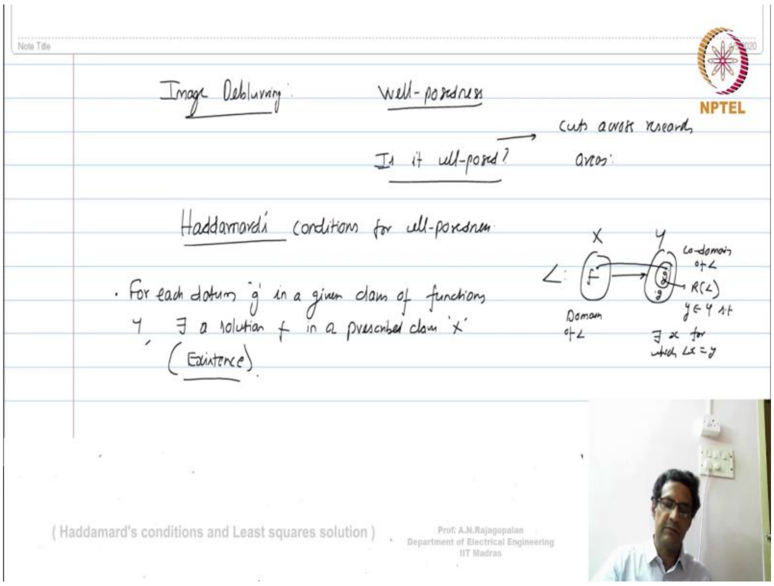
Image Single Processing
Professor. A. N. Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture No. 73
Haddamard's conditions and Least squares solution

(Refer Slide Time: 00:18)



So, so our focus, as I said before, is going to be on image deblurring problem, which is of course an, the inverse problem, a hard one at that. And we would like to solve this. We would like to know what it takes to solve this. And that brings in a notion of what is called Well-posedness. What this means is this problem well-posed. A problem that that has to be solved, for example, it does not have to be an image processing problem. Any problem for that matter that we tring to solve, one should first of all ask is our problem, or is this problem that has been assigned to me, is it well-posed?

And and this notion of well-posedness in the sense that, that I can really solve this problem and then (())(01:07), that is what it means. Now this kind of a well-posedness, it can, it cuts across across you see, research areas. This is not something specific to let us say image processing, it cuts across the research areas. And, and therefore, this is a generic question. It is kind of a generic question that can be asked anywhere, anytime for an offer any kind of problem. And from now on what we are going to do is, we are going to be looking at more abstract things.

These abstract things are not really not really per say for really a deblurring problem. This would be, they are generic in nature. For example, we are now, we are going to look at numerical stability issues, we are going to look at existence issues, we are going to look at uniqueness issues, all these issues are going to be generic in nature. And therefore, for example, any kind of a problem that you might want to solve tomorrow, be it the domain of image processing, be it the domain of wireless communication, whatever it is you want to do, you will find that some of these ideas will translate across these areas.

And therefore, when I would like to draw a relation to image processing or image deblurring, I will show that. But then otherwise, until such point, then I mention whatever we plan to do from now on is going to be generic in nature. It is going to be the, these are going to be generalized, general ideas and these are valid for any kind of problem, not necessarily the image deblurring problem. Now, if somebody were to assign you a problem and say that solve this problem, where he does not mean that you would immediately jump in and say that "Okay, right, let me go ahead and find the solution." What they will first of all ask is this problem well-posed.

Now, what is this? What is the abstract notion of well-posedness? So, among the various notions that exist for well-posedness, the one that is most commonly and are subscribed to is what is called Haddamard's Haddamard's Haddamard's Conditions for Well-posedness. A mathematician Haddamard conditions for well-posedness. So, according to him, according to him he laid down actually 3 conditions. And according to him a problem that satisfies all 3 conditions is well-posed. And even if it violates any one condition, it will, one or more condition than it is ill-posed.

