the eight times Z by doing eight forward simulations. There was a question.

Student:[Inaudible]

Instructor (Stephen Boyd):What's that? I'm evaluating what?

Student: So multiplying A by Z doesn't [inaudible]?

Instructor (Stephen Boyd):Doesn't save anything? What's that? Oh, how do I get A-squared Z?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, yeah, yeah. Exactly. So if I did this the stupid way, by storing A as a matrix and then actually doing a matrix vector multiply, right, which would be beautifully blocked out for me and optimized for my cache sizes and things like that, it would be fine. But the point is that you work out all the arithmetic and find out that the whole thing's over N-cubed. So you have to have a much faster method of multiplying A by the – in fact, the typical methods, instead of N-squared, might be N log N.

Less than N is unusual because you just have to write the answer. If someone – I have to give you Z, you have to read all the entries in Z, and you have to give me the answer back. So it's not going to be faster than N. If it's N-squared, you might as well just do it the old matrix vector multiply method. So all the action is in between, and it's usually N log N.

Student:I was going to say, you can store all of those vectors [inaudible].

Instructor (Stephen Boyd):No it wouldn't. Oh, sorry. You have a good point. Yeah, that's a good point. Yeah, so you'd add them up one by one. Right. That's a very good point. You wouldn't calculate all these vectors separately, in which case, you've just stored a million by a million. Of course, you couldn't do it. You're absolutely right.

So what you'd do is you'd actually evaluate the polynomial the right way where you'd accumulate. You'd multiply by one thing. What's that called, when you evaluate a polynomial that way? It's got some famous method. Luetin method. I don't know. Who knows? But thank you, that's a very good point.

Okay. All right. Well, that was just as aside, just for fun. Let's look at the proof of the Cayley-Hamilton Theorem, and you always start something like this by assuming – you warm up by assuming A is diagonalizable. So we say, we assume A is diagonalizable, we have T inverse eight. T is lambda. Characteristic polynomials in terms of the eigenvalues is just this thing worked out here. Then we work out pi of A, and that's going to be pi T lambda T inverse.

Now, when you take a polynomial of a similarity transform, you can pull the T out. You've seen that a couple of times. This is pi of lambda. That's a diagonal matrix. So we just have to show the pi of lambda, zero. But this thing is a diagonal matrix. Each entry looks like this. It's lambda I, minus – lambda one I is the first one.

So each of these – not one of these matrixes is zero, but each of them, the I one of these matrixes has a zero in the I I position. So when you multiply out all these diagonal matrixes, that's the same as multiplying – for example, the Ith entry is you multiply all the Ith entries. One of the them is zero. In fact, the Ith one. That means the whole thing is zero. So it was that simple.

Now for the general case, what you have to show, we've worked out – that's the Jordan Form. What you have to show now is that the characteristic polynomial evaluated at the Ith Jordan Block is zero. So that is this. Here, you have to multiply J I minus lambda one to the N I.

This is the first term. Actually, I'm not going to care about any of these matrixes, because this is the one that's going to do the trick, the Ith block. The Ith block here, when I subtract lambda I I, the zeros go away. I get these upper triangular matrix, which you should recognize as an – that's a downshift. I should remember these someday, but anyway, let me just do a quick experiment here.

Or is it an upshift? It's an upshift. Thank you. It's an upshift. I knew that. I wasn't confused. All right. So that's an upshift matrix, and you raise it to the N I power, which is the size. If you call upshift N times on an N by N matrix, there's nothing left – sorry, on a vector, there's nothing left. So this is actually the zero matrix here. So it works. It's just a little more complicated, but it works.

Okay. So that finishes up the Jordan Canonical Form. I have to say, I wish it were an issue – I wish it were something that was only math and didn't have any real consequences. Unfortunately, I can't tell you that's the case. So it does have consequences. In fact, systems with nontrivial Jordan Block occur all the time.

Maybe the most famous one – we'll get to it, but the most ubiquitous one, it's probably each of you right now. Probably, in fact, almost certainly, each of you has a piece of electronics with you that involves a nontrivial Jordan Block. So if anything you have with you, which would be an iPod, computer, anything, cell phone, there's going to be an FIR filter in there somewhere for something. Certainly for the audio and probably for other things as well.

If you have an FIR filter, I got news for you. That's a Jordan Block. Actually, for the same reason that it's an upshift. If you look at what an FIR filter is, it's nothing – well, I'm getting ahead of myself. It's something that looks like this, right? It's a bunch of delayers, like this, with an input, and then a bunch of coefficients multiplied like this. So a bunch of coefficients, and then these things are all added up. That's the output. There you go.

If you work out what the A matrix is for this, and X two plus one is A, X of T. A is the upshift matrix. So that means you're carrying, on your person, a nontrivial Jordan Block. Not that it matters, but, you know. Actually, it's good to know about these things because it means that there are things that hold for diagonalizable matrixes that aren't true for systems with Jordan Blocks. If you imagine that those things are true, you could easily get into serious and actual trouble.

I don't mean actual trouble in the sense that you'd have a mathematical misconception. Many mathematical misconceptions, actually, are harmless, except when you're in a math class. These would not be harmless because they would actually have real implications. You'd make assumptions about things, make predictions about how things would work or not work, and you would just be wrong.

Okay. Well, let's move on to the next section of the class which is, we're going to look at linear dynamical systems with inputs and outputs. So we'll look at this. It's not too much different. Some things are kind of interesting here.

So here, we're going to bring in this term, and that's the output – we didn't even have that before. I mean, we could have, but no reason to worry about it. So you have A X plus B U. So here, you've distinguished these two terms. People would call this, actually, the – it depends on the field. You would call that the drift term, would be A X, and then BU would be called the input term. The drift term, of course, that makes perfect sense. It's sort of what would happen if you were zero. So that's what the drift term is. There's other names for that. I can't remember them right now.

Okay. The picture is something like this. Up until now, the picture was this. You'd have a face plane, and you'd draw the state here. If there is no input, you simply calculate A times X. You get X-dot, which is actually – it's a vector, telling you where X is going instantaneously. You would draw that, rooted in X, here. Now you can see the direction X is going in and the magnitude that vector tells you how fast it's going. That's just X. The point is that what makes it interesting is that X then moves a little bit, and where it goes, the drift term, changes a little bit. So it's now undergoing some curve. Actually, you know exactly what it does now. It's sines and cosines, invariant planes and all that stuff. So you know how that works.

Now what happens is this. We have inputs, and inputs allow you to change the velocity vector in a very simple way. Here's a real simple example. Here's X-dot, if you have U equals zero, here. That's A X of T would be this velocity vector here. If U is one, so this is B, you actually go in this direction. If U is minus 1.5, you go here. Okay? So for example, if I – now you can actually visualize, at least on a vector field, the affect of messing with U. So imagine U as a joystick.

Suppose U, for example, were 10,000. What would be the velocity – what would be the direction of X in this case? It's pretty much aligned with B at that point. So in fact, you would say it this way. When U is 10,000, the BU term has completely overwhelmed the natural – you know, the natural dynamics or the drift term. Basically, X is now going in the direction B, and with a very big velocity. Okay? So that's the picture.

In fact, you can imagine – you know, you imagine your choice of U basically says that your velocity vector can lie – if it's rooted here, it can lie along this line here, and that's your choice in U. Okay. So let's get some interpretation of these things. If you write this out column-wise, you get X-dot is AX plus B one, U one down to – I guess it's M inputs and P outputs. I think these are converging on standards – I mean, they're just, of course, conventions.

These are one of the [inaudible] use of M, and what you can think of this is this. It says that the state derivative, that's X-dot, it's an autonomous or drift term, and you get one term per input. So you get this. So each input gives you, essentially, another degree of freedom for X-dot, assuming the columns of B are independent, which is often the case. Not always, but often the case.

So assuming the columns are independent, it gives you another degree of freedom for X-dot. So that's the picture. You can also write it row-wise. So you can say that XI-dot is – that's the drift term. It basically says it's an inner product of your input vector with – now that's a row of the B matrix, and that tells you that. So, for example, if you see that a – what would it mean if the third column of B is huge for that system? It has a meaning. I don't want details, just gross meaning. If the third column of a matrix B is huge, what does it mean about it?

Student:[Inaudible].

Instructor (Stephen Boyd):It says what?

Student:[Inaudible].

Instructor (Stephen Boyd):Exactly. So the third input, that says that system is extremely sensitive to the third input. The gain from the third input or whatever is high or something like that. It says that a small value of U three – that's the third input – is going to cause a huge deviation of X-dot from its drift direction toward the actual direction.

What if the third column of B is huge. What does that mean? It's got a meaning, too. Suppose it's way bigger than all the others. That says something. What does it say?

Student:[Inaudible].

Instructor (Stephen Boyd):It says what?

Student:[Inaudible].

Instructor (Stephen Boyd):No, I'm talking about the third row of B. The third row, sorry. The third row of B is huge, it basically says this. It says that all of the inputs have a tremendous affect on X three-dot on the third component. Now, X three, by the way, can couple back in and have an affect on X one, X two and X four an all those. No problem, but the immediate affect on where the thing is going mostly affects X three.

All right. A block diagram of the system is this. Looks like that. Your input comes in. This is called the feed-through term because I guess it just feeds right around here. That's the feed-through term. B converts the input to basically X-dot terms. X-dot has two components. That's X-dot, is the input to a bank of integrators. It's A X. That's the autonomous system there, plus B U. Then exit phi comes out, gets multiplied by C to form the component of the output due to the state. That's added into D U. That's the feed-through term.

So everything here has the obvious interpretation. So for example – a lot of these we've seen. For example, if I told you that C two five is huge, something like that. What does it mean? It means that the second output, Y two, is mostly dependent on X of five. Okay? You can go through it and make all the interpretations, but I think they're kind of obvious.

This block diagram is interesting when there's structure. So here would be an example. Suppose A is block upper triangular, and B has this – is also, in a conformal way, the bottom half is zero. You get something like that. If you draw the block diagram of this out, you get this. You get U coming it. It multiplies B one. I should say, that's the autonomous system. We've already seen that picture, actually, and it's kind of interesting. We interpreted this as saying that X two affects X one. But X one does not affect X two because there's no arrow going down.

Now when you – note that U also does not affect X two. So the real system looks like this, and you can see a lot of things here. You can see, for example, X two is not affected by U. This is actually fairly important because it says – if someone says, "Could you please take the state to zero," or, "Take it to this desired state," then you'd say, "I can't." They'd say, "Why?" You'd say, "Because with my input, I can't affect X two." They'd say, "I can't believe it. We paid so much for those actuators, and you still can't do it. We're going to find somebody." Anyway. You get the idea.

Okay. Now we'll look at the analogue of transfer function. I'm actually just curious how many people – I guess you've read, EE, you can't possibly avoid this, right? I don't know about other people in other departments. How many people have actually seen transfer function? Is that everyone? Come on. You're just holding back. You haven't heard of transfer function before? That's cool. What departments are you from? What are you in?

Student:[Inaudible].

Instructor (Stephen Boyd):CS. Okay, that's consistent with things I know. And your department?

Student:CS.

Student:CS. Um hm. A pattern is emerging. Okay. That's fine. No problem. It's not that big of deal. Actually, it was a very big deal last century. It's going to be much less of a deal this century. Of the things – if you're going to not know about something, it should be something whose D importance D T is negative. You made a good choice. Yeah. So all right. Good.

All right. So let's take the Laplace Transform. By the way, that statement holds for the Laplace Transform, too. Let's take the Laplace Transform, X [inaudible] equals A X plus B U. You get this. We did this before, but with this. We have the Laplace transform now of the input here. That's BU of S. We just solve for this. We get X of S. There's our friend, the resolvant. It's X of zero. That's the term we saw before, and if you take the inverse Laplace transform that, you get the matrix exponential.

Then here, I have something interesting. I have a product. Actually, I have a product of two things. I know what the inverse Laplace transform of this is. I know what the inverse Laplace transform of that is. That's U. Little U of T. Now, this is the product, and the inverse Laplace transform of a product is the convolution of the two.

So you get X of T is E to the T A X of zero, plus this interval, zero to T, E to the T minus tau, BU of tau, D tau. So that's the solution.

By the way, if you don't know about Laplace transform, it's not that big a deal. You could also get this formula directly, I think. They did, I believe, in the 18th century or maybe the very early 19th century using something called integrating factors or who knows. That's the solution. So I think it's that simple. You can also just check it by differentiating.

All right. We are going to interpret this. That's important, even if you're in CS, actually. It's very important. Erecting this stupid barrier, learning about Laplace transforms and transfer functions and stuff is a pity because this one is actually really important for everyone to understand, so let's look at it.

So it says that the state has two terms. One is this term. We're very familiar with that term. That's basically what happens if there's no input. This term is really interesting. It's a convolution, and it's a convolution of the input with something over here. The function, E to the T A B, that's called the input to state impulse matrix. We'll see why in a minute.

SI minus A inverse B, the resolvant times B, that's called the input to state transfer matrix because – we'll see why in a minute. Well, I can tell you why right now. If the initial state were zero, you could see that the Laplace transform of the state is the Laplace transform from the input times this thing. So this is called the transfer matrix, from input to state. Okay.

Now we'll plug in the readout equation. You get Y of S is C, SI minus A inverse, X of zero, and then you can rewrite the whole thing. This is quite familiar to us, in the time domain. That's this thing. Basically, that's the component of the output due to the part of X that's autonomous or something. The zero input X. I mean, every field has a different name for that. Yeah.

Student:[Inaudible].

Instructor (Stephen Boyd):This thing? Oh, no. I call that – that's the convolution. I was going to say I call, but not only do I call it, but everybody would call that the convolution. What made you suspect the legitimacy of this convolution? You said it was almost – because what?

Student:[Inaudible].

Instructor (Stephen Boyd):The [inaudible] the integral. Ah huh. Okay. Well, so my response – I mean, I agree with you. Often, you see a convolution where the integral goes from minus infinity, plus infinity, something like that indeed. However, here, we'd be in some – we're actually – I guess we wouldn't actually. Yeah, you could write it that way. The zero to T is only because you start from here, from X of zero. So I'll get to that later.

But this is the convolution. This is what – if you walk up to someone and say, "What's the convolution?" Actually, they'd give you two things. The minus infinity, plus infinity and this one. They coincide if you do things like agree that U is zero for negative T, for example. Question?

Student:[Inaudible].

Instructor (Stephen Boyd):Can you –

Student:[Inaudible]

Instructor (Stephen Boyd):Yes.

Student:[Inaudible].

Instructor (Stephen Boyd):Right. That is correct. That's right. Right. Now convolution in the scaler case is communicative. Right. Absolutely. In the matrix case, it's not, but it inherited that just because these are matrixes, so that's a very good point. But we'll get to these. Okay.

All right. So the output looks like this. It's got this term, and then it's got this term here, which is a function, which comes up a lot. C S I minus A inverse B plus D times U of S. In a time domain, it looks like that, here. You get all sorts of things. This function up here, this matrix, is called the transfer matrix or transfer function of the system.

This thing, here, well, if I evaluate it with T, so C E to the T A B plus D times the delta function at zero is called the – that's called the impulse matrix or the impulse response of the system. We're going to find out why soon.

Okay. So the simplest thing is this. If you have no initial condition, than this term goes away, and I have Y is H U. And it looks like that. By the way, this is – I don't want to make fun of it. In a time domain, you'd write Y equals H star U. This is, by the way – if anyone said, why on earth do you Laplace transforms and all that kind of stuff, this is why you do it. You do it because you have some huge structure or something like that with 40 variables. You have six inputs and things like that. Fantastically complicated. Each input affects this thing dynamically in some different way.

They all couple together. It's complicated enough, even if there are no inputs. You throw in the inputs, it's complicated. Very, very complicated. That doesn't mean we won't understand it, you can't write three-line programs that actually work out – write small snippets of code that actually do serious things, work out good inputs and all that kind of stuff. Bottom line, though, is that it's really complicated. However, when you take Laplace transforms and the input is zero, you get down to this beautiful formula. It actually goes back, oh, basically way into – because basically it said, "It's just multiplication." But you're multiplying complicated objects, Laplace transforms, so that's what it comes down to.

So this is kind of why you do this. Actually, it's interesting, this religious arguments we have, not real ones, but they are, actually. It turns out, in ISL, let's say, there were several faculty members who lines up and said that the Fourier transform and the Laplace transform are absolutely fundamental. They're fundamental. That's the real thing. Then others of us – I'm on the other side – said, "No, no, no. Convolution is the real thing." If we were smarter, if we had the ability to look at this, and actually, after a while you can get good at this. To look at it, then you can look at it and say – look at a convolution and go, "Yeah, that's gonna give a good ride," or, "I'm not getting in," or something like that.

Then you wouldn't need this, but they're just different representations of the same thing. Anyway. All right. Now here, HIJ, if H is this transform matrix, that C S I minus A inverse B plus D. If you take IJ, that's the transfer function from input UJ to output YI in the absolute undergraduate EE. But also ME and some other areas who've learned about transfer functions. That's the transfer function. From the Jth input to Ith output. Okay.

Let's look at this impulse matrix. That's this thing. Here's what it is. It's very interesting. It says that when the initial state is zero, the output – they have a beautiful number of – I mean, how many Askey characters – five Askey characters represents the input/output behavior of any linear dynamical system. It's just Y equals H [inaudible] with U. It's an integral, but that's it.

By the way, this has got everything in it. You've got the coupling from different inputs to different outputs. You've got coupling across time. That's what convolution is. So it's all there, five Askey characters, and it's basically this. It's a beautiful equation.

By the way, this is an equation everybody should understand, even if – whatever you think of Laplace transforms and transfer functions and everything, this one is unbelievably important. Actually, we should go over it and understand absolutely everything about it.

All right. So let me see if I can get this right. All right. Here it says – let's actually do scaler because we do have some people who maybe haven't seen convolution before. So we're going to do scaler first. This is review for everybody else, and I'm going to write is zero to T, H – I'm just trying to make it look like down here.

T minus tau, U of tau, D tau. So you want your scalers, H – everybody's a scaler. So what this says, I'll give my interpretation. For example, when I used to teach 102 – EE 102. So the interpretation is this. It says that the current output is actually an integral, but let's call that something like a linear combination. It's a mixture of the input in the past because I'm only going to refer to U of tau here, between zero and the current time. So it's a mixture.

The question is, "How much do you throw into the mixture to form the current output?" The answer is you multiply it by weight, which is H of T minus tau. So in fact, if T is seconds, tau here should be seconds ago. That's what they are. That's the actually formal unit of tau here, is seconds ago. To see if this makes any sense, I could ask you the following.

Suppose that H of T is – let's say that H of seven is really big and positive. What does it mean? What if H looks like this? It goes like this. What does that mean when you have a convolution? What does a convolution do?

Student:[Inaudible].

Instructor (Stephen Boyd):It says what?

Student:[Inaudible].

Instructor (Stephen Boyd):It basically says the following. It says that when you form the current output, this tells you how much the current output depends on what the input was. If this is tau, these should be seconds ago. So it means the following. It means that H – it does depend on what it was in the recent – in the last five seconds a little bit, but basically it depends a lot on what the input was seven seconds ago. Everybody got that?

Then it depends a little bit on what it was eight, nine, ten and less and less, but basically – so it says that basically in this case, the simplest thing is – this convolution integral is actually something like a seven-second delay. Everybody got this? Now, if H of T, for example, decays, it means that this system has fading memory. Why? Because if H of T – as T gets bigger, H gets smaller, the interpretation of H of 100 is how much the current output depends on what the input was 100 seconds ago. That's what it means.

If I make 100, 200 and 300, and it's getting smaller and smaller and smaller, it says the amount by which the – what happens, 100, 200, 300 seconds ago, affects the current output. It's getting smaller and smaller. It's got fading memory. Okay? Something like that.

Actually, once you see this, you should just sort of think of this as everywhere you go. For example, let's let Y be – let's make Y river flow, and let's let U be rainfall in a region. Let's talk. What does H look like? What does it look like? I mean, just grossly, obviously. I just want a gross idea of what H might look like. What do you think? Go ahead. What?

Student:[Inaudible].

Instructor (Stephen Boyd):I don't know. I want – I don't want – just want no exponential, no math. I don't want to hear any math. You can use words like big, small, positive, negative, no more. Okay, what? It should decay?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay. Well, it might do something like this. It might be small at first, and that would represent immediate run off. That stuff that got into the river first, but I think, actually, it would take a while because the water comes down, it goes down through tributaries into creeks and things. Then it builds up. So it might actually – actually, the flow might depend quite a bit on what it was several days ago.

If you can believe this story or not, for the record, I have absolutely no idea how this works. I'm making it all entirely up, just in case you – but I have no idea. I'm just making it up, but it could be right. It might look something like this.

First of all, could it be negative? It's probably not negative because you're basically – if it's negative, it says, "Hey, that's kind of cool. The height of the river today actually is suppressed by additional rainfall 2.2 days ago." So I don't know. It might look something like this. It might be small, and it might kind of go up like this, and then eventually it would die down again when all the water's come out. Something like that. Does that look cool? So that's H.

If this has a – boy, does this have a meaning. So I'm just saying, this is important. You should just think everything is this way, actually. You don't look convinced. How about pharmacokinetics? You inject – U is the amount of a drug you inject or ingest or something like that. Then Y would be the – what its concentration is in the bloodstream. What do you think it's going to look like? Probably something like this, but we'll just leave it that way. It depends how it's administered and all that kind of stuff. Anyway, you get the same story.

Okay. Back to this. There we go. Now this is a full multi-input, multi-output system. Here's the whole thing. By the way, this would be kind of interesting. This would be, if we went back to rivers, the height at the river, so YI of T is the height of the river at some place, I, at time, T. UJ of T is the rainfall in some sector, J, at time T. My question is, how does it affect it? This gives you the whole answer. This is assuming a linear model, which might or might not be the case for that.

Here's a linear model. It says that the total contribution, you sum across J, and by the way, these are not just random indices. When you're summing across J, you're actually summing – if you write code for this, it shouldn't be called J. You're summing across inputs. You're summing across the input labels, right? So that's really what's happening. That's what this sum is. This is the [inaudible] across the input labels, and then here, this integral, you can think of it as a sum, if you like. Some kind of a generalized sum, and it's actually integrating H, I, J of T tells you how much output I depended on what input J was tau seconds ago. I think that kind of made sense.

We'll look at some example, and I think this will make more sense. Okay. We have the step matrix or step response. This is not fundamental, but this is traditional, so it's good to cover it. It builds on the last centuries' stuff. So the step matrix or step-response matrix is this. It's simply the running integral of the impulse response. It actually has a simple interpretation. It's quite interesting, actually. It's when – if you want to know what SIJ of T is, it basically says you apply the following input.

UJ is one, and all the other Us are zero. So it says, "Apply an input where the Jth input is one," and this – if you multiply that, that will give you – and you look at SIJ of T. You'll get the output I versus time, okay? So SIJ of T gets YI when U is EJ.

This, you can actually – if A is invertible, you can actually work out a formula for that in the time domain, which is this. This is easy enough to verify or do from Laplace transforms. Okay. So let's look at examples because that will sort of make it all clear. So here's an example. It's three masses stuck between two walls. There are springs in between each of the masses and the walls, and there's dash pots or some kind of dampening mechanism.

We're going to apply two inputs. One is a tension between mass one and two, and the other is a tension between mass two and three, okay? So these might be implemented by a solenoid or a Piezoelectric actuator. Something. It doesn't matter. So we're able to – U one positive means we're pulling the two masses together, these two. U one negative says we're pushing them apart. Those are forces.

We'll take all constants one. So these are kilogram, one newton per meter, one newton per meter per second. Okay? So that's the picture. We're going to take – X is going to be six dimensions. The state, it's going to be some positions from some offset of these three masses and their velocities. Then you can work out what X-dot is. It's just this. X-dot is this top row. It's easy. That's zero I. It says that the block top – that's Y one, Y two, Y three differentiated – is equal to the block bottom. That's what this top – this is zero I.

Then these are various things I threw in at the right places, and they could well be – they might – I think they're correct. That's what they are. Actually, I'm sure they're right. Then here, for example, let's just audit one column of B. So the first column says – when you see that, it says something really interesting. This column, of course, is how U one affects X-dot.

What it says is really interesting. It says it has no affect immediately on the position, and that's correct. If you apply a newton to something, you will not see any immediate affect on the position. What will happen is when you apply a newton to something, it acquires a velocity. Then because I acquires a velocity, it acquires a displacement. So this kind of makes sense.

Down here, it says its affect on the first mass is one, and minus one on the second. That's exactly right. Look at this. In the first mass, if I pull with a newton here, I'm pulling this to the right, which is positive displacement, and I'm pulling this one to the left. This tells you how much velocity is acquired per newton. The fact that it doesn't touch – there's a zero here. This zero is because this force is only between these two masses and not that one. Okay? So that's just – that's my audit of just a random column.

By the way, you should do that. Just, always, when you look at equations, just audit them. Make sure you understand what it means and all that stuff.

Eigenvalues of A turned out to be minus 1.71 plus minus J 0.71. Minus one, and then minus 0.29. So we know already, qualitatively, what the dynamics are going to look like. That is something that decays in about two seconds. Even though, technology, it's oscillatory, it's gone. So that's a bump. That's a thud is what that is. That's more like a bump. That's more like a thud. Okay?

There's also one that – this decays in four seconds or whatever. It's not even done one cycle. So even though it's oscillatory, it would be kind of like a bump is what it's going to be. This one – that's interesting. This one decays, oh, let's say in about five over 0.29 – 15. Is that about right? I think this takes 15 seconds. I can cheat. Yes. Yes. About 15 seconds for this to decay in this mode. Meanwhile, it's period would be – we can calculate that from this. The period is going to be – if it's 0.71, whatever – that's 1.4 to ten. Does that sound about right?

Remember if I'm wrong, it reflects on you because there are a lot more of you than there are of me. What do you think? Ten? 15 second decay, ten second period. That's also, by the way, not an impressive oscillation, right? It means it's going to get one and a half oscillations in before it's gone. I think I did that right. We'll find out.

All right. So the impulse matrix from U one is this. You can organize impulse matrixes, by the way, lots of different ways. You can plot them different ways, and they're kind of interesting. You are, after all – in an impulse matrix, you are plotting something that depends on three things. It depends on I, that's the output index. It depends on J, that's the input index. And it depends on T, which is the time-lag variable, or it's tau in seconds ago or T in seconds ago.

So here we've plotted that, and this shows you the impulse response from U one. So let's even think about what the impulse response would be. The impulse response would be this. I would take – I would grab hold of mass one and mass two. I'd grab them, and I would apply an extremely large tug, tugging them together, but very briefly. That's what would happen.

Now, actually, you can integrate the dynamics of this in your head. You should be able to, right? It's very simple. If you pull these two masses and let go real quick, here's what would happen. Of course, they would acquire a velocity, and they'd start moving toward each other. But when they move toward each other, first of all, this spring is pushing them apart. There's damping from here. This spring is pulling this guy back to the wall. This one is pulling – is extending that spring. It's pulling this guy.

Then what happens is this guy starts moving to the left, and then these things reach some zero velocity part. Anyway, then the whole thing oscillates, and what do you think? I'm just saying – look, it's not a pretty picture, but this is how you integrate things in your head. So this is the right way. Anyway. So the thing oscillates for a while, and then the damping mechanisms remove a bunch of energy and stuff like that. Let's just see if this is sort of consistent with what happens.

This says that when you tug on these, the first mass rapidly moves to the right. This moves to the right. The second mass, that's this guy, rapidly moves to the left and, in fact, it looks like it moves a little bit more. So I'm just making that up, but it looks to me like a little tiny bit more. Then you have to go back here and explain why, when you tug these things, does this mass actually only move out – this one moves a little bit father to the left.

This is just for fun. This is just explanation, but if someone said, "Can you explain it," how would you explain it? What do you think?

Student:[Inaudible].

Instructor (Stephen Boyd):There you go. So this one is tied to a hard wall, so it's being restrained by a hard wall. This one, in fact, is being restrained by a mass connected through a spring to a wall, and this mass actually starts moving to the left, too. So the whole system is looser here, there's less pulling it back and so on. So that's why it goes a bit far.

By the way, I think I mentioned this before, when you do things like this, you have to be able to give an explanation like this. They don't have to be correct. If your story-telling skills are good enough, people will go for them. It's fine. You have to – I mean, first you just learn to do this, but then you have to be able to do the following. To get a story for this, and then have someone say, "Oh, by the way, it's wrong, and actually, these are switched." So the thing that you just gave a two-minute intuitive explanation for why one thing goes farther than the other, turns out it's wrong.

Then you have to actually not be embarrassed and go, "Right. I can explain that, too." That's what you have to do. So this is very important. It's not a formal part of the class, although it is possible, maybe we should have a story-telling portion of the final, just for fun. I mean, just a brief one. Maybe the right way – I mean, to really know if you're a master, you'd have to give a good story both ways. So we'd give you some phenomenon. We won't tell you which is which, and you have to give a convincing story.

I think that should probably be submitted in, in video to us because it's not something you write on a piece of paper. Then we would – I'd get a panel, and we would evaluate each one. Would we go for it or not. Probably, both of the things we'd ask you to explain would be false, actually, because that's the real – anybody can explain something that's true. That's super easy, right? But really, I think the way you know if you got this – you should start.

By the way, start by explaining things that are true. When you've mastered that, you can move on to the more advanced topics. We'll get into this later on various occasions. Okay. This shows the response from you, too. There's a symmetry here which is kind of obvious.

Now we'll look at our second example. It's an interconnect circuit. So it's quite real, quite important. It wouldn't be this simple, but it would look something like this. You have a tree of resistors and a capacitor at each node to ground, and I have a voltage source here. That's going to be our input, so I have a scaler input system.

I'm interested in the voltage at every, single node, like this. So I'm interested in voltage. This is quite useful if you wanted a picture for where this comes up. This could be a driving gate. You could include this resistor in the gate if you like and maybe some of that. This is a driving gate. The gate goes high, shifts from zero to one. Actually, it's just about right. You're lucky because VDD is one, right about now, on some processors.

So this goes – this flips from zero to one volt. T equals zero, and what happens here, of course, you can figure out. Again, if you have background in EE, you can integrate the dynamics of this in your head. That's easy to do. So for example, we'll make everything a one here. For EE, if this thing flips from zero to one volt, what is the voltage there right at T equals zero? What's the voltage here? These are all zero. It's zero, right?

What is it – that's one, one, one, one, one. What is it one millisecond later? My time scales are all off, but that's fine. What?

Student:[Inaudible]

Instructor (Stephen Boyd):It's still zero. That is actually, technically, a correct answer, but I said, no, I want the next level of accuracy from zero. All right, let me ask you this. Right at T equals zero, this thing flips up to one volt. That's zero. There's a volt across this resistor. It's one ohm. How much current is flowing?

One amp. So one amp is flowing into a one [inaudible] capacitor. What's the DVDT? One volt per second. Okay. So in one millisecond, the answer is the voltage here is one millisecond, about. It's not quite right because, actually, as the voltage here rises, the voltage – the charging current decreases slightly, you know. So that's fine.

The voltage here, though, it does nothing. It's not even – there's no current flowing here. Once the voltage here builds up, then you get current flowing across here, and that – by the way, that slows down the buildup here. This thing builds up, and then once the voltage here builds up, current starts spilling into this last one. So again, very roughly – and by the way, this description I'm telling you is true. I will tell you when I'm lying maybe. We'll see, but for this one, this is actually – so we should expect this one to pop up, linearly at first, but go up. It should slow down for a little bit because more current is being sloughed off to charge these guys.

This should be slower, and then this should be the last one. We'll see, in fact, that's going to be the case. Let's look at that. That's the step response, and sure enough, here it is. I claim that you could even, accurately go down here and describe things like changes and slope here this way. So you could even say things like – in fact, if you zoomed way in here, if you zoomed way in right at the beginning, what would S one look like? If I zoomed in the plot right here – we'll just do this, then we'll quit.

If I zoomed in, I wanna know, what do the voltages of these nodes look like between zero and one millisecond. That's for ones everywhere. We need somebody in EE. We already discussed it. This one looks like a line going up at one volt per millisecond. It's linear. What do you think the voltage in here looks like?

Student:[Inaudible].

Instructor (Stephen Boyd):Zero. That's actually a very good – that's a valid answer, but I want the next level of accuracy. What's the voltage here and here look like?

Student:[Inaudible].

Instructor (Stephen Boyd):What does it look like? It's quadratic. It's quadratic here. It's quadratic because it's an RC circuit being charged by a ramp. This goes up quadratically. Now, of course, when T is small, T-squared is really small. That's why I accepted the answer when someone said zero. That was perfectly cool.

What do you think it looks like here? It's cubic, and that means it's very, very small. I think I said that correctly. You know what I can tell from this? You need – we need to do exercises on intuitive integration. You'll leave this class, you can type in XM times T times A, make plots, all sorts of stuff like that. But you need to do this.

The only real reason is so that you can make up stories. You can't just say, "No, that's what it does," because people don't like that. They want the human element. They want you to look at this and say, "No, no. Sorry. That's one. Switching time is that." But they want you to say, "Yeah, but my –" And you say, well, see, at first it was charging quickly, but then work here was being sloughed off, and of course these were growing quadratically. But then, you see, they caught up. That's what they want.

We'll work on that. I think maybe for the final, I think we're going to – we'll do something. Okay. We'll quit here.

[End of Audio]

Duration: 78 minutes

IntroToLinearDynamicalSystems-Lecture15

Instructor (Stephen Boyd):Some announcements. Actually, you can turn off all amplification in here. Thanks. And you can down to the pad. I'll make a couple of announcements. The first is today's lecture, we're gonna finish just a few minutes early because I have to dash over to give a talk at 11:00 a.m. in CISX. In fact, you're all welcome if you wanna come. I don't know why you would, but it's a talk on circuit design and optimization. That's in CISX auditorium. So remind me, actually, if we get – if I'm not out the door and walking towards CISX at 10:45 a.m., you can do something like wave your hands or anyway, that kind of thing.

As a result of that, I'll be canceling my office hours today. A little bit late notice, but I will, and I'm moving them to Thursday. That doesn't really work really well with like, for example, Homework 7, but I'll also be around sort of on and off in the afternoon today, so if there is something you really thought you needed to get me today for, you'd be able to find me some time in the afternoon probably. Or you can send an email.

One other announcement is I got one inquiry. I think it had to do with someone planning to go away or something like that over the next week, and they asked could I possibly be so cruel as to assign Homework 8 this Thursday. What do you think the answer is to that? I'm just sort of curious.

Anyway, I don't even have to answer that. Of course we're gonna have a Homework 8. That was never in question, so we will indeed assign a Homework 8 on Thursday and it'll be due like maybe the Tuesday after the Thanksgiving week, something like that.

I saw someone – are you okay?

Student:I'll live.

Instructor (Stephen Boyd):You'll live? Okay. Her head kind of listlessly fell backwards. That's okay. All right. Okay, any other questions about this? No? I guess the people coming in now will be shocked when I get up and leave at 10:40 a.m. Okay, any questions about last time's material? Otherwise, we'll finish and then, actually, later today we'll get on to the essentially final topic in the class. We're gonna spend a good deal of time on it, but it will in fact be the last actual topic, so.

All right. We're studying linear dynamical systems with inputs and outputs, so systems of the form x dot is Ax + Bu and y = Cx + Du, like this. The transform matrix, if you plug in s = 0, is, of course, you just plug in C(si) – a inverse b, and you plug in s = 0 and you get minus C(A inverse) B + D. It's a famous matrix. It's one you'll see arising often and this is called – I guess the more descriptive term for this would be the static gain matrix is what this would be.

But kind of a rather cool retro term to use is it's the DC Gain matrix. That's direct current, so this goes back to like, I don't know, 1910 or something like that, but it's kind of – I use this, but mostly just to irritate people and stuff like that because it's so retro. All right. And I'll say a little bit in a minute about what happens if A happens not – if A is not invertible.

So what this describes is it actually describes the system, what relates the inputs to the outputs under static conditions. That is exactly what this does, so if you have static conditions, that means u, y, and x are all constant. Then of course, you have x dot and in that case is zero; x is constant. See if zero = A + Bu, y = Cx + Du and if you eliminate x from these equations by solving for x = minus A inverse Bu here and you plug that in here, you get this. Okay? So this is assuming A is invertible here, so this is what it describes.

Now, if A is not invertible, what it says is that there are inputs for which you cannot solve this equation here for any u, so it says there actually are no – there are u's, or there can be u's, for which there is no static equilibrium. You don't have to worry about that, but that's the meaning of that.

Now, if the system is stable, this is the integral of the impulse response, and this just follows from the Laplace transform, that the integral of a function is – well, this is e to the –s(t), so h of s is integral either the –h times this, 0 to infinity. You plug in s = 0 and it's the integral.

Now, that of course, requires h of t, this integral, to make sense and that would be the case if h of t decays.

Okay, it's also the same as the limit as t goes to infinity of the step response matrix, and the step response matrix here is, of course, the integral from 0 to t, and that's again, follow just from that's what the definition of the integral is.

Okay, and if you wanna know sort of what does the static or DC transfer matrix tell you, it basically tells you this. It says if the input vector converges to some constant u infinity, so the inputs can wiggle around, they can do whatever they like, but if they converge to a constant value, then the output will also wiggle around, but it will converge to a constant value and that constant value is obtained by matrix multiplication h of 0.

So h of 0 is very important. It's the thing that tells you, roughly, how the input affects the output when all the transients and so on have settled out. And if you work out – we can work out some examples and they should make perfect sense.

For a mass spring system, the DC gain matrix – our mass spring system is right here. Our DC gain matrix, and let's think about what it's gonna do – it's gonna map the two tensions you apply here into the displacements of these three masses. So that's what it's gonna do. That's what the DC gain matrix is.

So, of course, it's a 3x2 matrix and it would tell you, for example, the first column is obtained by pulling one Newton on this tension, letting the whole thing come to equilibrium and then recording the offset of the three masses. And it's kind of obvious what happens if you pull a Newton here, this thing displaces to the right, this displaces to the left. This displaces to the left a bit less. This one probably a bit more to the left than this one goes to the right. I'm making that up and I will change my story if we look at the actual numbers now and it's different.

And let's see. Did I get my story right? Yes, I did get my story right, so this says that if you apply one Newton, the left mass – that's the first position – moves a quarter meter to the left. The next one moves minus a half; that's twice the displacement, and the other one moves minus a quarter. And then you get a similar thing over here. There's a symmetry.

And this you can work out. Well, it's horrible, but I don't recommend doing it by hand, but for example, if you took this thing and C and worked out C times si minus this thing inverse times that, you would indeed get this matrix.

Now for the circuit, the DC gain matrix is actually quite simple. Again, you can work it out, but let's see what it means. I have to find it somewhere. Here we go.

So the circuit, the DC gain matrix, has the following interpretation. It basically says apply one volt here, wait for all the transients to settle, and then look at the mapping from the voltage here to the voltage at all the nodes, and of course, it's one, right? Because if you put a volt here, if everything is static, there's no voltages changing anywhere, no current disflowing, this is an equi- potential here. So these all have to be the exact same voltage. So that's the DC transfer matrix. Actually, in this case, that's literally the DC transfer matrix from that input to those outputs, it's like that.

Now, these are silly cases. Obviously, if you have something more complicated, it's interesting immediately. I mean it tells you immediately how something works in static conditions.

