Hello, everyone, welcome to today's lecture. We will talk about basics of optimization.

Outline: We'll start introducing concepts, including what is convex vs non-convex optimization, formulate the problem, talk about the optimal conditions for optimizers, and how to deal with constraints using Lagrangian functions. The purpose of today's lecture is not to introducing the whole field of optimization. As you can see, we can spend a whole semester to talk about optimization. It is a quite deep subject. The purpose of this lecture is just introducing enough mathematical background, so we can develop machine-learning algorithms. you're not required to master every single detail in the lecture.

- Concept
- Convex/concave function
- First-order condition
- Second-order condition
- Langragian and dual function
- KKT conditions

Motivation: The motivation for studying optimization problem in machine learning (ML) is quite clear since we have seen quite a few examples in previous lectures. For example, in clustering we're solving optimization problem to find the optimal centroid and examine function π to minimize the sum of the squared distances between data points and the centroid it's being assigned to. We developed a heuristic for solving this, which is the k-means algorithm. The problem itself, it's actually difficult to solve. It is a NP-Hard problem. PCA is also solving an optimization problem where you're trying to find the weight vector or the direction W to project the data to maximize the variance of the projection, which preserve information. Subject to the constrain, this weight vector itself is normalized. In maximum likelihood problem, we have seen examples so far. We're trying to find a parameter to maximize the log-likelihood function of the data and your research probability model. In the end, it also begins solving an optimization problem.

Clustering:

$$\min_{c,\pi} \frac{1}{m}\sum_{i=1}^{m}\left\|x^i - c^{\pi(i)}\right\|^2$$

PCA:

$$\max_{w:\|w\|\le 1} \frac{1}{m}\sum_{i=1}^{m}\left(w^\top x^i - w^\top \mu\right)^2$$

MLE:

$$\theta = argmax_\theta \log P(\mathcal{D}|\theta) = argmax_\theta \log \prod_{i=1}^{m} P(x^i|\theta)$$

Motivation: The optimization problem is everywhere in ML. It is a foundation to ML. with statistics and geometric approaches, optimization provides a way to model and solve the problem, such as clustering PCA, MLE, regression, SVM, deep learning, and how do we tune the parameters of neural networks to best fit the networks to data and how do we fit this quickly? These are all subjects being studied in optimization with various solution methods. The k-means, for example, is a heuristic for finding the clustering result for some NP-Hard problem. There are special methods such as PCA & EM. PCA resolves in closed-form expression in the end, we find the Eigen decomposition of the covariance matrix, or singular value decomposition of the big matrix. There are general methods such as gradient descent and stochastic gradient descent. These are the method that can be generally applied to various optimization problem. We'll talk about these later on as well And these general methods are probably the most widely used in machine learning field for solving optimization problem. There are three components involved in optimization problem, including the decision variables, for example, in solving MLE, decision variables are the data, so we're trying to find θ to maximize the log-likelihood function, the constraints, sometimes we can't play with θ, but we have constraints to our variables, and the objective function and maximum likelihood problem, the constraint function. The objective function is a lot like it function. The main watershed in optimization is the difference between convex and non-convex optimization problem because the complexity involved in solving this problem is quite different and we have their different strategies for solving them. Basically, when you have convex problem, we're pretty happy. We can have a polynomial solvable problem and Nino Southern pretty efficiently. For non-convex problem, we have to come up with a lot of times case-by-case strategy to solve them.

Optimization is foundational to ML
- Modeling (together with statistics and geometric models)
  - Clustering, PCA, MLE, regression, support vector machine, deep learning, ...
- Solutions methods
  - Heuristics (e.g., k-means)
  - Special methods (e.g., PCA, EM)
  - General methods (e.g., gradient descent, SGD)

Three components
- Decision variables
- Constraints
- Objective function

Main watershed in optimization
- Convex/Nonconvex optimization (different strategy for solving them)