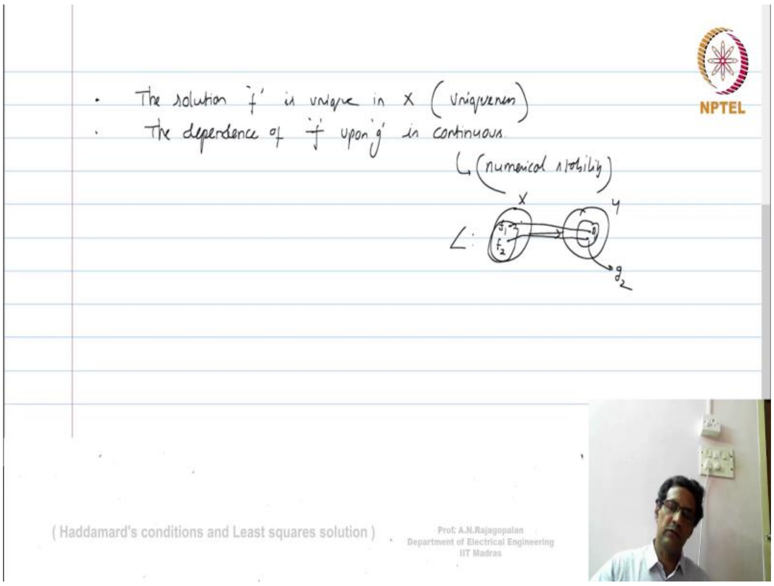
And of course, one who would like to look at well-posed problems, because (())(03:30), those are the kinds of things that you would like to solve, but then unfortunately most inverse problems are actually ill-posed. This is the reason why it becomes all the more important to understand what we mean by well-posed and ill-posed problems. The first condition that he lays down, that he laid on is as follows. Now, for example, imagine that you have some operator L that is that is going from a space x to a space y, so that kind of takes points from x and then maps them to points in y.

So, for example, so basically this is z domain of L, x is the z domain of L. y will be the co-domain of co-domain, yeah the co-domain of L. And then L maps these points from x to y. And of course, one can kind of think about space within y, let us call that the range space of L, which consists of all those points y belonging to y such that there exists an x for which Lx equal to y. That is what we call the range (())(04:47). So, Haddamard's condition for well-posedness are as follows.

For each datum, for each datum, g in a given in a given class of functions, functions y, there exists a solution f in a prescribed in a prescribed prescribed class x, where this condition is also called the condition of existence. That means, if I actually give you an observation, solution should first of all exist that means if I give you a g from the range space of L, then I know that in that case, our solution does exist.

And there should be some f here. Whether f is unique and all, we do not know as of now, but then existence, but for example, if I give you an l, we give you a g that is outside the range space of L, then we know that there is no f in x such then we know then we know that there is no f in x such that, so that L can act on it, act on that f to give you g. So, a solution, so Haddamard's first condition is that a solution should first of all exist.

(Refer Slide Time: 06:10)



Then a second condition is the solution if it exists, the solution f is unique in x. So, that solution, if it exists, should be unique in x. This is called the Uniqueness condition. And the point is, I mean, so its like saying that if you kind of go back to this diagram, I give you a g, which is in the space of L, because then you know that a solution does exist, therefore existence is satisfied, but then do you have just 1 f that will map to g or there can be multiple fs that map to g? Can you have an f1 that also goes to g, if you have also an f2 that goes to g, in which case uniqueness is not satisfied? And I can leave that, the multiple solutions for 1 g, for a given g.

That would that would then make this problem ill-posed. If a solution, if the solution is not unique, then of course, that would make a problem ill-posed. Now, if you think that, now so if you really combine conditions 1 and 2, then it looks like, this like one-to-one and onto, you have a map. If something satisfies both conditions, then then in that case, you have a one-to-one and onto map. And therefore you can think of L as being unique, but, and therefore, one would think where we even need one more condition. That looks like all should be fine then. If I have a one-to-one onto map, then I am done. And I know I give you an observation and because it is onto, I know for a fact that a solution exists.

On top of that, I know, I also tell you that the solution is unique, in which case you are done. For the g, you can always, you already know that, that f exists and that that f is unique. But the whole point is that Haddamard did not stop there, he introduced one more condition. He introduced a third condition, wherein he said that the dependence of f upon g of f upon g should be continuous, is continuous or should be continuous. Now, what does that mean? This has got to do with what we call as numerical stability of your problem.

All these are issues that we will examine individually. Not just from an image processing point of view, but in general. And that this numerical stability, what is actually means is if I kind of go back to that diagram that I drew, where we are going from x to y, this is x, this is y and we know that, that L is one-to-one and onto, therefore I give you a g from r of L. And then, I know that I am going to give you from r of L, which is right space of L, therefore we know that a solution exists.

