Some Mathematical Methods in Machine Learning

Jean Gallier

CIS Department
University of Pennsylvania

jean@cis.upenn.edu

December 5, 2023

This talk is based on some lectures given in the online course MCIT 515.

This talk is based on some lectures given in the online course MCIT 515.

This course was designed in collaboration with Jocelyn Quaintance, whom I warmly thank.

This talk is based on some lectures given in the online course MCIT 515.

This course was designed in collaboration with Jocelyn Quaintance, whom I warmly thank.

I also thank Carlos Esteves, Stephen Phillips and Yulai Wang. Without their help, this course would have never been completed.

1. Motivations

Two central problems in machine learning are

1. *Motivations*

Two central problems in machine learning are

(1.) *Data fitting* (or *learning a function*).

1. *Motivations*

Two central problems in machine learning are

(1.) *Data fitting* (or *learning a function*).
(2.) *Data classification*.

1. *Motivations*

Two central problems in machine learning are

(1.) *Data fitting* (or *learning a function*).
(2.) *Data classification*.

For this introduction we focus on the more classical problem of data fitting.

Fitting Points in the Plane

Assume we have some data points in the plane given as a list of m coordinates

$$((x_1, y_1), \ldots, (x_m, y_m)), \quad x_i, y_i \in \mathbb{R}.$$

Fitting Points in the Plane

Assume we have some data points in the plane given as a list of m coordinates

$$((x_1, y_1), \ldots, (x_m, y_m)), \quad x_i, y_i \in \mathbb{R}.$$

The figure on the next slide shows an example of 100 points in the plane.

Fitting Points in the Plane

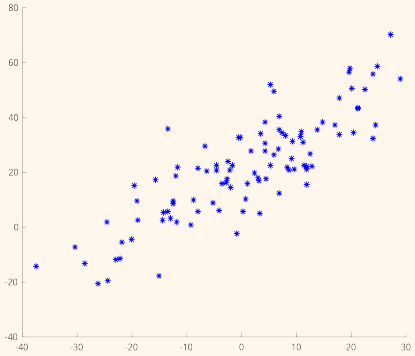

Figure 1: A data set of 100 points in the plane.

Learning an Affine Map

We are looking for a function $f\colon \mathbb{R} \to \mathbb{R}$ such that $f(x_i) = y_i$ for $i = 1, \ldots, 100$.

Learning an Affine Map

We are looking for a function $f\colon \mathbb{R} \to \mathbb{R}$ such that $f(x_i) = y_i$ for $i = 1, \ldots, 100$.

The simplest kind of function is an *affine map*, that is, a map of the form

Learning an Affine Map

We are looking for a function $f\colon \mathbb{R} \to \mathbb{R}$ such that $f(x_i) = y_i$ for $i = 1, \ldots, 100$.

The simplest kind of function is an *affine map*, that is, a map of the form

$$f(x) = wx + b,$$

for some real numbers w, b. The number w is called a *weight*.

Learning an Affine Map

We are looking for a function $f\colon \mathbb{R} \to \mathbb{R}$ such that $f(x_i) = y_i$ for $i = 1, \ldots, 100$.

The simplest kind of function is an *affine map*, that is, a map of the form

$$f(x) = wx + b,$$

for some real numbers w, b. The number w is called a *weight*.

The numbers w and b must satisfy the 100 (affine) equations

$$y_i = f(x_i) = wx_i + b.$$

Learning an Affine Map

In general, unless all the points lie on the same line, the above linear system has no solution.

Learning an Affine Map

In general, unless all the points lie on the same line, the above linear system has no solution.

We are asking for too much. A more promising approach is to *minimize the error*.

Learning an Affine Map

In general, unless all the points lie on the same line, the above linear system has no solution.

We are asking for too much. A more promising approach is to *minimize the error*.

But what is the error?

Learning an Affine Map

In general, unless all the points lie on the same line, the above linear system has no solution.

We are asking for too much. A more promising approach is to *minimize the error*.

But what is the error?

Gauss and Legendre proposed a method over 200 years ago: the *least squares method*.

What is the Error?

Every equation $y_i = wx_i + b$ can be written as

$$y_i - wx_i - b = 0.$$

Think of $y_i - wx_i - b$ as an *error*.

What is the Error?

Every equation $y_i = wx_i + b$ can be written as

$$y_i - wx_i - b = 0.$$

Think of $y_i - wx_i - b$ as an *error*.