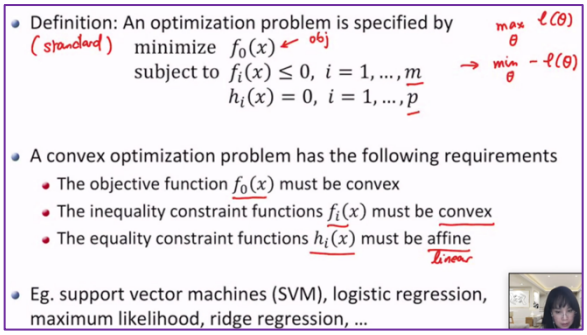
Convex vs. Non-convex: We have seen this concept in previous lectures. Let's make it more formal. for example, linear regression is a convex problem. The problem means a picture describing it. Your objective function is convex. you can roughly imagine picturize as a bowl, that's hot water. You can see that if you try to minimize a convex function, we're trying to find the state, this point here. If you don't have constraints, you're trying to find. But if you have constraints, if the constraint says somehow doesn't include this solution, you will end up solving a data point here at the boundary of your physical domain. But in general, you can see convex problem have a pretty nice property that you can find the global optimal solution, which is data point in polynomial time. this is the algorithmic term to say that the problem can be solved efficiently. For convex problem, gradient descent or many acceleration methods including Newton's method can converge to global solution. that's the good news. Convex problem is relatively easy, and you can find good solution. Non-convex problem is also very common in ML, for example, clustering, the first problem we see in this class is non-convex problem because you have to consider in principle, all the possible enumerations of data points to the cluster. It is combinatoric in nature. it's very expensive to solve. The MLE for GMM is also non-convex. We've seen it last time. we use EM algorithm which come up with a sequence of lower-bound approximation and we tried to solve that approximately. For non-convex problem, in general, we cannot find the global solution in polynomial time. if really fun to find an absolute best solution to clustering,

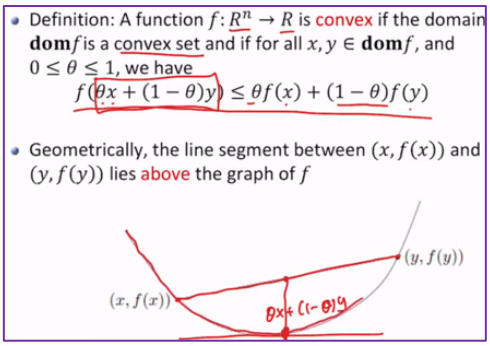
Convex problem
- e.g. linear regression
- Can be solved efficiently (find global optimal solution) in polynomial time
- Gradient descent (or many acceleration methods e.g., Newton method) can converge to global solution

Non-convex problem
- e.g. clustering, maximum likelihood for GMM
- Cannot find global solution in polynomial time (NP-hard)
- Use heuristics to find local optimal solution

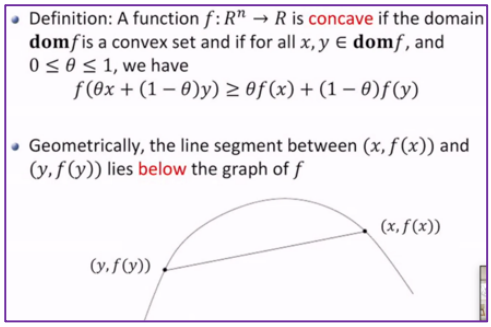
we have to numerate all the possible combinations of data points assigning to centroids. It is non-polynomial, we call them NP-hard for such problems, non-polynomial solvable. For many non-convex problems, in the end, we develop some heuristic efficient algorithm, not finding the global solution, but finding a local optimal solution. for example, in this case, you can see the objective function the shape is very different. it has a local value here. Research problem a naive for vanilla version going to descend, it's likely to be trapped if you start from here, then we trapped as local solution, and this is the actual global solution we want to derive. In the non-convex problem, you cannot easily get a global solution enters, likely that you will end up converging to one of the local minimizers. Sometimes it's good enough for your problem.