We're gonna cover a couple more topics on systems with inputs and outputs. So the first is dyschronization with piecewise input, so here you have your system x dot A + Bu, y = Cx + Du and I put in a sequence I'm gonna call that use of D, D is for discreet or something like that. And what we'll do is, well, the input is going to be – the continuous time input, that's the input to this system here, is going to be equal to – I'm going to index into the sequence and that's going to be over the time interval from kH to k + 1H.

So by the way, many people call it – there's all sort of names for this. This is sometimes called zero order hold is one name you'll hear for it. It just means the input is piecewise constant. There's probably some other names. I'm just not thinking of them at the moment, but that's basically what it is.

Okay, and now, we'll define sequences, so x[D], x of D, that's for discreet, that's a sequence. So in other words, it takes as argument an integer and it's gonna be x at kH. So x is this function from R + Rm; x[D] is a sequence. That's a function from z + to RN.

That’s what x[D] is. And the system came by sampling here, so these would be – you can call this a perfect sampling or something like that.

So here you’ve referred to h as the sample interval. Actually, that’s for x and y. Up here, it would be very common to call it the update interval. So that would be very common to just use a different name for it. And in fact, it turns out that in many systems, the update and the sample intervals are different. We’ll get to that in a minute.

So this would be the update interval. So for example, you’d say, oh, the update frequency is 50 hertz. The sample frequency is 400 hertz. U hertz did the same, but that’s the idea.

Okay. All right, so you can just think of these as sample versions of x and y. Let’s work out how they’re related. Well, the x[D] of k + 1 is x of k + 1H, but we can get that because we can propagate forward. This is what the state was at kH. This propagates it forward h-seconds. So this tells you how the state would propagate, actually, if there were no input over that interval.

Then this convolution here tells you what the effect of the input is, but it’s only over that interval. Here u only refers to between kH and k + 1H. Now over that time period, this thing is a constant and therefore, comes out of the integral. These are matrices. I obviously can’t pull it out on the left. I have to pull it out on the right. It’s the only correct place to put it and it goes here. B is also a constant; that goes out, and so you get something like this. I shouldn’t say that. You don’t get something like this. You get this.

So the terms make perfect sense. This is basically what would – this is what the state would be if there were no input applied. This, that’s complicated, but this thing here, that’s a matrix here and it just multiplies what the input is, its constant value over that interval and it’s an update, and in fact, if you look at this closely, you can write this this way. It’s a discreet time dynamical system. It says x[D] k+1 is AD, x[D] + BD times uD, and everything works this way. D is the same, C is the same. These don’t change, but A and B are given by this. And some people call this the discreet time system, so that’s what this is.

This is simple enough. This integral, by the way, can be evaluated numerically, but in fact, there’s lots of cases where it can be analyzed or you can actually just give an explicit formula for it from this thing. And I’ll give a case like that in a minute.

So A is invertible, which, by the way, need not be the case. There are plenty of interesting cases where A is not invertible. You can express that integral as this, and this makes perfect sense if someone walks up to you on the street and A for example was a number, you would know that the integral, either the TA between zero and h is something like it’s one over A times e to the HA minus 1. So you would guess at a formula like this would hold. And the questions would be – now, that’s still just a guess. That’s how these overloadings should work. You should just figure out what would this be if they were scalars. Then you have to put things in the right place.

Actually, it doesn't tell you whether the A inverse should go on the left or the right here. That it doesn't tell you. That's the first thing. And then there's the bigger question as to whether or not it's true because arguing from analogy from a scale of formula will often give you the right answer, but in a large number of cases, it just will give you something that's completely wrong.

So here, I can tell you why you're okay, or actually, maybe someone can tell me why. What is it about the terms here that actually gives us – what makes this safe? By the way, safe means you should still go and check it, but what makes this safe? What do you imagine makes this safe?

Everything commutes here. You imagine this as a power series in A. Everything commutes. A commutes with A inverse, obviously, because A-A inverse is A inverse A is i. When everything commutes, that's actually kind of the safe time. That's exactly when your scalar formulas are gonna actually correctly – they're gonna lead you to correct results. So that's how that works.

And if you wanted to show this, it wouldn't be hard. You'd actually just write out the power series for e to the Tau-A, integrated term by term, look at the new power series you have and say, hey, what do you know. That's the power series for this. Now, you might not wanna do that because of the A-inverse there, but still. Well, anyway, that would be that. So you have this.

Now, and interestingly, we can actually now point out something quite interesting here is that stability is preserved by discretization. So if Eigen value of A, that's an x dot = A + Bu lambda N, the Eigen values of AD are e to the H lambda 1 up to e to the H lambda N. Did we put that on like some additional – I have a vague memory we made an additional homework problem on this or something. Spectral mapping theorem. Well, I have a vague memory of it. Maybe you have or – a few people nodded. You weren't just being polite, right? You actually saw Spectral Mapping Theorem in this course sometime? Okay, good. All right.

So here you would know that the Eigen values of AD, which are given by just the exponential like this, would be e to the H lambda 1 up to e to the H lambda N. And that's interesting because it turns out that if you have the real part of a complex number is less than 0, that's if, and only if, the magnitude of its exponential is less than 1. And the reason is very simple. It's the magnitude of e to the – well, here – u, if u is a complex number, is exactly equal to e to the real part of u like that. So that's almost by definition. So if this here has a negative real part, then the magnitude of this is less than one. Now, this is just the simplest form, but there's all sorts of horrible little things. They're mostly horrible in bookkeeping, but you could actually, in principle, work out all of them now. Here are the kind of horrible things that could happen. You could have offsets. You could say, well, you know, we measured the state at these times, which are offset. They'd give you some horrible timing diagram, and then they would say the whole thing operates at 50 hertz, but the state sample and the update are offset by 8 milliseconds, or something insane like this. It's not hard. You just plug in the right e to the 8 milliseconds A in the right places and you could work out what happens and all that sort of stuff. It's not fun, but it can be done.

Very, very common is multi-rate and so here you could actually have different inputs updated at different times, different outputs sampled at different intervals. They're usually multiples of a common interval, so that's extremely common. So, for example, a jet engine controller on an airplane might run at 400 hertz and then something wrapped around that, the update might be at only 100 hertz, but the sampling might be 50 hertz. Your nav update, your navigation update might be running at 10 hertz. Your radar altimeter might be running at 2 hertz and all this kind of stuff, and it's just a lot of horrible exponentials flying around and it's not fun, but somebody has to do it. And I claim, in principle, you could.

Okay, the next topic, which is dual system, I'm gonna actually skip over because I decided I wasn't – I'm feeling slightly pressed for time, but only just slightly. So don't worry. You'll know when I get panicked. I'm not going there. I control it by the number of espressos I have before coming here. I'm at two right now, but I can go up to four if I need to. We don't yet. Next topic, causality. So I'll just say a little bit about this. Now, one interpretation as you write the state is this. It's basically the state of time, T; the state propagated forward T seconds. That's what this term is. Plus, that's if you do nothing, so some people call this the zero input term, or I don't know, something like that. This thing is the zero initial state input. This is the effect of the input. Now notice that the state of time, T, depends on the initial state and the input only over the interval between zero and T, time. So you would say something like this. The current state and the current output depend only on the past input. Well, that's what causality is. Causality says that the things you do now affect the future, not the other way around. Now it's a bit strange because you look at x dot = Ax + Bu, y = Cx + Du and I don't see any asymmetry in time in these equations. I don't see anything that would make things running backward in time look any different than running forward in time. I don't see any asymmetry here. Actually, the asymmetry is right here, in fact. That's the asymmetry, so it's a subtle point. It's probably worthwhile to mention. So the asymmetry doesn't come from these equations. These equations are very happy. They run backwards in time. They run forwards in time. No problem. It's actually our assigning – our considering an initial value problem. That's a problem in which the initial state is fixed, which makes it appear causal to us. For example, suppose you fixed the final state, x of T. Well, then I can write another formula that looks like this. It says that the current state is equal to the final state, propagated backwards in time to the current value, plus, and then an integral that has to do only with the future. So this is an integral. And I mean you can just check this formula. It's absolutely correct.

So in this case, if you fix the final state, the system is, in fact, not only is it not causal, it's anti-causal, or something like that. I mean this had to be. So these are both related to the concept of state, which we, so far, have used only to mean x of T in x dot = Ax + Bu. But in fact, there's an idea that when you say state, it ties into a much bigger idea and let me say what that is. So first, let me just say what state is abstractly. How many people have seen the concept of state, like in a computer science course, some very abstract computer science course or something like that? Actually, [inaudible] theory where you have to know that, right? You have to know the state of a processor. So in fact, the state of a processor, now that's not abstract. That's quite not abstract. The state of a processor or something like that is very simple. It's all the values of the registers and all the stuff like that. It's basically whatever you would need to do in that processor so that, if you restored the state and ran it forward in time from there, its behavior would be absolutely identical. That's what the state is. So for example, when you call a function, you push the state on some stack or something, and when you finish calling the function, you pop the state and the idea now is, aside from side effects that happen during the function call, it should actually continue absolutely as if nothing happened. By the way, if you forget to set a register, if you forget to restore a register or something like that, you're in deep, deep trouble, and that's because you failed to restore the state. There's also an abstract idea of state. Abstract idea of state is something like this. It is this and this is worth thinking about and understanding. This is gonna be abstract, so you don't have to worry about it. It's just for fun. The state is what you need to know about a dynamical system at a given time so that, if you knew the inputs in the future, you could perfectly predict the future behavior. That's what it means. So other ways to think of it is that it is a sufficient statistic for what has happened in the past in order for the purposes of predicting the future. So that's what state is. Okay, so for example, if you have a model of how prices are dynamically changing, they depend on certain other things, like interest rates and things like that, you would say that the state in that process is sort of everything you need to know so that moving forward you can make prediction of these prices correctly, given the future inputs. Have a question?

Student: So this would work for like a time invariant system?

Instructor (Stephen Boyd):No. No, it works perfectly well. The question was does it work for time variances. It works perfectly well for – the same concept works for time variance systems. And by the way, discrete systems as well. So that's the idea.

So there's lots of ways to think about it. You'd say something like this. The future output depends only on the current state and future input. Or you could say the future output depends on the past input only through the current state. Or you could the state summarizes the effect of the past inputs of future. These are all ways to sort of say the same concept. And another way to say it is the state is actually the link or the bridge between the past and the future. So another way to say it, you say this in machine learning, you'd say something like this, that the past and the future are conditionally independent given this state. Again, if you don't know what I'm talking about, that's fine. These are vague ideas, but that's okay. It's a very useful one to know about.

Okay. Okay, so that's the concept of state and that's just to sort of point out what it means.

Another topic is change of coordinates. We've already done this for an autonomous system. For a system with inputs and outputs, a non-autonomous system, the same sort of thing happens, except you'll see that some things don't transform. So if we choose to change the state to x ~, we write x is tx ~, so x ~ are the co-efficients of the state in the ti expansion of the state. Then you've got x ~ dot is. You get your familiar term here. That's the T inverse AT, that's the similarity transform, times x ~ + T inverse Bu. That makes perfect sense because Bu was the effect on the state derivative, but in the original coordinates, and that's what it is in the new coordinates.

Now, the linear dynamical system then looks like this: x~dot is A~x~+B~u, y to C~x~+D~u and here A is the familiar similarity transform. B gets transformed by this T inverse. C gets transformed on the right and D doesn't change at all. But these make perfect sense, absolutely perfect sense. And the reason is this. You're only changing the state coordinates, so there'd be no reason that D would change because D maps input to outputs. That's how you do the state.

This is a read-out map and this basically says how do you map x~. The T here transforms for the new coordinates to the state coordinates to the old ones, and so on. Now, when you do this, the transfer function is the same. In fact, the input and output have not changed at all here, have not changed at all, and you just work out what happens here. If you form C~si minus A~B – by the way this means the impulse response, for example, is the same. So you can check that C e to the TAB – well, I guess you'd say plus delta of T times D, that's the impulse response. This is the same as if I put tildes all over these things. It will be identical because in fact, u and y haven't changed. So this will be the same.

Okay, so you get this, and again, you have to work this out. This is not immediately. You have to throw T's and T inverses and then the usual tricks with T and T inverses help, for example, i you write as T, T inverse, for example, or something like that. You pull things out, mess with a few inverses and things, then there's a lot of smoke, a lot of cancellations, and in the end, you get this. So it's the same.

Okay, this allows you to have ideas like standard forms for linear dynamical systems. So you can change coordinates to put A in various forms, like you can put diagonal real modal Jordan. I'll say a little bit about the use of this. It actually has a huge use, which I'll say about in a minute.

So here if you put it in certain forms, then you get very interesting block diagrams because, for example, if A is diagonalized, you get something like this, and that is the so-called diagonal form of a linear dynamic system. That's the diagonal form, just as an example.

Now, you might ask why would you want to change coordinates like this. Well, you might wanna change coordinates of a real system just to understand how it works. So for example, this might be a modal – this would be a modal expansion in the middle, and then if one of these numbers were really small, you'd say something like, well, the input doesn't really strongly drive the third mode. That would be it. Or you'd say something like you look at a number over here, it might be small, or if there's a scalar, or if you know this is a matrix, it's small. You'd say the output doesn't particularly see a large contribution from the second mode. That would be the types of things you'd say.

Now, there's another real use of changes of coordinates and this is real, and it is absolutely not – you do not change coordinates in the purpose I'm gonna describe now, to things like Jordan canonical form or diagonal or anything like that. It's this.

If this represents some sort of signal processing system that you need to implement, for example, this might represent your equalizer, let's say, in a communication system, and you have to implement it. This change of coordinates is your friend. It is a degree of freedom because basically it's a knob. It says down here, the point is I can choose any T I like and I will have implemented the same signal. It has the same input/output properties, so I can change coordinates anyway I like.

I could do this for lots of reasons. I could change coordinates to get an A, for example, that has a lot of zeros in it. That might be very useful because when I implement this, it means it's much simpler to implement, for example. And there's actually forms you would use.

You might also, if this was actually some kind of analogue implementation, you would actually change coordinates to make the resulting system one that was less sensitive to perturbations in the parameters, just for example. These would be the types of things you would do.

Okay, that was just for fun, but it's important to know that the change in coordinates actually has real use beyond merely sort of understanding how a system works, which is the most obvious application.

Okay. So we'll finish up with discrete time systems. They're pretty simple because there's not even any differential equation. They're just linear iterations. This is xT + 1 is Ax of T + Bu of T. The block diagram looks like this. This is sometimes called – you can call it a register or a memory bank or something like that. That's what it is. Or a single unit delayer or something like that; a single sample delayer. Well, one over z, z inverse would be a typical one.

That means that's the output sequence. What goes into it is actually the sequence advanced in time because if you're looking through a delayer at a signal, what went in is something that was advanced in time on unit, so that's what this is. That's the block diagram.

You'll notice it's exactly the same, except the z is replaced with s, so there's really nothing here.

Oh, I should – I guess I'll also mention if you know about digital circuits and things, you can also imagine that there's a latch so that this thing doesn't – like you do two-face clocking or something like this, or there's a small delay or something like that so that on each – this thing doesn't race around and come back. It's basically there's a one-sample delay there.

Now, here the analysis is well, I mean it could've been done on Day 1. It requires nothing more than the knowledge of like matrix multiplication and that's about it. So x of 1 is this; x of 2 is that. You multiply this out and you get that. And the pattern is very simple. The state is A to the Tx of 0. That's basically what A to the T for a discrete time system, that's the time propagator operator. That's what that is. That's time propagation, A to the T.

So this propagates for the initial state and this is actually discrete time convolution. It's nothing more. This is just discrete time convolution here. And you can write it this way: Cy is CA to the Tx of 0. That's the input contribution from the initial state, plus H convolved with u starts discrete time convolution, and the impulse response in this case is D at T = 0 and then C, A to the T minus 1, B for T positive. So that's the impulse response. It's D, CB, CAB, CA2B and so on.

Okay. Now, suppose that you have a sequence – and we'll just cover the z transform very quickly. There's not much here. It's mostly just a review. Suppose you have a sequence. We'll just make it matrix value to cover all cases right off the bat and we'll make it a sequence defined not as a double-sided sequence, but a sequence defined from 0, 1, 2 on these indexes. Then the z transform is simply this function here. And this maps, this makes sense, and depending on how violently w diverges. You have a divergence no faster than an exponential, then this is guaranteed to make sense for large enough complex number z because if z is large enough in magnitude, these things are falling off here quickly.

So you referred to this as the domain of w and as with Laplace transforms, it's not a big deal; you don't really need – all that can happen there is things can get complicated.

Now, if you define a new signal, which is time advanced like this, so it's the same as another signal, but it's what the signal will be one sample later. You work out what the z transform is. It's again, it's like baby algebra. You multiply this out. I'm gonna change T to T bar, whatever, T + 1, and I'll write it this way where the z comes out, and then you can write that as zw of z minus zw of 0. It looks like the same thing for the derivative formula for Laplace transform. That's the only difference right there, is that z. But otherwise, there's no – it doesn't matter.

Okay. Discrete time transfer function is simple. You just take the z transform. I mean there's a big difference here. We don't need this to get the solution. We already know the solution. We just worked it out by multiplying matrices, so this is, in a sense, not particularly needed. I just wanna show you how this works. And there actually are people who are more comfortable in the frequency domain; so just from there, depending on their cultural upbringing, personality type, things like that, there are just people more comfortable. That's fine. No problem. And so this is addressed to them.

By the way, if you're perfectly comfortable with matrix multiplication, then I think everything we did over here, this described exactly what happens in a discrete time system. But anyway, all right.

So you take the z transform here. You get the usual thing. This is the analog of sX of s minus x of 0, except we have this extra z in there. You get this and you solve for the z transform of the state to get this formula here. And the only difference is there's an extra z here and you get the Laplace transform of the output is gonna look like this. It's h of z, u of z. H of z is Czi minus A inverse B + D. That's the discrete time transfer function. Got a question?

Student: Yes, [inaudible] rotation of that extra z?

Instructor (Stephen Boyd):That's a good question. I think the answer is yes. I can defend its existence, at least this way. Or I can't argue for it, but I can tell you why you shouldn't be shocked to see it there. If you sample something faster and faster and faster, this is, basically, -- where is it? I lost it. Here it is. Okay, that one. This basically says you're off by one sample in calculating the effect of the output, the effect of the initial state, for example, on the output. That's kind of what this says. If you sample faster and faster, the z has no effect because you're sort of moving something close. It's a smaller and smaller time interval, so it's okay. It would be like saying, well, no problem. I'm just using x at the end of the interval as opposed to the beginning, and as the interval gets small, the effect goes away. So that's not an argument as to why it should be there. It's an argument as to why it shouldn't bother us that it's there. That is just an additional argument because the main argument, why it shouldn't bother you that it's there, is correct, which is a strong argument. Not always completely persuasive, but that's it. Okay, so this finishes up a section of the course and we're now gonna enter the last section of the course, and in fact, we're gonna do essentially one more topic and then the course is over. It's gonna take a lot. We're gonna do a lot of stuff on it. It's very useful. It's really cool stuff. It has to do with singular value decomposition. You may have heard this somewhere, somebody. Actually, how many people have heard about things like singular value decomps? That's very cool. Where did you hear about it?

Student: Linear algebra.

Instructor (Stephen Boyd):A linear algebra class, so it's gotten there. It's funny. It's only 100 years old. Traditionally, material hasn't gotten into – that was taught in the math department?

Student: No, actually, [inaudible], but under them.

Instructor (Stephen Boyd):Oh, taught in the math department?

Student: No, it's EE.

Instructor (Stephen Boyd):Oh, okay, sorry. It was taught in an EE department. Okay. So you know, I think actually, it's about time. It's been around for about 80 years now, so it's about time you might see its appearance in math, linear algebra courses. Did anyone here actually hear it in a linear algebra course taught in a math department? Cool. Where?

Student: At the University of Maryland.

Instructor (Stephen Boyd):Aha. Cool. So that was, by the way, one hand in a sea of – for those watching this later or something. Okay, all right, fine. So we'll look it up. How about in statistics? Anyone hear about this principle component analysis? There's a couple, okay. What's that?

Student: We used it in machine learning.

Instructor (Stephen Boyd):P – so Machine Learning, you know about it through PCA? But other than that, I guess people in like CS never heard of this. Okay. Cool. I'm trying to think of some other areas where it's used. Okay, all right.

So we'll do the last topic. We'll start in with some basic stuff. You know, it's actually quite basic and I'll explain that in just a minute. We're gonna look at first, the special case about the eigenvectors of symmetric matrices, what you can say, and then we'll get to quadratic forms. That's actually a bit different. Actually, it's the first time – so far, a matrix has always meant the same thing to you. This is actually be a different – it's gonna mean something different here. We'll see that soon. We'll look at some of the qualities, the idea of we're gonna overload inequalities to symmetric matrices. We're gonna overload norm to matrices. These are not gonna be simple overloadings. These are not – they're gonna be overloadings in the sense that some things you'd guess are true would be true, and a bunch of things you would guess are true are false. And these are not simple overloadings. They're not what you think they are and they're really useful. And this will culminate in the singular value decomposition or principle component analysis, depending on your background. Okay. Let's start with the Eigen values of symmetric matrices. So suppose you have a symmetric matrix, obviously it's gotta be square. And here's the first fact. The Eigen values of a symmetric matrix are real. Oh, by the way, whole groups of people, for example, if you do physics, depending on what kind of physics you do, what happens is all the matrices you see are real. By the way, they could be either symmetric – there's another one where it's self-adjoint is what you'd call it. And it means that all the Eigen values you'd ever encounter would be real.

Or, by the way, sometimes there's an I in front, in which case, all the Eigen values are purely imaginary or something like that. So if you're in one of these fields, what happens is, after a while, you get the idea that all matrices are symmetric or self-adjoint. Then you actually start imagining things, like all of these, and they lose – even people who have done this for years and stuff like that, they get very confused then when you go back to matrices that are non-symmetric. Or they've even completely suppressed it and forgotten it. Okay, but for you – I should say this – for symmetric matrices, they're very special things that obtain, in terms of the Eigen values, eigenvectors, all that. It's very useful to know. Just don't spend all your time off dealing with these. If you're one of the other types, make sure you know what happens when matrices are non-symmetric. But anyway, let's look at it. Let's see how this works. Suppose you have AV as lambda V. V is non-zero, so V is not eigenvector and V is complex here. Let's see what happens here. I'm gonna look at V conjugate transpose. By the way, that's an extremely common thing. People write that as VH or V* and I should mention, although it's – I don't know – anyway, in Matlab if you type V prime, like that, and it's complex, you will get this. You will get the conjugate transpose. So this is V conjugate transpose and if you're very slow with these arguments, because they look – just one little mistake and – in fact, arguments when you first look at them, they look like it can't be, like you missed something or whatever. They look magic. But let's take a look at it. We're gonna say V conjugate transpose AV as V conjugate transpose – I mean a parser as AV here. But AV we know is lambda V, so I'm gonna write this as – the lambda comes outside. I'm gonna write lambda V conjugate transpose V. V conjugate transpose V is the sum of the absolute values of the complex numbers lambda i because that's what, well, because A conjugate A is magnitude A2 for a complex number. Now we're gonna do the same calculation, but we're gonna do it a little bit differently. I'm gonna take this thing here and I'm gonna replace these two with that expression, and that's fair. And let's see why. Well, we can do the transpose first. This is the same as AV – well, you can do either one. This is V conjugate transpose A conjugate transpose. That's what this is. Now, V conjugate transpose, that's what I've got here. A conjugate transpose is A transpose because I'm assuming A is real. So I get that. And AV is lambda V. I plug that in here, but there's still the conjugate up top and that comes out as lambda bar times this. Now these two are equal, so that's equal to that. This is a positive number and they're equal; that's equal to that, and the only conclusion is lambda is lambda bar. You can go over this and look at it yourself to check that I am not pulling a fast one on you because if you first do this calculation, and I would assume no problem, I just lost a conjugate somewhere. But in fact, no. These are two valid derivations. That's equal to that and therefore, lambda is lambda bar. It's the same as lambda is real, same thing. Now, we get to a basic fact about symmetric matrices and it's important to understand the logic of it. It's quite subtle. It says this. There is a set of orthonormal eigenvectors of A. That's what it says. Now, in slang, you would say things like this. You might say the eigenvectors of A aren't orthonormal. In fact, that's how you'd say it informally. But that is wrong. Okay? This is the correct statement, so as with many other things, you might want to practice thinking and saying the correct statement for a while, and after a while, when you realize people are looking at you weirdly and they're like why would you sound like that, then when it's actually causing social troubles, then you switch. People start thinking you're a [inaudible]. And then you switch to the slang and the slang is the eigenvectors of a symmetric matrix are orthonormal. That's wrong in so many ways if you parse it. It's sad. We'll go over the ways in which that's wrong in a minute. So let's see what that says. It says I can choose eigenvectors Q1 through QN, which are eigenvectors of A with Eigen values lambda i, and the Qi transpose Qj are delta ij. That's the same as saying there's an orthogonal matrix Q for which Q inverse AQ, which is the same Q transpose AQ, is lambda. So another way to say it is you can diagonalize A with an orthogonal matrix if A is symmetric. That's the condition. Okay, now, that says you can express A as Q lambda Q inverse, but Q inverse is Q transpose. Now this I can write lots of ways, but here's one.

I can write this in a dyadic expansion here. This is the sum lambda i times Qi, Qi transpose. We're gonna look at this and it's a beautiful thing. These are end-by-end matrices. Some people call these outer products, so it's a sum. They're also called dyads. And so this is sometimes called a dyadic expansion of A because you write A as a linear combination of a bunch of matrices or dyadic expansion. Now we have seen that matrix before. It is projection onto the line that goes through Qi, so this is A. You express A as a sum of projections onto these – in fact, they're orthogonal projections, these matrices. And I think – I have another vague memory of a homework assignment problem or something like this. Maybe not. Okay. Some of my vague memories are, well, wrong or something. Okay. So these are projections, so there can be a lot of interpretations of what this means. Before we go on, though, let's talk about the slang statement. So here's the slang statement. This is what you would say. The eigenvectors of a symmetric matrix are orthonormal. There's your statement. This is among friends, casual get-together; this is what you would say. You're just fooling around doing research, no one's looking, this is what you'd say. Actually, you could even say this at a conference. There'd be some social cues, though. When I hear someone and people like me hear someone say this, we get a little bit on edge and we listen for a little while to figure out either they have no idea what they're talking about or they know exactly what they're talking about and they're choosing to use informal language. It's usually clear very quickly. Okay. This doesn't make any sense and any sense that you could assign to this is completely wrong. First of all, you can't talk about the eigenvectors of a matrix, even though everyone does, because it doesn't make any sense. There's zillions of eigenvectors. Take any eigenvector; multiply it by any non-zero number, that's an eigenvector. So you don't talk about the eigenvector because that doesn't – let's start with that. That doesn't make any sense. Okay. So if I have a matrix – here's one – i, that matrix is quite symmetric. What are its eigenvectors? So what are the eigenvectors of i?

Student: All the non-zero vectors.

Instructor (Stephen Boyd):All non-zero vectors are eigenvectors of i. So let's make it two by two and I could say, okay, here's my choice of eigenvectors for i: 1 1 and 1 2. There, that's my choice of eigenvectors. That's V1 and that's V2. It is now false that these eigenvectors are normalized. They don't have norm 1. That has norm square root 2, that has norm something else, squared 5. So that's false. They're not normalized and they're most certainly not orthogonal by any means. Okay?

And in fact, if someone said get me the eigenvectors of i, and i returned these two things, no one can complain. These are eigenvectors of i, period. Okay? So it's absolutely the case that i times this is 1 times this, i times that is 1 times that. So that's fine.

Okay. Now, there's actually a situation in which you can actually say when something is close to this, so let's forget the normal because that's silly. Can you say the eigenvectors of a symmetric matrix are orthogonal, and this case shows the answer's no because it's not true. Here's a matrix in symmetric. That's an eigenvector, that's an eigenvector; they're independent and they're by no means orthogonal. I think that's enough on critiquing this thing.

So the right semantics is, the right statement is you can choose the eigenvectors to be orthonormal, and that statement is shrewdly true for i because, for example, I could choose 1 0 and 0 1. For that matter, I could choose 1 1 divided by square of 2 and 1 minus 1 divided by square of 2, and that would also be an orthonormal set of eigenvectors for i. Okay.

Let's interpret this formula. This is A is Q lambda Q transpose. Now remember, Q transpose is Q inverse. So let's look at some formulas. Let's look at some interpretations. So the first is to look at three matrices separately and it says that if you wanna operate by A on a vector, here's what you do. The first thing you do is you multiply by Q transpose. So the first thing you do is the x comes in, you multiply Q transpose x and you get Q transpose.

And now we know what Q transpose x does. We know exactly what it does. That's essentially Q inverse. Q transpose x actually resolves x into the Qi coordinates. That's what it means. It resolves x. So this vector is the coordinates of x in the Qi expansion.

Now we multiply by symmetric matrix – I mean, sorry, a diagonal matrix. That's very simple. It's very simple to visualize what multiplying by a diagonal matrix is because you're actually just scaling each coordinate.

By the way, if it's a negative Eigen value, you're flipping. You're switching the orientation. Okay? That's here.

Now, when we multiply by Q, we know exactly what that is. That's actually reconstituting the output. So if you like, you can think of this as a coding, a scaling, and a reconstruction. Your question?

Student: Yeah, sorry, still don't have any [inaudible] orthogonal eigenvectors. Why is it that for a symmetric matrix you can find orthogonal eigenvectors, but if a matrix is not symmetric, you can't necessarily find orthogonal?

Instructor (Stephen Boyd):It's a great question. I haven't answered it yet. But I'm going to. I think I am. Might be Thursday, but I'm gonna get to it. So we're just gonna push that on the stack and we'll pop it later maybe.

So the question was why and I haven't said so. So first, we're just gonna look at – I said it as a fact and we're gonna look at what does it mean? What are the implications? Then we're gonna come back and see why it's true, but we'll see why in a minute. Now, by the way, I have shown why the Eigen values should be real. I have not shown that you can choose the eigenvectors to be orthonormal.

Oh, by the way, one implication of this, it says that a symmetric matrix, the Jordan form is really simple. It's always diagonal. You cannot have a non-trivial Jordan form for symmetric matrix. So we're gonna get to that later, I hope. I hope we are. I think we are.

Okay. Now, this is actually a very interesting interpretation. Oh, and by the way, it's worthwhile knowing this; this comes up all the time. This is, well, roughly, actually, this exact operation is carried out, for example, in the current standard for DSL. It's also done jpeg. So jpeg, you do a DCT transform on an 8 x 8 block of pixels. You don't have to know this. I'm just saying this is not those blog diagrams of abstract things. This type of thing is done all the time in all sorts of stuff all around you.

In jpeg, at least in one of the older standards, you take 8 x 8 blocks of pixels, you form a DCT transformation, and then, in fact, you don't multiply here. In fact, what you do is you quantize in the middle and then you transmit these and then it's reconstituted when you decode the image. Okay? So pictures like this actually are all around you, widely used, and so on. It comes up in signal processing, compression, communications. I mentioned one in communications. It comes up all over the place.

So that's the picture is you resolve into the Qi coordinates, you scale, flip of lambda is negative, and you reconstitute.

Now, geometrically, there's a beautiful interpretation because we know an interpretation of orthogonal matrix geometrically is it's an isometric mapping. So it's something that preserves lengths and it preserves angles, and it preserves distances.

Now, it can flip. For example, you can have a reflection through a plane and roughly speaking, you should think of these as rotations or reflections. So this basically says – I'm gonna call it a rotation even if it's a reflection – it says rotate the input by, for example, round some axis by 30 degrees. Scale in that new coordinate system and then it says undo it, and that means rotate around the same axis 30 degrees the other direction. So that's the idea.

We've already mentioned this. Oh, by the way, when you diagonally real scale a vector, there's lots of ways to say it. There's, well, I found both dilation and dilatation, so somehow there's two. I thought for a while dilation was the only correct one. No, it also turns out, it's also English to say dilatation and I tried to blame it on some weird people in some field. I couldn't identify the field that committed this crime. Or country of origin; I also tried to pin it maybe like on the British or something like that. That seemed like a good, promising – that would be the kind of thing, that extra syllable, just have that Britain there. But couldn't blame it on them. Couldn't chase it down.

So you'll see dilatation. There's also another thing you should – so you'll also see this. And by the way, on a couple of occasions, I have had students and people say that actually these mean different things and one or two of them tried to explain it to me. They seemed – the distinction seemed very clear to them, but it never sunk in with me. There may be. There may be a difference, but if there is, I for one haven't got it. That might be my – probably me. So there might even be a distinction. There could be some field where they say, no, no, no, totally different things.

This is where you multiply each component in the vector by lambda i. So this is the picture. Now this is decomposition like this, we've already talked about. This is a dyadic expansion. You can call it – oh, by the way, some people call this simply the spectral expansion. This is a spectral expansion of A, that's what this is. This thing over here.

And these are called projectors and sometimes they even – in fact, a very common way to see this would be this. This would be very familiar, but actually, in a lot of cases, there would be an infinity here. In a bunch of fields, this would be very, very – you'd see things like that and you'd also see the same thing with an integral and all sorts of stuff, but it would look just like that. And they'd call that the spectral expansion of the operator A, depending on what field you're in. Okay.

So let's look at – this is just a stupid example, but just to see that something happens here. So here's a silly matrix. You clearly don't need anything to figure out what this matrix does to a vector. But as usual, with the examples, the boys and I even do this for a 2x2 matrix. The boys even do this for a 30x30 matrix, or for that matter, 3,000 x 3,000 matrix, where it is by no means obvious what a 3,000 x 3,000 matrix does symmetric at all.

So here you work it out. The eigenvectors turn out to be 1 1, 1 minus 1 – oh, did you hear that? That was slang, big time slang. So let me wind back and say it again without slang. But then I'll stop after this lecture and I'll go back to slang. Okay, so I'll say it precisely.

For this matrix, I chose the eigenvectors 1 and 1 divided by square 2, 1 minus 1 divided by the square root of two, which are orthonormal. Actually, that involved a small slang because I shouldn't say the vectors are orthonormal. I should say I chose the set of two eigenvectors consisting of 1, first eigenvector, 1 1 divided by square of 2; the second eigenvector, 1 minus 1 divided by the square of 2; end of set. And that set of eigenvectors is orthonormal. There, that was formal.

That's why people don't talk this way and why, if you see people who do talk this way, it's weird. But you should think that way, so you should speak casually. But maybe you don't know when the right time is to, but you should never think casually. That's called sloppy or something. Okay. All right. Or you could do it; you should just do it in private. You shouldn't write it down or something. And not think that it's clear thinking.

So here's the picture. There's Q1 and Q2, and here's some x that comes in. Here's x and so the first thing you do is resolve it until you project it onto the Q1 line and the Q2 line. That's orthogonal projections; that's these and these. Then you scale these by the eigenvectors. I guess let's see what happens.

This first guy, nothing happens. And the second one gets multiplied by two and flipped in orientation, so this guy gets flipped over here and doubled, and so you get something like that, and then that's the output.

Now, you sure did not need to do a spectral decomposition to find out what this 2x2 matrix does to a vector, but it's just to show an example.

Okay, now we're gonna show what I said I was gonna show, except we're not gonna do the full case. We're gonna do the case where lambda i is distinct. In this case, when the Eigen values are distinct, there is a statement that can be made somewhere in between the nonsense slang and the correct formal statement. There's actually a true statement that lies in between them. So we're gonna do the case of lambda i distinct and in this case, I can say the following – all right, let me see if I can get it right.

All right, it's gonna come out right. Here we go. I have to concentrate. Here it goes. If the Eigen values are distinct for a symmetric matrix, then any set of eigenvectors is orthogonal. Independent eigenvectors – oh, I was close. I was close. All right, let me try it again. If a matrix is real and symmetric, then any set of N independent eigenvectors is orthogonal. Yes. It came out right. I think that's correct.

So that's a different statement from the original that I made. Just the quantifiers were in different places or something like that. Notice I didn't have – so in this case, you don't have to choose. There is no choice.

And the slang reading of that would be – in slang, you would say this, in abnormal speech you would say the eigenvectors of a real matrix with the state Eigen values are orthogonal. And then if you simply choose to normalize them, then you could say "and therefore could be normalized to be orthonormal" or something like that.

So AVi is lambda Vi. Norm Vi is 1. Well, let's see what's gonna happen here. So here, notice I didn't have to choose the Vs in any special way. I just chose them to be the eigenvectors period. I have asked you to normalize them. That seems entirely reasonable. The eigenvectors are non-zero. You can obviously divide them by their norm and get something which is still an eigenvector and has norm 1.

So we'll work out Vi transpose AVj, this thing, and we'll do that two ways. I'll first parse it, I'll associate it AVj. That's gonna leave you lambda j transpose Vj, but I'm also going to rewrite it this way as AVi transpose Vj. Now you have to check. AVi quantity transpose is Vi transpose A transpose, and that's why I use the fact that A = A transpose and then that's the same as this.

Here, I get lambda i. Now when you do things like this, you have to be very careful. It's just like the calculation we did before. It's probably that you've made a mistake, but you can check here. There is no mistake and you see the following. It's actually quite interesting. If i is not equal to j, you get this statement. Just by saying this is equal to that, you get this. Now there's only two possibilities. I mean if i equals j, this is a non-statement because it says 0 = 0. If i is not equal to j, this is a number which is non-zero. That's our assumption that the Eigen values are distinct. Therefore, that has to be zero and we're done.

So in this case, you can actually say the eigenvectors, with a small bit of slang, you can say the eigenvectors of a symmetric matrix with distinct Eigen values are orthogonal. And then therefore, can be normalized easily to be orthonormal, so it's something like that. There's a little bit of slang.

Okay, now the general case, the distinction is this. You have to say the eigenvectors can be chosen to be orthogonal. An example would be the identity matrix where, of course, you could choose an orthonormal set of eigenvectors, but if you're just weird and perverse, you could choose any set of independent vectors and say, what? Those are my eigenvectors. Don't like them; they're independent; what's your problem? So that's that. Okay.

So let's look at some examples of this. The first example is a RC circuit. So here I have a resistive circuit. I pulled the capacitors out on the left like this and the state of this dynamical system's gonna be the – I can take the voltage on the capacitors and I can take the charge just as well, and the dynamics of this is CVk. CkVk is the charge on a capacitor k. CkVk is the current flow and that's the current flow into the capacitor, which is the negative of the current flow into the resistive circuit. So that's the dynamics of that.

And I have i = gV; g is the conductance matrix of this circuit and its maps, the voltage appearing at these ports to the currents that flow into the ports. So it's a conductance matrix like that.

So this describes the dynamics of the system and you can write that this way. V dot is minus C inverse gV and so we have an autonomous dynamical system. There's A like that.

Now, by the way, g is symmetric, the resistive circuit. That's actually a basic principle from physics, basically says that if this is an arbitrarily complicated thing with the resistors and things like that, just your terminal resistors, then in fact, this g matrix is symmetric. And you get similar things in mechanics and other stuff, and economics, too.