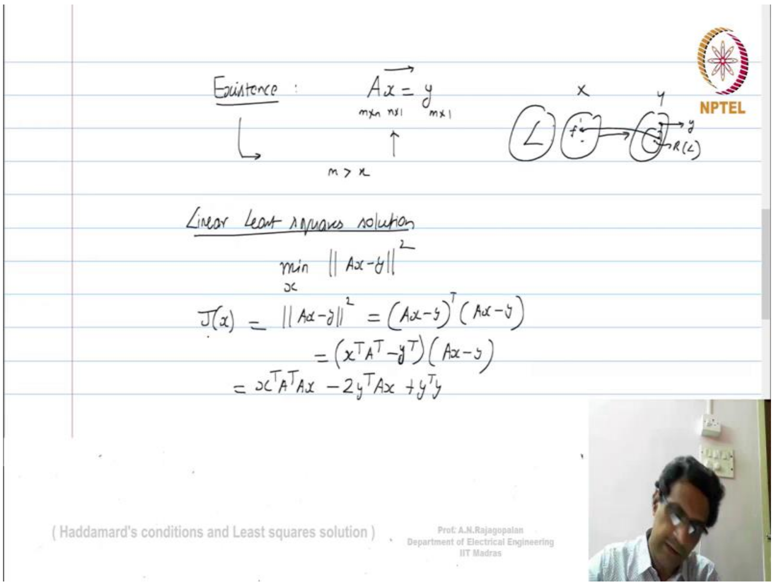
So, let us say, I give you something like g1, and I know that, that gives me a f1, the point is if it gives you a point very close to g1, another observation, just let us say, it is very close to g1 and that one is let us say g2. Now, because of the fact that this map is one-to-one and onto, I know that g2 will not map to f1 even though g1 and g2 are very close, I know that when I go back with a inverse map, I may get some f2. Now, now one-to-one and onto, this will ensure that g1 maps f1 g2, even though g2 is very close to g1 will not map to f1, it will map to f2.

But now this third condition, what it is trying to say is that if g1 and g2 are very close then my solutions f1 and f2 should also be very close to one another, otherwise it is a ill-posed problem, which makes a lot of sense, because what this means is that, what it means is that even if there is a small amount of noise that you add to the observation and that takes you away from the solution in a sort of significant way then it means that, it means that you have an ill-posed problem, because it is numerically unstable.

So, so you would expect that the small change in observation should lead to a small change in the solution, otherwise solving the problem makes no sense. So, these 3 conditions are what are called Haddamard's conditions for well-posedness. And even if one of them is violated then what you have what is called an ill-posed problem.

Now, most inverse problems tend to be one or through or all of there, they would fall, they would suffer from all these 3 conditions typically. And therefore, one needs to find a way out. The most general way out, we will find later, but then some of the simpler things, that we are aware of, that we kind of tend to do is, what we will see first.

(Refer Slide Time: 10:29)



So, let us first examine this notion of what is called existence. And the point is, if as solution is not only guaranteed to exist, like for example, you might know that it is like saying that if I am, so if I am trying to solve Ax equal to let us say y, and if I give you, now the existence thing is more like, now, , so suppose let us say, suppose A is m cross n, this (x) is n cross 1, this (y) is m cross 1, and then suppose let us, suppose this y is not in the column space of A then we know that a solution does not exist. So, we have an inconsistent set of equations. So, in such a case, you may not be able to solve for x.

Solution does not exist. But then my point is we do not really stop there. So, it is like saying that if I go back to the, to my example that I I drew earlier, it is like saying x y and the point is if I have given you a y that came from the right space of L, then of course the solution would exist, but then due to noise or some other reason from outside that I give you outside of r of L, then we know that that is no way, there is no, you saw vector here that can be acted upon by L in order to give you this y.

So, when y is not in the span of the columns of A, and of course, we try to, because for us it is all about matrix and then all the other operators, the linear and the blurring operators, everything is typically a linear therefore, we would want to look upon this L as a, it is kind of a matrix for us. So, we want to solve x equal to A, x equal to y. And, normally what can also happen is the other case that can happen is you have got like m, m much larger than n, which means which means which means that we are going to see overdetermined set of equations and therefore, I mean, a solution may not exist.