Optimization: A general optimization problem is typically specified by minimizing objective function and subjecting to some constraints, sometimes you don't have constraints, it's totally fine. If you have two constraints, they typically can be written into two types: inequality and equality constraints. Suppose you have m inequality and p equality constraints, not a big number of samples, dimension, or anything. What if we have a maximize problem to objective function? For example, in maximum likelihood, we're aiming to maximizing the log function. You can convert it into the equivalent, standard form by looking at minimizing, minus of functions. Let's call this to the standard form of optimization problem. What do we mean the optimization problem is complex? It has to satisfy the following requirement. The objective function f(x) is convex. The equality constraints and inequality constraints have to satisfy that inequality constraints are affine. this is a formal way of writing the fact that hi(x) is linear, This fi(x), the equality constraints have to be convex. that's the requirement. We'll define this more formally in the next few slides. Some examples of optimization problem including SVM for writing machine, maximum likelihood ratio regression, linear regression, so on, so far.



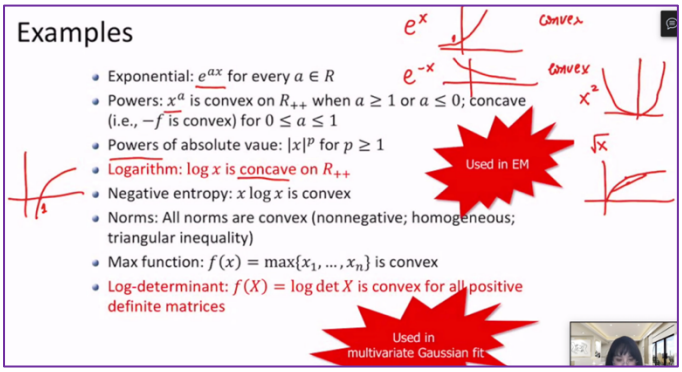


Convex functions: What is convex function? This function can be multi-varied as well and that's from the n-dimensional input to a scalar, its function is convex. If first the domain is a convex set, we'll define what is convex set in the next few slides as well and it also satisfy this requirement that f(θx+(1-θy) is less than or equal to the linear combination of the function value evaluated at data points (x, y). Geometrically, what does this mean? Suppose you have a pair of data: (x, f(x)) and (y, f(y)). if you find any point in between x and y (bottom point, θ is between 0 and 1) and if you look at the function value for any point between x and y, we can compare this point with a line segment connected the two points. That means, if we compare this function value at the middle point, The point corresponds to θx+(1-θ)y. it's always smaller than the point above. this is a general definition of a convex function. Pictorially, you can imagine you have a bowl-shaped function that holds water.

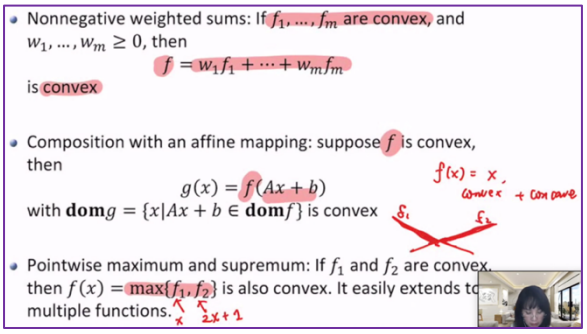


Concave Functions: concave function is the opposite of convex function. you can just imagine if I flip everything in my definition and I've had a concave function.

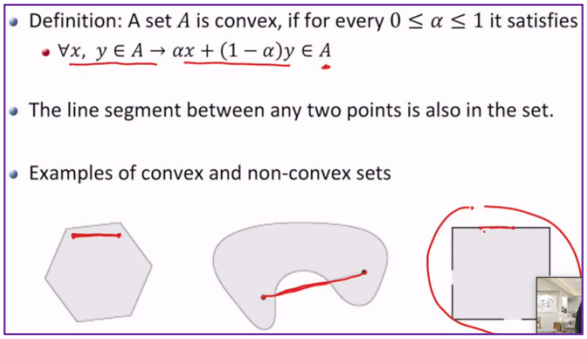
Examples: For example, exponential function e^{ax} is it convex or concave? consider a=1, and when x is equal to 0, this equal to 1. this is what e^x looks like, a convex function, when a is less than 0, you have exponential decay, also convex. exponential function is convex in general. power's function x^a, what a is greater than or equal to 1. let's look at x^2, a quadratic function. it satisfy the requirement for a convex function. what about when a is less than 1 is going to be concave? for example, the square root of x, and what this is like. This is going to be a concave function suspending upwards. if you form any linear combinations of data points, then the function value is going to be lying above the line segments joining that. Then power's function is going to be convex. The log function is concave. let's draw this log function. This is 1. it's bending backwards. the rest of it has more examples of convex vs concave function. This will help you to decide if the objective function is convex, or if you're facing a convex optimization problem or not. we have seen many of these instances being used, for example, in that in EM or multivariate Gaussian.