Okay, however, this matrix is not symmetric. But we're gonna change coordinates to make it symmetric and to change coordinates to make it symmetric, we use a rather strange state. It's the square root of the capacitive times the voltage. Now that's kind of weird because that's a reasonable choice of state. That's voltage. That is an entirely reasonable choice of state, this measure in volts. That's the charge, to use the charge. This is in Coulombs; that's in volts, but, in fact, you're using something that's halfway in between because this is Ci to the 0Ci. You're using square root CiV. It doesn't look right. These are very – I forget, I mean the physical units of these – this is really quite bizarre. I guess it's volts times square root farads, which you could write out all sorts of other ways. Whatever it is, it's weird. It seems odd. When you change coordinates like this, this is scaling. You actually end up with x dot is minus C minus half, gC minus half, so this is actually now a symmetric matrix here. We can conclude that the Eigen values of actually this matrix, which, however, is similar to this matrix, are real. So we recover something you probably know from undergraduate days or intuition or something like that. An RC circuit has real Eigen values period, so you can't have oscillations. Okay? Cannot. So for example, in an interconnect circuit, if it's well modeled by resistance and capacitous, you cannot have oscillations period. Okay? You have to have inductance to have oscillations. Okay, that's that. I will say a little bit about this. I quit in a minute literally, and I'll tell you why in a second. I should say there is, if you really do wanna know why you would use coordinates like this, there is actually something interesting here and I'll tell you what it is. What is norm x2 for this thing? For this? Well, it's sum square root CiVi quantity2, which is what? It's sum CiVi2 and somebody in EE tell me what this is. It's the electro statically stored energy in the capacitors times 2. Thank you. I was just hanging on that last thing. That is twice the electro statically stored energy. So these coordinates might look sick and indeed, they are sick, I think. However, these are coordinates in which the norm now corresponds exactly to energy, so if someone said defend that, you could say, well, it's quite natural from the energy point of view. I have to quit a few minutes early today. I told people at the very beginning because I have to rush over and give a talk at CISX auditorium, so we'll quit here.

[End of Audio]

Duration: 72 minutes

IntroToLinearDynamicalSystems-Lecture16

Instructor (Stephen Boyd):That's it. I should make a couple of announcements. The first is homework eight – we actually worked out last night. I wanna say two things about homework eight is – the first is when you look at it, you'll be horrified. It looks long. It's not short. However, a lot of those are just, like, quick little things – just basic stuff you should know about, actually, the topic we're doing now. So – and also, we got a big timely request from a student who suggested that homework date should be due not Tuesday after the break, but in fact Thursday. And we decided to take his advice. So it'll be due Thursday after the break. That's two weeks, so it's actually kind of a light homework set for two weeks. Or at least that's the way we see it.

And then we'll have one more – we'll have a homework nine, as well. And then that's it. So – make sense? Any questions about last time? Otherwise, we'll just jump into it. Okay. Last time, we looked at an RC circuit as an example. And we ended up with – if we chose the coordinates in this really weird way – and it was kinda weird that there wasn't – there was – I mean because it has really strange units, right? It's square root of capacity – farads times voltages – times volts. Very strange units. But these units actually are natural because a norm squared is actually an – it's electrostatic stored energy, so that's – this is not completely bizarre. In these coordinates, the system – the A-matrix – is actually diagonal. Sorry, symmetric. So the A-matrix in these weird coordinates are – is symmetric. That means lots of things. It means that the eigen values are real. They're actually real and negative. That's another story, but they're real and negative. And the eigen vectors actually can be chosen to be orthogonal. So if you express them in the voltage coordinates, the eigen vectors are not orthogonal, but they are what people would call C-orthogonal. So if you put the capacitants matrix in here, you get this. And that's the same as saying that SI transpose SJ is zero if I is not equal to J. That's what this statement is. Okay. So that's just an example of a lot of other uses of this. But now we're gonna start up the idea of quadratic forms. So this is actually a big break in the class. Up until now, a matrix generally was associated with something – with some concept like this. And this could be something like a measurement setup – something that maps your actions to the results. And even, in fact, when you write it this way, it sorta has that flavor because here it maps the state to the state derivative. So it still has the flavor of a mapping from one thing to another. We're actually, now, gonna use – well, in some ways, sadly – the – well, in some ways it's good – the same data structure, which is, say, a matrix, to represent something completely different from a transformation or linear function. It's a quadratic form. So a quadratic form is something that looks like this, and it's supposed to just generalize the idea of the square of something. So that's what it's supposed to be. And as the form X transpose AX – and if you work out what that is, it's some AIJXIXJ. So it's a sum over all possible products of pairs of the components of a variable like that. So that's what it means – and then times AIJ. Now one thing you notice right away is something like this. Of course, X3X4 is equal to X4X3. So the coefficient A34 that goes here and the A43 that goes here – they sorta do the same thing. So we're gonna see that. But this is what a quadratic form is. And this means, actually, you have to relearn a lot of things. Like for example, when you see a matrix that represents a quadratic form, and you see that the two-three entry is large, it used to mean, in something like this, that the gain from X3 to Y2 is large. Now, it means something – in the context of a quadratic form, if A23 is large, it means something very different. It means, somehow, that variable two and variable three are strongly coupled. If they're both large, you'll get a very huge – you'll get a large contribution to a quadratic form. So that's what it means. Okay? So again, you'll have to sorta relearn – I mean, it's not a complete relearning, but you'll have to relearn what it means that way. Now, you might – you can just as well assume that A is A transpose. In other words, if you have A34 and A43, these are the two contributions from I equals three, J equals four and I equals four, J equals three. You can see that these numbers are the same. So I can pull them out and make it A34 plus A43. And I might as well replace both of those with the average of the two. It doesn't change anything. So in matrix language, you write it this way. You say that X transpose AX – and let's do a quick calculations this first. Let's take X transpose AX, and that is a scaler. That's a scaler. Let's transpose it. Well, I get X transpose – that's this one transposed. I get A transpose, and then I get X transpose transpose, otherwise known as X. What this shows is that A and A transpose give you the exact same quadratic form. Okay? So if you have two quadratic forms, they are the same if, for example, they are transposes of each other. Okay? Now inparticulus means the following: it says that if I have any quadratic form, I can write it this way. I can write it as a quadratic form with A plus A transpose over two. Now A plus A transpose over two – that's got a name. That's called the symmetric part of a matrix, and it's extremely easy to describe what it does to a matrix. If you form A plus A transpose over two, you replace each entry in the matrix with the average of itself and it's transpose – associated transpose element. So that's what it is. So you replace A34 with A34 plus A43 divided by two. Okay? Now I'll say something about this. It turns out now it's unique. It turns out if two quadratic forms are equal for all X, and if the matrices defining them are symmetric, then A is B. And that's a homework problem, but I don't mind going into that a little bit to explain what it means. And I can say a little bit about what this – and you have to be very careful understanding exactly what it means. What it means is this. It does not mean that every time you see a quadratic form it will be symmetric. That's actually not the case. You will encounter expressions of the form X transpose AX where A is not symmetric. Okay? And you have to be very, very careful because some formulas for symmetric forms are gonna assume A is symmetric. And they will be absolutely false when A is not symmetric. Okay? Now, politeness suggests – I mean, when you're messing around with quadratic forms, it is polite to pass to others other methods or functions that consume quadratic forms. You pass out symmetric matrices because otherwise, it creates confusion. By the way, another method is – another standard is you could also assume, for example, another canonical form is to assume that all matrices in using quadratic forms will be upper triangular. That's also very common. It's not that common, but it's another standard form. Okay? When – if you're working on a project with someone or something like that, and they produce a quadratic form, if they're – in my opinion, politeness requires that the A should be symmetric if there's not storage issues or something like that. They should produce a symmetric A. And if you want to write sort of bombproof code, the first thing you might do when you get a quadratic form is to symmetrize it. Just replace – if a matrix is already symmetric, symmetrizing it has no effect. But now you've made your code safe against – or your algorithm or whatever you wanna call it – it's now safe against somebody passing you a quadratic form that was not symmetrized. Okay? So – and let me just – I mean, let me just point out what I'm saying here. Here's a quadratic form. There. And let's actually – in fact, somebody tell me what is that quadratic form? Just write it out in terms of Xs. It's X1X2. It's nothing but the product of X1X2. That's what it – that's the quadratic form associated with that matrix. Okay? Here's another one. I mean, it doesn't really matter, but you can have three halves and minus one half zero zero. Okay? What's that? Have I done it right?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it? I think I did it right. What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):That's X1X2. Okay? Even though this looks different, right?

So canonical form – reason – the canonical form for this would be one half, one half. That would be a standard form.

By the way, there are cases where quadratic forms come up in – naturally, in non-symmetric ways. But then you have to be very careful because a lot of stuff we're gonna do is gonna depend on the matrix being symmetric. Okay?

So I think I've said more than enough about that. Let me ask you a couple of questions, here. Just – what is – if A is diagonal, what does that mean about the quadratic form?

You know what A – it means if A is a square matrix, and it's diagonal, and you have Y equals AX, we know what it means. It means YI depends only on XI. Yet somehow, there's no cross coupling from inputs to outputs. What does it mean in a quadratic form to say A is – A is diagonal?

Student:[Inaudible].

Instructor (Stephen Boyd):What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):Correct. It has only squares. So it's a sum – it's a weighted sum of squares. That's what a diagonal matrix in the context of a quadratic form means. It's a diagonal sum of squares. You don't have these cross-terms – these XIXJ terms where I is not equal to J. Okay? So that's kind of the idea.

Okay. Let me talk about – let's talk about this uniqueness, even though it's a homework problem. It's not hard, but it's actually interesting. First of all, let's make sure we understand the concept of uniqueness. Let's make A symmetric – I'm sorry, not symmetric – square. And let's take Y equals AX. And you remember from day one, day two of the class, we said the following: that if you have a mapping that's linear, it's given only by one matrix A. There's no way two different matrices can produce – if A is not equal to A tilde, then for sure AX and A tilde X differs for some X. That's how we did it.

And how did we show that? Well, it was kind of easy. We plugged in X equals EI, and that gave us the I column of A. Therefore, if two people have two matrices, which are different and yet had the same mapping – in other words, for any X, they both – AX is A tilde X, it can't be because you can just go column by column and check.

So let's just talk about this. Yes, we're doing one of your homework problems, but that's okay. And I'm not gonna do it all the way anyway. I'll do it part way. Well, who knows.

Okay. Let's say we have two matrices A and B, and for any X, X transpose AX is A transpose BX. Now you know right away that does not mean A equals B unless we assume they're symmetric. So we're gonna assume they're symmetric. Okay.

Give me some candidate Xs to throw in, please.

EI, great. So let's find out what is EI transpose AEI. Now remember, if you wanna contrast this to your other model of a matrix, AEI gives you the Ith column of A. That's what it is as a mapping. But this is a scaler, and what is this?

Student:[Inaudible].

Instructor (Stephen Boyd):It's AII. Okay. So this tells us immediately if two – if there are two quadratic forms which agree for all X, it's as the diagonals of those two matrices have to be the same. That's because – just – if you evaluate the quadratic form AEI, you get the diagonal. Okay?

Now, how do you show – how would you show, now, that A12 is equal to B12, assuming X transpose AX equals S transpose BX holds for all X? How would you show this?

Student:[Inaudible].

Instructor (Stephen Boyd):What do you wanna add? What do you wanna put in?

Student:[Inaudible].

Instructor (Stephen Boyd):You wanna put in E1 plus E2. Okay. Let's try. If you plug in E1 plus E2 transpose A E1 plus E2, what comes out? Four terms.

Student:[Inaudible].

Instructor (Stephen Boyd):It's A11 – that's this one – plus A – maybe what is that? Is it A21 or something? A21 plus A12 plus A22. And if we believe that this is equal to the same thing with B there, we've gotta get B11 plus B21 plus B12 – oops – plus B22. Everybody cool on that?

Now because – if we plug in E1 and E2 separately, we find out that these are equal, and we find out that these are equal when we plug in E2 separately. So these go away, and you can see we kind of have it here.

So in a quadratic form, for example, if I gave you a method or a black box that calculates a quadratic form, it's not that easy – or it's not instantaneous for you to be able to get the matrix A that represents it. Or at least it's not something like with a matrix. If I gave you a black box with four BNC connectors on the left labeled X1 through X4 and five BNC connectors on the right labeled Y1, Y2 up to Y5, it's extremely easy to get the matrix. You apply one volt in turn to each of the inputs, and you measure the voltage on the outputs. Period. And you're reading off the columns. It's very easy. Okay?

Quadratic form – that's a different thing. You actually have to do a little bit of kind of signal processing and thinking to pull out – to figure out what A12 is. You'll have to actually first apply one signal, like E1, then E2, then put them together, and you're gonna kinda from the cross-coupling terms, you're gonna back out what A12 is. Okay?

All right. So let's move on.

Let's look at some examples of quadratic forms. Here's one – is the norm of A – B can have any dimension. B has any dimension. It does not have to be square. So if I take BX – that's some vector – and I form the norm squared of that, I just multiply out with that. It is X transpose B transpose BX, and if you'd like, I can finally rewrite that this way. Like that. And this is – that's the matrix that represents your quadratic form.

Does this matrix have to be symmetrized?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, yeah. I suppose you – a correct answer would be yes. Is this matrix symmetric already? I should have asked that question. It's symmetric already. So you don't have to symmetrize it. All right?

Okay. How about this one.

First of all, you have to get the feel – you have a very good feel of what a linear function looks like. You see any products of X1 and X6. You see a cosine or a sine of X1. It's all over. You have a – I promise you without – before you even step down and start doing math and all that kinda stuff, you know what a linear function looks like. Okay? You need to learn what a quadratic form looks like. And the things you're gonna wanna look for are things like squares. Now unfortunately, this does not fit immediately into the definition of quadratic form. But it has the sense and feel of a quadratic form, and in fact, it is one.

And so the question is what is this – by the way, it's a very interesting function. Can someone tell me what that function measures? I mean, just in English, some word that would describe it.

Student:[Inaudible].

Instructor (Stephen Boyd):It's something – exactly. It's something like – let me think of a good term. How about this – the wiggliness of the signal? It is the sum of the squares of the differences between the components of the vector and the previous one.

So actually, could this ever be zero? When would it be zero?

Student:[Inaudible].

Instructor (Stephen Boyd):If X in constant, this is zero. And what kind of vectors – for what kind of vectors would this be huge?

Vectors that – well, I mean, roughly high frequency ones. Ones that change sign every time.

So this is a – it's either – I was gonna say smoothness. It's a roughness measure or a wiggliness of the signal. That's what it is. So this has a use.

How – let's see. How are we gonna write that as a quadratic form?

You can just multiply this out, and you get XI plus one squared plus XI squared minus two XIXI plus one – something like that.

Now you do the matrix stuffing. Okay? And it's a little bit more subtle than matrix stuffing for finding – for writing a linear function. So a little more subtle. It's just – it's not too hard. Let's go through here and figure out what goes where. Or I'll do it. I don't know. I'll do a couple of these. No, let's just do three because I'm lazy.

X1, X2, X3. Okay? And this is X1, X2, X3 – sorry – transposed like that. Let's get this one-one term. What goes here?

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):One. Okay? Is that right? I think it's right. Okay. And then maybe you get – what's here? Two. And that's one, I think. Is that right? All right. I'm just winging it here.

What’s this entry up here?

Student: Zero.

Instructor (Stephen Boyd):Zero. And what’s the justification for that? How do we know that’s zero?

Because the one-three entry in the quadratic form corresponds to a product between X1 and X3. There is no product of X1 and X3 here. Okay? And in fact, I believe – are these minus ones, maybe? Okay. I’m trusting you. Remember this reflects on you, not me.

Okay. So it looks like that. And by the way, if you make this bigger, you get this tri-diagonal matrix down here. And the tri-diagonality also has this very strong meaning.

It says, “This is a quadratic function.”

But it says that, for example, X5 and X8 do not appear – there’s no product. There’s no interference between them. X5 interacts with X4, and it interacts with X3. But it doesn’t interact with X8. So that’s what the tri-diagonal means. Okay?

So this is – that’s how you write this out.

After a while, you get used to this, and you’ll even look at this – in fact, there are many fields where this matrix has a name or something like that, and they’ll just look at it, and they’ll say, “Oh, that’s the –”

This one actually has a name. It’s called the Laplacian or something like that on a line. But the point is, you would look at this and immediately know it’s that. In fact, people would just go back and forth without saying anything.

Okay. Here’s another one. Norm squared of A – this is norm squared of some linear function minus norm squared of another linear function. That’s gonna work because you’re gonna get F transpose F minus G transpose G like that. It’s just that. And that’s your – that’s what you need there. Okay?

Okay. Well, I should mention – and another way to fill this out here is you might wanna fill this out without worrying about symmetry. In fact, that might even be how this would be more natural or something like that.

So you work it out not worrying with symmetry, then symmatrize it. Then check because you can see here – you’re actually doing – when you fill this matrix in in a quadratic form, you’re actually doing operation – I mean, you’re doing operations. It’s not, like, blindly sticking in E1 and then reading off the output and filling that into a row – a column of a matrix. You’re actually doing real operations here. You better – you should check your quadratic form. Just for safety, check it. So, okay.

All right. So there's a norm squared – by the way, we're gonna later see every quadratic form has this form. We'll probably even see that later today. Okay? Or we can – we could see it later today if I pointed it out, but maybe I didn't. But anyway – so this is more than a mere example. It's – in fact, it's all examples. Okay. Now there's a couple of things that you define by quadratic forms. If you have a level set, that's the set of points where a quadratic form is equal to a constant. That's called a quadratic surface. And these would have things like – these would be things like in R2 and R3, this would be the horrible quadratic surfaces, like the – I guess the hyperbole and whatever they are and all these types of things. That's what these are. Some are interesting. I'll write down some of them. And a sublevel set is actually a quadratic region. And there's some important ones that come up, like a ball is this. This is the set of X for which X transpose X is less than one. That's the unit – that's called the unit ball NRN. What's the A matrix here?

Student:[Inaudible].

Instructor (Stephen Boyd):I. So in fact, I – which means something like a unit operator in Y equals AX – in the context of a quadratic form, I is simply sum of squares – norm squared. That's what I is – I represents. Okay?

Let me ask another question here. What's this? Some people confuse the English, including even some people who should know better.

That is the unit ball, and that is a unit sphere. But you'll see even people who should know better kind of referring to these the other way around. They might claim that they have a – I don't know – that they have some scheme. But I doubt it. Okay.

So what we're gonna do now is get a feel for quadratic forms. This is gonna be unbelievably useful. The same way you get a feel for Y equals AX, you're gonna get a feel for – well, we're gonna get another feel for Y equals AX very soon, but you're gonna get a feel for what does X transpose AX do? What does it look like? And we're gonna answer that.

Well, if A is A transpose, then you can write – you can diagonalize it with an orthonormal basis of eigen vectors. So that's – you can write A is Q lambda Q transpose, and we'll sort the eigen values so that they come out as the largest to the smallest.

By the way, it only makes sense to sort eigen values if it's known that they're real. But if a matrix is symmetric, its eigen values are real. So this is okay. But you have to watch out for that because in general, of course, the eigen values of a real matrix do not have to be real, and it would make no sense whatsoever to sort them.

Okay. So let's look at X transpose AX. We're gonna plug in X transpose Q lambda Q transpose, and I'm gonna reassociate this as Q transpose X lambda Q transpose X. Now the same way a diagonal matrix in Y equals AX is very easy to understand is basically you independently scale each component. But a quadratic form that is diagonal is also very easy to understand. It is nothing but a weighted sum of squares. There's no cross-terms. We just talked about that. It's a weighted sum of squared. So actually, this is really cool. This says the following: Q transpose X, which is a vector – and you should even know what it means – it's actually if you transpose X, is the vector of coefficients of X in the Q expansion? So it says it's the resolution. It's a resolving X. So this says, "Do a –." It says, "Resolve X into its Q components, and you get a new vector called Q transpose X." Now it says, "Form the quadratic form with the eigen vector matrix – eigen value matrix diagonal as the matrix defining the quadratic form." And that's this. It's got this form. It's just a weighted sum of squares. Now this is cool because now we can actually do – now we can say something. Actually, now you can really understand it a lot. So let's take a look. First of all, we can ask how big can that quadratic form be? Well, the answer has got to scale with norm X squared because if I double X, for example, what happens to X transpose AX? If I double X? It goes up by a factor of four. If I negate X, what does X transpose AX do? Nothing. It's quadratic. Okay? So here, let's see how big could this possibly be. Well, look. These numbers here, they're all non-negative. And I could replace them – these are all non-negative – then I can replace these numbers with the largest of those numbers – that's lambda one by definition. Oh, I should warn you. There are a few fields where they count the eigen vectors the other way. So lambda one is the smallest one and lambda N is this. This seems to be the most common, but there are other cases – for example, in physics it goes the other way around because lambda one is gonna correspond to something like a ground or energy state – the lambdas are energies. And also in mark off chains. So if you're in a depart – in a statistics department, but only in a course on mark off chains, lambda one will refer to the smallest one. Okay? Don't – but just normally, this is how they're sorted. So – but you may have to ask somebody. Okay. So this is no bigger than that, but this thing here – I know what that is. That – what's in these parentheses here – that's actually Q transpose X norm squared. That is the norm of X squared. That's one way to think of it. You can also say that this is something like – what do you call that? Vessels theorem or something – I don't know what. What do you call it when you do an orthogonal transform on something, and – or you expand something in some orthonormal basis, and the sum of the squares and the coefficients is equal to the sum of the norms squared to the original thing? Vessels something – is that it?

Student:[Inaudible].

Instructor (Stephen Boyd):Which one?

Student:[Inaudible].

Instructor (Stephen Boyd):What was it?

Student:[Inaudible].

Instructor (Stephen Boyd):Parcival. Thank you. So that's parcival there. There we go. Okay.

But this thing is that, and I get this: this says that X transpose AX can be no bigger than lambda one times norm X squared. That's very cool. And let me ask you this: could it be this big? And if so, how would you choose X?

Student:[Inaudible].

Instructor (Stephen Boyd):You'd choose X to be what?

Student:[Inaudible].

Instructor (Stephen Boyd):Exactly. The eigen vector says use lambda one, which here is Q1. So if I take X equals Q1, then what you guys are saying is the following: is that X transpose AX – well, let's just see if you're right. Well, what's AX if X is Q1?

Student:[Inaudible].

Instructor (Stephen Boyd):It's lambda one Q. So it's – we go Q1 transpose AQ1 is Q1 transpose lambda one Q1, and indeed, you're right. I don't know what that was. That was a little stray dot. There you go. This – the lambda one comes out, and you get one. Okay?

So now – actually, now you know something. If you want to choose the direction in which a quadratic form is as large as possible, you're gonna – that direction is gonna line up with Q1, the first eigen – how would you make this quantity as small as possible, given the fact that the – that this number in here has to be one?

You're gonna line it with QN, and this thing would come out as lambda N. So you get the same argument works the other way around. It says that X transpose AX is bigger than lambda N norm X squared, so you have this. Very important inequality here. Basically says that the quadratic form is no bigger than the norm squared times lambda one, but it's bigger than lambda N X transpose. And not only that, these inequalities are sharp, meaning that there is an X for which you get equality on the right, and there's an X for which you get equality on the left. Generally, those are different Xs.

So sometimes people call lambda one lambda max, and sometimes lambda N is called lambda min – very common. And the way you'd say that is if the maximum eigen value of A here – of course, you'd never say that unless A was known to be symmetric because then you would sound like an – idiot would be the technical term because a non-symmetric A can have complex eigen values, and then max doesn't make any sense. So you would never talk about the max eigen value – lambda max – like this of a non-symmetric matrix. For a symmetric matrix, that's fine.

Okay. I think we already pointed this out that you get these, so the inequalities are tight. Okay.

Now we're gonna do something interesting. I should point out – just to put a signpost in the ground to let you know where we are – we're just fooling around with quadratic forms, and let me be clear about this: so far, you have seen absolutely nothing useful you can do with quadratic forms. Okay? Just wanna make that clear.

That's gonna change. I assure you. But for the moment, we're going on trust. That is your trust of me. So – okay.

I think right after I said that, I heard the door close. But anyway, that's – okay.

You don't trust me?

Well see? He doesn't trust me. He knows it. All right. I shouldn't have said that. All right.

We're overloading equality now. We're gonna look at positive semi-definite and positive definite matrices. So you have a symmetric matrix. You see the matrix is positive semi-definite if X transpose AX is big – if the quadratic form is always non-negative. Okay? And it's denoted this way with an inequality like that and sometimes A with a little squiggle or something like that. I'll say when you might use a squiggle in a minute. Okay?

But this is the cool overloading. However, I have to warn you in many other contexts – in some contexts, A bigger than zero means the entries. This is the same as AIJ is bigger than zero. Just to warn you.

For example, in many languages if you have a high level language for doing linear algebra – like even, let's say, matlab – and you write A bigger than or equal to zero, what is returned is a matrix of bullions that tells you which entries are positive and which is not. And that would be true pretty much in any reasonable high-level language library for doing linear algebra. Okay?

So sort of like the exponential, you'd have to – this is sometimes – people call this matrix inequality is what they refer to this as. Okay.

Oh, and by the way, sadly, there are many applications where the matrix A is positive – or is non-negative. So you have – and that's called – what people call them is they say it's weird. They refer to those as non-negative matrices – those are non-negative matrices is what they – a matrix is element-wise non-negative. And they do come up. It comes up in economics. It comes up in statistics. It comes up in lots of areas, unfortunately. And then it's very confusing because you have to actually ask – stop people and say, "What do you mean by a non-negative like this?" Okay. All right. So a matrix is positive semi-definite if and only if the minimum eigen value's bigger than zero. So all eigen values are non-negative. That's the condition. And that is absolutely not the same as the elements in a matrix being non-negative. Now there are a few things we can say. For example, you can – the following is true. If A is positive – and this means in the matrix sense – then it is certainly true that it's diagonal entries are non-negative because, after all, this is EI transpose AEI. Okay? Converse is false. There's cases of all sorts – other than the diagonals, the off-diagonals can have any sign you like. Okay? So there's really no connection. Other than this, there'd be no connection. Now, you say a matrix is positive definite if X transpose AX is bigger than zero for all X non-zero, and that's denoted A positive or A squiggly positive, like that. And that says that the minimum eigen value is positive. Now I do have to say one thing. It's important to understand. Matrix – non-negative – being positive semi-definite or positive definite is not something a person can do. You can get rough ideas. You can say, "I think that's not positive semi-definite. It smells positive semi-definite." You can say all sorts. "I'm getting a positive semi-definite feeling from that matrix." But for a matrix like three by three and bigger, there is no way a person can look at it and say – you can look at it and say the matrix is not positive semi-definite in special cases. For example, if the three-three entry is negative, I can look at it and say, "It's not positive semi-definite." Someone can say, "Really? You can – without even – did you calculate the eigen values in your head?" To which, if someone asks you, the correct answer is, "I did." And they say, "Yeah? What are they?" And you say, "Doesn't matter. One is negative." That's all you have to do. And they'd say, "Which one?" And you go, "Does it matter? All you asked if it was positive semi-definite, and I said no." Okay. So all I'm saying is we've entered the place where these things are just not obvious. I mean, this is any more the case than you can look at a matrix and say what its rank is. Are there special cases? Of course. If there's, like, a zero column, you can say something intelligent about the rank. But you can't look at a matrix and say, "Oh yeah. That's rank seven. I've seen that before." I mean, you just can't – actually, if you can do this, please talk to me after class. But no, I mean, you can't do it. So this is weird that something as simple as our overloading a positive, which after all is not that hard to detect in a number, is very hard when you get to matrices. Or at least it's not obvious. Okay. So all right. So you have matrix inequalities. You say a matrix is negative semi-definite if it's – and you write that A less than zero. And even cooler is this: if you have two symmetric matrices, you write A bigger than B if the difference is positive semi-definite. And what this means is this: to say that A is bigger than B, for example – strictly bigger – means that the quadratic form defined by A is bigger than the quadratic form defined by B for all X not equal to zero. That's what it means. Okay? Oh, again. You cannot do this. I mean, this would imply this. This would certainly imply that AII is bigger than BII. But beyond that, good luck. It's not that easy to do.

Student:[Inaudible].

Instructor (Stephen Boyd):It does, indeed. Right. Although you don't have to because if you apply for X and sort of negative X here, you get the same thing in a quadratic form. So it does. Okay.

Now you know, you overload symbols so that, first, they're useful. And second that they suggest or the make, like, little neuropaths to other things that you know. Some of which – and then it turns out some of these are true and some are not true. So let's look at some of these.

So you can show things like this: if A is bigger then B, and C is bigger than D, than A plus C is bigger than B plus D and so on. And I'll go through some of these. Some of these are more or less hard than others to show. We may have – I think we actually spared you an exercise on this. I think – is that – did – I can't remember. I think we did spare them this. I don't know why we spared them this. It was just a moment of – anyway.

So – but these are things you can pick one or two of these and show them yourself. Okay?

However, a lot of things you'd know would be false. And I'll give you one, and that's – let me give you one. So here's one. If I have two numbers, A and B, then either A is bigger than B or B is bigger than A – period for two numbers. This is absolutely false for matrices. It's just not true. I think I've already gone back and I inserted the verb can't – or I put the model we can. So this should be – it's only partial order. You can have two matrices with not one – with two quadratic forms with one or one matrix not – or two symmetric matrices, one being – not being the other and so on.

By the way, in terms of quadratic forms, what this means is the following. It says that there's at least one X for which X transpose AX is bigger than X transpose BX. And there's some Z for which X transpose AX – now you know what, that should be a stretch like that. There we go.

So basically if two matrices are incompatible – incomparable – what you're saying is the following. Did I do this right? Yeah, I did. Okay. It says the following:

Student:[Inaudible].

Instructor (Stephen Boyd):Thank you. It was close. Very close. There. It says –

Student:[Inaudible].

Instructor (Stephen Boyd):Still wrong?

Student:[Inaudible].

Instructor (Stephen Boyd):Doesn't matter? Just forget it?

Student:[Inaudible].

Instructor (Stephen Boyd):What?

Student:[Inaudible].

Instructor (Stephen Boyd):Z – okay. Oh! Thank you. Thank you. Good, good, good, good. Okay. Thank you. Good.

You can't read it now, but that's okay. I'll write it Z transpose AZ – you still can't read it – is Z transpose BZ. There we go. Okay, there we go.

So what this says is if they're incomparable, then actually, the way I would describe it is this: basically, if they're comparable, you're saying, "This quadratic form is bigger than that one for any direction." That's what it means when they're comparable.

To say they're incomparable, you say, "Please compare these two quadratic forms."

And what you're really saying is, "Depends on the direction you look in. You look in that direction, this one's bigger. You look in that direction, the other one is bigger."

So that's what it means. That's exactly what it means to be incomparable. I can make that geometrically right now.

So ellipsoids – an ellipsoid is a sub-level set of a quadratic form. So if you write that – hang on – a positive definite quadratic form. So if I have a set that looks like this set of X transpose A is less than one, that's an ellipsoid if A is positive definite.

Oh, I do wanna mention one thing about matrix inequalities, and it's nasty. Okay. So if – normally, when someone writes this, okay? And let's assume you're not in the sixth week of some economics class where you're studying matrices with non-negative elements. So you're not in one of these weird places where people – where this means element-wise. This means matrix inequality.

All right. Let me just say a few things about it. If the matrices A or B are not square, then this is a syntax violation. Okay? If they're not symmetric – well, it really depends on the social context in which you make this utterance. Okay? So in many social contexts, it is considered impolite to use the matrix inequality between matrices that are not symmetric. Okay?

But in others, people do it. And they do it all the time. So I see it in papers. I actually see it in software systems and things like that where they do it. And they would say things like – if they're writing out some horrible block matrix, they say, "Well, no. It's easy because I can write down the non-symmetric matrix inequalities." But what it means here like this is it saves them the trouble of writing the lower triangular, or something like that. Anyway. If – so if A – if either A or B is not symmetric, this is either an error – but I would call it not a syntax error. It would be called a social error because you've said something that is impolite in the company where you find yourself. Okay? So if you do this – On the other hand, if it's clarified that you're in company where it's okay to use matrix inequality between symmetric – non-symmetric matrices, this is okay. And what it means is this. That's what it means. Okay? So – and let me just show you one – I'm just gonna do one example. Actually, what I'm saying now – I realize we have absolutely no applications of this material, so there is the question as to why you should be interested at all. So we'll fix – I'm gonna fix that. I promise. But meanwhile, let me just say something. This trips many people up. I predict some of you – it goes like this. You wanna check if that matrix is positive semi-definite. Okay? But it came from somebody else, and they came from a culture or a place or a field where it's okay to give quadratic forms in non-symmetric – with non-symmetric matrices. Okay? And there's nothing wrong with that. I'm not saying there's anything wrong with it, but let's just say that's what happened. But you grew up in a culture where people give symmetric matrices. Okay? And you said, "How do you check it" And you say, "Well, I remember that guy said you can't look at a matrix and tell. Diagonals are all cool. They're all positive, so it's a candidate for being semi-definite. I'm gonna have to find the eigen values." Everybody following this? Because that's how you do it. That's one way. So you would go to matlab, let's say, and you'd write Ig of A. And it would come back, and it would say something like this: one two one three. Okay? And you would say, "Well, that's it. That matrix is positive semi-definite." Okay? Any comments on this?

Student:[Inaudible].

Instructor (Stephen Boyd):What?

Student:[Inaudible].

Instructor (Stephen Boyd):If you don't know A is symmetric, this is completely wrong. It is absolutely and completely wrong. Let me show you something better that could happen. If you ask for the eigen values, and you got that, that's – it's still wrong, but it's better. It was so wrong that your mistake – it kinda – at this point, you'd say – well, at that point, you'd say complex. Wait. First of all, you – then you'd – someone is asking you. The next part is to check whether the eigen values are positive, or something like that. And you'd – this is gonna tip you off if something's wrong. Everybody see what I'm saying here? But the really insidious, horrible mistake is this – is you just form Ig of A, and it comes back like this. Okay? Just to let you know, there are examples. I can construct them easily where Ig of A are all positive, Ig of A plus A transpose over two, which is the symmetric part, has got a negative eigen value. Hey, we should make that a homework problem – maybe construct such a matrix. That would be fun. I'm gonna write that down as a homework problem. Not for you, don't worry. Future generations. So – okay. So back to ellipsoids. So an ellipsoid is a sublevel set of a positive definite quadratic form. Oh – the attribute positive definite applies both to symmetric matrices – in certain cultures, it also refers to non-symmetric square matrices. But you also use positive definite to apply to the corresponding quadratic form. So you say a positive definite quadratic form or a negative definite quadratic form or something like that. Okay. So this is an ellipsoid – a sublevel set, here. A special case is A equals I, in which case you get the unit ball. Okay? And this has got semi-axis. This is kinda what an ellipsoid looks like. In many dimensions, it looks kinda the same, except that it's got a whole bunch of mutually orthogonal semi-axis. The semi-axis – you can actually work out here – are given by the eigen vectors of A, and the lengths are given by one over the square roots of the lambda I here. So this is the picture. And you can check because if you go – what it basically says is if – I can explain, at least, the monotonicity property. If I go in the direction Q1 in X here, then X transpose AX is as big as it can be per unit length that I go. That means I will very quickly get to a point where I hit one, and I'm at the boundary of the ellipsoid. That means in the direction Q1, the ellipsoid is thin. Does this make sense? So the large eigen vectors – eigen values correspond to the thin parts of the semi-axis. By the way, there are many other people – ways people write ellipsoids. One other very common form is something like this. They put an inverse here. They're basically using the inverse of this matrix. When you do this, large eigen values here correspond to large directions in the ellipsoid. So – and you'll see all sorts of different things. There's even other ways to describe ellipsoids. Okay. All right. So now you can say a lot of things – I mean, about an ellipsoid – it's actually very interesting. It depends on A here. So the square root of the maximum eigen value divided by the minimum eigen value – that's called the – that gives you the maximum eccentricity in the ellipsoid. It tells you the max – how – it says, actually, how non-isotropic the ellipsoid is – or non-symmetric. An ellipsoid is fully symmetric if you go in any direction, and it kinda looks the same. That's true for the unit ball. If it's just a little bit off – like a little bit – like 5 percent fatter in one direction than another, you would say the eccentricity is, like, 5 percent. This would be like the Earth, right? Which is a little tiny – it's pretty spherical. But it is not spherical. I mean, it's actually got a little bulge. Okay? I forget what the order of the bulge is, but it's pretty small. It's not 5 percent, that's for sure. But nor is it zero. Okay? So – but you would call that something that has low eccentricity. If the eccentricity gets up to, like, ten, you get very interesting things. It means that – at least in R2, it means you got a cigar. That kinda thing. And in R3 and R4 and R5, you can't really say what the shape is just from the eccentricity. You can just say that there's – in one direction, it's got – it's way thinner than it is in another. So for example, imagine in R3, an ellipsoid that looks like a pancake. That would have there – in fact, tell me about the eigen values of an ellipsoid that looks like a pancake.

Student:[Inaudible].

Instructor (Stephen Boyd):It's too what?

Student:[Inaudible]

Instructor (Stephen Boyd):Well, let's see. A pancake is big in two – it's got two semi-axis that are long and one that's small. So the eigen vectors would be what?

Student:[Inaudible].

Instructor (Stephen Boyd):One what?

Student:[Inaudible].

Instructor (Stephen Boyd):One big eigen value and two roughly equal smaller ones. I think. Did I say that right? I think I did. Anyway, that gives you a pancake. Okay?

How about a cigar in R3?

You laugh. These are very important things. You know why? These things are gonna – ellipsoids are gonna do things like give you confident sets when you do statistical – when you do estimation. Statistical estimation or, for that matter, even without statistics, it's gonna give you confident sets. And it's a very good thing to know whether you're talking about a tiny point, a giant thing – but then the geometry of it – are we talking cigar or are we talking pancake? And then in R10, it's also interesting.

So what are the eigen values of a cigar?

Student:[Inaudible].

Instructor (Stephen Boyd):It's what?

Student:[Inaudible].

Instructor (Stephen Boyd):Two big, one small. Exactly. Okay. So that's the idea.

Okay. Now we're gonna – I mean, ellipsoids do come up, but we're gonna get to the next topic. And there's actually a bit of an abrupt change here. So up till now, I've been using A as a symmetric matrix that represents a quadratic form. Okay?

So let's just sort of end quadratic form for the moment. And A – I'm restoring A to what it had been until the beginning of this lecture. So A had been general non-symmetric, non-square matrix represents a mapping. Okay? So restore your old interpretation. Pull back in your old interpretation of A – of Y equals AX. Now we do this: we're gonna define norm AX divided by norm X. That's gonna be called the amplification factor, or the gain of the matrix in the direction X. So if you have Y equals AX, it basically says, "How big is the output, if you think of Y as the output, compared to how big is the input?" Notice that if I scale X by five, this doesn't change because X – this goes up by five, so does this. Even minus seven – so it's really – it's not a function of X. It's really a function of a direction. Okay? So that's the amplification factor. Now sometimes it doesn't depend on what X you choose. For example, if A is the identity, the amplification factor is one in all directions. I mean, that's kind of obvious. Okay? But actually, the interesting part is gonna be – in general, it's gonna vary with direction – the amplification factor of the mapping Y equals AX. And so we're gonna ask questions like this: what is the maximum gain of a matrix? And for example, what would be – what input would be amplified by that maximum factor? That would be the question. What's the minimum gain of a matrix? And what's the corresponding minimum input gain direction? Okay? And we want questions like this: how does the gain of a matrix vary with direction? That's what we want to know. Now I wanna point out – actually, this looks simple. It is extremely profound, and it is unbelievably useful. So early on in the class, we talked about things like null space. So let's talk about null space. How do – what would you – what does it mean, in terms of a gain of a matrix, to be in the null space of the matrix?