In such a case, then what we do when one would expect that if I cannot find the solution, because of the fact that this outside of r of L, what I could then do is find something that is closest to y in r of L. And therefore, when I, once I land here then I can always go back and find out, and the, I can always go back and find out an f for this, because an f for this will certainly exist because once I (fo) I come from y to y dash and y dash rise in there, lies in there (())(12:58) space of L, then I know that y dash will have some solution f dash.

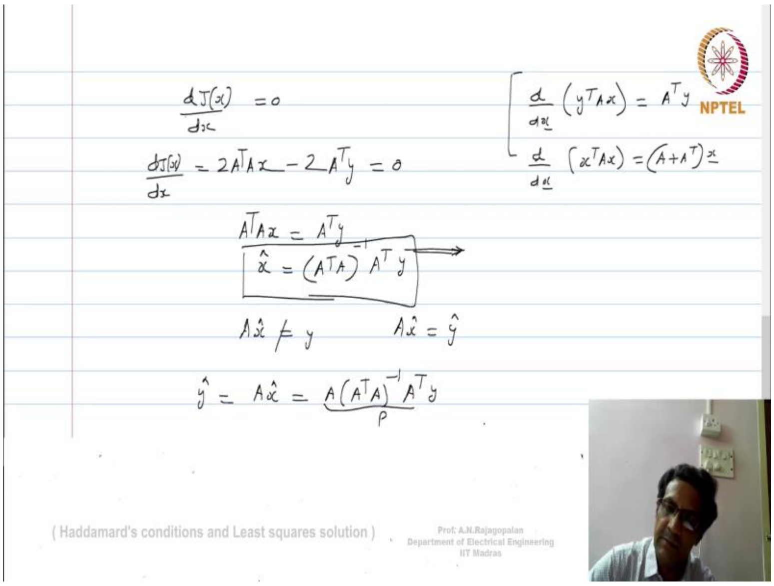
Now, whether the solution f dash is acceptable to us or not is another issue. I mean, how close y and y dash are, and therefore, whether this f dash is something that we can accept is some other issue. But the point is, we typically do not leave a problem unsolved just because it is ill-posed. We try to make every attempt to at least get some kind of an approximate solution.

Now, the most common approximate solution, when you open, when you have a situation like this is what is called Linear Least Squares, linear least squares solution. What is this linear least square solution? So, it says that if you cannot find an x such that Ax is equal to y, if you cannot do that, then try doing this, try minimizing with respect to x, this function norm of Ax minus y square. So, it is trying to say that try to pick a, pick an x such that the norm of Ax minus y is as small as possible. They have been trying to pick x such that Ax is as close as possible to y.

Now, we can actually define this cost as J of x, suppose we define this cost as J of x. And then we write it as Ax minus y norm square, then this becomes Ax minus y transpose into Ax minus y, which in turn becomes x, x transpose A transpose minus y transpose into Ax minus y, which in turn becomes x transpose A transpose Ax minus of course, you should realize that x transpose A transpose y and y transpose A transpose x are both scalars and each is a transpose of the other.

They are both scalars in fact. Therefore, you can write this as minus 2y transpose Ax plus y transpose y. This is y of course, this is y transpose. And in order to find an optimal x, we can do doe j by doe A by doe x, let us equate that to 0. And because of the fact that x is a vector, please note that x is a vector here, even though I have not underline it, underlining it per se, x is a vector.

(Refer Slide Time: 15:13)



And therefore, when you, when you do, when you take Ad derivative with respect to x that is a vector, what you will get is doe j by doe x, for that j, it will turn out to be, see, for example, you should know that in general, the, in general, if you have, if you do d by dx, where x is a vector of y transpose Ax, where y is not, y is not a function of x, y is a vector, A is a matrix, then this will be equal to A transpose y. This is something that is easy to show. We can just take a small example, take a 3 cross 3 or 2 cross 2 and show this.

On the other hand, if you d by dx of x transpose Ax, where instead of y, you got x transpose here. This is equal to A plus A transpose x. This is again something that which you can easily show. So now, if you apply the same sort of principle here, why do we take doe, we can might as well take dJx by dx equal to 0, and therefore, dJx by dx will be, the first term is x transpose A transpose Ax and A transpose A symmetric, therefore this will give you 2 times, A plus A transpose is 2A transpose Ax. And then we have got minus, minus 2.