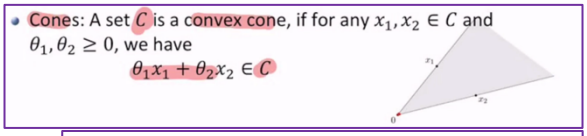
Operations that Preserve Convexity: there are some operational preserved convexities, for example, if you form a linear combination of convex function, the result is also convex. Composition with an affine mapping, if f is convex, and you apply f(Ax+b) on a linear function, the result g(x) is also convex. The linear function is both convex and concave. if we consider f(x) = x as a linear function, this is both convex and concave. This is only instance example that is a function both convex and concave. So, If you take the maximum of two functions, for example, two linear functions, f_1 and f_2, let's say f1 = x and f2 = 2x+1, then basically you have a situation like this (drawing), this is my f_1, f_2. taking the maximum, you end up being taking the higher of the two lines, you end up having a piecewise function that is convex.

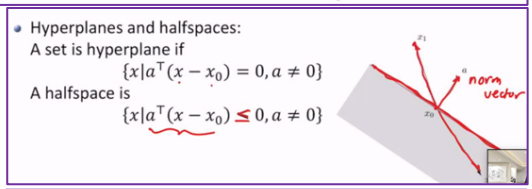


Convex Set: we talk about the concept of convex function, what is convex set? The definition is important to understanding constraint optimization problem because essentially the constraints defines a set. A set is convex if you take any two points in the set, and you connect these two data points and the points in-between on the line segment connecting the two points is also inside your set. if you take any two points in your set and form a linear combination, means look at any points in-between them. If that's also inside your set, then you're set is convex. this is example of convex set. this is the example of an uncommon set because I take these two points, I connect the point and I end up having one segment of my line outside of my set. it's not convex. look at this example, is this convex or non-convex? here I have a gap. This gap means this particular boundary doesn't belong to my set. this set is actually nonconvex because, if I take two points here, I join them, then I end up having points not going onto my set in some special cases.



Common Convex Set: here's some examples of common convex set way where typically see in machine learning. One is called the cones. so, a set is a convex cone if for any x_1, x_2 in this cone, and you form this linear combination for θ and θ2 get non-negative, and the new combination also belongs to your set is called a cone. you can imagine, basically this is like an array that lights out that forms the cone. Imagine you have a flashlight here and you turn it on, then the light itself defuse, that forms a cone.

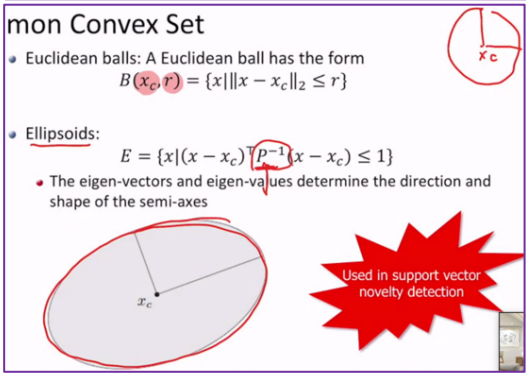




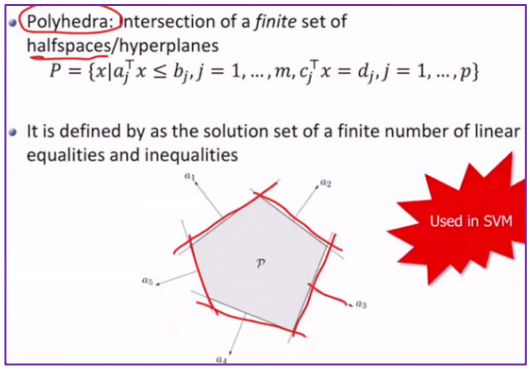
Another example is hyperplane and half space. a set is hyperplane if you consider anything, lies on a line. this example is a hyperplane. this a is normal vector. you're looking at any point that are the difference of x and x_0 is orthogonal to your norm vector. that's basically this one. This defines a hyperplane. a half space is defined using this new expression as well, but it's one side of your hyperplane. For example, if I write it this way, that means the vector have to be lying on one side of your norm vector. This is your norm vector. in this example you can see this x_2 will be in the half space, but x_1 is not half space. it defines half of the space, entire space.