Student:[Inaudible].

Instructor (Stephen Boyd):It means the gain is zero. So the null space, in this language, is a direction in which the gain is zero. Okay? That's fine. No problem. It's useful. It's – and so on.

We're actually, now, gonna get the language to – what if I had a matrix which, in fact, is full rank. Let's say it has no null space. Okay? It's – there's zero null space, but the gain in some directions is, like, 1E minus ten. Okay? Of course, that depends on the context of the problem. But let's suppose that, for practical purposes, that's zero. Now you have something interesting. You have a matrix, which actually has zero null space technically. It has no null space because there's no vector you can put in and have it come out zero.

There are, however, vectors you can put in and have them come out attenuated by some huge factor – attenuated so much that, for all practical purposes in that context, it might as well have been in the null space. Everybody see what I'm saying?

So what we're working towards here is a quantitative way to talk about null space – in a quantitative way. So instead – so that's gonna be the idea.

I'm just saying this is where we're going. All right.

So we can actually answer a bunch of these questions, like, immediately now that you know about quadratic forms.

So matrix norm – if you talk about the maximum gain of a matrix, that's called the matrix norm or spectral norm. It's got other names. It's got – one other name is the – it's not the Hilbert – is this the Hilbert-Schmidt norm? Somebody from math? It's not Hilbert-Schmidt – no? Okay. It's a spectral norm – oh, L2 norm. That's another name. And it's probably got other norms – names in other fields. All right.

So maximum gain of a matrix – that's its norm. And guess what? It's overloading. So we overload it with just two – with the norm symbol. Okay?

So we simply write norm of a matrix this way. Now when you overload something, you have – there's one – there is one thing you must absolutely check. You must check that in a context – so overloading means that you assign a different meaning to something depending on the syntactic environment or context. That's what overloading means. Okay?

So you use the symbol equals. You use plus. Even though in the same page in your book, sometimes it's plus between vectors, sometimes matrices. Okay? But the point is, technically, those plusses mean different things because it depends on the context. Inequalities with matrices X now means a different thing and so on.

So the norm is – we're gonna also overload here. Now here's what you must check when you propose an overloading. You must check that when there is a context which could be interpreted two ways, the two meanings coincide – extremely important. So for example, if you have an Ig concept like vector edition – it's just vector edition, and then you have a concept like scaler edition. If the vector is one by one, it can be considered a scaler. And so you better be sure that, in fact, either way you decide to interpret it, you get the same thing. That's very important when you do overloading. You have to check consistency.

So here, for example, the matrix A could be a column vector, which we would consider a column vector, in which case the norm there is very simple. It's the square root of the sum of the squares. It could be a row vector. We haven't actually talked about that, and up until this moment for you, it was actually syntactically incorrect to write the norm of a row vector technically because we define them for vectors.

I mean, you could wave your hands in public and get away with it or something like that, and people would know what you meant. Now it's actually correct, and it actually agrees with what you thought it did. Not that any of you has ever done that, but I'm just saying had you done that.

Okay. So you have to check that when A – let's talk about a one – let's talk about an N by one matrix. So here's A. And let's talk about its gain as a function of direction. So I write this – well, everyone here knows exactly what this does. X is a scaler. It does nothing more. Okay?

And the gain of this – well, it's silly. There's really only one direction in R because remember, if you go one way or the other way, that's just a negation. It's the same. There's only one direction in R. So the gain of an N by one matrix – we can have a very short story about that. Ready? Here it goes. It's the norm of this that is interpreted as a vector. There, that was it.

There are no two directions in R, and so discussing the gain as a function of direction is not actually really interesting. Rather, it's a very short conversation. I don't know if that made any sense, but that's okay. So it works.

Now, let's figure out what it is. If you wanna maximize the norm squared of AX, we just square that out. You write it – it's the maximum over X non-zero of X transpose A transpose X divided by X transpose X. We just worked out what that is. It is – that is exactly the maximum eigen value of the matrix A transpose A, and I actually really want to warn you here. A – on this page, A is not square and need not be symmetric. It doesn't even have to be a square.

A transpose A, however, is two things. No. 1, A transpose A is symmetric. I should say No. 1, it's square. No. 2, it's symmetric.

By the way, it's also positive semi-definite. How do I know that? Because if you form the quadratic form associated with A transpose A, that's X transpose A transpose AX. That's norm AX squared, and a norm squared – always non-negative.

So A transpose A is positive semi-definite. Okay?

And so now you have a formula. There it is. The norm of a matrix – also called the matrix norm, spectral norm, L2 norm – there's one more name. It's gonna come later. We'll get to it. One more name – maximum singular value, in other words – but this is – and it's a square root of the largest eigen value of A transpose A.

By the way, this is a very considerable generalization of the following. Let me just see if I can write it right. Yeah. Let's try it.

For a scaler, this is basically saying it's the square root of A squared. There. It's a very considerable generalization of that formula. Okay? So that's what it is.

Okay. So that's the maximum gain. You can't figure this out by looking at a matrix. You can say a few things about it. We'll look at some of the things you can say about it. You'll look at some others in homework, but you can't look at a matrix and say, "Oh, that's got a gain of around three."

Well, you can make some guesses, but you – it's like the eigen values. You can't look at it and go, like, "Oh, nice eigen values."

You just can't do it. You're gonna – it requires computation. Okay.

Let me say a couple things about this. Well, I should say that in matlab, if you type norm of a matrix, you will get this. So that is actually overloaded there. And many other systems would do the same thing.

Okay. Well, the minimum gain is you wanna minimize that, and that's the square root of the minimum eigen value of A transpose A. That's this. Okay.

And we also now know what the directions of minimum and maximum gain are for a matrix. And they are, in fact, the eigen vectors associated with A transpose A associated with eigen values lambda max and lambda min. And this is pretty cool. If A has a null space, then A transpose – if A has null space, A transpose A is singular and semi-positive definite. So that means that it has eigen values that are zero. Those eigen vectors that are zero – they correspond exactly to the null space. Okay?

So now you have a method for finding out how the gain of a matrix varies with direction. Or at least you know it's extreme values, including zero if it's got a null space. So that's fine.

Let's just look at an example.

And by the way, we can – before we launch into the math of it – I mean, also I should add as usual, these ideas are not useful for problems with, like, N equals two and M equals three. That's a joke. You don't need any of this stuff to figure out what this does to a vector. You do need this to figure out what a 2,000 by 500 matrix does, which has a million entries in it. You do need it to find out what that thing does to a vector because there's no way you can eyeball it and say, "Oh, that can have a gain up to about 20GB. But oh, look at that. There's some directions where you're gonna get minus 50. That's pretty uneven. I mean, it just – you can't do it. This is kinda obvious, but before – actually, before we start, I wanna look at this and just say a few things. What is the gain of this matrix in the direction E1? Somebody tell me. You don't have to give me an exact number. Or you could just tell me qualitatively. There. Thank you. What's the gain of that matrix in direction E1? Well, if you plug in E1, what comes out? That. What's the norm of that? I mean, I don't know. We could figure it out exactly. What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):Square root of 35? I'm gonna trust you on that. Fine. What's the square root of 35?

Student:[Inaudible].

Instructor (Stephen Boyd):I mean, come on. If we're doing this with just hand, what is it? It's like – I hear that. The correct answer is almost six. Okay?

So the gain in the direction E1 is almost six. What is it in that direction?

Hey. You need to do this. I'm not gonna do it.

Student:[Inaudible].

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):The square root of what?

Student:[Inaudible].

Instructor (Stephen Boyd):56? What's that?

Student:[Inaudible].

Instructor (Stephen Boyd):What? A little more than seven? Seven and change? Is that – that's about right. Let's say seven and change. Okay.

So the gain in this direction was – what was that? Almost six. The gain in this direction – almost seven. So we're kind of – we're actually – we've got this matrix, and we're kinda – we poked it in one direction, we got a gain of, like, almost six. We poked it in another – a little bit more than seven.

By the way, you know what that kind of suggests, just on the basis of those two samples, is it's kinda homogeneous, right? Well of course, you can look and see that it's not, but hang on. I'm just saying based on those two samples, you might get the wrong impression. You might get the impression that the gain is always between six and seven, okay?

Now, if I told you that the maximum gain of this matrix was 1,000, what would you say?

Well, you wouldn't believe it from any reasons, not least which is that the maximum gain is sitting on the lectures light below you. But you would be quite dubious because you'd look at this, and you'd say, "Now come on. You put in two numbers here with norm one. Okay? That's two numbers whose squares add up to one. You plug it into that, and you get a matrix with three – I mean, you get a vector with three entries whose norm is 1,000. Not seeing it." I'm just saying just, like, having the intuition running here and all that. Okay. All right. Let's just see how, if you just work out exactly what it is, you form A transpose A. That's this symmetric matrix. And everything should be going – everything should be working here. These diagonal entries are positive. They have to be. And just so you're just checking everything. Here's the eigen values of this matrix are 90 and .265. Okay? Obviously, this matrix is completely made up, but the point is this actually kinda shows you how these things – why these things are good. I'll say something about that in a minute. The eigen values of this – and that says, actually, that the maximum gain is 9.53, and actually it turns out it's the eigen vectors associated with that – that's that, by the way – this is T lambda T transpose. You put in – you apply to A this vector, which has norm one, and the result comes out and has norm 9.5. Okay? By the way, that's consistent because we took E1 and E2. We got a norm that was, like, a little under six – a little over seven. We just took the first two easiest things to calculate. So it's not that surprising that by taking some weird combination of E1 and E2, you could actually get – scoop the gain up a factor of 30 percent higher – 20 percent or something like that. Whatever it is – 25 percent higher. That's not implausible at all. Okay. Now let's look at the minimum gain. The minimum gain is gonna be, actually, the square root of this thing. And it turns out that's .5. And it says if you put in this vector, then A times this thing comes out scrunched down roughly by a factor of two. So for example, for people in electrical engineering, you would say that the gain of this matrix varies from plus 20 DB, roughly, to minus six. Okay? Okay, so maybe it's like minus four and plus 19 DB. But that's roughly what it is. Everybody see this? Now I wanna point something out. That's a range of 20 to 1. You know what? I take it back when I said you don't need this to understand something this simple. You know what? That's not obvious. There's no way you could look at that matrix and say that this thing has a gain – I mean, guessing the maximum gain is to be nine. Okay, cool. And you could do that. Okay? That's fine. Guessing that the minimum gain is .5 – I don't think so. Okay? You could look at that matrix, and you could say, "One column is not a multiple of the other. Therefore, that matrix is for sure full rank. It's got rank, too. It has no null space. There is no X that has a gain of zero." In other words, there's no X for which A is equal to zero. But it is not obvious until you do this analysis that there is an X which gets scrunched by a factor of two. Okay? That's what I'm saying. And if you think it's not obvious here, you gotta try it again in a 1,000 by 300 matrix or something. It's not obvious at all. Okay. So let me say a few more things about matrix norm. So first of all, it's consistent with vector norm. I mentioned that before. If you worked out what it is, you get lambda max of this one by one matrix. I know what the maximum eigen value of a one by one matrix is. Of course, this holds always. This is – this holds, and this says – it says that the norm of the output is less than or equal to the norm of the input times the maximum gain of that matrix. That's my reading of this equation. By the way, you haven't seen anything like this except sort of like – Koshi Schwartz is like the closest thing you've seen. You've seen something like A transpose X is less than norm A norm X. Okay? You've seen something – that'd be something – the closest analogy I can think of to something you've seen is this. But – okay. Scaling – if you scale a norm, it's the same as scaling with the absolute value. And you get the triangle inequality here. Actually, no one would have overloaded the norm of a matrix had the triangle inequality not held. And the triangle inequality says that if you add two matrices, it's less than the norm of A plus the norm of B. By the way, you can get a very good idea for this if – basically, from a sort of an operator point of view, it basically says the following: take X and apply it to both A and B, and then combine the results like that. This is the block diagram representing A plus B. What this says is let's figure out – let's put in an X citation here and see the result, and let's see how big it could possibly be – the gain. Okay? What this – norm A says that when you put something in of a norm here, then what comes out can't possibly be more. The maximum gain of A is norm A. Down here, norm B. So there's no way that two things, each of which amplifies something by, for example, two and three separately, can amplify something by more than five. It can be less, of course. Okay? Definiteness means if the norm of a matrix is zero, then the matrix is zero. You wanna watch out on these things because we've already seen weird things like this. The eigen values of a non-symmetric matrix can all be zero, and yet the matrix is not zero. We've seen that. But that can't happen for a symmetric matrix. For a symmetric matrix, all eigen values is zero, the matrix is zero. Okay? And you have norm of a product. And this, again, you can figure out. That's a different block diagram, a very simple one. It's BA – it says operate with B first, then A. And it says that the gain – in this signal processing, the maximum amount by which an input can be amplified going from here to here is definitely less than – less than or equal to the maximum amount here multiplied by the maximum gain here. And I can ask you the following question: when would you get equality here? And you can tell me in terms of most sensitive and blah, blah, blah input directions. You would only get equality here if the following happened: if when you probe B with the maximum gain input direction – that's the eigen vector associated with B transpose B of the largest eigen value – what comes out, then, would have to line up exactly with the highest gain direction of A. [Inaudible] would be further amplified by the maximum amount, and the total amplification you'd get would be norm B norm A. Did that make sense? Okay. So I'm gonna quit here and just say a few things. First of all, have a good break. Enjoy homework eight. Feel free to read ahead. Now come on, some of it – there's flights involved. You can do this on the airplane. It's fine. Feel free to read ahead, but also feel free to have the sense to not do problems in homework eight on material we haven't covered, although we've covered a whole lot of it. So, okay. Well have fun.

[End of Audio]

Duration: 73 minutes

IntroToLinearDynamicalSystems-Lecture17

Instructor (Stephen Boyd):Welcome back. I've got to say, the schedule is a bit bizarre where you're off for a week and then come back for a week and a half, basically.

Let me go down to the pad here. I'll start by making just a couple of announcements. You might want to zoom in a little bit here. The first one is, just to remind you, the final is going be next Friday to Saturday or Saturday to Sunday. And as always, if for some reason these don't work for you, just let us know and we'll figure out an alternative time for you to take it. Other administrative things are homework, so I guess homework eight, which you have now, right, and that's due this Thursday, I think. Is that right? Male Speaker:

Yeah.

Instructor (Stephen Boyd):So that's due Thursday. But we're going to pipeline a little bit and we're going to launch homework nine later today. That's after we figure out what it is and all that sort of stuff. So we'll launch homework nine later today and that's due next Thursday, that's obviously, the last one. Except, of course, with you final you'll be given a homework ten, which you can turn in during the first week of January.

Okay. Let's, if anyone remembers what we were doing, it seems, this is the topic we were doing before our quarter was interrupted by the schedule. We were talking about the idea of the gain of a matrix in a direction. So this is actually very, very important. This is sort of at the heart of the overloading Y equals AX. If you have Y equals AX where A, Y, and X are scalars, it's, you don't, I mean, it's very easy, that's simply multiplication. When you have a matrix here, it's much more interesting. Because in fact, the amount by which the size of Y divided by the size of X, that's the gain of the matrix A in the direction X, this varies with direction. That's exactly what makes it interesting.

Now there are cases when it doesn't vary with direction. You've seen one. One is when A is orthogonal. So if A is square and its columns are orthonormal, then Y equals AX, the norm of Y is the norm of X, the gain is one in all directions. So watch, you're going to get is the complete picture today, on that.

So last time we asked the question, what is the maximum gain of a matrix. And we answered it. And the answer was well, it's, first of it it's got a name we gave it a name. The name is the norm of a matrix. And it is given by the square root of the largest, whoops, there we go, there it is the square root of the largest eigen value of A transpose A. Now, you have to be very, very careful here, because A is not necessarily symmetric. In fact, it's not even necessarily square. A transfers of A, of course, is always square, however, and positive semi definite. It's nothing norm of a matrix and that tells you exactly what the maximum amplification factor of a matrix is when you consider it the mapping of Y equals AX. But we know more. We know things like this, the maximum gain input direction is the eigen vector associated of A transpose A, associated with [inaudible]. So that would be the input, which is amplified by the largest factor. The input that is amplified by the smallest factor, input direction is QN. That's the eigen vector of A transpose A associated with [inaudible]. Okay. And of course, this can be zero, which is just a longwinded way of saying A has a null space. Because if you're in the null space, it means you've come in non zero, you come out zero; it means the gain is zero. So this is a quantitative idea that generalizes the qualitative of null space. Okay. This brings us to the whole story. And the whole story is the singular value of decomposition, otherwise known as the SVD singular value of decomposition. Now, it's been a while. Historically it's been around for quite a while, maybe 100 years, that's certainly 100 years. It was certainly well known in the '20s and '30s. It's been used in statistics. You'll hear other names for it. The one you'll hear most is PCA, which is principle component analysis. And there's probably a couple other words for it in other strange fields. That came out the wrong way. That was not to imply that statistics is a strange filed, although, it is a bit odd, actually. So but anyway, so you'll hear other names for it. Certainly, in some areas of physics it's got some name, and I guess, in other contexts it's called things like the carbon and low ebb expansion, that's something else you'll hear about the KLX. You'll hear all sorts of names. But the main ones are singular value of decomposition and principle component analysis. Okay. And the singular value of decomposition is a triple matrix decomposition. By the way, we've seen a bunch of them by now. Well, not that many, actually, but we've seen a bunch, and it is very important to keep your matrix decompositions clear in your mind. We've seen diagonalization, that's something like A equals T, land to T inverse. We've seen QR factorization. We have seen an orthogonal eigen decomposition. That's like A equals Q land duct Q transpose, and this is yet, another. By the way, there's lots of others as well, maybe not a lots but five or so other common ones. So here it is. It says that any matrix can be written out this way. It's A is U sigma V transpose where A is end by end with rank R. This intermediate matrix sigma is diagonal and its size is R by R. So it's exactly equal to the rank its size. And the singular values, that's the sigma one through sigma R, these are ordered, always, from largest to smallest. So sigma one is the largest and sigma R is the smallest here and these are positive. U and V are both matrices with orthonormal columns. So U transpose U is I and V transpose V is I. So that's what these are. So it looks like this, U and then sigma, which is diagonal, and then V transpose can look any size, like that, for example look like, for example. Okay. That would be a typical way this might look. Okay. So this intermediate dimension is the rank of the matrix. Now, you can write this many other ways. If you write the columns of U as U one through UR and the columns of V as V one through VR, then you can light this new sigma transpose out as what people call as a dyadic expansion. So that's a dyadic expansion, which of course, means you're writing the matrix as a linear combination of dyads and the dyads are UYVI transpose. Dyad is just a word for rank one matrix, also called an outer product, though, in other fields. Actually, there are no other fields. This is math. So – So sigma I here, these are called the singular – these are the non zero singular values of A. So there's actually a bit of confusion and we'll get to that in a bit, and I'll warn you about that, as to whether singular values can be zero. In the definition I've just given you, the singular values are all positive period. So it doesn't make any sense to say sigma five is zero. But we'll see that, in fact, there's a lot of confusion on the streets about this and people do refer to zero singular values. But we'll get to that. Okay. Now, these VIs are called the right or input singular vectors of A, we're going to see Y in a minute, and the UI are the left for output singular vectors. And it sort of makes sense. If A is U sigma V transpose, you can see that when you sort of operate, but when you form AX, the first thing you do is multiply it by V transpose X. But you know what that is. That's expanding V in the VI basis. That's the first thing you do. Then you scale and then you send the output along the UI basis. So these are really the basis sort of for the input that actually comes through the system and the output. So we'll get more into that in a minute. But it's clear V is associated with input and U with output. That's clear. Okay. Well, these things are, right at the moment, they're quite mysterious but I think we can clear up the mystery very quickly. If you form A transpose A and A is U sigma V transpose times U sigma V transpose, if you transpose this you get V, sigma is diagonal and therefore symmetric, so sigma transpose is sigma, then you get U transpose U sigma V transpose, that goes away, and you get V sigma squared V transpose over here. And what that says is the following: It says that V sigma square V transpose is an eigen vector decomposition of A transpose A. So A transpose A is a symmetric positive semi definite matrix, if you write out it's eigen vector in respect to decomposition, you get V sigma squared V transpose, that's this thing, and so those are the input singular directions. And that says that these are eigen vectors of A transpose A. It also tells you exactly what—now we know what the singular values are. They're no longer mysteries. The singular values are the square roots, sigma I is the square root of the I eigen value of A transpose A. Okay. And that's for I equals one to R. Those are only eigen values of A transpose A that are positive, then they become zero. And in particular, sigma one is nothing but the norm of A. It's the square root of the largest eigen value of A transpose A. So sigma one is the maximum gain. That sort of makes sense. We'll see why that's the case in a minute. We can actually see it right now, but I'll wait a minute. Now, if you multiple A by A transpose in the other direction, you get AA transpose, then you just multiply this out. This time the V goes away and you get U sigma square U transpose and what you find are the UI are eigen vectors of AA transpose. So that's what they are. These are the output directions. And the sigma IR, again, the square roots of the eigen values of AA transpose, and here a question comes up, and now you have two formulas of the sigma I, which is the right one? Well, they're both correct because it turns out that the eigen values of, let's see, CD are the same as the non zero eigen values of CD, are the same as the eigen values of DC, whenever these are both square matrices. Was there a homework problem on that or something? I have a vague memory of this. Yes. Okay. There was. Good. So in fact, AA transpose and A transpose A, which by the way, generally have different sizes, but the non zero eigen values are the same. By the way, this is super useful. I can give an example. If I ask you what are the eigen values of PP transpose, or something like that, that's a symmetric – I don't even need that, I can just do it the other way around. Let's make a general thing PQ transpose, there you go. So PQ transpose is a rank one matrix and I asked what are the eigen values of PQ transpose. You would use this theorem to answer it. Where you'd say, well, they're the same as the non zero eigen values correspond to the eigen values of that. That's a one by one matrix. I know what the eigen values of a one by one matrix are. It's just the number. So that says that the eigen values of PQ transpose are Q transpose P and N minus zeros if that M by M. Okay. So I'm just giving you an example but anyway, that was just a little aside. Okay. Now, we can actually see what these Us are. Now, obviously, if you write A is U sigma V transpose, if you plug anything into A like AX and some vector here, and it gets multiplied by U. So anything of the form AX is a linear combination of the columns of U.

Therefore, the columns of U, actually certainly, are a subspace of the range of A but in fact, there' exactly equal to the range of A, because I can make this thing be any vector I like, including like E one through ER here. Okay. So U one through UR are an orthonormal basis for the range of A. Now, V one through VR are an orthonormal basis for the null space of V orthonormal complement, not the null space of A. In fact, it's sort of the opposite of null space of A. These are the input directions that basically are not in the null space. That's one way to say it. So if you're orthogonal to V one through VR you're in the null space of A. So that's what this means. So this connects, once again, the all sorts of things you've seen before. So let's get some interpretations now of what this means, A equals U sigma V transpose. Well, it means this, you can write it out as a dyadic expansion or you can do it as a block diagram. As a block diagram, it works like this. It says take a vector N and the first thing you do is you partially expand it. Of course, the Vs are not a basis, because there's only R of them. But it says you partially expand V, you calculate the V coefficients where I equals one to R, that's V transpose X, that's an R vector, this is R vector with the coefficients of X and a VI expansion, then you scale those by positive numbers. You scale this by positive numbers. That's all you do. That's diagonal, no interaction. Then you take those positive numbers and you reconstitute the output by multiplying by this matrix's columns orthogonal. Now, by the way, when you move from here to here, that's an isometric mapping. In other words, distances going from here, this is distance in RR, that was big R, big bold R, with a superscript little r, and then this is in our, I can't remember now, M, is that right? Yes. This is NRM. So this mapping is isometric. The distances are preserved exactly, angles preserved, everything here, norms, everything. Okay. So this is kind of the idea. Now, this looks exactly the same as if A is square it looks just like an eigen value decomposition for symmetric A. Actually, it's one difference. It is symmetric A, this is diagonal and real, but they can be negative, because you can have negative eigen values. And in that case, U and V are the same. The input and output directions are the same. Part of that sort of makes sense. That's what symmetric means. Symmetric means something like you can swap the input and output and it looks the same. Because when you transpose a matrix that's what you're doing. You're switching the roles of the input and output. So this kind of makes sense. So this is the picture. By the way, you can see all sorts of things now about you get a completed picture of the gain as a function of direction now. And let's see how that works. If you want to find an input here for which the output is largest possible, that's here, that'd be the maximum gain input direction, you would do this. You'd say, well, look, norm here is the same as the norm here. So if you want to maximize the norm here, you don't even have to worry about the U, just make this as big as it can be. So it says what you really want is you want this vector, then scaled by these numbers sigma one through sigma R, to be as big as possible. Now, when you put in a vector here of unit norm, here you're going to get a V transpose X is a set of numbers whose norm is less than one. I guess that's vessels theorem, or whatever, and I believe you had a homework problem on that, too. So because the partial expansion and the VI coordinates. How would you make this come out so that – what's going to happen to this vector is this vector's going to come out, it's going to be a bunch of numbers whose squares are going to add up to less than one, less than equal to one, and then they're going to be scaled by positive numbers. If you want that to come out as big as possible, you have to put all, all of this into the first component because it is going to get the greatest amplification factor. And that means you want V transpose X to be E one. And there's only one way to do that. And that's to chose X equals V 1. That's the first column of V. You can check this, you can verify this many other ways. I'm just explaining how this works. Okay. So that's the idea. U one is the highest gain output direction. And of course, you have AB 1 and sigma one and U one. In fact, of course, you have AVI equals sigma I times UI. So it basically says that the I's singular input direction is amplified by gain factor sigma I. The sigma I's are gain factors. So let's see, actually, if we can figure out how this works. Am I missing something here, Page 30? No, I'm not. Here we go. All right, here we go. So let's take this four by four matrix. Obviously, that's so small you can figure out what happens yourself. Actually, that's on the boundary, to tell you the truth. Even a four by four matrix, you can't look at it, no person can look at a four by four matrix, except in totally obvious cases like when it's diagonal, and say something intelligent about the gain varies. I mean, you can say a few things, you can mumble a few things and wave your arms, but you really can't say much more. So you can imagine what happens when you have a 500 by 500 matrix. But anyway, let's take A is in our four by four, and the singular values are 10, 7, .1, and .05. So the first thing, we can all ask sorts of questions about what does that mean about the matrix and its gain properties. Well, the first is this, the rank of A is four, because you have four singular values. So A is U sigma V transpose, U, V, and sigma are all four by four in this case. And what this says is that the gain varies from as small as .05 to it as big as 10. So that is a 200 to 1 ratio of gain. So if you want to know how isotropic is this mapping, in the sense of how much can the gain vary with direction, the answer's 200 to 1. We'll talk, actually later this lecture, about what that isotropy means and isotropy means, lack of isotropy. So the gain varies from .05 so it has no null space. However, there is an input direction, which is scrunched by factor of 20 to 1. And there's another input direction, which is amplified by a factor of 20 to 1. Okay. So we can be much more specific about it. We could actually say that if you have an input, if X, mostly, is in the plane given span by V 1 and V two, these are orthogonal, that's the two most sensitive input directions, if you're in that plane, then you get amplified by somewhere between seven and ten. That's what it means. Okay. So there's one plane of input directions for which this system roughly has a gain on the order of ten, you know, between seven and ten. By the way, the outputs come out along the directions span by U one and U 2. Those are the two most sensitive output directions, or the highest gain output directions. Now, at the other side, there is another plane, by the way, orthogonal to the original plane that requires, by the way, some serious visualization skills, advanced visualization skills, since this is an R 4. But you have another plane orthogonal to the original high gain plane, now you have a low gain plane inputs that along in this low gain plane they get scrunched by factor between zero, 5, and .1. Okay. And they're going to come out, well, they'll hardly come out, but when they come out, I mean, they'll be greatly attenuated, but they will come out along U three and U four span by that low gain [inaudible] plane. That's the picture. Okay. Now, depending on the application, you might say that A is rank two. That's going to depend entirely on the applicant. Obviously, it's not rank two. Mathematically it's rank four, right there. However, in some cases you might say that A is rank two. As someone said, it's sort of the gain, if this is some kind of communication channel or wireless system, that's a gain of plus 20 DB, that's like minus 26 or something like that. You might say, you know what, that's below the noise floor. That's worth nothing. Okay.

Let's look at another sort of baby example here. Let's take a two by two matrix. Now, for a two by two matrix you do not need singular value composition to understand what it does, that's for sure. That much is for sure. Okay. With the singular value of 1 and .5, basically what it say is you'd take something like X here, there'd be an input coordinate system, that's V one and V two, and you would resolve X into its components. That would be this and this. This V one gets multiplied by sigma one and then comes out along U one. So whatever that is, if that's V one, let's say that about .55. Then you've got .55 times U one, where's U one, that's over here. This should be bigger, shouldn't it? A little bit. I didn't draw it right, I think. Well, anyway, it's about right. That should come out about here and then for U two you get, maybe, I don't know, .7 V two, .7 V two should get scrunched by .5 and you should get .35 V two, and indeed, that's, at least visually, a third. So that's about right and that's the output. That's the picture. Okay. That's the singular value decomposition. It has spread, in fact, to almost all fields. Then any field that uses any math, any field that does any computation. So you will hear about it in different contexts, you'll hear about it in the statistics, you hear about it in the finance, you'll hear about it in basically anything. Signal processing control, wireless stuff, networking it goes on, and on, and on. So you'll hear all about it. So it's actually a very good thing to know. It's probably it's the last real topic for this class and it's very important and you'll be seeing it unless you – well, anyway, you'll be seeing it in many, many contexts. There's actually still some weird backward fields where it hasn't hit yet. It's coming to those fields. So there's a bunch more. I think, actually, let's see, the last couple of years it's sort of hitting fluid mechanics. So that's the one I know about most recently. Which is amazing, because it's this incredibly complicated very sophisticated field and they just didn't know about this and it's a big deal now because it explains all sorts of stuff there. Anyway, it's also used in tons and tons of other things. It's used in, let's see, it's used in the current standard for DSL, is basically straight out of the SVD, so all multi-input, multi-output wireless systems straight from SVD. It's a lot of methods for compression and things like that, straight out of the SVD. Another one, and this is – it's only a small stretch, but it turns out, in fact, things like Google's page rank. You may have heard this, straight from the SVD, the zero of order of the page rank, SVD. It's nothing else. So anyway, I just wanted to point out that this is actually an incredibly important topic. I want to say that because others, obviously, are much less important, for example, our friend, the Jordan canonical form. But we teach that just so you'd know about those things and can defend yourself, if needed. If you're caught in a dark alley late at night with a bunch of mathematicians. So, okay, who are pissed off. Okay. So let's do a couple of applications, but like I say, you go to Google and type this or some of it's synonyms like PCA and there'll be literally millions and millions of hits. The whole field's based on this so you'll be seeing a lot of this. The first thing is, actually, I can now tell you the whole story on the pseudo-inverse. This is for any A not equal to zero. I've already fixed the notes, I just haven't posted it, but I just noticed this mistake this morning. So for any A non zero has a singular value decomposition so people don't really say that the zero matrix doesn't have a SVD or something, or it gives you a headache, I don't know, something like that. I mean, V and U would be zero by, they'd have to be zero by something. So any non zero matrix has a SVD. So if you have a non zero matrix and its SVD is U sigma V transpose, then the Moore-Penrose inverse is V sigma inverse U transpose. That's it. And you can check this agrees exactly with the two formulas for the Moore-Penrose inverse or pseudo-inverse you've seen so far. So you've seen when A is skinny and full rank there's this formula, and you've seen when A is fat and full rank there's this formula. But this formula, here, works for any matrix that is non zero, any at all, and it basically does what U think U do. If you want to sort of invert this, you'd sort of make a V transpose over there. Sigma square invertible diagonal can be, of course, inverted, and then you do something like inverting U, it would be inverting U, if U were square, you'd make it U transpose and you get this. You switch the role of the input and output directions, invert the gains along these directions and that's what you get. Okay. So that's the general Moore-Penrose inverse. It coincides with these, in these two cases, but it's now generalized to other situations. And in fact, in the general case here's the meaning of it. So let's look at the general case when A is not full rank. Okay. So A is not full rank. If A is not full rank and you want to do least squares, least norm things get tricky. So let's see how that works. If A is not full rank the first thing you do is you say, well, how close can you get to Y with something of the form AW. This a least squares problem except now A is not full rank. And so up until now, you actually did not know how to solve this. You inform your trusted A transpose, A inverse and right at that point you have a problem because A transpose A is not invertible if A is not full rank. Okay. However, if you want to minimize this, there are still – you can form this least squares problem and ask how close can you get? Now, because A is not full rank the answer's not going to be a single point, it's going to be actually the whole set of points an affine set. So it's going to be the set of points for which AZ minus Y, the deviation is as small as it can be. That's an affine set. It's a set of least squares solutions. Now, in that set, you can then ask for the smaller, the minimum norm least squares approximately solution. And that's exactly what A dagger Y gives you. So this is quite complicated but it's worth thinking about and you should be able to sort of see what it does now. So this is the picture. So X zero inverse is the minimum norm least squares, and I really should put approximate here because I don't like writing, I don't even like, I'm going to fix that. I didn't even like saying least squares solution. Because the whole point is it's not a solution. I know that that's what everyone says, but it irritates me. So I'm going to change it. It's an approximate solution. Okay. Now, we can also do the pseudo-inverse of the regularization. So let's see how that works. So let's see how that's going to work. So let's let U be positive and then this is the regularized, this is ticking of regularization, or ridge regression you call it in statistics, or something like that, and here you have two things. You have sort of your mismatch and then you have something that penalizes the size of X. By the way, it's clear, this is going to be related to pseudo-inverse, because pseudo-inverse finds among the Xs that minimizes this, it finds the one of smallest norm. So it's clearly related, these are clearly very close. Now, this is a perfectly good formula. It's simply A transpose A plus Mu I inverse, A transpose Y. This inverse is valid, period. A can be fat, skinny, full rank, it don't matter, it can be zero here, it's fine, no problem, okay, because the Mu I takes care of everything and this is always positive definite. Okay. Now, as Mu goes to zero this limit gives you this pseudo-inverse. And in fact, it turns out if you want to get the pseudo-inverse this is another formula for it. It's the limit of A transpose A plus Mu I inverse times A transpose. So it's the limit. Another way to understand what the pseudo-inverse is, if you don't like the SVD, you can think of it this way, it's the limit of regularized least squares in the limit as Mu goes to zero. Now, in some cases this is super simple. If A is skinny and full rank, then this makes sense when

Mu equals zero and it just converges to our old friend, this thing. But this formula's quite different in other cases. If A in fat and full rank, A transpose A is definitely not invertible. Okay. But A transpose A plus Mu Y, as long as Mu is positive, is invertible. And this inverse makes sense and this will actually converge to this A value. In fact, in that case it would have converged, of course, to A transpose quality inverse, which is the other formula for it. But the other cool thing is that even if A is not full rank, this converges to that, to the pseudo-inverse. That's what it is. So the pseudo-inverse is something like the limit of infinitely small regularization or something like that. As usual, the important part to know what it means not to know simply that it exists and so on. Okay. So now you know the full story on the pseudo-inverse. Now, like a lot of other factorizations, like the QR and I think we've seen a couple others, maybe just QR, you can extend, in factorization, by zero-padding matrices on the left and right. You do that to make certain a zero-padding some and then padding out other things in convenient ways. One way is to like, for example, fill out an orthonormal basis. So you can do that with SVD, too. So if you take – that's the original SVD, like this, so R is the rank, these are all positive numbers. What you do is you find a matrix U two, which is complimentary to U one, and you find a complimentary matrix to V one. So that's V two. Complimentary means that the columns of the V two are together with the columns of V one, form an orthonormal basis. And you know lots of ways to do that. One way is to run QR by appending, for example, the identity matrix after U one through UR that would do. So what you're going to get now is U one U two is actually now a square matrix and it's the output size M by M. And you'll get a matrix V one V two. By the way, these look, when written like this, they're fat, they're actually square because U is tall here. Right, so this is actually a square matrix, although visually when printed it doesn't look like it. That's square as well. That's the input direction and both are orthogonal. So now what you do is to make up for the fact that you've just added a bunch of extra columns on U and V, no problem, you zero pad sigma and you get something like this. Now sigma is no longer square. It's diagonal in the weird sense that off the diagonal N equals zero. But generally people don't talk about diagonal matrices that are not square. I mean, like, I don't know why they don't, but they generally don't. And then these are just the zero padding the square. By the way, some of these could be zero, depending on A. So for example, if A is skinny and full rank, like this, it means it's rank R and that says that when you write it as U sigma and V transpose, one of these is going to be square. I guess this one. That's your V transpose. So in this case, if A is skinny and full rank, you don't have to actually add anything to V. For example, if A were fat and full rank, you wouldn't have to add anything to U. Okay. I hope I got those right. I might not of, but too bad. Okay. So you zero pad like this and now you can write, once again, A is U sigma V transpose. So you have A as this thing and it works out this way. And people call this the full SVD of the matrix. In this form what happens is these are actually orthogonal matrices U and V. But sigma now has exactly the same size, as the original matrix A is diagonal, I mean, in the sense that's the only place where it can have – but it can also have zero entries. And the one we looked up before, if you want to distinguish it from this, is called the compact or the economy SVD. Oh, I should mention this, it's actually quite important in matlab, for example, if you just do this I think you get the full thing. Although I don't remember. Is that correct? Okay. And if you want the compact one, I think you write something like this. That's econ, the string econ, I think. Anyone have a laptop with them? I know Jacob was, he's not here, so anyway. Okay. By the way, this is quite important. Let me mention a couple of reasons this is important. You don't want this – if A is, for example, I don't know, 100 by 10,000 then just forget the computation, let's talk storage, here. So you can be in big trouble if you type this. If you type this it's going to return a V, which is 10,000 by 10,000. Okay. And if that's not big enough of a dense maker to make trouble, you know, make this 100,000. And now, for sure, you're in trouble. So unless you want the big one, you could actually be in big trouble. Here, this one would do the right thing. I would return something that was, if this was full rank, it would return 100 by 100 matrix, no problem, 100 by 100 diagonal matrix, definitely no problem, and in this case 100,000 by 100 matrix, again, that would be no problem. Okay. But if you try this one it won't work. Okay. So I just mention this. But it is important. Okay. Now, here, by the way, is where the confusion on zero singular values comes in. So it's up there and let me explain it in this context. Now, let's just take a matrix A that's in R like seven by seven. And let's say it's rank six. Okay. So the SVD, or the compact SVD, or economy SVD looks like this. It's U, that's six Y, and you get a six by six diagonal, and then you get a V transpose, which is going to be six by seven. Okay. V is seven by six. Okay. Now, this is what happens in that case. And the singular values are sigma one through sigma six. However, when a matrix is square a lot of people will talk about sigma seven, in this case, and they'll say it's zero. And there are actually applications where you want to talk about that. Because, in fact, what you might be interested in is what the minimum gain of this matrix. Well, it's rank six. It's got a null space. The minimum gain is zero. And then people will talk about V seven. V seven is, of course, nothing but, in this case, describes null space of A. So it turns out while sigma one, V one, U one, these things are sort of unambiguous. But if someone writes something like sigma min, you have to be kind of careful. You mean the minimum of the positive ones or you mean the actual minimum, in which case it would be zero or something like this. So and things like this you just have to ask because it just depends on the context as to whether a person's referring to the smallest positive one, or the smallest one period, one zero. Of course, if things are full rank or whatever there's no issue here. There's no ambiguity. Okay. So let's look at the geometric interpretation of how this works. So this is a full SVD. That's to make it simple to do the interpretation. You have A equals U, sigma V transpose. We know exactly what orthogonal matrices do geometrically. They either rotate, or they reflect, or something more complicated, but equivalent. They preserve angles, they preserve distances, they preserve norms. Right. So it's an isometry. Anything related to distances, or angles, or whatever, is preserved by V and U. Now, like rotations. Okay, so this says to operate by A on a vector you carry out three operations. First, you multiply by V transpose, then sigma, then U. So the first one is something like – and I mean rotate loosely in the sense of – of course, a rotation is given by an orthogonal matrix, but it could also be a reflection or some combination of the two. Okay. So the first thing you do is you rotate. Then you stretch along the axis by sigma I. Now, you put in the differing gains. So far there's been no difference in gains anywhere because if you multiple by an orthogonal matrix it's actually complete isotropic in the sense that the gain is one all directions. Now you put in the gain and isotropy. So you put in the unevenness in the gains now. By the way, some people call this, I've heard dilate and, I don't even know how to pronounce this, but it might be dilatate. Does anyone actually know how to pronounce that? For a long time I thought it wasn't a word. But then I found it in enough books that I realized it was a word, and then I thought it was like a weird variation. Maybe something like from the U.K. or something like that. You know just some weird thing. But it turns out no, it appears to be a word. I don't know if there're exact synonyms or not, but anyway. But you'll hear this operation when you stretch or scrunch something along the axis is a dilation. That makes sense to me. Dilatation, actually, I heard people say it. Even people who seem to know what they're talking about say it, so I suppose it is a word. Okay. You also zero pad or truncate, because when this middle action, here, actually is going to map something of different – it's going to map of different sizes. You zero pad or truncate, depending on whether the output is bigger than then input or smaller. So that's what you do. And the final thing is you rotate by U. So this means that we can calculate what happens if you have a unit ball. You can calculate exactly what happens. Take a unit ball. First, multiply a unit ball by V transpose. If you take a unit ball and you apply an orthogonal matrix to it, nothing happens. I mean, the individual vectors move. This point might move over here. But the ball is invariant. The ball is the same. So anything that's in the unit ball, if the image of it is the same unit ball. Not at the detailed level, not on a vector-by-vector basis, but on the set basis, it's the same. Okay. Now, you take this ball and you apply a dilation to it. And in this case, it's a dilation along the axis because you multiply one here, there's a singular value of two, and a singular value of .5. You take the two, that's a long the X one axis, or whatever, and you stretch it by two, and you scrunch it by a factor of two and X two. Okay. Now you take this thing and you rotate by this output, these output directions, and you get this. And you can see all sorts of cool stuff here. You can see, for example, that, indeed, you won with the largest gain output direction because the unit ball and all the things that went in here, were things of norm one. The biggest that any of them ever came out is that guy and that guy, by the way. So those are the two points that came out with the largest possible gain. And look at that, sure enough, they're aligned with U one. And what input direction, by the way, ended up here? What input direction was it?