In the method of least squares, *the error (or loss) is the sum of the squares of the errors*:

$$\sum_{i=1}^{100} (y_i - wx_i - b)^2.$$

Least Squares Solution

Here the least squares solution for our data set of 100 points.

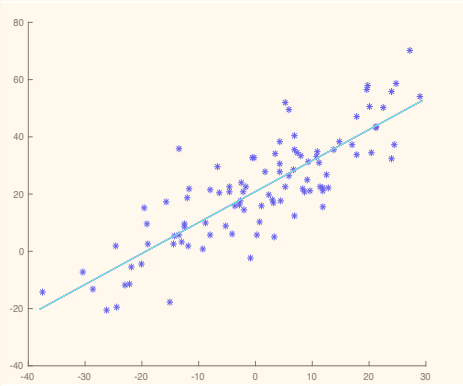

Figure 2: The least squares best fit.

Fitting Points in $\mathbb{R}^n$

We can generalize the problem to data in $\mathbb{R}^n$.

Fitting Points in $\mathbb{R}^n$

We can generalize the problem to data in $\mathbb{R}^n$.
Assume we have some data given as a list of m pairs

$$((x_1, y_1), \ldots, (x_m, y_m)), \quad x_i \in \mathbb{R}^n, y_i \in \mathbb{R}.$$

Fitting Points in $\mathbb{R}^n$

We can generalize the problem to data in $\mathbb{R}^n$.
Assume we have some data given as a list of m pairs

$$((x_1, y_1), \ldots, (x_m, y_m)), \quad x_i \in \mathbb{R}^n, y_i \in \mathbb{R}.$$

We wish to learn an affine map f: $\mathbb{R}^n \to \mathbb{R}$ of the form

$$f(z) = w_1 z_1 + \cdots + w_n z_n + b,$$

with $z = (z_1, \ldots, z_n)$ and where $w_1, \ldots, w_n \in \mathbb{R}$ are *weights*.

It is convenient to denote the quantity $w_1 z_1 + \cdots + w_n z_n$ (an inner product) as $z^\top w$.

The Euclidean Norm (or ℓ^2-Norm)

The *Euclidean norm* (or *ℓ^2-norm*) of a vector $z = (z_1, \ldots, z_n) \in \mathbb{R}^n$ is defined as

$$\|z\|_2 = (z_1^2 + \cdots + z_n^2)^{1/2} = (z^\top z)^{1/2}.$$

The Euclidean Norm (or ℓ^2-Norm)

The *Euclidean norm* (or *ℓ^2-norm*) of a vector $z = (z_1, \ldots, z_n) \in \mathbb{R}^n$ is defined as

$$\|z\|_2 = (z_1^2 + \cdots + z_n^2)^{1/2} = (z^\top z)^{1/2}.$$

The *least squares problem* is find $w \in \mathbb{R}^n$ that minimizes

$$\|\xi\|_2^2,$$

where $\xi = (\xi_1, \ldots, \xi_m)$ is the vector given by

$$\xi_i = y_i - x_i^\top w - b.$$

Pseudo-Inverse

It turns out that there is a *unique solution* $\begin{pmatrix} w \\ b \end{pmatrix}^+$ *of least* ℓ^2*-norm*.

Pseudo-Inverse

It turns out that there is a *unique solution* $\begin{pmatrix} w \\ b \end{pmatrix}^+$ *of least* ℓ^2*-norm*.

Furthermore, this solution $\begin{pmatrix} w \\ b \end{pmatrix}^+$ is expressed in terms of something called a *pseudo-inverse*.

Pseudo-Inverse

It turns out that there is a *unique solution* $\binom{w}{b}^+$ *of least* ℓ^2*-norm*.

Furthermore, this solution $\binom{w}{b}^+$ is expressed in terms of something called a *pseudo-inverse*.

In our case

$$\binom{w}{b}^+ = A^+ y,$$

where A^+ is the pseudo-inverse of the matrix

$$A = \begin{pmatrix} x_1^\top & 1 \\ \vdots & \vdots \\ x_m^\top & 1 \end{pmatrix}.$$

Pseudo-Inverse

The pseudo-inverse of a matrix A can be computed in terms of its *singular value decomposition* (or *SVD*).

Pseudo-Inverse

The pseudo-inverse of a matrix A can be computed in terms of its *singular value decomposition* (or *SVD*).