Euclidian ball is also a convex set and as you can see, it's defined to be anything centered around the center with certain radius r. you can easily validate the satisfied requirements for compact set. Euclidian ball is a ball centered at x_j, so the shape is isotropic in different directions.

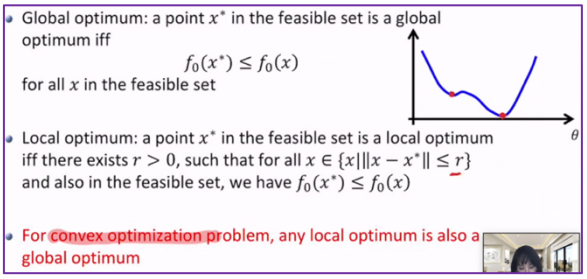
Ellipsoid is also convex with non-isotropic shape. Ellipsoids is more general. you have ellipsoids in your space. along different directions, it may have a different radius, basically like this. the P inverse defines the shape of ellipsoid. in fact, the eigenvectors and eigenvalues of the p here will determine the direction of the shape of your signing axis. have you seen this before? If you think about so-called Mahala Nobis distance, we actually have seen a metric like this. we're going to use this in some machine learning algorithms such as novelty detection and SVM when we talk about them.



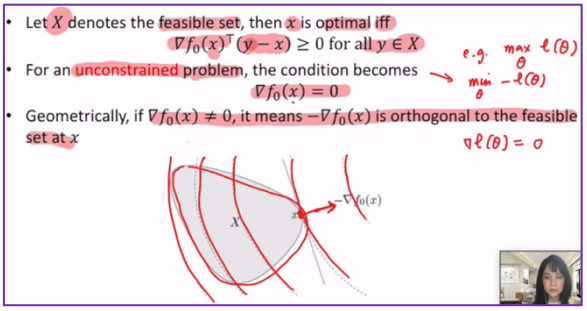
if you have a bunch of a finite sets defined half spaces or hyperplanes, that defines so-called polyhedra. This is a general form of a convex set. this is an example of a polyhedron. You have five different hyperplanes, defines the half space in the middle. that's a linear equalities (LE) convex set. And this is also used in SVM derivation.



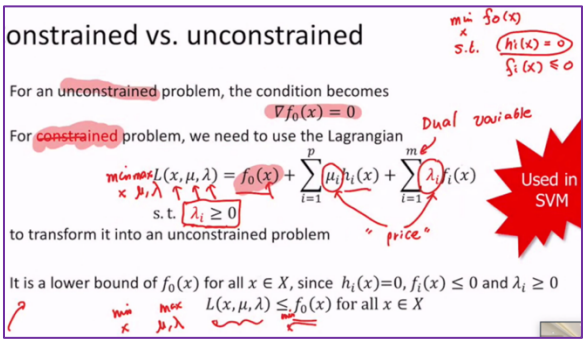
Global vs Local Optimum: A global minimum is a data point in the feasible domain inside to satisfy the constraint. if we do have constraints, it is a global minimum means that it is better than anywhere else. Any other points, it achieves absolute smallest value on your objective function. A local minimum is a point that is better than any other point in a neighborhood. Neighborhood was with this r. If there exists such r that you're better than anybody else in this particular neighborhood, then this is a local minimum. this and this are both local minimums. a global minimum is also a local minimum. It has to be first satisfied local minimum condition to be a global minimum. the nice thing about convex optimization problem is that any local minimum is also a global minimum. That's why in convex optimization, it is fairly easy to find the global optimal solution because you can just define a local solution.