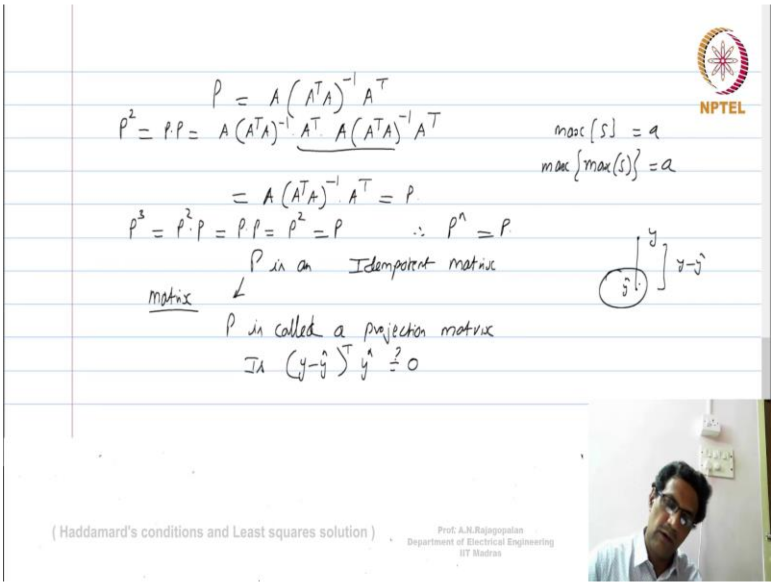
This is y transpose Ax, and we know that d by dx of this is A transpose y. So, minus 2A transpose y. And then their last term is a function of y and therefore, and this simply goes to 0. And therefore, what you have is A transpose Ax is equal to A transpose y, which in turn means that you have got x to be equal to A transpose A inverse A transpose y. Now now this is your x hat. So, the optimal value, the optimal x hat, the optimal x hat that you will get under the least squares condition is this. So, your linear least square solution is this.

And and, of course, at the hand of course, its not true that Ax hat is equal to y. It would not happen because of the fact that y is not in the span of inter-columns panel A. So, Ax hat is not equal to y, but then of course Ax hat will give you some y hat. Ax hat will of course give you some y hat. And then this y hat is such that it discloses to 1. So, if you go back to this figure, you have got y here, you have got y hat in the range spaces of L, therefore y and y hat, so y hat is such that y and y hat and Y hat are very close.

So, this error vector has a smallest norm, because this can be looked upon as a error vector. And therefore, this solution x hat is such that this error vector has a smallest norm. Now, there can occur a situation where, let us say, when, where you may not be able to see invert A transpose A. When you cannot invert x transpose A, it means that means that the that the columns of A are not linearly dependent. When that happens, we had to find another way out, but for the time being, let's just assume that A transpose A is invertible.

And then when to, how to solve the case where the A transpose A is not invertible or in other words, the columns of A are not linearly dependent. Then that case, we will see, we will see soon. So, something like this is, it is called (())(18:25). So, this is your least squares estimate. Now, its interesting to see. No, its interesting to see that y hat. If you see J times x hat, and therefore, this becomes A to A transpose A, the whole inverse, if I just substitute for exams from here, it becomes into A transpose y. Now this is your y hat, lets call this matrix system is P, because P has some say, has some interesting properties. Look at P, now P is A into A transpose A whole inverse into a transpose.

(Refer Slide Time: 18:58)



So, P is equal to A into A transpose A the whole inverse A transpose. Lets try finding out what will be will be P square. P square is P into P, A into A transpose A the whole inverse A transpose is P and then followed up with another p which is A into A transpose A the whole inverse into A transpose. Now, these 2 will cancel each other off and you will end up with A into A transpose A the whole inverse into A transpose, but this, we notice this part is P itself.