Student:[Inaudible].

Instructor (Stephen Boyd):V one, precisely. So V one is some vector, here. Say that might be V one. V one got rotated to here or here, by the way. It wouldn't matter, it got rotated either to here or to here. And those are the two points, on this ball, that suffer, or on this sphere, that suffered the largest expansion here, and ended up here. Then it got rotated over here or here. So that's it. So it says that the image of the unit ball, image means you overload matrix vector multiply to apply to sets. You multiply a matrix by a set of vectors. You just multiple all the vectors and you get a new set. That's what this is. And it says that the image of the unit ball is an ellipsoid and the principle axis's are exactly sigma IUY. Okay.

Now this allows you to understand exactly the game properties. And let's do an example. For fun let's do a couple of examples. In fact, let's start with this. Suppose the singular values are 5, 4, and 0.01, there's your singular values. What does the image of the unit ball look like? What does it look like?

Student:[Inaudible].

Instructor (Stephen Boyd):It's a pancake. Exactly! Slightly oblong pancake, but it's a pancake for sure. Everybody agree with that? And this says something like this. There is an input plane, let's say it's a mapping from R three to R three, it says there's an input plane, which is a plane where this thing kind of has a high-gain, like between four and five, and another gain, you might even call V three, you might even call that like an almost null space, because while AV three is not zero, that would qualify for full no space membership. It is not zero. But it comes out attenuated by a factor of 100. Okay. So it's something like almost a null space. In fact, this is exactly the kind of ideas you want to have when you do this. In fact, you'll find that all of these ideas about null space, range, all that stuff, it's wonderful, you have to understand all of it. However, in practice, these things are quite useless and quite silly, very silly, and very useless. What the SVD will do is it will give you quantitative ways of talking intelligently in the context of some application about things like null spaces, and ranges, and things like that. So here the range of this matrix is R three. It's on two period. Up until, maybe, the third week of the class it's on two period. Now, in this context, depending on you know the details of the application, you could say that matrix actually has a range which is two dimensional. It's a lie. It has a three-dimensional range. But it's third direction is so feeble, the gain is so feeble, that, again, depending on the context, you might say for practical purposes that matrix is rank two. Okay. So this is what this is going to allow you to do. We'll see a lot of examples of this so this will become clear. Okay. So that's the pancake. What's this? What's the image? If the singular values were 5, 4, 3.9, what would the image look like? Just roughly.

Student:[Inaudible].

Instructor (Stephen Boyd):Egg, sure. Yeah. Maybe not even quite an egg. No, maybe an egg, I don't know. Something like an egg, right. So basically it says that there's one cross section that looks kind of like an egg. Maybe not even as much as a football or something like that, but an egg. That's that. Okay. How about this?

Student:[Inaudible].

Instructor (Stephen Boyd):It's what?

Student:[Inaudible].

Instructor (Stephen Boyd):A cigarette toothpick. Okay. I'll take toothpick. Maybe I miss heard that. Did someone say cigarette? Okay. Yeah. No. I'll take toothpick. Yeah. So it's a toothpick. That's what that is. And basically, actually, what you can say is, we'll see this quite precisely soon, you can actually say in this case that matrix is nearly rank one, depending on the application. I want to stress that. Matrix is nearly rank one depending on the application. So I want to stress that. So that's the idea. Okay.

So let's see how the SVD comes up in estimation and inversion. Y equals X plus V, where V is the measurement noise, like this, and X is something you want to estimate. Now, probably the best way to really get more deeply than we've gotten into this, although, to tell you the truth, the stuff you know is going to work quite well in practice and go quite far. So what you would do is you might do something like this. You might do least squares, here. You might take Y, choose to estimate X, you would take something a dagger Y, okay, which would be A transpose A inverts A transpose Y, and you would say that that's the X that minimizes the discrepancy between what you actually measured and what you would have measured had the perimeter been X had, or something like that. Better models, well, maybe not better, I don't know, different models are obtained by looking at probability and statistics. So here you assume V has some distribution. But there's another model, and the other model says this, that this noise, I don't know anything about this noise, but I'm willing to give you a bound on its norm. Okay. So some people call that a norm bound model or something like that, and some people call it a worse case model or something, and it's a lot more forgiving than a statistical model. We're not looking at statistical models so it doesn't matter, but a statistical model basically says V is random. So V is not out to get you in a statistical model. In this one, V can be out to get you. V can include someone intentionally trying to mess up your measurement. So V can actually include things like somebody intentionally trying to jam you. It can include something related to X. For example, V could include quantization error, in this case. So it'd be another model for quantization error. Something like that. But anyway, that's the model where you know nothing about the noise except you have a norm bound on it. Okay. Now, let's put an estimator in, a linear estimator, and let's make it unbiased, that mean BA equals I, and you know what that means, it means that if you form X minus BY, if there were no noise, you'd get prefect reconstruction no matter what X is. So this means it's perfect reconstruction with no noise. Now, the estimation error is this. If you form X hat, that's what you guess, minus what X really is, it's BV. Now this makes sense. We've seen this before. The key now, is to do this, among left inverses of A. What you want is a small one. Why does it want to be small? It wants to be small because B is what amplifies the noise. So B does two things. It maps a measured vector into an estimate, but it also maps measurement noise or error into estimation error. So you want B small. B can't be that small because, after all, BA is I. So for example, you can't take B as zero, as an extreme case, obviously. Okay. Now, we can then analyze, using this model, the set of possible estimation errors and that's an ellipsoid. Actually, there's a more refined ellipsoid you can get that's smaller but we'll just do this one. So here you can get a more refined ellipsoid but for now, we'll just this. We'll just say that if you get BV and V ranges in some unit ball, then in fact, there's an ellipsoid, which is BV, as V ranges over all vectors of norm less than alpha. This defines an ellipsoid, and we're going to call that B uncertainty ellipsoid. Note that has nothing to do with Y. Actually, that's the key to the more refined estimate. But this is fine for now to give you a rough idea. Then it says that this is where the estimation error lies, but an ellipsoid's original model is symmetric, so you can say X is X hat minus X utility, that's X hat minus, it's N, X hat minus the uncertainty ellipsoid, but minus and plus ellipsoid are the same because an ellipsoid is invariant under negation. And that says that the true X has to align an uncertainty ellipsoid, this E uncertainty, it's centered at the estimate X hat, and you can say exactly what the insert into the ellipsoid is. B here, at the singular values of B gives you the semi-axis of uncertainty ellipsoid. So you want B, if, for example, these were the semi-axis of B, then we can talk about what it means. In this case, if these were the singular values of B, therefore, these are the semi-axis lengths, when you multiple them by alpha of the uncertainty ellipsoid, in this case, very different. This one says your ignorance is pretty much isotropic. It means that when you guess X hat, where X can be compared to where X hat is, it's uniform, it's kind of your ignorance in this direction is kind of, you know, it's not too much different from your ignorance in that direction. And if you want, you can imagine a GPS system or something like that where you're estimating position. And it basically says, I guess if someone forced me to guess where I am, I'd say I'm here. But if someone says how far off are you, you say, well, I can be, you know, plus/minus three millimeters this way, it's about equal in all directions. Everybody see this? If, however, this is the story, it gets really much more interesting. This says well, in fact, this is in an estimation context this is singular values of B remember. This says, in that context you would say, well, in two directions I have uncertainty on the order of plus/minus four millimeters, let's say. In another direction I've nailed it. I have way, way better. Better, I'm much, much better, estimate of the uncertainty. So in this case the uncertainty ellipsoid is a pancake. Remember for an uncertainty ellipsoid small is good. This is better still. This says, in two orthogonal directions I have a very, very high confidence in my estimate. But in one direction, it's really way, way worse. So that's what is says. Okay. So that's the picture. Okay. So that's the idea. Okay. So you can look out, for example, the worse error is the norm of B, for example, and it turns out, in fact, that the best thing, again, by this measure, is actually suing the least squares estimators. So if you use least squares estimator here, in fact, you're uncertainty ellipsoid will be given by this formula will be as small as it can possibly be. So if what you care about is uncertainty ellipsoid, or if you believe this analysis and so on, then once again, our friend least squares is the best estimate you can come up with. In other words, if someone says, no, I'm not going to use least squares I'm going to use some other fancy thing, you'll get an uncertainty ellipsoid with some other left inverse of A. You'll get an uncertainty ellipsoid that's actually bigger in all directions. That's what will happen. And that comes out this way, that's B least squares, B least squares transpose less BV transpose. And that's that. So once again, we see that least squares is actually a good thing to do. So let's look at this example, our navigation using range measurements. So here I have four measurements that estimate two positions and we look at two positions. One was the just enough measurements method where you take the first two measurements, you say, look, I've got two unknowns, all I need to two measurements, I'll use the first two, you don't even need the last two range estimates, don't even need it. So you form this left inverse like this. It's got a two by two block and a block of zeros. And then we'll compare that to least squares, which is X hat is A dagger least squares. That's a two by four matrix. A dagger is this matrix that blends your four ranges into your position of your X and your Y position. Okay. And from the same example, here's what we get, this is where alpha equals one. In the first case, you get this. And in the second case, you get this. Now, what is correct here, is that if you were to translate this over so they have the same center, they'd lie on top of each other, in fact. Oh, sorry. This one completely covers that one. Okay. So that's the picture. Now, there is something a little bit confusing about this and I'll say what it is and it has to do with the fact that our analysis here can be sharpened a little bit once you have Y. This idea that this set, this uncertainty ellipsoid, the idea that this set contains X minus X utility, that's completely correct. However, this does not depend on Y. Note it's independent of the actual measurement you made. It turns out, once you have the measurement you can actually get a smaller uncertainty ellipsoid. And it's not very hard to show. I won't do it now. But now we're going to explain why, for example, if one estimator says you have to be in the shaded set, another one says you have to be in this shaded set, then for sure you have to be in the intersection. And the intersection of those two is some weird thing that looks like this. It's this little guy here. Everybody see that? And then you'd say, wow, so although this was not a good method, it actually did give us information because it ruled out, for example, all these points over here that this model allowed you to keep. It turns out that if you do the more subtle analysis of uncertainty ellipsoid, you get something that's the same shape as this but it's smaller, like it's scaled down. Something like that. So, okay. I just mention that because it's, well, I confused myself this morning when I looked at this. That's what it is. This doesn't matter. The main point is you can understand uncertainty and how a measurement system works now by looking at the singular values of B. That's what you can do. Okay. I think what I'll do is I might actually, I'll skip this thing, which is just a completion of squares argument, that is a method to show, and if you have any left inverse it's going to end up with BV transposed bigger than the B least squares B least squares transpose. So I'm just going to skip that because it's just a completion of squares argument, because I want to get on to one other topic. Now, what we're going to do is – I guess, I can make some other comments about, I'll make just a couple more points about estimation and inversion. So in estimation and inversion, let's go back to that and look how that works. You know Y equals AX say plus V, like that. Now in some cases, in fact, I'll just tell you in a case where, I don't know, that I was involved in, and it was geophysical inversion. So X had a dimension around 10,000, something like that, and Y was a bunch of magnetic anomaly data, and it was something like 300,000 or something, if these numbers are not right, the story's going to make the right point. So 10,000 variables, 300,000 measurements like, no problem. In fact, that sounded like a 30 to 1 ratio of measurements to perimeters you want to estimate. That's a pretty good ratio, right, 1.1 to 1 is not so great, 30 to 1, that's sounding pretty good. You've got about 30 times more measurements than you need. Everybody cool on this? Okay. So they'd say, no problem. A is full rank, by the way. And so you would imagine you would form this. You would form that. Something like that. Okay. And in fact, you can do this and you would get things. Things would come out and it would be shown in beautiful color that would show you the substructure, the presumed substructure and things like that. So that would be it. Okay. No problem. Now, this matrix A, I can tell you what its' singular values were. Okay, its' singular values were this. Actually, how many singular values does it have if it's 300k by 10k and it's full rank? So how many singular values does it have?

Student:

[Inaudible].

Instructor (Stephen Boyd):It's 10k, 10,000. So I'll just jump into the story. It had about 20 significant singular values. After that they dropped off precipitously and were way small. They were like below the noise floor. Okay. Everybody see what I'm saying? So the sad thing, at that point, people had to accept the following: They paid a lot of money to fly an airplane with this squid, you know, magnetic anomaly detector around. They got

300,000 measurements, 30 times more measurements than they had on those, 30 times. Which you would think would be way, that's a lot of measurement redundancy, 30 to 1. And it was very, very sad, because they thought they took 300,000 measurements and in fact, they didn't know it, but they actually took only 20. So they thought they were in the 30 to 1 measurement to data position, but in fact, they were in something like a 1 to 500. So what it meant was this X hat, though very pretty, very nice when rendered in color, basically, was completely meaningless. Now, you can estimate about 20 perimeters in X if you wanted, but certainly not 10,000. Okay. So this is an example of the kind of reasoning or understanding you would have when you fully absorb the idea of SVD. And it's sort of this step beyond what happens when you first just sit around and just talk about rank, and range, and null space, and things like this. This matrix has no null space. There was no arrangement of subsurface, there's no X for which AX is zero. None. None. So in fact, without the noise you can get X exactly, in fact, by many methods. Because there's lots of left inverses of A. This is your understanding as of week 2.5. Okay. Now, it's much more subtle and it basically says for all practical purposes you found A was full rank, it was actually like rank 20. You have a question?

Student:

[Inaudible].

Instructor (Stephen Boyd):I'd like to say yes, but the answer is no, I don't know why. You know, I can make up a story though, about it. It'll be pretty good, too. Yeah, I could make up a story. I could probably even get the number 20, I mean, with a long enough story. Right. But no, I don't know why that is. It just is. So that's a very good question. Actually, it's because of that that you'd want to then – once you've come to the realization that although you have 300,000 pieces of data, you actually only have 20 measurements, independent measurements.

The next question is what are you going to augment it with? Do you want to drill a couple of exploratory wells, you want to do magnetic anomaly, gravitation anomaly, or do you want to set up some explosives. These are all getting other different measurements. And it would be interesting there to find out which one of those would augment these to give you. And all you do, completely trivial, is you add some more rows to A. Any measurement is a row to A. You add more rows to A and find out what happens to your singular values. And if you go from 20 non singular, you know, significant ones to 50, you just added 30 more measurements.

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, absolutely! Yeah. The additional measurements, beyond the number of the perimeter you're estimating, that goes into smoothing, blending, averaging, this kind of thing. I mean, that is, the matrix we just looked at, I don't know if you remember it, I remember it, the two by four matrix, I mean, that kind of says it all. You should think of a left inverse in that case and actually, I love the names, sensor fusion is one, it's a great name, sensor blending. And basically, it's a fat matrix kind of with all relatively small numbers. It takes all of those measurements in, scrunches them together, and gets very good measurements. So beyond any measurement beyond 10k, would have gone into averaging out the noise and stuff like that. I don't know if I answered your question. As a polite way, he said, no, that's what he said, but oh well, that happens. Okay. Our next topic is, also, a topic on which there are entire classes, like whole ten week classes, and it's a huge area. It's the idea of a sensitivity of linear equations to data error. So I'll say a little bit about that. It's not something we've looked at in this class. In this class, when you use metlab, for example, which in turn uses LA pat, metlab does nothing, right. When you type A backslash V, it doesn't actually matter what's happening, but something like the equivalent of forming A transpose A inverse A transpose times B is happening. And you know that that's not done in perfect precision. It's done with floating point numbers or whatever, and that means that the numbers that come out might not be exactly right. And you're use to that because, I guess, when you solve some problem and you look at the X is supposed to be equal to Y, you'd take the norm. You'd look at X minus Y and then entries would be on the order of A minus 15, and that'd be your hint that what you're seeing is just the artifacts of round off errors and stuff like that. That's a huge filed. It's call numerical analysis. And we'll see a little bit of it now, about how this works. But singular value becomes positional central to the whole thing. So let's just do a square matrix, just for fun. Y equals AX. Well, of course, that means X equals A inversed Y, end of story. Not much more to say. Actually, there's going to be more to say, and it's actually quite interesting. Now, suppose you have a noise or an error in Y. For example, Y equals Y plus delta Y. Now, in numerical analysis, delta Y is like a floating point round off error, for example. So there it's really small, right, it could be on the order of, you know, 1E minus 10, or, I mean, it depends on the context, but we're talking errors out there in the 10th, 15th, 20th digit. Right. Actually, in an engineering context, you want to imagine delta Y, it's generally much bigger. If Y is a measurement, for example, and you use a 12 bit A to D, that's only 4,000 numbers. So what that says is, the fourth digit, did I get that right, 4,000, yes, the fourth digit is absolute nonsense, if I have a 12 bit A to D. If I measure something with 12 bits, anything, it doesn't matter what it is, and then it says basically that whatever the true Y is, I don't know, but a delta Y was added to it that was on the order of one in 4,000. Okay. By the way, that's pretty good. A lot of high-speed stuff operates with much smaller precisions like 8-bits and things like that. Okay. So when you're doing engineering, if you're doing economics delta Y is probably on the order of Y, basically. I mean, it's like friends in economics say, but never on record, you know, when we talk about accuracy of things, they'll say things like, well, I'll tell you the truth, if we get this sign right, we're happy. But they'll never say that on the record. I don't know why. It has to do with the sociology of the field or something. Anyway, so they're happy of they get the sign right. That means that this delta Y is on the order of Y. But anyway, all right. Well, if Y becomes Y plus delta Y, X, obviously, becomes X plus delta X, where delta X is A inverse delta Y. That's simple enough. And that's says that how much X has changed, because Y changed, is simply less – the worse case change is norm A inverse times norm delta Y. That's it. And what that says is the following, if the inverse of a matrix is large, and I mean large in norm, then it's says small errors in Y can lead to large errors in X. By the way, it does not say does lead. I guess, do would be English, actually, but anyway. It does not say it must lead. That's not the case. For you to actually suffer the largest increases here, delta Y has to be in the largest gain input direction of A inverse. That's what has to happen. Okay. But unless you can rule that out, it's possible, and it says you can get larger. And now, this says, basically, you really can't solve for X given Y with small errors. And that means A can be considered singular in practice. Okay. So I mean, we can do a quick example here. And by the way, this clears up something a bit odd, which I will explain now. Take a matrix that's singular. So it's N by matrix and it's singular. Then you ask someone here's a measurement, Y equals AX. I've measured it. You know, this Y equals AX, and X and Y plus V, so there's a small noise. V is delta Y here. That's it. So that's your measurement system. And then you say, please estimate X. And someone who's taking the first two weeks of [inaudible] will say, you can't, your A is singular. Of course, I can't estimate X, it's ridiculous. In fact, you know if it's rank N minus one, there's a whole line of things with no noise, give you the exact same answer. It's impossible, so forget it. Get yourself a better measurement setup. Okay. So you go back into the lab, and it's a fact, actually, that if you'd take a matrix A and you add, if you pick sort of entry at random or if you add a small number to any random number to the entries, with probability of one, the matrix will be non singular. So you go to the lab. All you have to do is find some physical measurement thing, tap on it with a hammer. I guarantee you A is now non singular. So a very gentle tap, just tap it once, I'd say, it's taken care of, no problem. And they look at A, A changed in the fifth digit. And I say, check it, it's non singular, and it will be. By the way, it will absolutely be non singular. Okay. Now, and I'd say, go ahead, and then I'd walk away. But actually I should walk away very quickly because it's obviously not going to work. A inverse, A had a singular value just exactly zero. After I tapped the measurement system with a hammer, A had a singular value that was one E minus eight. Okay. A inverse now has a singular value that's ten to the eight. The norm of A inverse ten to the eight and that says that, if you go down the pad here, it says that small errors in Y can be amplified by as much as a factor of ten to the eight. Okay. So if you wanted say, a couple of digits of that, you know, if you wanted X to be accurate to like, you know, to .01, no problem, Y has to be accurate to ten to the minus ten. Okay. Which is not with a 16-bit A to D is going to give you or anything like that. Okay. So I mean, this all completely obvious. You know, it would be idiotic if taking a little rubber hammer and tapping on a measurement setup, you know, or if I told the pilot in the magnetic anomaly thing to just do another thing and I'd pushed on the control stick or something like that, and that made it non singular. Anyway, it was non singular there, anyway. But that doesn't affect anything. In fact, it's singular value analysis that tells you this. Okay. So what this says is that a matrix can be invertible, metlab, which is to say LA pack, will happily invert the matrix for you. By the way, if it gets too close to singular, then LA pack will issue a warning and say something like R con warning matrix is singular to working precision. At that point, you're so far out of it, it means that basically, with double precisions it's finding out. But the bad part for engineering, in fact, for all applications, is – I guess it's the good part, is that the really evil ones are when you invert it, you see entries like ten to the nine, ten to the twenty, you know something's fishy. But the worse part is when it's kind of plausible and it's giving answers that's plausible but completely wrong. That can happen, actually. Okay. So – all right. Now, I think I'll mention one more thing and then we'll quit for today. What you really want to do is it doesn't really make an sense to say that the norm of delta Y is – I mean, if I ask you, if norm delta X is .1, is big or small, well, it depends on what X is. If the norm of X is 100 then the norm of delta X is .1, it's pretty good. That's like a .1 percent error. If the norm of X is .01 and norm delta X is .1, that's terrible. That means that your delta X is ten times bigger than the X you're interested in. So you should really be looking at relative things that develop errors. Now to do that you do this, you say, if Y is X and norm Y is less than norm A, norm X, and therefore, if you work out delta X over X this thing is the relative change in X. That's less than equal to the norm of A times the norm of A inverse times the relative change in Y. And this number comes up all the time. The product of the norm of A and the norm of A inverse, that's called the condition number and it's traditionally called kappa of A. And it's the maximum singular value of A divided by the minimum, always bigger than one. And this is read informally something like this. It says that the relative error in solution X is less than the relative error in the input data times the condition number. So the condition number is what amplifies or can amplify relative error. So that's the picture. Or, again, there's a misspelling here. But in terms of, if you take the log two of this, and again, very roughly, you'd say something like this. The number of bits of accuracy in the solution is about equal to the number of bits of accuracy in the data, that's the long two of this thing, minus log two kappa. So when you solve Y equals AX for X given Y, you lose accuracy. By the way, if the condition number is one you lose no accuracy. Matrices that have condition number one, I'll give you an example, would be an orthogonal matrix. They have condition number one. All singular values are the same, gain factor all directions equal. That's why people who do numerical work love them. And by the way, it's also why a lot of transforms you'll see, like DCT, FFT, or DFT, a lot of transforms you'll see, actually, are orthogonal. I mean, there's many reasons for it, but one nice part about it is when you use transform like that, and then the inverse, and stuff like that, you actually are not losing bits of accuracy. Okay. So this is the picture. So we'll continue from here next time.

[End of Audio]

Duration: 78 minutes

IntroToLinearDynamicalSystems-Lecture18

Instructor (Stephen Boyd):Let me start with a couple of announcements. One's half announcement, half apology. The current homework, I think the last problem requires material that we haven't covered yet, but I'm going to cover it right now. I sent an email out about that, so I hope that – but there were a lot of people panicked, asking me yesterday, how can I do this and this sort of stuff. I guess for a lot of you, this didn't bother you.

The other announcement was that someone pointed out that last time, I said something about homework eight being – I think I said it was due Thursday, wrote down Friday or something like that. For full inconsistency, we should've had the web site make it due Wednesday or something. So I didn't achieve full inconsistency, but near. Of course, in that case, the rules are simple. The precedence rules. The web site is right.

Then I think that worked out perfect. Usually what I say is wrong. That would be the least reliable. What I write would be slightly more reliable, but I hope that didn't cause any confusion. The other thing is just to make sure you know, we posted homework nine. Obviously, that's the last homework, so that's posted now. We're going to finish up what is essentially the last topic in the class today. We'll finish it.

We'll do another bunch of material, which is very interesting, but it is nothing but applications of this stuff we've done so far. Any questions about last time? If not, you can go down the pad. I guess I've been trying to motion to you that you're focused way off in the wrong place there. You're going to want to move over here.

There we go. So if you go down to the pad, we'll finish off our coverage of SVD. So we'll go to matrix solving square system of equations. Y equals AX, so you have N equations, N unknowns. That's Y equals AX, and of course, the solution is nothing by A inverse Y. So that's the solution.

Now we're going to look at what would happen if Y were off a little bit. Y might be off a little bit for many reasons. I could be as innocent as a floating point round off, or it could be that Y is measured with finite precision, which, of course, is always the case. Is that twisted at an angle of ten degrees for you? Okay. Just curious.

Getting closer. Well, we'll just not worry about it. It's still twisted, but that's okay. So we'll look at what happens when Y varies. Of course, if Y varies a little bit, then X will vary a little bit, and the change in X will be A inverse delta Y. Last time, I think, I pointed this out, but if you have a matrix, which is invertible, nonsingular, but where the inverse is huge – and of course this is exactly what you'd get if you had a matrix which was, for example, singular, and then you perturbed it slightly to make it nonsingular. You will have a matrix that's now nonsingular, but it’s inverse if going to be huge.

Now, when I matrix has an inverse that's huge, you can solve Y equals AX, no problem. That's what it means to be invertible. Here's the problem right here. It says the tiny errors in Y can become extremely large errors in delta X. So that means that it may mathematically be invertible, but it is of no particular practical use, unless you're willing to certify that there are no errors in Y out of the 15th digit or something like that. So that's the difference.

If you want to bound that, you can say that the norm of delta X is going to be less or equal to the norm of A inverse times norm of delta Y. So in this sense, it says that when you see that the norm of the inverse matrix is huge, that's a hint that small perturbations in Y can lead. It does not mean it's guaranteed to. It means can lead to large changes in X.

Now, when you're looking at a perturbation, for example, delta Y or delta X, as to whether or not it's big or small, a far better measure than simply the absolute number is the relative error. Relative error is something like this. It's delta X over X. So if this is, for example, 0.01 delta X over X, you can basically just say that means it's off by say, one percent. The same here.

For example, if you have a ten-bit A to D, when you measure Y, assuming Y's a measurement, then this would say over here – this number here would be less than or equal to something on the order of 0.01 percent. That's two to the minus ten, roughly. Okay.

Now by noting that Y in norm is no bigger than the norm of A times norm of X, and then simply dividing this inequality by the previous one, you get a very, very important inequality. We looked at this last time. It's this. It says that the relative error in X is no bigger than the relative error in Y times the norm of A times the norm of A inverse. Now, this comes up all the time, norm A, norm A inverse, and it's got a name. It's called a condition number. It's sometimes written cond A. So that's – it's a very famous quantity. It's often denoted using kappa, and you can write it in terms of singular values easily because the norm of A is, of course, the largest singular largest value of A. The singular values of A inverse are one over the singular values of A. Therefore, the largest singular value of A inverse is one over the smallest singular value of A.

It's the spread in the singular values, the range. By the way, geometrically, this is an N isotropy. That's what this is. If this is two, it says that the gain of the matrix varies no more than a factor of two with direction. That's the meaning of this. It says that the image of the unit ball under A will map into an ellipsoid that is not too skewed. No semi axis is more than a factor of two if two is what this is – just two, period. No semi axis is a factor of two longer than any other semi-axis. That's what this says.

So this, you can think of the condition number as a measure of being nonisotropic. In other words, it's how much the gain varies as a function of direction. That's exactly what the condition number is.

A couple things to note about it. You will have some homework problems on this, but some are obvious. One is that this number is always bigger than one. So it's always bigger than one. The second is the – that's obvious because it's the maximum singular value divided by the minimum singular value. Another factor is it's homogenous. If I multiply A by ten, the condition number doesn't change. So the condition number in no way changes. That makes sense because we're doing relative error here.

See, if I multiply it by ten, each of these goes up by ten, or if you like, this goes up by ten, that goes down by ten, and it preserves the condition number. So this normally interpreted something like this. The relative error in the solution, X, is less than or equal to the relative error in the data, Y, multiplied by the condition number. That's a [inaudible].

However, in the '60s, maybe early-'70s, things like this weren't appreciated deeply enough. As it spread, it became, in fact, almost a little counterproductive because this sort of became almost a religious thing. In other words, people would simply say, oh, condition number's too high. Forget it. In terms of, this is just an inequality, and in fact, there's lots of cases where, due to special structure in A, this actually works. You can have a condition number of ten to the eight, and actually, the solution's are computed reliably. In other words, there's actually no change in the relative error.

Of course, there you have to look – this is just a bound, and there you have to look more into the actual properties of A. But there are many problems like that. In fact, some that were abandoned, there were methods abandoned in the '70s on the basis of large condition number. So people would say, oh, the condition number of those matrixes are huge. You can't do that. Everyone knows that's bad practice. It turned out, actually, the solutions were being computed quite accurately, but these methods were then out of circulation for 30 years and then rehabilitated five, ten years ago.

So a looser way to say this is you take the log base two of both sides, and you actually get something that's – it's a very good way to put it. It says something like this. It says that the number of bits of accuracy in the solution is about equal to the number of bits of accuracy in the data minus log two of the condition number. Let me say a little bit about what that means. What I'm saying is something like this. If this number is, let's say, one percent, if that's 0.01, it basically says, I know X to one percent. If I know X to one percent, it takes about – oh, somebody help me out. Eight? Seven? Six and a half. Okay. So that's six and a half bits. I should've taken 1,000, should I?

So let's make this 0.1 percent. That one I know. That's two to the minus ten. So if this number is 0.001, then this thing here, it basically says, is about ten bits of accuracy in these. That's not quite right because this is norms of vectors and it's not the same as the norms of the individual entries and the [inaudible] entries, but roughly speaking, we'll say there's about ten bits of accuracy in this at that point. Certainly, the 15th bit in any component of delta X is meaningless. That is certainly the case here.

So if you accept that definition of what it means to have the number of bits of accuracy, then this is actually correct. What this says is that when you solve linear equations, when you invert equations, what happens is the following. I should mention, by the way, the exact same thing works for mapping X into Y because note that the condition number is independent. You can swap A and A inverse and it's the same. So in fact, multiplication by A, in fact, that same analysis holds. It gives you exactly the same decrease, potentially, the same bound and decrease in accuracy in solving. So A and A inverse are kind of the same from that point of view.

What this says is in terms of this bound, you can only decrease accuracy. The only matrixes that will not decrease accuracy are ones with condition No. 1. Condition No. 1 means sigma max equals sigma min. That means all singular values are equal, and that, in fact, only occurs if a matrix is a multiple of an orthogonal matrix. It's actually kind of obvious because you write U sigma V transpose. If, in fact, all the numbers on the diagonal of sigma are equal, which would happen if the condition number is one, sigma is actually – it's something like sigma max, which is the same as all the sigmas, times I, this thing.

That comes out, and it basically has the form sigma max I times UV transpose. That's going to be an orthogonal matrix. So this is why people who do numerical work and people at one end love orthogonal matrixes. They'll do anything to – so in their algorithms, they're multiplying for orthogonal matrixes. For that matter, also solving equations with orthogonal matrixes.

It's not just because it's easy to invert an orthogonal matrix because it's the same as the transpose. It's not just that. It also has to do with liability. So various transforms you'll hear about DCT, the Discrete Fourier Transform, all sorts of the common transformations you'd see actually have this form. The other people who love it are people in signal processing for the same reason, although the values of these things are off considerably. If you're doing numerical analysis, you're worried about errors in floating point numbers. So you're worried out there at the 14th, 15th digit. If you're doing signal processing, the third digit is probably suspect, but it's the same principle. That's the idea.

So the way you'd say this is if a matrix has a small condition number, you'd say it's well conditioned. It's called poorly condition if kappa's large. Whether it's small or large, that depends entirely on the context. If around you are people who do numerical analysis, and the types of errors they're worried about are things like floating point round off, but that's all they're worried about, then a matrix with a condition number of 1,000 would actually, in many cases, be considered well conditioned, from a numerical analyst point of view. That would be well conditioned because at that point, that's not a big enough number to have you worry about how floating-point errors are going to propagate forward.

If you're doing signal processing, or you're doing anything in an engineering context, then a condition number of 1,000 means that you can potentially lose ten bits of accuracy. It means you can start with 14-bit signals, and you'll end up with something with four bits of accuracy. Or if you start with a 12-bit signal, you'll end up with something with two bits of accuracy, at which point, you're kind of right at the boundary – so you're at the "why bother" boundary because you're just computing stuff that may not make any sense.

So poorly conditioned, if you do numerical analysis, that starts maybe to round ten to the six, goes up to ten to the eight, things like that. Ten to the six and ten to the eight in signal processing or estimation or statistical context, we have a name for a matrix like that. It's called singular, meaning there's just no point. Anything you compute, if the number ever had anything to do with any actual real measurement or something like that, certainly can be complete nonsense. They might be right, but that's because this is just a bound here.

But it says that you cannot get a mathematical argument at what you're computing – it could just be noise, and artifact of your ADD or something like that. By the way, the same analysis holds for lee squares. If you have a nonsquare, you have sigma max over sigma min. Here, this is simply a dagger is put here, and it's exactly the same. So the same story analyses these things.

Like I said, you can take an entire quarter that's on this topic and related topics at Stanford. There's multiple quarters that you can take that does this kind of stuff. This is very important to know.

Our almost last topic in the singular value decomposition has to do with low-rank approximation. Actually, I mentioned this way early in the class. We talked about rank, and I said, well, if a matrix is low-rank, you can write it as B times C. You can make a skinny/fat factorization of the matrix. We talked about applications of this and what it would mean. For example, multiplying by A, if you have this factorization, is way fast now. So if this were a million by a million matrix, and I factored it as – first of all, I couldn't even store a million by a million matrix. Let's start with that. Much less could I multiply a vector by it. But if it happened to have rank ten and I factored it as million by ten, ten by a million, no problem. Not only can I store it, I can multiply by it, and the speed-up factor is just something absolutely unimaginable.

So we talked about the idea of what does it mean if something's low rank. That's just one application. At the time, I said this would become much more powerful when you have a method for actually calculating – by the way, we found a method for calculating B and C this way. QR factorization was one way. So QR factorization did it right off the bat. So QR gave us a skinny factorization.

I mentioned at the time that later, we're going to have a method that does this. Then you have something like this. This is actually really interesting because it turns out there are lots and lots of matrixes whose rank is a million but which are almost rank ten. You already know what that means. It means you have a million by a million matrix, and if you work out singular values, they're all positive. Therefore, its rank is a million.

But if, in fact, the singular values of a million by a million matrix are big, up to ten, and the precipitously drop, but remains positive, it means that matrix is almost rank ten. We're going to see how that works right now. That's this topic. The use of this is – I mean, there are so many uses, it's amazing, and there are many more, I'm sure, still to find. So that's what we're going to talk about now.

So let's take an N by N matrix with rank R, and that's its SVD expansion here. So that's the SVD expansion. What we're going to look for is a matrix of rank P, which is less than R. We want this to be a rank P approximation of A. How are we going to measure approximation? We're going to measure it by the norm of A minus A-half. That's the matrix norm, so that means it's the worst-case gain of the error.

By the way, there are other ways to measure – there's other norms on matrixes. You'll encounter one, I guess, like on upcoming homework. Another one is something called the Frobenius norm. The Frobenius norm is nothing, but it's actually this. I'll write it this way. I think this works. It does. So this says string A out as a vector and treat it as a vector in RMN. This is nothing but this. It's the sum over I and J of the entries, separately. It's that. You'll have a homework problem on that, but basically it says treat the coefficients of the matrix as a vector, and take the ordinary Euclidean norm. So that's another one.

But this is the matrix norm. Here's a solution, actually, for both cases. It's this. The optimal rank, P, approximation of a matrix, A, is given by this. You simply truncate the SVD. You write out the SVD. You gather P diads. If you want rank P, that is, in fact, the optimal rank P approximation. This norm, A minus A hat, what's left over is – since A hat is the first chunk, it's the head of the SVD expansion, what's left is the tail of the SVD expansion. That's this. This matrix here has norm sigma P plus one.

So that's actually quite beautiful because it gives you an interpretation of actually what sigma three means. It's actually quite beautiful, sigma three. This gives you a perfect interpretation. It says it's the error by which you can approximate A by a rank two matrix. That's what sigma three means. We'll get to that in a minute.