First order optimality condition: This is a condition that the optimizer need to satisfy. as you can see, this is also the way to find the optimal solution. In fact, we have used this for solving our problems. For unconstrained problem, the optimality condition becomes to find the gradient of your objective function and evaluate this at particular x. Set the gradient to be 0 and find x, x is going to be a minimizer, and at optimal point, the gradient basically will vanish. that's the requirement. for example, we have used this for solving the maximum likelihood problem. to maximum likelihood, we try to maximize the log-likelihood function (max l(θ). so, this is equivalent to minimizing the minus of the log-likelihood function (min -l(θ)). If you want to write this in standard form, we have suggested you find the gradient of the log-likelihood function ∇l(θ) and set it to be 0, then solve for the x for this to hold. That gives the maximum likelihood estimator. that's exactly using the first order optimality condition. In general, the first order optimality condition read as follows; If you have a feasible domain or feasible set, this is a set containing all the possible physical solution. these are the candidates for solution. If you have constraints in your problem and these acts are going to be all the points that satisfy your constraint. Then x is optimal if and only if this satisfies. it says you can find the gradient of your objective function (a vector). if the inner product of the gradient vector at the point x with any deviation from your x. y is another point different from your x inside your visible set. If the inner product with the deviation from your x is always non-negative, then this x is optimal. intuitively, what does this mean? This means that if you deviate from x, then you will end up increase your objective function value. that means your x is absolutely the smallest of possible points you can have for the objective function. if you look deeper, basically, geometrically, this means that the minus of the gradient vector is orthogonal to the feasible set X. imagine this picture. Imagine the high-dimensional problem, these contour plots are the equal value the contour lines of your objective function. this x is your physical domain. You consider all the possible points inside this feasible domain as possible solution. this is the gradient of your objective function, then you can see at optimal point, you're trying to minimize your app, then minus the gradient vector should be orthogonal to your feasible point at this particular point as the geometric interpretation. the first outer optimality condition is a very general condition, and you can use this to find the optimal solution in general.

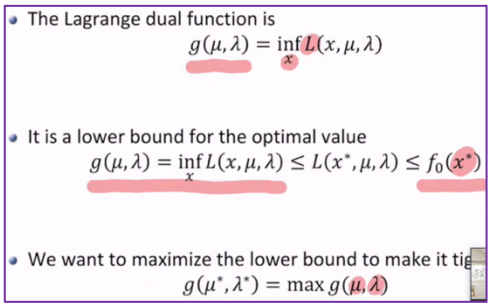


Constrained vs unconstrained: how do we use it to solve the optimization problem the ML form? And let's make it more specific. for unconstrained problem, like we mentioned, the condition is relatively simple. We just look at objective function, set it to 0, and this gave us a set of equations. From this, we can find the optimal solution. For constrained problem, we have constraints. How do we deal with it? For example, in EM algorithm for GM models, one of steps in the middle of the derivation, we have to find the optimal parameter updates for the πi, for the rate of the Gaussian components and the π need to satisfy constraints, you have to sum up to 1. that gave us a constraint. we have used the so-called Lagrangian function to solve that problem. the same thing here. the Lagrangian function is based on the idea of, let's try to combine the objective function with the constraints together in one function, so we can solve the unconstrained optimization problem. if you have two types of constraints. one is the inequality constraints, you have to satisfy fi(x) less or equal to 0, then the equality constraints are hi(x) = 0. λi is the Lagrangian multiplier for the inequality constraints to satisfy the requirement they are not negative. combining the objective function with linear combinations of the two types of constraints will give you one function. In terms of the x, the decision variables, and two sets of multipliers that we introduce. you can see μ and λ, depending on how many constraints you have you can have equal number of **dual variables**. The dual variables introduced into your problem, and they became the variable to the Lagrangian function as well. this way we're going to try to solve the one problem involves x, μ, and λ without any constraints. The only control you have is these have to be normality, but that's fairly easy to deal with constraints. what's the intuition behind this? basically, we're trying to convert a problem with constraints into a non-constrained problem. think about when you write it down here. you're trying to minimize

