

**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  $\|AX - Y\|^2$  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 over XNRN and we evaluated  $J_1$  and  $J_2$ , 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,  $J_2, J_1$ , 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  $X$ s 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  $J_1$  and  $J_2$ . 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  $J_1$  and  $J_2$  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  $J_1$   $J_2$  plot or  $J_2$   $J_1$  plot, and that's this way. If I look at level curves of this composite objective – that's  $J_1$  plus  $\mu J_2$  – in the  $J_2$   $J_1$  plane, these are nothing but lines with slope minus  $\mu$ . If you were to minimize  $J_1$  plus  $\mu J_2$ , here's what you're doing. You are doing nothing but this. You're moving this line with a slope of minus  $\mu$ , 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  $J_1$  plus  $\mu J_2$ , 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  $J_2$  than I said before. Presumably, we'll find a new point where  $J_2$  is smaller. You're gonna pay for that.  $J_1$  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  $J_2$  and to pay for that, it has increased  $J_1$ .

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  $\mu$ s 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  $\mu$ s, store the design and the  $J_2$   $J_1$  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.  $J_2$   $J_1$  and I'm gonna put a slight but small tradeoff like that. Now, let's talk about minimizing  $J_1$  plus  $\mu J_2$ . What will happen when I minimize  $J_1$  plus  $\mu J_2$ ? 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  $X$ s 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  $\mu$ s. 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  $\mu$ s in the middle, you have a lot of  $\mu$ s 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  $J_1$  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  $AX - Y$ . The bottom part is  $\sqrt{\mu} \sqrt{FX - G}$ . 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  $\tilde{A}$  or something like that. It just means in [inaudible] it's even simpler. It's something like  $\tilde{A}N$  – 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  $\tilde{A}^T A$ , one of the inverse, times  $\tilde{A}^T$ . Well,  $\tilde{A}^T A$  – you can work out analytically what that is. That's it. That's  $\tilde{A}^T A$  like that. You see something actually quite beautiful here. You see  $\tilde{A}^T A + \mu F^T F$ , and that's how this works. Over here, you get  $\tilde{A}^T Y + \mu F^T G$ . 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 table. We have a ten second period. We apply forces  $X_1$  through  $X_{10}$ , 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,  $FT = 10$ , and so we have  $Y = A^T X$  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.  $J_2$  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 overshoot. 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  $AX - Y$  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  $J_1$ , that's just least squares. That's classic least squares. We know the solution. If you only cared about  $J_2$ , let's get that out of the way right now.

If you only cared about  $J_2$ , 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  $A^T A + \mu I$  inverse  $A^T Y$ . 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  $A^T A$  – 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 Tikhonoff 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  $AX - Y$  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  $10^{-6}$ , 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  $A^T A^{-1} A^T Y$  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  $X_K$ , then I would have  $R$  of  $X$  is about equal to  $R$  of  $X_K$  plus – that's the Jacobean – times the deviation. Here, if you wanted to, you could say something like provided something like  $X$  is near  $X_K$ . 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  $X_k$  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  $n$ th root of that number where  $N$  is the number of parameters, everything is cool, because a million simulations in  $R^{10}$  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  $Y = AX$ , but  $A$  is fat now. That means  $M < N$  – 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  $N - M$  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  $X$ s to satisfy  $AX = Y$  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  $X$  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  $Y = AX$ , and you can say roughly that the dimension of the null space of  $A$ , which is  $N - M$ , 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  $A^T (A A^T)^{-1} Y$ . It should look very familiar but be a little bit off.

You are used to  $A^{-1} A^T Y$ . 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 mnemonic. My mnemonic 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  $A^{-1}$ , your syntax alarm would go off unless you know  $A$  is square. But here,  $A^T (A A^T)^{-1}$ , 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  $A^T$  on the right. You can also form  $(A A^T)^{-1} A^T$ , 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  $A^T A$  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