Now, this is completely intuitive. It's not obvious, but it is – of course, there are a lot of things you have to remember that are intuitive and false. That's actually why you're here. If everything intuitive were true, then we wouldn't need to do all this stuff. This makes sense. It basically says something like this. You rank the diads in order of importance. The sigma gives you order of importance. You take as many as you can afford in your rank budget, which is P.

This gives you the rank of P, approximately. There's lots of applications of this immediately. It says here that if you have to multiply by a matrix many, many times, let's say in some real-time communication system or in anything else, you need to multiply by a matrix many, many times. What it says – for example, that might be some forward simulation. Who knows why you need to multiply by a matrix. Let's say you do.

It says what you might want to do is look at the SVD of that matrix. That's offline. That could take minutes or a day. It doesn't really matter to compute. You look at the SVD of A, and you decide how many singular values you feel are significant for your application. What kind of error will you accept? Then you form the leading – you choose a rank P, you get a rank P approximation like this, like a rank 10 approximation. Now you have a method of multiplying by A, approximately. If you truncated anything, you're no longer multiplying by A. By the way, you can bound the error.

You can never be off in a worst-case sense by more than sigma P plus one times the norm of what went in. So this is not just approximation. It's approximation with a guaranteed bound. So you already have lots and lots of applications of this, of getting a low-ranked matrix.

For example, how do you get a rank-one approximation of a matrix? What's the best rank-one approximation of a matrix?

Student:[Inaudible].

Instructor (Stephen Boyd):How'd you get it?

Student:[Inaudible].

Instructor (Stephen Boyd):Rank one means I want to write my matrix A is BC transpose, like that. I want to write it as an outer product. How do I get the best outer product measured?

Student:[Inaudible].

Instructor (Stephen Boyd):You'd take the largest singular – you'd take U one. In fact, it's this. Here, I'll just write it out. It's sigma one, U one, V one transpose. That's it. That's the best rank-one approximate of a matrix. Let me ask a question. What would be the properties of the singular values of A under which it can be well approximated by a rank-one matrix?

Student:[Inaudible]

Instructor (Stephen Boyd):What is it?

Student:[Inaudible].

Instructor (Stephen Boyd):That's it. So if, in fact, sigma one is much bigger than sigma two, sigma three and all those, and those are small. By the way, what is the exactly condition that makes A rank one? Then there's sigma one equals zero, and if you – well, I use the loose interpretation of sigma two. Sigma two, sigma three and so on are zero.

But if sigma two, sigma three are small, it says that will be well approximated by rank one. Actually, I'd encourage you, in anything you poke around in, to look at these things. It's absolutely worth your while. There are shocking things that occur all the time. So when you work out a matrix that means something, some mapping, if you're doing something, look at the singular. In fact, it's just good practice to know them and look at them all the time, just so you know what's kind of going on.

Also, you'd be shocked. A lot of matrixes you think you've measured a lot, or you think you have a lot of degrees of freedom or a lot of control or something like that, you have an airplane and you put all sorts of thrusters and things like that on it and little flaps and vector thrust, you think you are way over actuated. A quick check of an appropriate SVD will tell you, actually, whether or not you really have eight or twelve actuators. You might only have three or something like that, roughly, but this is the kind of thing that would tell you that. Okay.

So let's prove this. We're going to show this in the case where the norm is the matrix norm. I should mention, this is from maybe 1920 or something like that, roughly. I could be wrong, but I know that it can be traced – I think it's associated with a name, Jan or something like that. It's also a mainstay of a lot of semi-modern statistics. Semi-modern statistics are statistics after Fischer in the '20s, but before the most recent one based on a lot of L1 and convex optimization methods. But that's not the story. So semi-modern was in the middle, that'd be '60s through '80s, and a lot of that is based on low-rank approximation, SVD and all this. There, it's called PCA.

I should say, this is actually a highly nontrivial result. There are basically no other examples where any optimization problem involving rank constraints can be solved. Absolutely none. The most minor variation on this problem, and it's impossible to solve. I point this out because some things are close to solvable. Others are not. For example, if you say – if you had any other condition on these, it's just no one knows what the rank approximation is. It's just this one isolated result.

So let's look at the proof of this. It's actually quite interesting how it works. It works like this. Let's let B be any other matrix whose rank is no more than P. Then let's look at the dimension of the null space at B. Well, that's got to be bigger than [inaudible] to N minus B because the rank of B plus advection of the null space is going to be equal to N. That's the preservation of rank. So in other words, if your rank is small, that says that your null space – if you have an upper bound on the rank, you have a lower bound on the dimension of the null space. They add up to a number, N, so it's kind of obvious.

What we're going to do is this. I'm going to take V1 through VP plus one. These are from the SVD. These are the P plus one most sensitive input directions. They're mutually orthogonal. This here is – I'm going to give a name for this. It's actually very important to get the right intuition here. Certainly, span of V1 is simple. That's a line, and it is the input directions which have the highest gain for the matrix, A. Span of V1 and V2 is a plane. It is the plane on which A has the highest gain. Now, that's rough, and you have to say exactly what that means in the right way, but that's sort of the idea.

So this is something like the P plus one dimensional subspace along which A has the biggest gain. That's what this is. I'm just giving you an interpretation. Nothing of what I just said is going to be needed in the proof. It's just to interpret what this is. Now the dimension of this is P plus one.

Very interesting, we're in RN, and we have the following. I have one subspace that's null space in B with dimension N minus P, and another subspace with dimension P plus one. They have to intersect not just at zero. In fact, all subspaces intersect at zero, but they have to intersect at a point that's nonzero. So there has to be a unit vector Z in RN, which is in both this subspace and this one. Actually, if that were not the case, then if I take the vector sum of the two subspaces, then I get a subspace of dimension N plus one in RN, which would substantially decrease my credibility. That's worse, even, than saying Thursday and writing Friday.

Let's see what it means. This says the BZ is zero, but Z is in the span of this thing. You can actually see what's going to happen now. It basically says – remember, what I want to show is that basically, B is not a good approximation of A. So B actually annihilates Z. But Z also happens to be in this high-gain subspace. So because Z is in the span, it's going to get amplified by A by some minimum amount. That's going to give me a bound on how good you could be. So A minus BZ, well, BZ is zero. That's AZ, but AZ is this.

Z actually, here, only comes into the first P plus one of these, like that, but this thing, here, sum VI transpose Z squared from I equals one to P plus one is actually the norm of Z squared, which is one. The reason is Z is in the span of V1 through VP plus one. So that's what happens here. So these numbers add up to one. These are non-negative numbers that add up to one, and this is a combination of sigma Is from one to P plus one squared.

So that's certainly bigger than the smallest one, which is sigma P plus one squared times norm Z.

This says – I found a vector Z whose length is one, but when multiplied by A minus B, it came out with a norm sigma P plus one, at least or bigger. That says that the gain of the matrix, A minus P, is bigger than sigma P plus one. I should mention that probably you've done that on this homework a couple of times, but I'll just mention that. If I have a matrix A and I say, what is the norm of it? If it's bigger than two by two, then you'd say, I can't answer that. I need a computer, or something like that. Then the person says, no, give me a bound or something on it, roughly. Then there's one thing that always works. If you find any nonzero Z and you just calculate this, that's a lower bound on the norm of A. But I think maybe you've done that on the homework that's due today. You've seen arguments like this. That's a lower bound on the norm, so that's the argument I'm doing here.

So this establishes this low-rank approximation idea, and this allows us to say all sorts of stuff. It actually gives a beautiful interpretation of sigma I. It says that the Ith singular value is the minimum matrix norm distance to a rank I minus one matrix. So it's quite beautiful. That's what it says. For example, sigma one – have I got this right? Does that look right? That's right.

So if I ask you for – I'm looking at this. Is anyone else confused, or is it just me? It looks completely wrong. Let's just go through it and see what it really is supposed to be. Is it N minus I plus one? It's this? I'm just confused? It happens occasionally. Actually, more than occasionally. Okay. Let's just try it here. Let's try sigma two.

So sigma two is the minimum two a rank – this should be N minus one or something matrix, actually. I'm – is this right? What? What I have here is right?

Student:[Inaudible].

Instructor (Stephen Boyd):It is right? Okay. Sorry. My confusion. It happens. We talked about this before. Sigma three, you want to know what is sigma three, the interpretation is it's something like – if sigma three is small, it says you're very close to a rank two matrix. That came out right, didn't it? That's consistent with this. I wonder why I was confused. I was just temporarily confused. All right.

Very important one is if you have an N by N matrix, the minimum singular value is the distance to the nearest singular matrix. So that's what that is. So that's what it means. You can see, again, if you have a very small, singular value for a matrix that's invertible – a square matrix that's invertible, many ways to say it. One way is to understand and say, your inverse is huge. That's one way. Another way to understand it now is you are very close. That matrix may not be – it is singular. It's not singular. It's not singular, but it's very close to a singular matrix.

So if you have a minimum singular value of ten to the minus eight, it says that matrix is not singular. But I can perturb that matrix by another matrix, which is on the order of – it has a norm of no more than ten to the minus eight, and my new matrix will be singular. That's what it says. So that's the interpretation here.

Let's look at an application, and I'm going to do one more application after this to finish up. So the first – that one, you just modeled simplification. This is kind of obvious, but suppose you have something like one equals AX plus V, where A is in 100 by 30, and it has singular values of ten, seven, two, 0.5, 0.01, and they go all the way down here. That's sigma 30 right there, and it's 0.0001.

Now, norm X is on the order of one. Let's say this unknown noise here has norm on the order of 0.1. So that's my model. It's supposed to be rough, so don't worry about the exact numbers or anything like that.

This is actually quite interesting because if – well, we can do some quick analysis here. Geometrically, we can say the following. If norm X is on the order of one, and you don't know anything else about X, let's propagate X through A. Well, you can imagine a unit ball being propagated through A, and you're going to get an ellipsoid. In fact, a very good name of that ellipsoid would be the signal ellipsoid. In communications, that exactly what you'd call it. Is the signal ellipsoid. It shows you kind of what would be the possible received signals if the input varies over a unit ball.

By the way, this might come up through some transmitter power constraint or something like that. So this is quite close to a lot of real problems. So you get this ellipsoid, which is a signal ellipsoid. Now, that ellipsoid looks like this. It's got some big semi-axis, ten, seven, two, 0.5, 0.01. What's interesting about it is this. To that ellipsoid, that's the signal. But it basically says that then the signal is corrupted by a norm of 0.1. What you can see immediately is that basically, all of these signals are sort of lost. They're corrupted by the noise. They're bigger.

In other words, this thing, you can imagine as a ball of a radius 0.1. Then you have this ellipsoid, which actually has positive volume. It's got some really thin dimensions. It's got 26 thin dimensions. In fact, 26 of the dimensions are much thinner than this. What that really tells you is for all practical purposes, this A is rank four. This make sense. You can even talk about the signal to noise ratio. You can talk about the signal to noise ratio here. It's 1,000,

Down here, you're down to a signal-to-noise ration of about five to one. Here, it's the other way around. It's one to ten or minus 20 decibels. Whatever you want to call it over here and so on. So you basically say that for all practical purposes, this simplified model here is going to be absolutely equivalent. This might be depressing because this might have been a measurement system, and you might've though you had a 3.3 to one measurement to parameter you want to measure advantage. Turns out you're wrong. It's the other way around. You actually measured four things, even though you thought you measured 100. This make sense?

Don't read too much into it. This is sort of the typical application. Now, I want to do one more application that I realize this morning should've been in the notes. I don't know why it's not. I guess it will be next year, depending on if I remember to put it in. It's this. I want to go back to the beginning. Let's rewind back to the third week of the class. I'm going to ask you this. Suppose I had a bunch of vectors, A1 through A – let's just make it specific though. So it's A100, and these are in R10, okay? So I have 100 vectors in R10.

By the way, these could be snapshots of something I've measured. Who know what they are? It could be ten stock prices over 100 periods. It could be anything. It's just a block of data. Then I'm going to ask you the following question. I could've asked you this question week three. I could ask you the following. How would you determine if these 100 measurements, these 100 vectors in R10, actually live in a subspace of dimension three? First of all, let's find out what that means. Let me ask you this. Could any human being detect that?

Totally out of the questions. Absolutely out of the question. If I showed you ten numbers, unless it's stupid like, for example, the first number is always the same, like it's always one. But let's assume it's a generic case. I just show you – I say, look. Here's some prices. 100 long, 100 last trading days. Here it is. We need comments. No human being can look at it and go, whoa, those are in a three-dimensional subspace. Can't do it. Out of the question.

How would you do it, computationally?

Student:[Inaudible].

Instructor (Stephen Boyd):QR. Exactly. You simply stake this as a matrix, and you run QR. You know what'd happen? You'd pick a first one, second one, third one. You'd run the QR that doesn't dump core when the matrix is less than full rank. You'd run a modified QR. You'd run it, and I would run through 100 vectors, and it will have generated a rank only three – you know, Q1, Q2, Q3. By the way, the implication of this is very interesting. You would say, you know, that pi statement was really interesting. In terms of it varies in a three-dimensional subspace. By the way, what would that mean? What would be the interpretation? Suppose prices vary in a three-dimensional subspace? What does it mean?

Student:[Inaudible].

Instructor (Stephen Boyd):You can do what?

Student:[Inaudible]

Instructor (Stephen Boyd):There you go. Yeah. That's a perfectly good way to say it. It basically says that instead of just treating these as a bunch of random – it would say, this looks like random ten vectors. No human being can look at ten vectors and say something was fishy about them. What this would say is that this analysis would say these prices depend only on three underlying factors. It would allow you stunning abilities.

For example, given – I guess the right way to say it would be three – seven. If I gave you seven of the prices, you could predict the other three. Did that come out right? Yeah? Okay. That's what it is.

So you want to know what's the practical consequence of knowing this fact? It's amazing. It's basically, given seven prices, you can predict flawlessly the last three. I think that's right, isn't it? Other way around?

Student:[Inaudible].

Instructor (Stephen Boyd):Wow, that's funny. I had a double-expresso this morning. Wow. I'm back to three shots. That's it. Not all the cylinders are firing. Well, it's even more stunning, then. Given three, you can predict the rest of the other seven. But what I said, technically was true, just for the record.

Now, do you imagine that stock prices are exactly in a three-dimensional subspace? That was rhetorical. No, of course not. Do you imagine received signals in some multiple-antenna system are in three-dimensions? Of course not. But now, you now know how to detect if these vectors are almost in a subspace of rank three or if – well, let's see. The range of these is definitely R10, but you can now detect whether the range of these is almost R3 or something like that. So somebody tell me how do you do it?

You just take the SVD of this matrix. You just take the SVD of this data matrix, and what will you see if you take the stupid data and you type SVD of A? What would tip you off that that data is almost in a three-dimensional subspace?

Student:[Inaudible].

Instructor (Stephen Boyd):You'd see three substantial singular values, and then you'd see a nice strong drop. Sigma four of the sigma ten would be small but positive. By the way, what that would mean is if you took these points and if you think of it geometrically, it says take these points and make a cloud in R10. I don't know how your R10 visualization skills are, but I'll tell you what mine are. Since I appear to not even be able to get – did you notice that both the mistakes I made today were of the same nature? It's an N minus I confusion? If it continues, I'll go to a neurologist, and they'll find out there's some lesion in some part of my brain.

Let's move on. Maybe that was the lesion talking. Let's move on here. If you were to visualize these in R10, if you could visualize vectors in R10, you could say, oh, let me see some more price data, and you could go, hang on just a minute, and you could go like this or whatever. And you'd be like, wow, that's in almost a three-dimensional subspace, but no one can do that. But if you could do that, you would see a cloud of points.

The cloud of points would actually – the SVD tells you something about this. It tells you that in some directions, the cloud would have a big extent. But in seven directions, it would be quite small. That's what you'd know. So that's how you'd be able to say that this thing is roughly in R3. By the way, how would you then actually calculate the so-called factor model in this case? How would you calculate an orthonormal basis of that three-dimensional subspace of the price subspace?

Student:[Inaudible].

Instructor (Stephen Boyd):First three Us. End of story. First three Us, U1, U2, U3, that's an ortho basis for that subspace. I guess it's not in the notes because it's basic and obvious, and this is just the definition of what SVD is, but still, this is the kind of thing you can do and will do using the singular value decomposition.

So that actually finishes our last real topic in the course, so that's it. The next topic we're going to look at has to do with this idea of controllability and solvability. It's interesting stuff. Actually, more than anything, what it shows is that once you understand everything up till today's lecture, everything else is nothing but a trivial application of it. So all other topics and stuff like that just follow from everything we've done up till now, period.

So essentially, the class is over now. You can treat the other topics as just interesting applications. Just cool stuff you can do with these ideas. Yes?

Student:[Inaudible].

Instructor (Stephen Boyd):I did.

Student:[Inaudible].

Instructor (Stephen Boyd):Are you asking what's the difference between the –

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, what's the relationship? Yeah, I can say a little bit more about that. If these price vectors – let's make them price vectors. If these are price vectors or price change vectors or return vectors or whatever, if these are price change vectors like that, and they're not exactly in a three-dimensional subspace here, and you run the QR factorization on it, what do you get?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, you get a Q that's N by N. The thing's full [inaudible]. You get a Q that looks like this, and you get and R, which looks like that. That's it. That's your QR factorization. It doesn't really tell you a whole lot in this case. Whereas the SVD, if you write out the full SVD, you get the same thing as this. The same shape, not the same thing. So the SVD is what would allow you, whereas the rank-revealing QR would not.

By the way, if sigma four gets small enough, it will trigger your rank-revealing algorithm to decide that it's rank three. That's a parameter in that method, and it has to be set right. It would normally be set way low. It would not be triggered by anything that came from data.

So that's the difference between a rank-revealing QR. And rank-revealing QR is something we looked at week three. So that wasn't yet quantitative. Week three, when someone said rank to you, it means what rank means. It has a mathematical definition. In that case, this would be rank ten. This real price would be rank ten. Now that you know about things like the singular value decomposition, rank is a fuzzier concept, although I should say something about that.

Rank by itself means rank. It's the mathematical definition of rank, and if anyone says, what's the rank of this data matrix, the answer's ten. If you're going to use a fuzzy definition of rank, you must qualify it. You must say something like – if someone says, what's the rank of that? You could say ten. You can say, but it's awfully close to rank three. It's well-explained by three factors, and it's almost perfectly explained by four factors, but you can't just say the rank of this is three. Everybody see what I'm saying?

Of course, once you're doing this kind of stuff and people on the streets, they say things like that all the time. Singular means singular. Non-singular means singular. Rank means rank. Null space means null space. However, if you have a couple of singular values that are really tiny, you can call that the almost null space. If you were writing or there were lawyers present, I would not call it the null space. You can say it's the practical null space or for the purposes of this problem, for the purpose of this application, it's the null space or whatever.

But I'm wondering all around, and I'm not answering your question. The question was, what's the difference between A equals QR and Q equals U sigma V transpose? Was that the question?

Student:[Inaudible].

Instructor (Stephen Boyd):Yeah, roughly? Today? I'm going to take that as encouragement. Okay. I will tell you what the relations are. Let's do this case where it's full rank and fat or whatever. That's orthogonal and that's orthogonal. V transpose and R have the same size. That's pretty much it. You can say a few more things. There are some subtleties you can say about the size of the elements about the diagonal of R and sigma, but they're much more fancy and advanced. Basically, there's not much you can say about the two.

There are, but they're more advance topics. They're not simple things. Did I answer your question, which was not actually your question, but you were polite enough to suggest that it was close enough? Yes? Good.

Had that been your question, would I have answered it? Good.

Next topic, unless there's any other questions about SVD? Did you have a question?

Student:[Inaudible].

Instructor (Stephen Boyd):That's an orthonormal – the first three vectors in U is an orthonormal basis for the three-dimensional subspace that the As are approximately in. I think that came out right. Is that the question? You might ask, actually, then what the three Vs are. V1, V2, V3. That's interesting. I'll let you think about it. That's what it is. It's actually quite interesting in this context of, say, if these were price changes or return vectors, let's say. You might ask what V1, V2 and V3 mean. Now, if they're transpose, then they're like this. You would get something that looks like this and then something that looks like that. There's some sigmas in there, but you can shove the sigmas either way you like.

In fact, go ahead. Let's go ahead and answer it. What do you think these are? Let's say this is time, so the index represents time. It's closing-day returns. What do you think these three are? They're very interesting. Do you have a guess?

Student:[Inaudible].

Instructor (Stephen Boyd):Well, this basically says what's driving the returns on the prices. There's just three underlying factors. They're not totally random and unrelated. There's three underlying factors. This is the time history of those three segments of the economy. That's what it is. You can look at the first one, and you can say, oh, that's related to interest rates. I'm totally making this up, by the way. There's probably people in here who know much more about this than I do. I know enough to know that's what that is. So this actually gives you the time history. If this means time, this is history.

In signal processing, these would be the actual recovered signals, and this would be the subspace in which those signals lie. So that would be the example. But these are best understood by just doing problems. Yeah?

Student:[Inaudible].

Instructor (Stephen Boyd):No. It would not be. If you did the singular value analysis of this, and all ten were significant, sigma ten, for example, was not small enough that you could say anything about it, that means, basically, that this cloud of points again, if you were able to do R10 visualization, kind of extends about the same in all directions. That's kind of what it says. It would be like, no – by the way, this might be a good thing, it might be a bad thing. It's bad if your goals it predict the value of seven of these components when given three. That's bad, but for other applications, it might be good.

In that case, it actually depends – no, it wouldn't be the same. I'll just leave it at that. There'd be no connection between them, no. Yeah?

Student:[Inaudible].

Instructor (Stephen Boyd):These? No. These are the ten vectors, and that's an orthonormal basis. Basically, the assertion is this. Every one of these 100 vectors is very well approximated by some linear combination of these three. That's what it says.

Student:[Inaudible].

Instructor (Stephen Boyd):No. It transposes here. So this thing here is V1 transpose, V2 transpose, V3 transpose, like that. The Vs are 100-long vectors. This theta block is ten by 100, so the V is also ten by 100. So V transpose – oh, it came out wrong. V transpose is also ten by 100. Yes?

Student:[Inaudible].

Instructor (Stephen Boyd):Y, and it would be terrible. So if all the singular values of a matrix are the same – actually, we talked about that earlier today. That is a matrix with condition No. 1, but you can say a lot about that. It should be or was or will be a homework problem for you, I think, if we were on the ball. We may have accidentally not assigned that, but basically the homework problem – it may not even exist, but it would go like this. I'm losing it.

It would go like this. I'm making sense, condition No. 1, only if it's a multiple of an orthogonal matrix. Why is that? Because it looks like U sigma V transpose, but all the sigmas are the same. Therefore, sigma is a multiple of the identity, and there for it pops outside. It really look like lower-case sigma, UV transpose. That's a multiple of an orthogonal matrix. So that would be great. Matrixes like that just bring tears of joy to the eyes of people who do numerical analysis.

For signal processing, that's the best channel you could possibly have because you're not going to lose anything. You thought you had ten dimensions. You pay for ten antennas, ten receivers, they're all paying off. If that were data, and you were analyzing it, you would say, so much for that. If you're analyzing data, it just means, well, it's all over the place. It's worse. It's in a balanced way, it kind of goes all over the place. You haven't figured out any underlying structure in there. I don't know if this makes sense.

If A actually is a channel in a communication system, for example, and it has only three significant singular values, that's bad because basically it says that the gain in three directions is significant, but then after that – so, but you think, but I paid for ten transmitters and ten receivers. What this says is you paid for ten. For all practical purposes, you're getting three.

I think I'm answering questions all around yours. I'm sort of surrounding it. What do you think? I think – what was your question?

Student:[Inaudible].

Instructor (Stephen Boyd):You've forgotten now?

Student:[Inaudible].

Instructor (Stephen Boyd):Yes, that would be correct. If all the singular values of a matrix were the same, then I had a long incoherent discussion of what that means. It means it's a multiple of an orthogonal matrix, right? Then it says – this is not a good candidate for low-rank approximation because basically any approximation you make is going to be terrible. Why? Because the thing kind of pokes in all directions, so approximating it by something of lower rank is going to give you a huge 100 percent error, basically, is what it's going to give you. I think we're done.

So, let's look at the next topic, which, like I said, is an application. It's controllability and state transfer, so we'll look and see what the idea is. So we consider a linear dynamical system – the truth is, we've already done these problems. This just kind of formalizes a bunch of it. These could be homework problems, really. So let's take X dot equals AX plus BU for a Discrete time system over some time interval, TITF. That's initial and final.

Now, of course, these are integers in the case of a Discrete time system, and these are real numbers, and that's an actual real number interval in the case of a continuous time system. So you say that an input is a trajectory. It's a function that maps this interval, the time interval into RN. You say that it steers or transfers the state from the initial state to the final state for that time interval. Then you have immediate questions like, for example, given an initial state, what can you transfer it to at the final time? What can you do? What are your degrees of freedom? If you think of U as a joystick or something like that or a possible input, the question is, what can you pull of over that time interval?

If you can transfer it to some target state, then the question is how quickly can you do it? Can you do it in three seconds? Can you do it in one? What's the minimum time it takes to transfer a given state to a target state. By the way, often, the target state would be zero. Remember, this is all – zero means some equilibrium position like some vehicle, an airplane flying at some trim position.

Another question would be how fast can you bring, after some disturbance like a wind gust – how fast can you bring it back to zero state, zero meaning it's back at the trim condition, for example. It can be an industrial process or something like that or network flows or an economy or something like that. It's all the same, in this class. If you're in those individual classes, it's not all the same.

Another question would be how do you find one that transfers you from some initial state to a final one. Another would be how do you find a smaller, efficient one that transfers you from one state to another. These are the types of questions – you can make up zillions more, but these are the types of questions that we can actually answer. In the Discrete time case, by the way, we can answer it all now, basically. You know all the answers. You don't need to know anything else.

Let's look at reachability. Reachability studies the question of taking an initial state of zero and going to some state, X of T. Then you say X of T is reachable. If it's a Discrete time system, you say in T seconds – sorry, it's continuous time, it's in T seconds. If it is Discrete time, or whatever the time units are, you'd say in T periods or T epochs, is what you would say. We'll let RT be the set of points reachable in T seconds or epochs.

That would be something like this for a continuous time system. It's the set of all N vectors that look like that. This thing here is X of T. When you start it, X equals zero. That's the formula for mapping an input over an interval, zero T, into the final state. You multiply by B, you put the exponential there, and you get that. This is a huge, infinite-dimensional set here. It's the set of all possible inputs that map zero T into RM.

Discrete time system, it's actually much simpler. It's the set of all things of this form where U of T is in RM. So the parameter that parameterizes this description is finite-dimensional. So here, I should say something like this, maybe T equals zero up to T minus one, I think. Is that right? Yeah. So that's what this would be.

It actually has dimension TM, this thing. Okay. These are subspaces, and it's easy to show directly. It's also easy to – basically, it's easy to show directly. It has an interesting implication to say it's a subspace. To say it's a subspace says the following. It says if you can reach a state in three seconds, let's say, and you can reach another state in three seconds, it says you can reach, for example, the sum state. By the way, what input do you think you'd use?

The sum of the corresponding input. You'd have to check that and all that, but it would work out. That's how that works. The other thing is that if you can reach – you have this. RT is a subset of RS, if S is bigger. This says that the points you can reach in 3.6 seconds is a superset of the set of points you can reach in 1.5. I'll write those down, so I don't forget them.

Actually, let's argue that. Do you not believe it?

Student:[Inaudible].

Instructor (Stephen Boyd):No, it's true in Discrete time, too.

Student:[Inaudible].

Instructor (Stephen Boyd):No, that's my next question, but it's a great question. I'll summarize your question. It wasn't a question. It was a statement, like, that's completely wrong. That's maybe not – put much more politely, but that was the question. So let's figure this out. I claim if you can hit a state in 3.6 seconds, you can hit it in – no, sorry. It's one of those days. You have them.

Let's try it again. This is where I'd really like to be able to edit those videos. It'd buffer up real fast. It'd be about 12 minutes. Let's try it again. If you can reach a state in 1.5 seconds, you can reach it in 3.6. Your intuition was that it might not be the case. Someone, tell me how to do that. How do you do it?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay. So first you reach 1.5, and then take your hands off the control?

Student:[Inaudible].

Instructor (Stephen Boyd):Oh, okay. I'm getting two different – you said you want to get there in 3.6 seconds. Get there in 1.5 and take your hands off the accelerator. U equals zero. You take that back?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay. It's formally retracted. The reason that doesn't work is because you get there at 1.5, great. You're at the target state. Take your hands off the thing, and what's propagating you forward? E to the TA. So you're going to drift. 3.6 seconds, you will not be where you want to be. But there's a minor variation which I heard in a kind of – actually, I heard ten answers coming at me, but they were kind of in a two-dimensional subspace.

Sorry. They were. Anyway, the correct way to do it is to do nothing for – do you think I can subtract these? 3.6. That's tough. You do nothing for 2.1 seconds. There you go. I got my stride back.

If you do nothing, normally you drift. But here, you started from zero. You get the distinction?

Student:[Inaudible].

Instructor (Stephen Boyd):You got it. It will not work. In fact, if you don't start from zero, you're absolutely right. If you can get from one place to another in 1.5 seconds, there's no reason to believe you can do it in a longer time period. But if you start from zero, that works. Everybody got that? That's it.

You say the reachable set is a set of the points that you can get to at any time. So if you can get there at all in any amount of time, that's called the reachable set. We can actually analyze this completely, about 15 seconds, for Discrete time linear dynamical systems. This uses all the ideas we know and love at this point. The Discrete time system, [inaudible] AX equals BU of T. Then I can write this. The state at time T, I will stack my inputs over zero up to T minus one, I'll stack them. So that's a big vector in RTM, like that.

I'll just write it out in a matrix, and that matrix looks like that. It's got a name. I guess it's called the controllability matrix. Some people call this the reachability matrix or something, but it's got names like that. The matrix looks like this. It's B, AB, up to AT minus one B. By the way, there should be absolutely no mystery as to what these are. Notice that I chose to stack the Us in reverse time. Why? I don't know. Just so this matrix would come with ascending powers of A, not descending powers of A.

So I've stacked my Us in reverse time. So if you do that, this says – how does UT minus one, which is actually the last input that has any affect of X of T, how does it affect X of T. The answer is just by B. The dynamics makes no difference. This says U of T minus two. That's your penultimate action. It's the action before the last one. It's UT minus two. That gets multiplied by AB. That should make perfect sense to you. B tells you the immediate action on X of T minus one A, then propagates that affect forward in time, one step. So that's what this matrix is. It's a very famous matrix.

It looks like this. By the way, the matrix comes up in other contexts. It's a huge topic in large-scale scientific computing and things like that, except they call it a Krylov matrix there. I'd mentioned this just because you'll see this matrix, actually, not with a vector – V will be a vector in that case. Okay. So in terms of that's real simple. We have a name for it. The reachability subspace at time, T, is actually the range of this matrix. The matrix grows, if I increase T, and now I can ask you any question at all. I didn't need any of this to ask you, but now I can say, I have a Discrete time system. I start at zero. I have you a target state to hit, and I ask you, give me the minimum number of time steps it takes to hit that target.

One possible answer is you'll never hit it. We haven't quite gotten there. How do you solve that?

Student:[Inaudible].

Instructor (Stephen Boyd):Could've done that in week three. How'd you do it?

Student:[Inaudible].

Instructor (Stephen Boyd):Okay. So what you do is this. You take – first I ask you, can you hit the target vector in zero steps? Can you? There is one case you can do that. If the target's state is zero, then the person who asked you this, you go back and say, pardon, you're already there. It's done. Then they'd say, amazing. I'm glad you took that class.

Now, let's say T equals one. Can you hit it in one step? The only things you can get to in one step is of the form BU of zero. That's what X of one can be. So the answer is this. You can hit a target, X sub target can be hit in one step if and only if it's in the range of B. Notice that A plays no role whatsoever. Can you hit it in two steps?

You check if X target is in the range of B and then AB. If it's in the range, done. That's it. You can hit it. If it's not, you can not. So you simply keep increasing these until X target is in the range of them. Make sense? That's it. Everybody – and this is basic, so that's the idea.

That'll give you absolutely the minimum time required to hit a given target state. Now, by the Cayley-Hamilton Theorem, once you get to A to the N, A to the N is a linear combination of I, A, A squared, up to AN minus one. That means, basically, when you get A to the N, B here, you're adding M columns, but for sure they're linear combinations of the previous ones. Actually, Cayley-Hamilton Theorem now has a very interesting implication in a dynamic system.

It says the following. It says that after N steps, the range of RT is equal to RN. So it basically says this. In principle, the set of points you can hit grows with each T. By the way, it need not, but it grows with each T, but once you hit N, it will never get bigger. That's it. So when you take N here, that's called a controllability matrix, and people say that the system is controllable if the controllability matrix is rank N.

That basically says you can hit any point in N steps. That's what it says. Actually, it may not be a good idea to hit it in N steps. You may want to hit it in more, but at least in principle, you can hit any point in N steps. I think we'll stop there, which is a good idea, unless I can think of one last gaff I can get in. But I can't think of any, so we'll quit here.

[End of Audio]

Duration: 77 minutes

IntroToLinearDynamicalSystems-Lecture19

Instructor (Stephen Boyd):– email – for example, if you don't live in the Bay area, you should email us to let us know when you want the final emailed to you. That's the first announcement. And I guess, even for people in the Bay area, sometimes traffic is a big pain or something and in which case this is an easier option. Second announcement is homework nine – we'll post the solutions Thursday so Thursday evening after homework nine is due. And I think we've now responded to maybe 10, and growing, inquiries. I guess there is a problem involving – the title is something like time compression equalizer, does this strike a bell? Vaguely. You look worn out. No? Okay. It's just early. Okay. All right. So we fielded a bunch a bunch of questions about the convulsion, we didn't put the limits in the sum, in the convulsion, but you're to interpret, I think it's W and C as 0 when you index outside the range. So a bunch of – maybe 10 people pointed this out to us or something like that. An important announcement, sadly, I have to leave tomorrow morning to go to Austin. I don't like doing that, but I have to go. So I'm off to Austin, and that means that Thursday's lecture, which is the last lecture for this class will actually be given this afternoon. And I think it's Skilling Auditorium 415 this afternoon, but whatever the website says that's what it is. And that's on the first page, the announcements page. So that's where. If you were are around this afternoon and want to come, please do come. You should know that it is every professors worst nightmare, maybe second or third worst, but it's way up there on the list that you should give a tape ahead and no one would come. This would cause you to give a lecture to no one. It's never happened, but it doesn't work. So at least, statistically, some of you should come. My guess is someone will come. We've had long discussions about this. Several colleagues have suggested that we should do tape ahead's from wherever we are, sort of like a nova show or something like that. So you could say hi, I'm here in Rio and we're gonna talk about the singular value decomposition or just something like that, but we haven't actually approached SCPD to see if they can pull that off, but I do want to do that sometime. Anyway, this afternoon is a tape ahead. Please come, statically. So as long as some of you come. My guess is that some people will come anyway. All right. Any questions about last time or administrative stuff? Oh, I have to say that one of the problems is because I'm actually in between this lecture and then Thursdays lecture, which is this afternoon, I also have to give a talk at NASA Ames so I'm gonna have to leave my office hours early today around noon. I have to be walking out the door by noon. So I feel quite bad about that. In fact, I'll even be gone when you get your final. That might be a good thing. But I'll be back Saturday morning. I'll be on email and I'll be contact, let's put it that way. And I'll be back Saturday. And we have a couple of Beta testers taking it; I think one in about an hour and a half. So someone is gonna debug it for you. It's already been debugged pretty well. Okay. Any questions? Then we'll continue on reachability. So last time we looked at this idea of just reachability. Reachability is the following state transfer problem. You start from zero and the question is where can you go? So it's a special state transfer problem. You start from zero and you want to hit something like, in states base, at time T. And we said that our sub T is the reachable sub space. This is sub space. If you can hit a point in T, seconds, or epics, you can certainly hit twice the point and it's a sub space, if you can hit one point or another, you can hit the sum. So it's the sub space. And it's a growing family of sub spaces. So we'll know exactly what the family is. Actually, we already know for discrete time. For discrete time it's interesting, but it's just nothing but an application of the material in the course. It's basically this. Our sub T is the range of this matrix, CT; this is the controllability matrix at time T. I think I mentioned last time that this matrix, you will see in other courses. I mean, it comes up in, for example, scientific computing, in which case RT is actually called a [inaudible] sub space. I may have mentioned that last time, but [inaudible] you will see that this matrix doesn't come up in just this context. It comes up in lots of others. So this matrix here and I think we discussed it last time, as you increase T it gets fatter and fatter, in fact, every time you increment time, the matrix gets fatter by the width of B. That's the number of inputs, which is M, is what we're using here. So what happens is you have a matrix, you start with B, that's where you the range of B in one step, then the range of B and AB is where you can get in two steps and that was parched very carefully and I guess I shouldn't have said it so quickly. When I said the range of B and AB, it means the matrix B space AB. So it's the linear combination of columns of B plus columns of AB. That's where you can get the two steps together. Okay. Now we noted by the Cayley Hamilton Theorem, once you get to N steps, A to the N is a linear combination of I, A, A squared up to A and minus one and so the rank of CT or the range, does not increase once you hit above N. So for example, the range of CN and plus 1 is also the range of CN. So it doesn't grow. Okay. Now that means we have a complete analysis of discrete time system where you can get starting from zero in T epics. The answer is just this. You can get to the range of CT for T less than N, and then after that, once you hit N, it's the range of C. And C is just CN. That's called the controllability matrix. And the system is called controllable if CN is onto. So in other words, if it's range is RN. So that's the idea. And so you can say, you get something that's not totally obvious, it's this, you have the following. In the discrete time system any state you can reach in any number of steps, can be reached, in T equals N steps. Now, that doesn't mean that's a good idea. We will see why very shortly, but nevertheless, as a mathematical fact, it says that if you can't reach a state in N steps then you can't reach it ever. So giving you more time to hit the step is not gonna help at all. Okay. In the reachable set, that's the set of points you can hit with no limit on time, is simply the range of C. It's the range of this matrix. Okay. Now a system is called controllable or reachable, now, unfortunately there are people who distinguish between reachable and controllable, sadly, so sometimes controllable means something slightly different, but don't worry about it for now. A system is controllable if you can reach any state in in steps or fewer, and that's if and only if this matrix C is full rank. So that's the condition. And we'll just do a little stupid example here is this. You have XT plus 1 is this matrix zero 1 1 0 X of T plus 1 1 U of T, now, we can just look at this and know immediately what it does. It does absolutely nothing but swap the roles. That's the swap matrix, I mean, if you ask me to describe it in English, that's a swap matrix. It simply swaps X1 and X2. The input, and this is the important part, acts on both states the same way. So the point is there's a symmetry in the system. It's just a stupid simple example. There's a symmetry in the system and it basically says that whatever you can do to one state, and I'm arguing very roughly now, it will do the same thing to the other. So that's a hint right there that there's gonna be some things you can't get to. We'll wait and see what they are. The controllability matrix is B, that's AB, and sure enough, B AB is not on two. It's singular. And the reachable set is all states where X1 is equal to X2. So no matter what you do here, no matter how you wiggle, you will never reach a state that doesn't have the form of a number times the vector 1 1. It just can't happen. And it's obvious here you certainly didn't need controllability analysis to see this here. And to be blunt about it, that's often the case in almost all examples. I mean, sometimes you don't know, you actually have to check, they'll be something, and in fact, not only that, but most lack of controllability comes down to symmetries like this. They can do much more sophisticated in large mechanical systems and things like that or after the fact you'll realize that something symmetric in your actuator configurations is symmetric and of course, you couldn't do something after the fact. We'll see actually there's a much more interesting notion of controllability that we're gonna get to of quantitative work. Okay. Now let's look at general state transfers. So general state transfers, that's a general problem. We're gonna transfer from initial to a final time, from an initial state to a final state, and of course this is the formula that relates the final state to the initial state and of course, this is completely clear, that's simply the dynamics propagating the initial state forward in time. That's nothing else. So this in fact what would happen if you did nothing, if you were zero over the interval? This is the effect, I stacked my inputs in a big M times TF minus T1 plus 1 vector and I multiply it by this controllability matrix here. And this gives you the effect of the input, how it changes your final state. Okay. So what this says is this equation holds, if and only if, I'll take X desired to be the state you want XTF to be, so I take XTF minus this is in the range of that because this is in the range of that and there's your answer. So it actually makes a lot of sense. It's actually quite beautiful. It basically says something like this. If you want to know if you can transfer from an initial state to a desired state, then it's really the same as the reachability problem, what you want to reach is an interesting state. You don't want to reach X desired. You want to reach X desired minus what would happen if your initial state were propagated forward in time. That's what it comes down to. Okay. So this is simple, but it's quite interesting. So I guess another way of saying it is something like this. The U, if you want to transfer from T initial to X of T initial to some X desired, it says don't aim at X desired. What you do is pretend you're starting from zero and aim for this point, which takes into account the drift dynamics. Okay. So that's kind of what you want you want to do. Okay. So general state transfer reduces to reachability problem, and now I believe last time somebody asked the following question. We talked about reachability and your ability to get from one state to another, let's say over some fixed time interval. And the question is if we made the time interval longer, can you get to more points? Certainly if the initial state is zero, that's true. If the initial state is not zero, that's false. It's just wrong. So it is entirely possible in general reachability to be able to hit a state from one initial state in four steps, but then in five steps to be unable to hit it. Okay. That's entirely possible. It does happen and so that's entirely possible. Now, there's a very important special case. Some people think of it as the dual of reachability and sometimes people call this controlling, I mean, if you distinguish between reaching and controlling, that is driving a state to zero. So sometimes the problem of taking a state that's non-zero and finding an input that manipulates the state to zero is called regulation and sometimes it's just called controlling. I can tell you the background there. The basic idea in regulation is that X represents some kind of – your state actually represents what we call X here represents an error. It's an error from some operating conditions. So you have some chemical plant, you have a vehicles, you have whatever you like, X equals zero means you're back in some state that you want to be in, in some target state or bias point in a circuit or trim for an aircraft or something like that and then regulating or controlling means there's been a wind gust or something's happened, you're not in that state and you want to move it back to this standard state which is zero. This equilibrium position, which is zero. So that's why it's called the regulation problem or control problem or something like that. And here you can work out exactly what that is, here it turns out this is just zero so it depends on whether or not, and of course, that's a sub space so I can remove the minus sign here. If I give you a non-zero state, let's just even just check that. So how would we do the following? I give you a system, I give you A and B and I give you a non zero state and I ask, "What is the minimum number of steps required to achieve X of T equals zero?" That's the minimum time control problem or whatever you want to call it. How do you solve that? So this is what you're given. I'm gonna give you A, I'm gonna give you B and I'm gonna give you this, X zero. How do we do it? How do I minimize T for which X of T is zero? Let's handle a simple case. If X zero is zero, then we're already done before we started and the answer is T equals zero in that case. Okay. How can you do it in one step? What do you do?