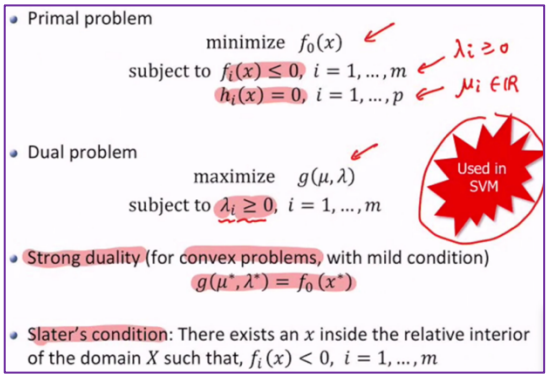


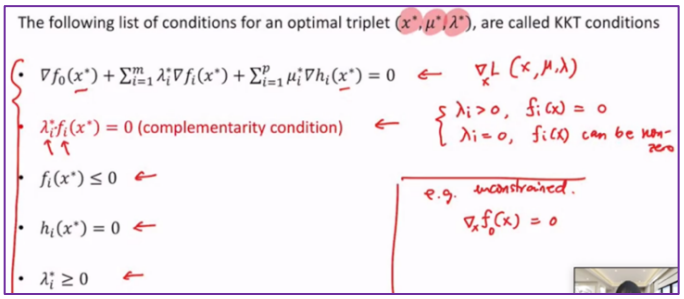
$f_0(x)$, subject to the constraints that hi(x) = 0 and f(ix) is less or equal to 0. if imagine that if I can allow your x to violate these constraints a little bit, if I could pay a price, if I log into while it is constraint a little bit if I pay a price. I'm going to try to solve this problem allow it to some violation. this μ and λ have the interpretation of the price of the violation. we can see if h_{ii} is not equal to 0, then this new i can interpret it as how much penalty I will charge on you for that amount of violation. similarly, if your fi(x) becomes positive, then I will start to charge you a price. These are called the shadow prices in some contexts as interpretation of that. as you can see, of course, I don't want to evaluate these constraints. I want to satisfy these constraints exactly. the motivation is that let's try to solve the problem to minimize the Lagrangian function, because that's not original Jackie want to minimize map to 0. meanwhile, I will try to charge my violation by setting the price as high as possible. If I charge the penalty to be large in the end, there's no incentive to deviate or valuate any of these constraints. If that happens, then I end up having the same solution as before satisfying the constraints. Exactly. that's the idea of the Lagrangian function. in fact, mathematically, you can also show that this Lagrangian function provides a lower bound to the original objective function. you can see if I try to find a tightest possible lower bounds, I should choose these two variables to maximize my Lagrangian function as large as possible. that's the motivation, then if I minimize on both sides respect to x, then maximize μ and λ, sometimes I will be able to get the same results. bottom line is, Lagrangian function is introduced for you to handle problem with constraints. Convert a problem into an unconstrained problem by introducing these new variables, also known as the Lagrangian multipliers. this way will make it easier to solve.

Lagrange dual function: This is one of the most important techniques in solving optimization problems to introduce this branch of functions. first, we're going to introducing this Lagrange dual function. Suppose I look at Lagrange dual function, I minimize this respect to x. as I minimize it, I evaluate at Lagrange function at the minimizer. in the end I end up having a function that's only a function of the multipliers. g, μ, and λ. you can see that, this g, which is the minimum of the Lagrange function. it's still going to be a lower bound of my optimal value evaluated at the optimal solution. this g is a lower bound. like we mentioned that we can try to make this lower bound as tight as possible by make it large. we're going to try to solve the problem of maximizing the g, the lower bound, which is back to this set of two variables, so that's the idea here.