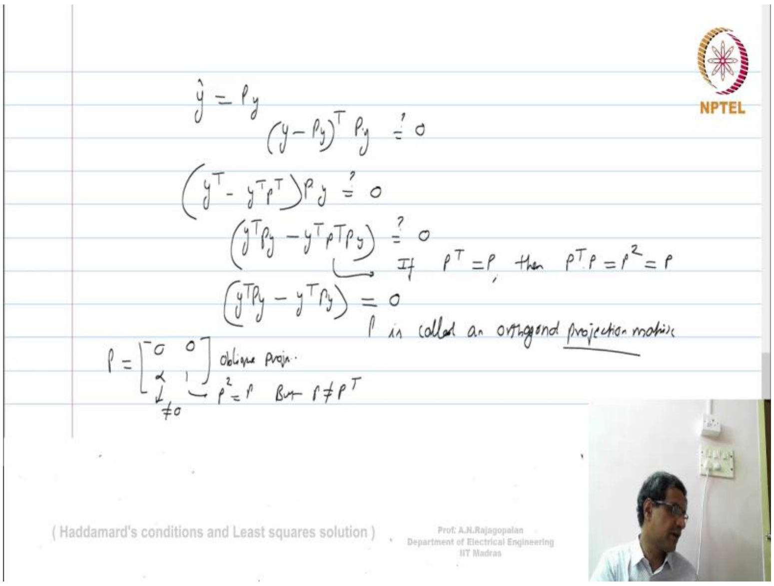
Now, what about P cube? P cube will be P squared into to P but that P square is equal to P therefore this is P into P, this is equal to P square with P square is equal to P. Therefore, any P raised to n is equal to P. And such an operator is called an, is called an Idempotent Operator, it is called the Idempotent Matrix, it is called the idempotent matrix. This is called an idempotent matrix. P is an idempotent matrix because of the fact that P raised to n is equal to P. Now, this idempotent is a variant notion, which is more generic, it does not have to be for matrices.

For example, you can also have nonlinear nonlinear operators, which are idempotent for example, taking max, taking min, and max of a set, max of a max of the set. So, for example, if you have max of S, max of S, if this is equal to A, then max of max of max of s is also max of max of S. It is all see, so if you say the max of max of S is equal to A. So, max, min, all these are the idempotent operations. Now, when because of the fact that P is a matrix, so P is also a matrix for us, it is not simply some idempotent operator, this is a matrix.

And therefore, because of the fact that this (())(20:49) is a linear operator, so P is called is called a Projection Matrix, the projection matrix, a projection matrix. Why? Because of the fact that, it really takes y, y for example in this figure, it takes y and and, it, so P projects y to y hat in r of L. Now, one more interesting thing is, one would want to one would want to check, “Is this error now orthogonal?” So for example, here, in this diagram, where this (())(21:30) is y minus y hat, so one can ask, “Is this y minus y hat or orthogonal to r of L?”

Now, we now have to check that what we can actually do is, if we just look at this and I will take this example. So, suppose we asked the following. Is y minus y hat, y minus y hat transpose? y hat, this equal to 0. Because of the fact that y hat, y hat is an r of L. So, we are asking, so we had this, we had a y which is outside, we dropped it, we took this P, P operator P matrix, that is a projection matrix applied it on y, got a y hat. And here is your error which is y minus y hat. So, we are asking, “Is y minus y hat orthogonal to y hat?” When is this orthogonal?

(Refer Slide Time: 22:33)



Now, substituting for Y hat, so y, so y hat we know is but P P operating on y because P operates on y to get, to give you y hat. Therefore, y minus Y hat is Py transpose y hat, which is actually Py. Is equal to 0? This means that y transpose minus y transpose P transpose Py this is equal to 0. Or in other words, y transpose Py minus y transpose P transpose Py, is this equal to 0? But then we know that, now if P transpose is equal to P, in case P transpose is equal to P, if this is true, if this, if it so happens that P transpose is equal to P, then that P transpose P is equal to actually P square.

Now, if you go back to this matrix, it is easy to check that or verify that P transpose is equal to P. If and when P has this form, so clear. Now, it is clear that that P transpose is equal to P.

And therefore, P transpose P is equal to P square. But then we know that for PP square is equal to P. And therefore, it so happens that you get, you have y transpose Py, and this while thing becomes y transpose Py. And therefore, if this is indeed equal to 0.

Therefore, because of the fact that this error is orthogonal to y hat, such a P, P is also called an orthogonal projection. It is called an orthogonal projection matrix. You can also have something like a, like an oblique, the projection matrix and so on. But then we are more interested in this, which is really an orthogonal projection, projection matrix. For example, something which is oblique would look like, oblique will be like 00 Alpha 1. You can do the check for example, P square is equal to P, but P is not equal to CP transpose, alpha not 0.

Of course, if alpha is, alpha equal to 0 then, then you are back here to an orthogonal reaction. So, such a thing is called oblique, oblique projection, but then this is not something that you will be interested in doing, but just for the information sake.