Student:

[Inaudible]

Instructor (Stephen Boyd):It's interesting. What you want to do here is the following. You want to check whether A to the T times X0 is in the range of B up to A T minus 1 B. That's it. I think. Make sense? This is what you need to check and you simply increment T now to check. You try T equals 0, we just did that. You try T equals 1, so you hit AX0; you want to check if that's in the range of this. Okay. Now, if you test this and you get out T equals N and the answer is still no, what do you say?

Student:[Inaudible] Instructor

That is cannot be done. Actually, because of this term, that actually requires a little bit of argument, but that's correct. So that's the basic idea. We have a homework problem that's actually a more, it's actually a more sophisticated version of this. I think. Good. Okay. All right. Okay. Now, again, just applying all the stuff we know, because this is nothing but applied linear algebra. There's nothing interesting here. Let's look at least-norm input for reachability. That's actually much more interesting. So let's assume the system is reachable, although, now that you know about SVD it wouldn't matter if it weren't, but let's assume it is. And let's steer X of 0 to an X desired at time T with inputs user of the UT minus 1. I'll stack them in reverse time. That's just so I can use CT this way. So I stack them in reverse time and I get X desired is this matrix, that's a fat matrix times this is my control, my controls stacked or you could actually call this a control trajectory. That's a good name for that vector. I want to put out one thing about that vector. It runs backwards in time. That's just indexing. I could've run them forward in time, too, but then I would've had to of turn CT around to start A T minus 1B, A T minus 2B….down to B. But everyone writes this as B, AB, A squared B. So time runs backwards in this vector. Okay. Now, in this state C is square or fat and it's full rank so it’s on 2 and we want to find the least-norm solution of that. The norm of this by the way is the sum of the squares of the norms of the components. That’s true actually for any vector. If I take a big vector and I chunk it up, if I divide it up, any way I like, the sum of the norm squared of the partitioned elements is this norm squared to the original vector. So that’s what this is and you just want to get the one that minimizes this. This makes a lot of sense. Some people would call this the minimum energy transfer. That would be one. That’s, generally speaking, a lie. It generally has nothing to do with that. It’s extremely rare to find a real problem where the actual goal is to minimize the sum of the squares of something. They do come up, but they’re very rare. Okay. Well, this is nothing. We know how to do this. So that’s called the least-norm or the minimum energy input that affects the given state transfer. And if you write it out in terms of what CT is, you get something very interesting. CT of course is B AB A squared B and so on and when you line that up with C transpose C, you get B transpose on top of A transpose B transpose and so on and when you put all the terms together you get a formula that just looks like that. There it is. So that’s the formula. And again, there’s nothing here. You’re just applying least-norm from week three in the class. That’s nothing else. But it’s really interesting. First of all, notice that it’s just a closed form formula for the minimum energy input that steers you from zero to a desired point in T epics and it just looks like that. And everything’s here. The only thing in here is a matrix inverse and you might ask, “Why do you know that that matrix is invertible?” What makes that matrix invertible? This matrix in here is nothing but CT CT transpose. It’s a fat matrix multiplied by its transpose. That is non singular if and only if C is full rank. And in that case, it corresponds to controllability. But in the case where it is controllable, C dagger is in fact this whole big thing here. By the way, it’s really interesting to see what some of these parts are. Let’s see what they are. There’s actually one very interesting thing is you see something like this. There’s sort of a transpose here and the really interesting part is that its running backwards in time. So we don’t have any more time left in the class so I’m not going to go into more detail here, but it’s just an interesting observation. By the way, this is related to things like you may have seen in other contexts, in filtering you may have seen single pluses, you may have seen matched filters, which is basically where the optimum receiver is sort of the same as the original signal but running backwards in time. If you’ve seen that, this is the same thing. It’s identical. So this is not exactly sort of unheard of. Okay. Now, this is the minimum input. By the way, these are the things that I showed on the first day, as I recall, you were completely unimpressed. So this is where we’re just making inputs to some, I don’t know, 16 state mechanical system to take it from one state to another in a certain amount of time. They were pretty impressive. We’re just using this formula. Absolutely nothing else. Just this. And all I was doing was varying T to see what the input would look like. To see what it would require to take you to a certain state. This is much more interesting. We can actually work out the energy, the actual two norm squared of this least-norm input. Now, if you work out what that is, I mean in general what the least-norm input is is actually it’s going to be a quadratic form. And the quadratic form is very simple. It turns out when all the smoke clears I’ll just go through all this. When the smoke clears, it’s this. It’s a quadratic form. This makes perfect sense that the minimum energy – let me explain what this is. This is the minimum energy, defined as the sum of the squares of the inputs. By the way, this is the minimum energy. So this is the energy if you apply the input to hit that target state if you do the right thing.

You are welcomed to use inputs that use more energy than this and many exist. Well, actually, unless C is squared, in which case if you hit it, there's only one way to hit it in that, and oh, I'm sorry, C is squared which means there's a single input and T equals N. If C is square there's only one way to hit it so all inputs are minimum energy. But if square is fat, and real simple, there's lots – you can go on a joyride and burn up a lot of energy and still arrive at X desired. That's it. This is the minimum. It's a quadratic form. And that quadratic form looks like this, and it's actually quite pretty. Inside here it's a sum of positive semi-definite matrixes. Now, I know they're positive semi-definite because each term looks like this. It's A to the tow B times A to the tow B transpose because this part is just that. But whenever you take a matrix and multiply it by its transpose, you get a positive semi-definite matrix. That's what you get. So it's a sum of positive semi-definite matrixes. Well, sums of positive semi-definite matrixes are positive semi-definite. And in fact, you can even say this and as a matrix fact, it's correct. When you increment T you add one more positive semi-definite term to this positive definite matrix once T is bigger than N or at some point and that makes the matrix bigger. And I mean now in the matrix sense. So this is a matrix here, which is getting bigger with T, and I mean in the matrix sense. That means, by the way, the inverse is getting smaller. The inverse is getting smaller. That means that the minimum energy required to hit a target in T seconds, as a function of T can only go down. Well, it could be the same in there. It could be the same. Actually, normally it goes down. All right. So it's actually quite interesting here. It says that we now have a quantitative measure of how controllable a system is or reachable. The reachable is sort of this platonic view that says, "Can you get there at all," and this one is much more subtle. It's less clean but it says basically this. It says oh, I can get to that state, no problem. I can get there, but what it'll do it tells you if for some example, getting there is something that takes a huge amount of input, a very large input is required to get there and for all practical purposes, you can say, "I can't get there." So that's the idea. Then we do beautiful things. I can ask you things like this. I can give a target state and I could say that the energy budget is 10 and I can say, "What is the minimum number of steps required to hit this target and stay within my input energy budget?" I could ask you that question and you could answer it by incrementing T until this goes below 10. One possibility is this will never go below 10. In which case, you announce that, well, you can announce several things. You can announce that is too little energy for me to get there no matter how long you let the journey be. So that's one option there. You can actually solve a lot of very sophisticated problems. So what this does it gives you a quantitative measure of reachability because it tells you how hard it is. It also allows you to say things like, "What points or directions in states base are expensive to hit," and expensive means require a lot of control. Cheap means, you can get there with very little control. And it's actually quite interesting. These are lipoids of course, and they basically show that the set of points in states based are reachable at time T with one unit of energy if that's a one. Actually, let's go through the math first and then I'll say a little bit about how this works. So as I said before, if I have T bigger than S then this matrix, that's a matrix in equality is better than that one because the difference between the two is the sum of a bunch of terms of the form, you know, FX transpose between time S and T. So that's what this happens here. Now, you know that if one matrix is bigger than another, the inverse actually switches them. So the inverse is less than the inverse here. Now we're done because if this matrix is less than that, and anytime you put Z transpose Z here and Z transpose here and Z here, this inequality becomes valid. It's an ordinary scalar in equality and it works. And that says it takes less energy to get somewhere more leisurely. So that's the basic idea. It all makes perfect sense. Now, I should mention something here for general state transfer, the analog is false. Absolutely, or is it? Ewe. Wow, and I put all the intensifier up in front, didn't I. Well, I think it's false. But all of a sudden I had this panic that – I think it's false. Let's just say that. That's what I think. I think it's false. I retract my intensifier at the beginning. It's probably false. There we go. We'll leave it that way. So I think with general state transfer, it's false. Okay. All right. I'm gonna have to think about that one for a minute. I'm pretty sure it's false. Okay. Let's just look at an example. So here's an example. It's a 2 x 2 example because that's the only states based I can draw anyway so here's a 2 x 2 example. And here's some system. It increments like this. There's an input, and I want to hit this target state 1 1. I just made it up. There's no significance to any of this. It's all just made up. And what this shows is the minimum energy required to hit the target point 1 1 as a function of time. And you see a lot of interesting things here. You can see that if you hit it in two samples it costs you an energy of over nine. If you say three, you can get there in almost half the energy. I guess it's half the energy if you double, if you say, instead of two steps, do it in four, and so on and you can see. And it goes down. Now, what's interesting is it appears to be going to an asymptote here, which means that to get to that point, with infinite leisure, it still costs energy. Now, I can explain that. That's actually reasonably easy to explain. If a system is stable – someone have a laptop open. So anyway, no never mind, you don't even need a laptop. Can someone work out the item values of this for me? I need a volunteer. Can you do it? Do you have a pen? So he's working on the item values, which he'll get back to us in a minute. I put him on the spot. We'll let you work on that for a bit and then – it's just because you have to write out a quadratic or something like that. So the conjecture is that this is actually – well, no. Cancel the item value thing. What ai was going to say is if this is stable, then in fact, you have to – if a system is stable and you have to get somewhere, you actually have to fight the dynamics to take it out to some place because if you take your hands off the controls, this is very rough. If you do nothing, the state will just decay back to zero. So you're swimming upstream when you're doing reachability for a system that is stable. Okay. Now, if it's unstable, let's talk about reachability. Let's say a system is violently unstable, so basically, all of the eigenvalues for a discrete time system have magnitude bigger than one. So what that means basically is if you do nothing, the state is gonna grow step by step anyway. Now, let's talk about what happens when I give you more and more time to hit a state. What's gonna happen? If I give you, like, a hundred steps and you have a system that's highly unstable or just unstable. If I give you a hundred steps to hit somewhere, what happens is all you have to do is push X of 0 away from the origin. All you do is you push X away from the origin the tiniest bit and then take your hands off the controls and you let the drift, which is the unstable dynamics, bring the system out to where you want to go. Does this make sense? So you kind of work with the different – there, you're not fighting the stream, it's actually on your side for reachability. Does everybody see what I'm saying? So what that suggests is that for an unstable system, as you give more and more time to hit a target, the energy is gonna go down, in fact, it's gonna go down to zero. So we'll get to that now. It is very hard to hit an isotopic target point like that. It is very easy to hit a target point like that. It's very cheap to hit this one and very expensive to hit that one. So the controllability properties are not isotropic in this case. Okay so let's examine this business of this energy going to zero. That is a sequence of a function of T, that is a sequence of increasing positive definite matrixes. And I mean increasing in the matrix order. That is a sequence of positive definite matrixes, which is getting smaller. Now, a sequence of positive definite matrixes that are getting smaller at each step converges just the way a sequence of non negative numbers that are monotone and decreasing converges. This converges to a matrix. That matrix has a beautiful interpretation. It's called P here, that's actually called the controllability gramian, this matrix. And actually it's the inverse of the gramian, but that doesn't matter what it's called. So this matrix comes up and actually it's beautiful. It's a quadratic form that tells you how hard it is to hit any point in states based with infinite leisure. That's what this matrix tells you. And by the way, if the system is violently unstable, P can be 0. That's extremely interesting. So it takes, basically, 0 energy to hit anywhere in a system that is violently unstable. Let me just do a simple example. Let's take B to I and let's A be 1.01 times the identity. It's a very simple system. U just adds to the input. The dynamics is you just times equal the state at each step by 1.01. So basically it says, "If you do nothing, the state just grows by 1 percent each step." That's all that happens. It's a violently unstable system. All the eigenvalues are outside the unit disc. They're all equal to 1.01 and now it's completely obvious that the longer you take, you name any point you want to hit, and what you do is if you take T samples you go back, I guess, by 1.01, you actually find out what input is required to hit that and you take that point and divide it by 1.01 to the T and that's the U that you've set on the first input. That's a sequence of inputs that just kick it out and then let the dynamics take it there. Those inputs will have, as T gets longer and longer, the energy will go to zero and the – by the way, if P is zero, it does not mean that you can hit any point with zero energy. The only point you can hit with zero energy is the zero state. So when you interpret Z transpose PZ, you'd say that that's the energy required to hit it with infinite leisure. It's really a limit. It says that you can hit it. When this is zero it basically says that you can hit that point, not with zero energy, but with arbitrarily small energy by taking a longer and longer time interval. That's what it really means. Okay. Now, it turns out that if A is stable then this matrix is positive definite. That follows up here. If a matrix is stable, well, what it means, is its power, that's A to the tow are going to zero geometrically. In fact, they go to zero at least as fast as the spectral radius, the largest magnitude and eigenvalue of A to the T. So that means this is a converging series. This thing converges to some positive definite matrix. The inverse of a positive definite makes a positive definite and you have this. So if A is stable, you can't get anywhere for free. But if A is not stable, then you can have a zero null space. Zero null space means just what we were just talking about. You can get to a point in the null space of P using the use of energy as small as you like so that's it. And all you do is just kick if a little bit and let the natural dynamics take you out where you want to go. You have to be careful doing this, obviously that this is way it works. So this is actually used in a lot of things. For example, it's used in a lot of what people call statistically unstable aircrafts, so if you look at various sort of modern fighter aircraft, some of the really bizarre ones will actually have the wings swept forward slightly and it just doesn't look right. It just looks like it's flying backwards actually, and it just doesn't look right, and sure enough, it's not right because it's open loop and unstable. That's what they mean by statically unstable. Most other ones are stable. Commercial ones are, at least so far, stable. I think they're probably gonna stay that way, but who knows. So with forward swept wings or statically unstable aircraft, you might ask why would anyone build an airplane, which basically sitting at a trim position, in some flight condition, is unstable. So let's think about what this means. It means things like your nose goes up and instead of there being a force or moment that pushes your nose down, when your nose goes up, actually, there's an up torque and your nose goes up faster. First of all, why on earth would you ever do this, that is the first question. So and this is just for fun. Someone give me a guess. By the way, I made a guess and it was totally wrong when I talked to someone who knew what they were doing.

Student:

[Inaudible]

Instructor (Stephen Boyd):Yes, that's the idea. You want to get a nice snappy ride. Okay. And you do. You get a very – as you can imagine you do. Right. You pop your elevator down a little bit or whatever it is and your nose is now going to go very fast. So is the idea that you can just do it with a small U so it's efficient? Okay. So what's the objective? Well, I assumed it was – I don't know. I actually finally talked to someone who knew what they were talking about, at least on this topic, and they told me in fact why you do this. The main reason, actually, has nothing to do with efficiency or anything like that. Obviously. You want small control surfaces for smaller radar cross sections. So the reason you want small control surfaces, obviously if you're flying at mock two or something like that, you're not really worried about energy efficiency or anything like that. What you want is a small control surface because control surfaces reflect radar stuff. So that's the real reason. And I actually found out how they work. They have, like, five back up control systems because, let's remember, you flip up, but you better be very careful with this, right, and you flip up with a tiny, very small, little subtle control surface that just goes like that. You flip up, and when you get to where you want, you better have just the right input to make you stabilize there and all that kind of stuff because if you lose it, I guess in this case, it's all over in three seconds. It's in under three seconds that whether the pilot likes it or not that explosive bolts go and you're out. So that's the way it works. And the way it works is I think that there were four redundant control systems. So I guess if the first one fails, the second one is all ready to go, if the fourth one fails, you're out the top whether you push the button or not. And that's the way this is and they actually do this. And actually now there's a move to do this for some chemical processes, too. By the way, there's a name for a chemical process that's statically unstable. What would be the common name for it?

Student:[Inaudible]

Instructor (Stephen Boyd):Yes, it's called an explosive. Yes, that's correct. So I don't know if these things are good or bad or whatever, but that's the – and people are doing it. They just said, no, we operate this process at an unstable equilibrium point because it's more efficient in terms of the overall operation. So that's it. All of these obviously require active control to make sure everything's okay. Right. Everything will become –

that's the whole point of an unstable system. Things will become not okay very quickly. There was a question back there.

Student: No.

Instructor (Stephen Boyd):Maybe no? Just stretching. Okay. All right. So. Okay. Let's look at the continuous time case and see how that works. It's a little bit different but there's nothing here you wouldn't expect. And in fact, this allows me to kind of say something that I should've said earlier but that's good. Now I get the excuse to say it. To make a connection between the conditions – there is a question.

Student:[Inaudible]

Instructor (Stephen Boyd):Right.

Student:[Inaudible]

Instructor (Stephen Boyd):Really. It's a homework. I can't do the homework, generally, just like that. I had a discussion once. Some people came to my office and I started explaining something, 10 minutes, dead end. I tried again, dead end. And then after 25 minutes they said, "Do you think it's fair to assign homework that you can't do? And I said, "Yes, absolutely because I said at one point, clearly, I could do it, and at that point, it obviously was trivial then." So all right. So let's answer your question. What was it? I can try, but I'm just – I can't do it. I'm not embarrassed in the slightest, but go on.

Student:[Inaudible]

Instructor (Stephen Boyd):That's a good problem. I wonder who made it up. No, I'm kidding. All right. Okay. So you're given an initial state and you want to steer it, not to the origin, but to within some norm of the origin with what, with a –

Student: Minimum amount of input.

Instructor (Stephen Boyd):– with a minimum amount of input. That's a great problem. Is it continuous time?

Student:[Inaudible]

Instructor (Stephen Boyd):Okay. Fine. All right. So I don't know. Can you solve that? I guess the answer is no. That was a rhetorical question. Let's talk about it. Right. It's safer for me in case I can't solve it. So what happens is you want to – let's fix a time period. Okay. So then it's a linear problem. Right. As to where you can get. So I guess it's sounding, to me, like a bi-objective problem. Am I not wrong? It's sounding to me like one. Right. So the final state is what? Let's just say if you go T seconds, it's T epics, it's A to the T X 0 plus and then something like CT times – I'll call it U, but everyone needs to understand U is really a stack of the times in reverse time. Is that cool? This is actually a sequence of U. The whole trajectory. Right. That's what you got and then what did you want to do? The condition is that this should be less than some number. What was the number I gave?

Student:.1.

Instructor (Stephen Boyd):.1. Good. A nice number. There we go. So we have that. And what did you want to do? You wanted to minimize the norm of U. And then your point is that we never did this, right? Is that your point?

Student:[Inaudible]

Instructor (Stephen Boyd):It seems to be. So we didn't do this. That's true. You can look through the notes and you won't find this anywhere. Any comments?

Student:[Inaudible]

Instructor (Stephen Boyd):What?

Student:[Inaudible]

Instructor (Stephen Boyd):Yes, thank you. Okay. So yeah. We didn't do this. Absolutely true. This is a bi-objective problem. This is a perfect example of how these things go down in practice, right, because basically, you go back and look at like week four, it was all clean. It was, like, "Yes, let's minimize AX minus Y with small x and then we drew beautiful plots and all this kind of stuff, right?" Here, it's clouded by the horrendous notation of the practical application. In this case, the practical notation is steering something from here to there so it doesn't look as clean. But it is the same. So you make a plot here trading off – I don't remember how we did it before, but you would trade off these two things like that and there's an optimal trade off curve here. There we go. I know one thing to do, you could set U equals zero, there, I got one. You could nothing and run up a very small bill here. So how do you solve this? How do you solve this? Anyway, I've already said enough. Are we okay now? So now what happens is you make the trade off curve here and then on this plot what do you look for? I find the point here, which is 0.1 and I go up here and I'm looking for that point and that will solve it, right? Are you convinced?

Student:

Yeah.

Instructor (Stephen Boyd):Okay. So that's it. All right. So it's true. You didn't do that before. But we did things that allowed you do it. So. Okay. Are you happy now? Okay. Good. Okay. Let's continuous time reachability. So how does this work? Well, it's actually in some ways trickier and in some ways it's actually much simpler. It's gonna be interesting, actually. So here's the way it works. Actually, in some ways it's gonna be uninteresting. That's the interesting part about controllability in the continuous time case. Okay. So we X. is AX plus BU and the reachable set of time T is actually now an integral and this, it's parameterized by an infinite dimensional set. It's the set of all possible input trajectories you could apply over the time period zero key. Absolutely infinite dimensional. Okay. Now, it turns out that this sub space is super simple. It's just this. It's actually much simpler than the discrete time case. In a discrete time case you can get weird things like this state you can hit it in five steps, but not four. This state you can hit in seven, but not three. You can get all sorts of weird stuff. I mean, all the weirdness stops. Once you hit N steps, you can hit anything you're ever gonna hit, you can hit. That's starting from zero in the discrete time case. In the continuous time case, it just bumps up to anything you're ever gonna be able to hit, you can hit. You can hit anywhere, you can hit it in one nanosecond, at least according to the model. So it's basically this. You form the matrix B AB up to A and minus 1B, that's the controllability matrix. And it basically says if this matrix is full rank, you can hit this set is all of RN for any positive T. And in continuous time, it says any place you can hit, any point you can reach in any amount of time, you can actually reach infinitely fast. That's what it says. And this makes perfect sense. You have to have your input act over a smaller, and smaller time. And it really couldn't have been otherwise. I mean, it would've been really weird if there was a state here you could reach in three seconds, but not two. That would've been kind of weird because you'd think, "Well, like, what exactly happened?" And in fact, because that's a sub space, it's dimension is an integer, so had this other thing happened, it'd be, like, you know, the dimension of the reachable set would've gone up to equals, you know, T equals 2.237 it would've jumped to three or four. And you think now, "What on earth would allow you, all of a sudden, at some time instance to manipulate the state into some other dimension?" I mean, it makes no sense at all. So in fact, it kind of had to be this way. So this is it. So that's the result. And we'll show it a couple of different ways. Actually, there's a bunch of ways to connect it up here to the discrete time case and see how it works. Now, one way to see that you're always in the range is C is simple. Let's start from zero, E to the TA is a power series, but I could use the Cayley-Hamilton as a back substitute with powers of A starting at N, N plus 1 and so on. I can back substitute powers of smaller powers of A. And I'll end up with this, it says that basically E to the TA is for sure, for any key, it is a polynomial in IA up to A and minus 1 period. [Inaudible] polynomial of A, a degree less than A. Okay. Now, X of T is just this integral, but now I'm gonna plug that in and I get this thing and now I switch the integral and the sum and I get the following. It's the sum from I equals one to N of this. But that is just a number. You could actually work out how these are exactly, but it doesn't really matter for us because that's a number and that's our friend the controllability matrix. So what this says is if you have a continuous time system, no matter you do with the input, and you start from zero, you will never leave the range of the controllability matrix. Ever. Now, we're gonna have to show the converse which is that any point in the range of the controllability matrix can be reached. First we'll cheat a little bit and we'll do that with impulsive input. If we're gonna use impulsive inputs we have to distinguish between zero minus and zero – well, T minus and T plus whenever T is a time when there's an impulse put. So let's just say before the impulse, we'll put zero and we apply an impulse, which is A. It's distributed across the inputs by a constant vector F, that's F1 through FM and it's multiplied by this K differentiated delta function.

That's what it is. And here, the laplace transfer of that, is S to KF. The laplace transfer of the state is SI minus A inverse B is S to the KF. I'll do a series expansion on this, I think that's a called a law expansion. Did I say that at the time? I don't think I did. No, I didn't think I did, but that's what it is. I think we used it to do the exponential. So if I expand this, I take out the powers that are going to multiply the S to the K and I get things like this. A bunch of them look like this and let's look at this very, very carefully. When I take the inverse laplace transform these correspond to violent impulses in X of T. This S inverse is gonna be the first one. That's sort of like a step term. This is all the stuff that happens between zero minus and zero plus. This is what happens right after zero plus. It makes perfect sense. It says that if you apply an input differentiated K times, it has an immediate effect on the state and the state is to move it to A to the K B. But now, you know how to transfer the state to anything in the range of C because if I make an input that looks like this, it's a delta function times F 0 up to a delta function differentiator and minus [inaudible] and I multiply this if I apply this, then X of 0 plus is C times this vector and now we're done. Now if it says that at least using impulsive inputs, I can reach anything in zero time using impulsive inputs. That's what this says. So that's the picture there. And the question is can you maneuver the state anywhere starting from X equals zero. Is the system reachable? If not, where can you get it? Well, you can kind of figure out what it is, but to kind do some of the calculations we can actually work out what it is. You work out the controllability matrix. It's A AB A squared B and you get this matrix here and you look at it for a little bit and you'll quickly realize its rank two. All right. Let's move on to a much more important topic, which is least [inaudible] reachability in the continuous case. It's gonna be very similar, except it's gonna be kind of interesting now because it's gonna be that we'll have this possibility of actually affecting a state transfer infinitely fast. And that's gonna come out of this. Let's see how that works. That's your minimum energy input. If you have X. is AX plus BU and you seek an input that steers X of 0 to X desired and minimizes this integral here. Now, this is not anything we did before. In fact, this has got a norm. People would call this, by the way, the two norm – just the norm squared of U. Okay. But this is not anything you've seen before and when this was discrete time, U was sort of a stacked version and it was big, possibility, but it was finite dimensional. That's an integral, were in the infinite dimensional case here. Actually, it's not anything you need to be afraid of. Some of you, depending on the field you're in, will have to deal with infinite dimensional things. I might even just be in continuous time or something like that. My claim is if you actually understand all the material from 263, none of the infinite dimensional stuff has any surprises whatsoever. Absolutely none. I mean, a few details here and there, some technical details, everything we did has an analog. And a simple, elementary one. Now, don't dress it up and make it look very fancy to justify, I don't know, just to make it look fancy, right, but you'll see the concept for example [inaudible] so instead of calling it something symmetric you'll have a self a joint operator. That's the other thing. You're then welcomed to call linear transformation an operator, which sounds fancy by the way. Or some people think of it as fancy. So you can talk about linear operator and you can find out, for example, a symmetric one can be diagonalized. There are some things that get more complicated, but if the operator is what's called compact, then it's gonna be exactly the same. It's gonna look exactly the same. It's gonna be something to the SVD also works, at least for compact operators. I'm just mentioning this because some of you will go on – if you ever have to do that, I mean, it should be avoided of course, dealing with these things, but if you find you've already chosen or are too deep into a field where these [inaudible] dimensional things do appear, don't worry because I claim if you understand 263 you can understand all of that just with some translations. There are a few additional things that come up that you don't – you'll have continuous spectrum and things like that, but otherwise it's fine. Has anyone actually already encountered these things? I think there's a lot of areas in physics where you bump into these things, so okay. All right. This is your first fore ray into that. So let's just discretize the system with an interval T over and. Okay. And later we're gonna let N go to infinity so that's what we're gonna do. So we're actually not gonna look at first over all possible input signals. We're gonna look at input signals that are constant over consecutive periods of length H which is T over N. So that's what we're gonna do. So we're not solving the problem. So we'll let them be constant and we'll just apply our various formulas from various things. It turns out [inaudible] exactly what we had before. Now, it's finite dimensional and this is now the controllability matrix of the discretized system. And remember, these have formulas, like, AD is E to the H A and BD is this integral here. Okay. And the least norm-input – now, this is all finite dimensional so there's no hand waving, nothing. It's week four of the class. The discrete least-norm input is given by this expression here. Now, if I go back and express this in terms of A using these powers of these things, after all, A is an exponential and powers of exponentials is just the same as multiplying the thing by that, you get something kind of interesting. What happens is BD turns into T over NB, so you get the following. That's this expression here. That's this first expression here. As N gets big, that converges to something that looks like that. Now, the sum is nothing but a ream on sum for an integral and the integral is that. Now, you put these together, in other words, you take this thing and then multiply by the inverse of that. Notice that the N conveniently drops out. That just goes away. So does the T for that matter. And I get a formula, and this is in fact different, it's this, it's B transposed times this [inaudible]. By the way, if you compare this to the discrete time case you will see that it is essentially the same, well, you have to change integrals and things like that. Now, what's really cool about this thing is the following. Now that it's completely and horribly marked up and no one can read any of it, but imagining that you could read it, the cool part is this matrix is non singular as long as T is positive. I can make T 10 to the minus nine and this matrix will be non singular. By the way, it's gonna be non singular, but if you integrate something – again, you have to assume some reasonable time scale and things like that, if I integrate something from zero to 10 to minus 9, that integral is gonna be very small. So that says that this inverse is going to be absolutely huge. And so what this says is oh, I can steer the input, I can steer the state from zero to a desired state in any number of steps. Sorry. In any amount of time I can do it very, very quickly, but it's gonna take a huge input. That's what this says. It all makes perfect sense. It all goes together and it makes absolute perfect sense here. Now, in the discrete time case, you might want to know why breaks down and what breaks down is real simple and it's for a simple reason. Let's see if I can say this and not sound like an idiot. The problem in the discrete time case is the time is discrete. This is the problem. Here, time is continuous. I can make it as small as I like. But here, what happens is I'll decrease T. When T equals N, I'm still safe by Cayley-Hamilton, but the minute I drop T below N, then there will be – I can take T down and at some point, this matrix can become non singular, in which the case, the inverse doesn't work. By the way, if I replace the inverse with a dagger, and make that a pseudo inverse, you get something very interestingly related to our famous homework problem. If I put a dagger in here, I'll get something really interesting. I'm gonna get you the least-norm input that will get you as close as you possibly can get to the desired target. Did this make sense? So that's what C dagger will do. And that's not the dagger from lecture four. That's not CC transpose C inverse C. Sorry. C trans – help me with this one. C transposed – whichever it is. C transposed quantity CC transposed inverse. Yes, that was it. It's not that dagger. It's the general dagger that requires the SVD. So that's what happens. Okay. Now, the energy required to hit a state is give by this integral. This integral from zero to T. And the cool thing about the integral is no matter how small T is, Q is positive definite. It's invertible. And I'm not gonna go over a lot of that, but that's sort of the basic idea. Let's see. And I'll just make the connection to the minimum energy [inaudible]. The same story happens. I have an integral, a positive semi-definite matrixes here. If I increase the time T that you're allowed to use to hit a target, this matrix goes up, this one goes down, and that's the quadratic form that gives you the minimum energy so you have the same result again. Okay. Let's quit for today. For those of you who just came in, I think I announced at the beginning of the class there's a tape ahead. It's today. It's today, 4:15, Skilling Auditorium, but as usual, you cannot trust me. Whatever it says on the website is what it really is. And statically, some of you should come because otherwise I'd be put in the terribly awkward position of giving a lecture to no one. It's never happened. Hopefully, this afternoon won't be a first. Okay. We'll quit here.