Primal and Dual problems: The primal problem is the original mild problem we are trying to solve minimizing the objective function satisfy the possible inequality and equality constraints. we can convert this to solving the dual problem, which is to maximize the lower bound of the original objective function subject to the constraint that these dual variables, the crunch multipliers for inquiry constraints. this λ is one variable for each constraint. you have to be non-negative. satisfy these constraints. μ are the dual variable for the equality constraint. this doesn't need to be exactly non-negative. they can be any scalar. the dual problems solved the problem of finding the height is the lower bound or finding the highest possible price you can charge to make your lower bounds tide such that there's no incentive to deviate and violate your constraints. that's roughly in the intuition here. I'm trying to solve the two sets of problems. primal and dual, and how are they related? As you can see, if I solve the problem using the primal. I have these constraints. it's not very easy to deal with it. The solution quick happening on the boundary or maybe happens in interiors. These constraints are going to change where my solution is going to occur. But if I look at the two problems, sometimes it's going to be a little bit easier because my constraints is relatively easy to handle. sometimes what you can do is you can write the problem in the Lagrange function form like this, then let's try to find the dual function by minimizing the bunch of function with respect to the original variables, x. From there we can convert this into a dual problem, then try to solve the dual problem. Once we have the dual problem, typically the x optimal solution is expressed in terms of these two variables. you can convert the optimal dual solution into the optimal x. that's the strategy. In fact, this is strategy is going to be used in solving SVM, the support vector machine, as you can see. these two are closely related. In fact, there's something called the strong duality. that basically means that by taking this approach, finding the tightest lower bound and the strong duality condition, you will recover the exact original optimal solution to the primal problem. the optimizer, that the optimal g value at the optimizer is equal to the optimal of 0 at the optimal x is to meet basically. again, talking about high-level, basically, as I can find a set of prices for the μ and λ such that in the end, there's no deviation from these constraints will end up getting the exact optimal solution to your primal problem. This is also called the zero-duality gap sometimes. And. you'd say, where is this strong duality is going to satisfy? for convex problem, convex optimization is typically easy to satisfy this constraint. In general, there's something called the Slater's condition. That's the sufficient condition for you to have a strong duality. Meaning that you solve this due problem. that's going to recover the original exact solution to the primal problem. we're not going to the very deep mathematics here, but just to talk about the idea, you know what mathematical tools and theories are available there. If we ever going to need something deeper, that's the place to look for.





KKT Optimality conditions: The last thing we're going to talk about is the completely general condition for finding optimal solutions involving constraints. this is called the KKT condition. KKT is a shorthand, an acronym for three authors who discovered the conditions. the set of condition basically is required for optimal triplets, the optimal dual variables, which will be satisfied for them to be the optimal solution. Meaning zero duality gap. Meaning strong duality. the first condition is basically requiring the Lagrangian function to have a vanishing gradient at the optimal x. you can see this basically is the gradient of L with respect to x. And the second condition is a little bit peculiar, is called the complementarity condition. this basically says the pair of durable with corresponding inequality constraints have to be 0. If you multiply them, they have to be 0. what does this mean? This means that if λi is greater than 0, if you're charging a price for that constraint, then the constraint had to be 0. if you're charging a price, and that means you have to satisfy that constraint at this binding has to be exactly 0. If λ_i=0 and there's no price for any deviation, then this can be non-zero. Basically, λi the durable and the corresponding inequality constraints have to be complimentary to each other. these two are nothing but the original constraints that x need to be satisfied. this is nothing but the constraint that's deliverables for the inequality constraints are satisfied. why did we introduce this condition? For example, if we don't have any constraint for unconstrained optimization, we all know how to solve it. You say I am going to find the gradient of my f 0, the original objective function respect to x and set it here. From here, I can solve for the optimal x. We have constraint is going to be more complicated. that's contribute this KKT condition. You can read this KKT optimal condition as equivalent of generalizing this very simple, that the gradient to 0 and solve for the optimal solution. This is the generalization of that in the case where you have constraints. if you have constraints, you have to solve the KKT conditions to find the optimal solution, including optimal x and optimal dual variables. this typically involves a set of equations, equality, and inequality. that converts, optimum problem, solving optimization problem into solving a system of equations and equalities.

Summary: that's it for this part of the lecture. like I said, optimization itself, it is a very deep theory, very deep topic. We can easily spend a whole semester talking about it. What we're doing here is hopefully provide you enough background information to, derive machine learning algorithms. As you can see, they have used some simple optimization in the past and moving forward. I use a lot more as well. if you feel that you couldn't understand everything in this lecture yet, that's totally fine because you're going to see this show up a few times in different places. this is starting point. That's it for today.