[End of Audio]

Duration: 74 minutes

IntroToLinearDynamicalSystems-Lecture20

Instructor (Stephen Boyd):I guess we're on. This is the last lecture – you can turn off all amplification in here. I don't need any amplification here. Thanks. We'll finish up a couple of topics in reachability and I'll just do a couple of topics in observability. It's not a hugely important topic. And then I wanna reserve some time at the end of this class to just say a few things at the very highest levels about what the class is, how all this stuff fits together, and all this sort of stuff, so that's what we'll do. So continuous time reachability we looked at last time, which in fact was I guess for the people here in real time was this morning, and we looked at what's the reachability subspace for a continuous time system, so that's the set of all points you can hit in T seconds starting from zero in a continuous time system. The answer turned out to be real simple in this case. It's not interesting or subtle like in a continuous time system where at each step for a while anyway interesting things can happen. You can sort of hit a few points and some more, and all this kind of stuff. It just all happens at once. What happens is if T is positive, then the set of points you can hit is the range of controllability matrix period. It doesn't change with T if T is positive. What will change of course is the size of the input required to hit a target state if you do this real quickly, as opposed to if you do it on a long leisurely timeframe. So we also looked at this time. If you wanna look at the least norm input for reachability, you're looking for the input that minimizes this integral of the square of the norm of U. And we worked through that by discretizing it, and the formula that came out was this, quite straightforward. It was this integral of a positive semi-definite matrix. The integral itself is positive definite. There's this inverse, and then that's multiplied by this thing over here, which is B transpose time E to the T minus tau. Now this is actually – that's the system running backwards in time. I think I commented on it this morning. And all the parts of this correspond perfectly to the discrete time case. Instead of having an exponential, you have a power. These are just time propagators for example. The other thing I should mention is that this whole thing is really something just like C transpose C C transpose inverse. The difference is C is now like this kind of big complex operator. It's not a matrix as it was in the discrete time case. But this is the analog of CC transpose here – or sorry, CC transpose, and that's C transpose here is what this is supposed to – this is the analog. What happens then is that you can work out a formula for the minimum energy to hit the state in the amount of time T. It's a quadratic form, and it's a quadratic form that depends on Q of T inverse where Q is this integral, so it's an integral of E to the tau A BB transpose E to the tau A transpose, very familiar here because this is nothing more – this is sort of a – that's the time propagator for what an action at now – what effect it has on the state tau seconds in the future. You integrate this action – that's sort of like the same as summing in the discrete time case – and you invert. And this in fact – I guess you had one number problem on this – this is a Gram matrix in fact. This is an integral of – that's a Gram matrix, and it's an integral of something times something transpose. That's a Gram matrix. Now we can conclude all sorts of interesting things here. For example, if T is bigger, you integrate, you add more positive semi-definite terms or whatever you wanna call an integrand to this thing, and Q gets bigger, and that means Q inverse gets smaller. So here again, you have the idea that as the time horizon gets longer, the minimum energy required to hit a target in longer time goes down. The difference now is that T can be arbitrarily short, and you may need a very large input. By the way, as T gets shorter and shorter, this generates inputs, which in fact converge into something like the impulsive inputs that we looked at earlier. So there are impulsive inputs that get you to transfer U to a state immediately. Of course, an impulsive input is extremely large. Here, this will actually construct the impulsive inputs. Now we could also look at what happens when T goes to infinity. This matrix always exists because this thing is monotone increasing in the matrix sense. That's non-decreasing, and therefore the inverse – this is a positive semi-definite matrix – sorry, a positive definite matrix, and it's decreasing as a function of T, so this goes down. It's the quadratic form that tells you how much energy it requires to hit a point, any point in state space. This goes down. It has a limit therefore, and we'll call that limit P. It's the same as before. And that gives you the minimum energy to reach a point arbitrarily leisurely. So that's what that means. It's the same story. If the matrix A is stable, then this integral converges here. Nothing goes to infinity here if A is stable. Why? Well, because if A is stable, E to the tau A, all the terms in it look like polynomials times decaying exponentials. So this term simply converges, and therefore P converges to the inverse of that, and is therefore positive definite. And that says basically if the system is stable, then the amount of energy required to hit a point arbitrarily leisurely is gonna always cost energy. So there's no point you can get to for free for example. There's no point you can get to with arbitrarily low energy. It makes sense. If it's stable, that sort of means that left to its own dynamics, the system state will go back to zero. So it means that if you wanna get to some state, you have to fight against that, and that's what tells you that P is gonna be positive definite. So that's basically the idea here. We can look at the idea of general state transfer. That's transferring a state from an initial condition TI to some final state T desired. It's all very similar. It's the same as the discrete time case. What you do is you say this. You say that the final state is equal to this. It's the initial state then – that's the time propagator. That propagates you forward in time TF minus TI seconds here. That's this part. So this is where you would land if you were zero. If you did nothing, this is where you would land. This is the effect of the input over that interval like this. So the whole thing, you realize the following. This is equal to that. This is equal to X desired if and only if you put this on the other side, if U steers X of zero to X of TI minus TF to this thing here which is X desired minus this time which sort of compensates for the drift term. And in fact, all of this kinda makes sense. This says that if you wanted to steer something to a certain place, what you'd do is this. You first find out what would happen if you did nothing, and that would be over here. I subtract that from this, and then I act as if I'm going from zero to hit that point, and everything will work out. That's by linearity, so this all works out. Now for continuous time it's a lot easier because basically the system is either controllable or it's uncontrollable, and that means you don't get any of the subtleties. It means basically if it's controllable, you can get from anywhere to anywhere else in arbitrarily small time, possibly with a very large input, but nevertheless you can go from anywhere to anywhere else in any amount of time as long as it's positive. You can do it in zero time if you're allowing yourself to use impulsive inputs, so that's how that works. Okay. So let's just look at an example to see how this works. Here's three masses stuck between two walls, and we have some springs, and we have some dampers, and we can apply two tensions here. I think we've seen this example before. So okay, and your state is six-dimensional. It's the positions and the displacements of all – sorry, it's the displacements and the velocities of the three masses, and the system looks like this. The blocks here make sense. This basically says that – this top three are the positions, and it says the position dot is equal to – you read this as zero I, so it says that the top three DDT are equal to the bottom three. That's by definition because the bottom three states are the derivatives. And then these two come from the stiffness and from the damping. And then these two tell you how the two tensions apply forces to the different masses here. That's what this is. So we wanna steer the state let's say from E1 – so E1 means the following. E1 means that this mass displaced one unit to the right, and the other two are at zero, and everyone is at rest. That's what E1 is. And we wanna take that to the zeroth state. By the way, if we do nothing, it's gonna go to the zeroth state asymptotically anyway, and I can't remember what the eigenvalues are, but we looked at this example before and I think the eigenvalues are on the order of one. Maybe there's a slow one on the order of 0.2 or something like that. It doesn't matter. The point is that the whole thing will decay to zero at some rate. By the way, that builds the intuition here easily. It says that left to its own devices, this is stable. The state is going to go to zero anyway. Therefore, if I give you a long enough time period, long enough that just by through natural dynamics the state has decayed, it's gonna cost very little to drive it exactly to zero. By the way, with its own natural dynamics, the state is not gonna go to zero exactly. It will just get small. They get very small. But the point is once it's small, it takes a very little kick to actually move it directly to zero, and therefore we expect it to take very little energy. So what happens is the following. Here's a plot of the energy required to steer it from one, displacement one, then zero zero, and zero initial velocity to zero in T seconds. Very interesting plot, it says basically for six seconds and more, the amount of energy required is extremely small. By the way, is it zero? It can't possibly be. If it were zero, it would mean you'd apply no input. If you apply no input, it never gets exactly to zero. But the point is it's extremely small. So this is explained by the fact that the natural dynamics has made it small anyway. What's interesting here is that this thing now ramps up very aggressively. By the way, does this thing go to infinity or stop like in the discrete time case? No, this keeps going. So you could do this 0.01 seconds. There's an input. The energy required is absolutely enormous. You can work out what it is, so you can do it arbitrarily fast. But you can see exactly what's happening here is that you can move the whole thing to zero as quickly as you like. It just takes more and more energy, and this is quite obvious here. By the way, the practical use would be somewhere around here where you would be pulling this thing back to zero a lot faster than if you just waited for it settle out. Actually, I should mention something about this. There are lots of applications of this, tons. A lot of them have to do with things where you have a mechanical system where you move something from one place to another. I just thought of another one that's not mechanical. You move from one place to another. When it arrives there, it's sort of wiggling a little bit, but you can't use it until it stabilizes. The first example is a disk drive head positioned. So you say please seek Track 255 or something. So this thing zooms over here, lands close, but when it ends there, it's sort of shaky. It's gotta track the track very carefully. You can't use it when it's shaking. So in fact, what you do when you land there is you do something like this. You actually actively apply an input that will basically do active damping, that will remove all the wiggling in it and allow you to track faster. So things like this are actually used now. The other applications I know in X Y stages, so in lithography. These are extremely expensive machines for printing integrated circuits and things like that, and they move over a couple centimeters at a time, and they move, and then they align themselves to accuracies that are just unbelievable. It's like at the nanometer level. It has to be if you're looking at print technology, which is actually quite ridiculous to imagine something that moves two centimeters then – this state isn't even wiggling at the nanometer level, and stabilize itself absolutely to something at the nanometer level. So the way that's done – oh, and by the way the reason – it's of course stable, so you can move over there and just wait for the whole thing to stop wiggling, and then expose your wafer. The problem is this machine costs some huge sum of money, so you can have the cost divided by the throughput. So if 90 percent of the time, you're waiting for the thing to stop wiggling so that you can do your lithography, that's not a good investment, so in fact they use active control to do this where you move it over two centimeters, and you definitely apply inputs that will do active damping, and will cause it to stop shaking early, so that's one. And I'll mention one more just for fun from MRI. So in MRI you do these things. I don't know any of the details. I don't know any of the physics, but you line up all the magnetic axes, and then you let go, and they slowly drift down like this, and you don't do another scan slice until they've totally relaxed again, or something like this. That's the method. You can actually use a method where you act – instead of waiting – I guess they have times like T1 and – does anyone here do MRI? Too bad, someone could've got me out – they could've told me what T1 and T2 – there's like T1 and T2 are the two relaxation times or something like this. You wait one of these amounts of times or something like that to do it again. If you wanna do fast throughput MRI, you don't wait. You actively force the spins down to zero. How? Using these formulas here like this one. And you improve the throughput. And again, you have a big expensive machine whose throughput is now better because you're actually not waiting for something to stabilize out. You're forcing it to stabilize it out fast, and therefore it's ready for reuse again. Sorry. That was just a weird aside. If you wanna look at what some of these inputs look like, we can do that. Pretty much almost for the dot that I drew which was in three seconds, this is the input that you would apply. These are the two forces. It's symmetric, but then the whole problem is symmetric. These are the two tensions you would apply. I don't think they're obvious, so this is what it looks like for – this is the least norm input for bringing the system to zero in three seconds. If you relax that just to four seconds, it gets a lot smaller. Why? Because now the natural dynamics itself is helping you, so that's four. And if you look at the output for U equals zero – this is the – I guess now we have our answer. This says if you just kind of let the thing go with no input, you'd see that it's pretty small in eight seconds. Of course, it's not zero, but it's already kinda – so when you're taking it to zero in three seconds, you can see you're actually doing something. Depending on the accuracy with which this thing has to be zero before it can be reused, you can see that you're actually doing something, and you're doing something that is substantially better than just letting this thing settle out by itself. So I think these are all kind of obvious here. Let's see if I can – yeah, these just show things like the output for three and four, and things like that, but they're kind of obvious. So I think that finishes up this topic of controllability. I wanna emphasize that except for the fact that some of it was continuous time, there's absolutely nothing new in controllability. It's just applied linear algebra. If you know what range is, if you know how to solve linear equations, if you know how to find the least norm solution of a set of linear equations, then all of this is just an application of that. Okay. Now we'll look at the last topic in the class. Like controllability, it's not important. I think you've even done, or maybe – controllability stuff I think you were doing well before it had a name, I think. Didn't we have a problem on the midterm where you had to find an input that took you somewhere? There was one. Yeah, okay. So that's my point. All we did now is just give a name to it. That's all. So the same is true of the observability and state estimation. These are just exercises. So what we'll do is we'll look at this idea of state estimation and discrete time observability and stuff like that, but let's get the setup here. So we consider a discrete time system – this is in the general case that the setup would be something like this. You'd have X of T is A of X of T plus B U of T plus W of T, and W of T is called a state disturbance or noise. Another name for this is a process noise. That's another one you hear. And you have all of this in economics, too, and it's got a different name there, and I forget what it's called, but it's got – in things like chemical process control, this would be called the process noise. It's got all sorts – I'm trying to remember what it is in economics, in econometrics. I can't remember, but maybe I'll remember during – it doesn't matter. It's got some other name, but it's identical to this. And here also you have a V of T, which is a noise on your measurement here. So that's called sensor noise or error. It's also called measurement noise. It's got all sorts of names. The assumption here is that A, B, C, and D are known, but what you want to do is you're gonna observe input and output over some time interval, and you want to estimate the state, and there are lots of problems about estimating the state. You might wanna estimate the current state, the initial state. You might wanna estimate the next state – in finance, that would be of tremendous interest, right? To estimate what the returns for tomorrow would be. That would be of great interest. So there are lots of problems you wanna do, and this is gonna be based on U and Y. And this is basically gonna be a particular application of just this, and I'm gonna do some horrible notation. What I'm writing down now is notation from Week 5 or something like that. It's gonna be just a big problem of this form, which basically is you have some parameters you wanna estimate. You call it X. You have a linear mapping that maps X into some measurements, plus a corrupting noise, and then you ask a question like based on Y, what can you say about X. Okay? Well of course, there's the – we can do the platonic case. We'll get that. That's easy. The answer is it depends on the null space of A or something like that, but we'll get to that later. Then once you introduce the noise, it gets more interesting, and now you can say interesting things about how to do this. For example, you might use least squares here, and we'll see that's what's gonna be the same here, but it's gonna be more complicated, but you could map all of this big complicated thing into something that looks like that. It's bad notation because why? We're already using here – X is here and A is different from what it is or something like that. So let's see how that works. So the state estimation problem is this. You want to estimate the state at some time S from the inputs up to time T minus one from the outputs up to time T minus one. And if S is zero, you're estimating the initial state. That's initial state estimation. If you're doing S equals T minus one, that's current state estimation. If you're doing S equals T, that's called the one step ahead predictor is what you're doing because you're trying to guess what it's gonna be on the next step. And there's all sorts of other variations on it. If S is something like T minus five, some people call that a smoother or something like that because you're sort of using future observations to go back and make your most intelligent guess about what your state was five samples ago. If you were actually trying – if S is T plus five, you're doing a five state predictor, a five epic predictor. You're trying to predict what will the state be in five samples, five epics in the future. By the way, in a lot of these problems, despite using all sorts of fancy methods needless to say, your answer could be useless. That kind of goes without saying. If I ask you – if you can't predict tomorrow's returns, you certainly can't predict next summer's returns or something like that. That's silly. And that's fine. Of course, that comes up here. We don't dwell on it really there, but it's a fact of course. If I don't give you good enough measurements with which to estimate the parameter, then of course you can be as sophisticated as you like, and you won't predict anything that's worthwhile at all, and the same is true here. Maybe here it's kinda more dramatic. So if you have something that estimates a state, it's sometimes called an observer or a state estimator, and it's got other names in other fields, and it's used in lots of other fields. Now a generic notation which is pretty widely used is something like this. You would call X hat of S given T minus one is an estimate. It's denotes an estimate of X of S given input and output information all the way up to T minus one. Now the notation of course is meant to suggest something like a conditional expectation. This is again if you have that background, and there are in fact cases where X hat of S given T minus one is nothing more than the conditional expectations of X and S given input and output information up to T minus one. But here I'm just using it to say it is an estimate of X of S based on that information. It doesn't have to be a statistical method you use. We'll show you how to use least squares or something. That's the idea. Now by the way, I claimed that you can just do this right now because all you would do is put it in this form. I mean you would have to stuff the right matrix A and it would be a pain, and there'd be some debugging you'd have to get right, but I promise you all of those state estimation [inaudible], they look just like this. You have to figure out what X is. You would have to stuff A in the right way, and all that kind of stuff, and then you'd use least squares, and that would be that. Actually, that wouldn't be a bad – you'd actually make estimators that would be pretty good. We'll go through the more standard discussion of all this. Let's start with a noiseless case. Actually, that's probably a good idea to start for anything because if you can't do it in the noiseless case, then you shouldn't be trying to do it when there's noise, so we'll start with this. Let's just find the initial state with no state or measurement noise. Then you have X T plus one is A X of T plus B U of T, Y of T is C X of T plus D U of T, and what you can do is I can write down the output as – first of all, I can write it down as a function of the initial state. That's what we wanna estimate. That's a matrix OT. OT is this matrix here because the output Y of zero is C X of zero. Y of one is C X of one, and a portion of that due to X of zero is C A X of zero. That's this, and so on. Notice this thing is OT. O is for observer as opposed to C, a script C which is for controller. And it looks very similar to your friend B, A B, A squared B, and so on, except this one is stacked vertically. The C is on the other side, and it's C – and in fact, the components should be totally obvious here. Like for example, C A to the sixth is a block row in this, and it basically says that – it's easy to describe exactly what C A to the sixth is. It's this. It takes the initial state. It propagates it forward six samples, and then it takes it at C, and then operates to map a state to an output measurement. So C A to the sixth is the matrix which maps state now to what the output will see in six samples. I think I got that right. That's what C A to the sixth – so that's what all these are. It's a time propagator followed by a measurement matrix, nothing else. Now T – it may have been on like your first homework or stupid like that. T is basically – this describes the mapping of how the input maps to the output. I think it was – was that on the first homework or something? I think so. Something like this was anyway. That's easy to work out. Again, you just have to stuff the matrices in the right way. T is for Toeplitz because that's a Toeplitz matrix, and a block [inaudible] and it looks like this. This is what tells you how Y of zero depends on U of zero, and the fact that this is lower block triangular tells you – it means something. It means causality because this says that basically Y of zero – so I put U of zero down to U of T minus one. By the way, note that I've decided now to go back to writing Us in order, not in reverse time like we did before. I can write them any way I like. I just have to be clear about it. So if I do this, then this first row says that Y of zero is D U of zero and has nothing whatsoever to do with U of two, U of one, up to U of T minus one. That makes perfect sense because the output at time zero has nothing to do with what you put into the system at one, two, three and so on. It's causal, and that's what this is. And all of these things should make perfect sense. B C A to the T minus two B, that's B takes you from an input, has an immediate effect. A to the T minus two propagates you forward in time. And C maps the effect on state to the effect on an output, so this makes perfect sense here.

Student:[Inaudible] dynamics [inaudible]?

Instructor (Stephen Boyd):No, it's actually causal because we fix the initial state.

Student: So you estimate this only if they are looking at the initial state?

Instructor (Stephen Boyd):That's right. No, sorry. No, it's not. It's as if the initial state – the term for the initial state is here, so I've separated out – well, no. I guess actually I do accept what you said because we're looking at the initial state, therefore the initial state appeared here. If we were looking at the final state –

Student: Then we'd be on [inaudible].

Instructor (Stephen Boyd):You've got it. Exactly. But in any case, what I can say is this. This is just an exercise in matrix stuffing. You just need to put it in a form that looks like this. A measure that you know is equal to – in this case, it would go something like this. An output which you know is equal to a matrix times the state you're looking for. I wrote it down for X equals zero – for T equals zero, but it could've been T equals five. It doesn't matter. It could've been T plus five, meaning look five states in the future, so that would be that matrix is known times the thing you wanna get. And then plus another term which is something you know. So this thing here has a very strong meaning. This term here is very interesting. This is the effect on the output due to the input. Now you know the input, and you know that, so in fact you know this term. It's actually your predicted output is what it really is. It's what you'd predict the output was if the initial state was zero. So that's what this is. And actually, that'll just go away in fact, that term. So you can rewrite these equations this way. This way you write OT times X of zero – that's what we want – is equal to something we know – the output, we measured it – minus – and then this is a beautiful thing. That's something we know times something we know, so this is whole right-hand side is something we know. But the interpretation of the right-hand side is quite nice. This is what we directly measured. We also know this. Possibly we measured it. When we measure this and propagate it through the matrix, presumably that exists as 15 lines of C somewhere, so this actually goes down – this goes down in a computer. And this basically is your estimate of what you would see if the state were zero. You subtract that from what you did see, and this is – now it may exist only in some processor. It may not have to exist in practice as an analog signal. It may exist only in a processor. This signal over here on the right is beautiful. It's the output kind of cleaned up from what you – from the effects of the input or something like that. Let me try that again. It's the output – I've got it. I'm gonna try it again. It's the output with the effects of the input removed, but it was removed synthetically. It wasn't done in analog. It was done in a processor. That's what this is. Then this says okay what's left is only the component of the output which is due to the initial state, although this would be X of S – it'd be some other state of it, if you're estimating something else. And now you're back up to Week 3, which is to say you have a skinny – you have A X minus B with A skinny. Presumably you have more measurements than you have unknowns, and that's it. And you can do all sorts of things with it. There's no noise at all if you can just solve the equation with your favorite left inverse. If there's noise, you might wanna do least squares. That might be the right thing to do in some cases, but you wouldn't do it always, or you might not do it always. Okay, so with no noise we can answer everything. It goes like this. It says we can determine the initial state of the system only if the null space of this observability matrix is zero. That says the observability matrix, the columns are independent. That's what it says. By the way, note the nice kind of duality to what we looked at early with controllability. So in controllability you have B, A B, A squared B, and so on, and everything came down to – this idea of controllability came down to – in that one it came down to something very simple. It was that this fat matrix should be full rank – should be on to. In this case, it's a skinny matrix and it should be full rank. The meaning in the first case is there's an input which will take you from anywhere to anywhere else, at least in N steps. In this case, it says something like this. If I observe N or more samples of the input and output, I can predict the state exactly if there's no noise. That's what this says. So that's the analog of what it means to be full rank. In this case, you call the thing observable. Notice that the input here has no effect whatsoever on estimating the state. Now there actually are cases when this is not quite right because basically it can just be subtracted off immediately. This is very – now people would say that doesn't make any sense. I would rather get it – it's better if you're not doing anything – someone else would say no, you gather better data. You can estimate the state better if you're wiggling the control stick or something. No, it turns out these are all just wrong. It makes no difference because you can subtract it off. There is one issue. If it turns out that your model is not quite what the thing is, then here the true T is not quite what you think it is, and when you do the subtracting you might get left with some residuals or something. That's a secondary issue. In this case, it doesn't affect it at all. Okay. So again there's a dual to all of this, and I think I'll speed up because it's actually kind of boring, and there's some things I wanna mention at the – there's some stuff I wanna cover at the end. So by the Cayley-Hamilton theorem, you know when you go C, CA, C – when you get up to CA to the N, you've just added a bunch more rows which are linear combinations of rows above. So it says basically if you can figure out – if you can work out what the state is from measurements from U and Y at all, you can do it in N steps. As in the control case, it doesn't mean it's a good idea to do it. You may want to take much more data and then average – I guess so far we have no noise, so that comment doesn't make any sense, but when you add noise it will make sense. So I think we'll go over – I think what I'll do is just go forward to what would be obvious here. I've lost that [inaudible] least squares observers. Thank you. There we go. I got it. So the most obvious thing to do in the case when there's actual measurement noise is to set it up exactly a least squares problem, and you'd have this. You'd estimate your state to be the observer matrix pseudo-inverse here, which in this case if it's actually an observable system is O – I guess it's skinny, so it's O transpose then quantity O transpose inverse times this thing like that times this thing. And this thing has got all sorts of names. I don't know. Let's see. This is something like it's your guess of the output due to the initial state. That's what this signal in here is because this is your guess of the output that would be due to the input that you actually measured if there were no initial state. Then this is what you really did measure. The difference is the output that's due to the non-zero initial state. So that's how you get all of this. I can mention something about this, and I think I'll even – but I won't go on and do the rest. I just say this alone here, if you wrote this in recursive form, you would actually have a very rudimentary form of something called a Kalman filter which you may or may not have heard of. But this thing which is – this is just an exercise from like Week 4 of the class. It's very simple to state this. You get things that will work – these will work unbelievably well. I mean this beats anything, any kind of intuitive scheme you could make up if you wanted to track a vehicle or something like that. Just something this simple which would end up being just a handful of lines would actually make estimates of states and positions that would beat any intuitive method you could possibly imagine, but that's kinda the theme of the class. So actually, I think I'm gonna quit there because I claim all of this is just sort of variations on applications. We could look at one application but I don't know. It was gonna be – it's an application with our friend the range measurements. Maybe I'll just do that very quickly and we'll look at that. So here's the example. It's just to give you a rough flavor of how these things work. By the way, there are entire classes you can take on state estimation, so you can take one in finance, or you can take one in aero-astro, where it's used to track vehicles, or it's used for navigation. So you can take entire classes, probably three or four in wildly different fields, and it's just on this material. Some of them are kind of cool. Actually, a lot of them are kind of cool. So we'll just do a very simple baby example. It's a particle that's moving in R2 with uniform velocity. I have linear noisy range measurements from various directions like that. You can see they're all kinda concentrated from here. By the way, if I have range sensors from here, that's not great. If someone gave you a bunch of range sensors and then arranged them to make your estimation job easy, you'd spread them around. In fact, you might even do something like put – you'd probably spread them around evenly, maybe even at four cross points like this because then you're sort of measuring different things. A bad thing to do would be to put all the range sensors in the same place kinda like this. Right? Because if you put all the range measurements in the same place, are these getting making different measurements? Oh yes, they are, but they're not that different, so you're sort of – you can imagine now that your estimation might not be that good. So that's it. And we'll make the range noises have an error on the order of one. And we'll assume that the RMS value of the velocity is not much more than two. So actually, we can write this out as a linear system. To describe a particle with uniform velocity is just basically this. It says X T plus one is gonna – the bottom block says that – the bottom block of X T which is the velocity is equal to the identity times the velocity. It means it doesn't change. So the bottom two entries of X of T are constant, and the top two change – actually, they're [inaudible]. Then your measurement matrix is gonna look like that. So that's what you have. The true initial position and velocities are these. This makes no difference, but then here would be an example of a track here. What's shown here are that line shows you the true to the particles moving at a constant speed. That would be the range measurement if the range measurements were perfect, but you can see we put a lot of noise on the range measurements, so basically any single range measurement would be – a snapshot of range measurement would be completely useless. You'd make an outrageously bad estimate of the position and velocity, obviously from this. What you can do now is actually just carry out this least squares observer fitting, and we can actually work out the actual RMS position error, and you get something like this. It's hardly surprising what you see, but the velocity error actually goes to zero, and the position error actually will – it won't go to zero, but it will go to a small steady state number. So that's what it is. That's how these things work. And this is just a taste of it. Technically, you could write all the code to make this plot. You could've worked everything out and so on because it's just an application. So that officially finishes the material for the class. The truth is that the last two topics we looked at were just sort of applications, so a little bit more than an application, but they were not real material. The singular value decomposition was the last real material. What I wanna do now though is actually – I wanted to have a bunch of comments about the whole like how does it all fit together and things like that. So that's what I wanted to look at next. The first thing is I wanted to say a few things about linear algebra, not that I haven't been talking about linear algebra for a quarter, but I thought I'd just say a few things about it. So just to organize my thoughts, so it doesn't come out totally randomly, I wanna zoom out a little bit here. The first thing that should be kind of obvious – if you haven't seen it, you will see it or something like that, but the point is it comes up in a lot of practical contexts. It comes up in EE and mechanical engineering, civil engineering, aero-astro, operations research, economics, machine learning, vision – it goes on and on. What's interesting is that actually that wasn't always the case. So signal processing 25 years ago basically went on and on with how you process a scalar signal for example. You would barely – you could go grab an issue of [inaudible] signal processing. You won't find a matrix in it. Now totally the opposite. Everything involves ideas like – it's extremely rare to find something that doesn't. The same is true in economics, and in finance, and in things like that. In aero-astro and control 25 years ago, the idea of sort of a multivariable system with multiple – that was this exotic thing that only PhD level students would study. That's completely wrong. All real systems are designed this way, no exceptions. So that's just the way things work now. You don't design a control system for a modern airplane or something like that, or a process or any – you don't even make an ABS system or anything like that – you don't do it without matrices. And this is a change. This was simply not true 20 years ago. 20 years ago, there were people who talked about matrices, and they were made fun of in fact by the people who were designing systems that worked anyway. I wasn't one of those people. So the point that's – it should also be kind of obvious is that nowadays linear algebra you can actually do it. If you're fiddling around, you can play with MATLAB. MATLAB is a toy and it's not – I have arguments with people about this, but in fact no real implementation uses MATLAB, although there are people who say that's not true. It is a toy in my opinion, and real implementations are not done this way, and that's often true. Now 20 years ago a class like this, there would be zero competition component, none, absolutely zero. By the way, you can imagine how boring it would be. I just – we won't go there. It was kinda boring actually, to tell you the truth. Luckily for you, you were born later than this point. Now of course, you have MATLAB. You've been playing around with that. By the way, there are other variations on MATLAB you can use. There's Octave and things like that. And the truth is that these days if you know what you're doing with some object-oriented system, you can make it look like MATLAB except that it will actually be a real language. So Python could easily be made to look like MATLAB for example. So you could write things like Y equals A X times – you could write Y equals A X plus V and all the right things would happen. There's real codes. LAPACK is something. You should've just heard that name. I don't want you to leave this class without hearing that name. Even though we talked nothing about how to actually carry out these confrontations, it's important that you know that things by the way that MATLAB does, which does nothing but parse what you type in, the actual numerics is done by open source public domain software written by professors and graduate students. It's LAPACK. Then when you get into large systems, there's other schemes like that as well, but this is a good thing to know about. And just to give you a rough idea, I'm sure you kind of know this, but maybe no one made a big deal about it. If you have let's say 1000 variables, and you wanna solve – I don't know. Let's say – let's do 1000 variables – let's do a least squares problem that looks like this, 1000, and let's make this like 2000. It can be anything you like. It can be I have 2000 measurements to estimate 1000 parameters. By the way, this is obviously not a small problem. That's two million real numbers to describe this matrix. Everybody see what I'm talking about here? This is serious business. You could make a big story about how sophisticated this is. You'd be taking in 2000 measurements and doing the optimal – you could make a big long song and dance about how you're doing – I'm blending 2000 measurements to get 1000 estimates of blah blah blah. How fast do you think – how long does it take to do that? You might have some rough idea. Does anyone know? You type A backslash B in MATLAB for this.

Student:[Inaudible].

Instructor (Stephen Boyd):What? You're right. About a second. So the answer's about a second. Might be two seconds, I don't know, something like that. What? 2000 by 1000. Do it twice though so it gets loaded into – okay, sorry. But you know, it's on the order of a second, two, something like that.

By the way, this is just stunning, the fact that you can do this. This is something unheard of. This is what happens when you ignore Moore's Law for let's say 20 years or something, and then come back and check again how fast it is. This is the kind of thing that happens. Now in fact – and this is a beautiful topic. It's something actually some of you might wanna look at. By the way, I don't – I think things like this have not really propagated too far, so there's a lot of times when you might wanna solve – do least squares with 2000 measurements to estimate 1000 things. That's a big thing. That's like a big system. The fact that you can do this in seconds with total reliability I think actually has not diffused into general knowledge. Like I had a professor in my office, and I asked him to estimate how long this would take, and he said half an hour. I won't reveal his name.

Student:[Inaudible].

Instructor (Stephen Boyd):How many? Four. On your laptop.

Student: No, on [inaudible].

Instructor (Stephen Boyd):Okay. Four seconds. There we go. So it's four. That's because it's too warm. Okay, fine. Four seconds gives you a rough idea.

Actually, an interesting thing that we haven't talked about at all – as it turns out, if you wanna do things like least squares, solve linear equations when – you can do this for much bigger, way bigger matrices provided they have special structure. Now, special structure – the typical one would be sparsity. Sparsity means a lot of the entries are zero, in which case there's methods that are just amazing and extend to things like 10,000, 40,000, a million depending on your luck. There's other structures like for example if it's banded or tri-diagonal and all this kind of stuff. And there's just a whole world of knowing the structures and how they imply fast methods. And this would take you to things like how people solve PDEs and all sorts of other stuff. That's worth knowing. Actually, I encourage you in the future to learn about these things. So please don't live your life just in MATLAB. There's something – even if you do, please look at – know what's under the hood. And actually, know then what would happen if – even if you have no intention of ever doing it – of actually figuring out what a real implementation would do as well. That's a good thing. I won't say much more about that. There's whole classes you could take on this at Stanford, entire ones, literally two on numerical linear algebra that I know of. Actually, I know of a couple others, so there's maybe two or three. You can even take one in geophysics and all sorts of other – Now I wanna say something about linear algebra and sort of my view of how it all fits together. I think there's really like several levels of understanding. You could add a fourth one, but the most basic is this. The most basic level of understanding is kind of at the high school level, and it goes like this. What you should know is this. If you have 17 variables and 17 equations, there's usually a solution, and that's when you were tortured by solving whatever three equations and three unknowns for no particular reason whenever you were tortured doing this. And then maybe sometimes 17 equations and 17 variables, there wasn't a solution, or there was more than one, and somebody waved their hand or something like that. And the same high school view would say something like this. It's actually just degree of freedom accounting is what it is. So for example, if you have 80 variables and 60 equations, you just do some accounting and you say, "Well, I've got 20 extra degrees of freedom." You could use those degrees of freedom for anything you'd like, but there's basically a 20 – this is roughly – a 20-dimensional subspace of freedom. You could use it to minimize the two norms or the Euclidian norm, which is what we've been doing, and you get a least norm solution. You could use it to minimize something else. You could use it to do anything, other stuff. There's just degrees of freedom you can absorb. Of course, if that's an estimation problem, this is probably not good news because it basically means that the only thing you can do is – you don't have enough measurements to estimate what you want. Now so it's very important to always start I think when you look at a problem from this point of view, just do the basic accounting. What are we talking here? Do I have more measurements than unknowns? Less? How many degrees of freedom do I have less than all that? Now of course, what you know at the next level, you learn things about singularity, rank, range, null space and so on. And what that would be for example would be here. If someone asks you, you'd say, "Look, I have 17 equations and 17 – but there's no solution." Why is that? And you go, "Ah well, you see that 17 by 17 matrix was singular." And I go, "What does that mean?" And you say, "Well, what it means is this. There's two ways to view it. The first is you thought you had 17 different equations. Actually, you don't. You have 16 because equation No. 13 is derivable from the others." Everybody know what I'm talking about here? So in fact, I'm talking about the left – I'm talking about the rows being dependent. They could say something like this. They'd say you thought you had 17 variables, 17 knobs to wiggle, or whatever you wanna call it. That's what you thought you had. Actually, you had 16. You know why? Because the action of this fourth variable can be exactly replicated by an action of these other variables. Okay? So you thought you had 17 knobs, 17 design variables, 17 inputs, whatever you wanna call it to mess with, but you didn't. And that was hidden in this matrix, which was 17 by 17 so it wasn't obvious to see by the eye. Okay? So that's the next step. Now, the platonic view – this is really basically called math. It's very important because – and it basically says something like this. It says that this matrix has Rank 5. It says this is full rank, so it really certifies that there really are 20 extra degrees of freedom here. That's very important to understand. Now I'll tell you the good thing about it. Now the good thing about this is everything is precise and unambiguous. There's no such thing as – there's no waffling, no nothing, and you don't – there's no nonsense. It's either singular or it's not, period. Its rank is 17, or it's 15, but it's not like anything in between, and it doesn't depend on who's looking at it, or anything like that. Things are true or false. What's nice about this at least – it's nice because it's very clear and unambiguous. Unfortunately, it can be misleading in practice, and we've already seen at least one example of that. For example, if I have a 17 by 17 matrix and it's singular, if I pick literally any random entry and perturb it by an arbitrarily small amount, the matrix becomes nonsingular, and I say done. It's all done. Now you can invert it. And then you'd better leave quickly before they actually – but theoretically they can now invert it, and there's no problem. In this case, it would be misleading to say for example to make a statement, which you could on the basis of just a platonic mathematical view – you could make the statement that nonsingularity is not problem in practice. Then they'd say, "Why would you say that?" You'd say, "No problem. Any matrix that's singular, if I perturb its entries by random numbers so small that they couldn't possibly affect that practical application, I guarantee you with probability one that the resulting matrix is nonsingular. Done. End of story." At this level, you can't argue with it. It's correct. That's interesting. That's a mathematical argument for why you don't need to know the math. Did you notice that? That's what it was. Now at the next level – this is Level 3. These are sort of levels of – the next level we have the quantitative – now the quantitative level, it's based not on qualitative things like Rank 3. It's based on ideas like least squares and SVD. So there you'd say things like – someone would say, "What's the rank of that?" And you'd go, "Ah, it's around six." And they'd say, "What are you talking about? How can the rank be about six?" And you'd go, "It depends on you know how big your – what's your noise level, and what do you consider small and big." So it's a bit wishy-washy, but it's actually more useful in practice. As long as you don't think when you're talking about this that you're actually doing math, which you're not at that point, so you have to keep them very separate. So this is very useful, and in this case no one would be fooled by someone coming along and offering to sprinkle some perturbation dust on your matrix to make it nonsingular. No one would be fooled by it because they'd say, "Yeah. Go ahead. Sprinkle all the dust you like." You sprinkle all the dust on it, and all that happens is one of the singular values, the zero now becomes ten to the minus ten, and you say, "Are you done?" And the person says, "Yes. It's nonsingular. I'm leaving now." And you say, "Well, it may be nonsingular, but the minimum singular value is ten to the minus ten. I can't do anything with it that I would want to do with a nonsingular matrix because for example the inverse is gigantic." The condition number is still ten to the ten or more. Now the interpretation in this analysis, it really depends on the practical context, and these ideas are extremely useful in practice. So I think these ideas by themselves I think are – if you sort of have arrested development here, it's maybe not so good, arrested mathematical development I'm talking about. Actually curiously, arrested development here is probably okay in my opinion. It doesn't lead you to – because you know you don't know a lot of things, and it's no problem because you're not expecting to work, and you type A backslash B, and it says singular [inaudible], you go, "Oh well. I'll try something else. This is not a big deal." Here you can actually be a little bit dangerous because you can walk around and talk about things like controllability and things like this, and singularity, and rank, and so on. This is actually quite – this stuff is actually very useful in practice. And I wanna talk about one more level above this, and that would be the algorithmic and computational sophistication. So that would be – another level beyond this would be to actually know how fast and how well you could actually carry out those calculations in practice. And that would be for example what it would take for example if you were going to let's say start Google. Let's just say because that's nothing but a singular value decomposition, but it's one based on an eigen in a billion by billion matrix. And I can tell you right now you can't type a billion by billion matrix into MATLAB and ask it – and type SVD of it. Okay? So you have that next level where you know how fast you can do things when you do them in real time. You can know things like that's a big matrix, but you know what? It's very sparse. I bet we can do this. We didn't talk about any of these things, but you can do things like calculate the – let me just give one example of that, and then I'll move on to the next topic. Here's the last example. You remember early on, there was this idea of representing a huge matrix as an outer product. If you did this, for example you had a method to speed up simulation of it by some huge amount if you've succeeded – and later, early on you learned that the smallest factorization you could do depends on the rank. It is the rank. And you learn a method for doing it, QR. Fine. Later in the class, when we studied SVD, you learned a method for getting an approximation that's low rank. Then you can take a matrix that's absolutely full rank, giant, and you could say, "This three rank approximation is good enough," and you win big because you can now multiply it so fast, it's amazing. You can do compression. You can do all sorts of crazy stuff now. Now that works if A is up to say let's say 1000 by 2000 or something like that, maybe 2000 by 2000, maybe a bit more.

But the real interesting one is when A is a million by a million. You can't store a million by a million matrix, let alone compute its SVD using the LAPACK routines obviously. And yet that's exactly the case where a low rank approximation would be like super cool because then you can say, "Hey, you know your million by million matrix" – people would laugh at you. They would say, "It's not a million – you can't even store a million by million matrix, so don't tell me about it." And you go, "Yeah, but what I'm telling you is I have a Rank 3 approximation of it." So there you can't just form the matrix and call SVD. That's ridiculous. There are methods that do the following, that will actually calculate the Rank 3 – that will calculate the SVD sequentially, so they'll – and they're extremely efficient and fast. The only thing they need of A is you have to be able to multiply A, and you have to multiply by A transpose. You have to provide routines for doing that. So if A for example is the forward mapping in for example MRI, or PET, or anything like that where you have a fast method for example based on Fourier transforms, you can actually do low rank approximation. I can tell no one has any idea what I'm talking about. That's fine. All I'm saying is there's another level of sophistication where all of the ideas here you can use on very large systems. So let me go on and just answer one more question, or address one more issue. It's like what's next. So there's lots and lots of classes – now of course, after you take the final, you may have a different view of all these things. Oh, I forgot to tell you. Since we give our final, or rather – I do think there's no way we can construct our final to be within the – it's a blatant violation of the registrar's rules, by the way – so at first I thought we could just call it Homework 10, but it counts for a lot, but it turns out you're not allowed to assign homework nonetheless – Anyway, so I should mention when you're doing the final, if you wanna take a break, you can go fill out the teaching evaluations which is fine with me. That's why we have tenure, so you can do anything you like, but if you get really pissed off, you can always do that just to unwind for a little bit. And that's actually I think a – it's possible because of our flagrant and open violation – and now on video violation of the registrar's – sorry. We'll go back to that. Yes. So there's a sequel to this class, but it won't be taught until next year. Now I am teaching this class, and this class 364A and B winter and spring, so I should say a little bit – I mean there's zillions of other classes you could take on all the topics I've been talking about, but let me say a little bit about what this class is about. There's a simple way to do it. This class, a lot of the techniques here relied on sort of least squares – least squares, least norm, and some variations on that, and so there were analytical formulas like A transpose – and so I can tell you what this class is. This class basically says what if you – it extends least squares to other things like the maximum value or the sum of the absolute values. Also, in 263 we don't really have any way to handle things like to say that a vector is positive. You see what I'm talking about? Like if we do Y equals A X plus V, and X is a power or something like that here, if we do least squares and then X would be a vector of powers, Y would be some observed temperatures or something like that, that's measurement error. We have no way of telling least squares that X is positive. We would just calculate a least squares solution, and if we're really lucky, the estimated X will come out to be positive. If not, it'll be negative and we'll have a bit of a problem. So it turns out – by the way, people have dealt with this for years using all sorts of ad hoc methods, in fact using all the ideas you know about like regularization and things like that, and they're completely ad hoc. It turns out you can handle those flawlessly. You lose your analytical formulas, but you can calculate just as fast as you could before with least squares. It's even more so when you do things like when you design inputs, which we've been doing. We just did least squares inputs. The least squares input that takes a system from here to a zero state, or from one state to another, you could do things like add conditions like this, which is actually kinda cool. And these are quite real now. You could add a condition like that. That's a real condition. You will not find if you go up to an MRI machine a note that says under no circumstances shall the sum of the squares of the RF pulse be more than this. It will not say that. Or when you buy a motor for a disk drive or something like that, there'll be current limits, there'll be things like that, and they're gonna look like this. It turns out you can solve those problems exactly, and that's what 364 is. When you go to 364, there actually is a lot more fun problems you can solve. You can actually really do all sorts of things like finance, and machine learning, and stuff like that. So that's what this class is about. I'd be happy to answer any questions if anybody has any. Actually, it's a bit embarrassing. I believe next week I'm gonna have to give the first lecture for it because I'll be in India when the actual class starts, so I've never done that. I've never taped ahead the first lecture before. I'm deeply embarrassed, but it was gonna happen sometime, so it will happen. All right. So with that, I'll say the class is over. I have to thank Jacob, Yung, and Thanos for an enormous amount of help putting it all together and all that kind of stuff. And unless there's questions, we'll just quit here. I mean I'll be happy to answer questions about these or other classes and stuff like that, but otherwise thanks for slogging through it. Oh, and enjoy the final. I forgot to say I'll be in Austin while you're taking most of it, but then I'll be back Saturday early morning, and I'll be in touch. I'll be around Saturday. Good.

[End of Audio]

Duration: 69 minutes