The SVD and the pseudo-inverse will be discussed extensively later.

Pseudo-Inverse

The pseudo-inverse of a matrix A can be computed in terms of its *singular value decomposition* (or *SVD*).

The SVD and the pseudo-inverse will be discussed extensively later.

The solution given by the pseudo-inverse is not always desirable or too expensive to compute.

Pseudo-Inverse

The pseudo-inverse of a matrix A can be computed in terms of its *singular value decomposition* (or *SVD*).

The SVD and the pseudo-inverse will be discussed extensively later.

The solution given by the pseudo-inverse is not always desirable or too expensive to compute.

Another method is to *penalize the ℓ^2-norm of w*.

Ridge Regression

The problem to solve is the following minimization problem known as *ridge regression*:

Ridge Regression

The problem to solve is the following minimization problem known as *ridge regression*:

$$\begin{aligned} &\text{minimize} && \|\xi\|_2^2 + K\|w\|_2^2 \\ &\text{subject to} \\ & && y_i - x_i^\top w - b = \xi_i, \quad i = 1, \ldots, m \end{aligned}$$

where K is positive constant.

Ridge Regression

The problem to solve is the following minimization problem known as *ridge regression*:

$$\begin{aligned} &\text{minimize} && \|\xi\|_2^2 + K\|w\|_2^2 \\ &\text{subject to} && \\ & && y_i - x_i^\top w - b = \xi_i, \quad i = 1, \ldots, m \end{aligned}$$

where K is positive constant.

This time there is a unique solution given in terms of the matrix X whose rows are the (row) vectors $x_i^\top$. For simplicity assume $b = 0$.

Ridge Regression

The unique minimizer is given by the expression

$$w = X^\top (XX^\top + KI_m)^{-1} y.$$

Ridge Regression

The unique minimizer is given by the expression

$$w = X^\top (XX^\top + KI_m)^{-1} y.$$

The matrix

$$XX^\top + KI_m$$

is particularly nice because it is *symmetric positive definite*. There are more efficient methods for solving linear system involving SPD matrices. We will study such matrices extensively.

ℓ^1-Norm and Lasso Regression

One of the weak points of ridge regression is that when the dimension n of the data is relatively large, the weight vector w is *not sparse*, which means that very few weights w_i are close to zero.

ℓ^1-Norm and Lasso Regression

One of the weak points of ridge regression is that when the dimension n of the data is relatively large, the weight vector w is *not sparse*, which means that very few weights w_i are close to zero.

A remedy to this problem is to *penalize the ℓ^1-norm $\|w\|_1$ of w* instead of its ℓ^2-norm $\|w\|_2^2$.

ℓ^1-Norm and Lasso Regression

One of the weak points of ridge regression is that when the dimension n of the data is relatively large, the weight vector w is *not sparse*, which means that very few weights w_i are close to zero.

A remedy to this problem is to *penalize the ℓ^1-norm $\|w\|_1$ of w* instead of its ℓ^2-norm $\|w\|_2^2$.

The *ℓ^1-norm* of a vector $z = (z_1, \ldots, z_n) \in \mathbb{R}^n$ is defined as

$$\|z\|_1 = |z_1| + \cdots + |z_n|.$$

Lasso Regression

Lasso regression is the following minimization problem:

Lasso Regression

Lasso regression is the following minimization problem:

$$\begin{aligned} &\text{minimize} && \|\xi\|_2^2 + \tau \|w\|_1 \\ &\text{subject to} \\ & && y_i - x_i^\top w - b = \xi_i, \quad i = 1, \ldots, m \end{aligned}$$

where τ is positive constant.

Lasso Regression

Lasso regression is the following minimization problem:

$$\begin{array}{ll} \text{minimize} & \|\xi\|_2^2 + \tau \|w\|_1 \\ \text{subject to} & \\ & y_i - x_i^\top w - b = \xi_i, \quad i = 1, \ldots, m \end{array}$$

where τ is positive constant.

This time, there is no closed-form solution. However a solution can be computed using an iterative process (ADMM) which solves a sequence of linear systems involving SPD matrices.

Elastic Net Regression

There are still undesirable features of lasso, especially when the dimension n of the data is much larger than the number m of data.

Elastic Net Regression

There are still undesirable features of lasso, especially when the dimension n of the data is much larger than the number m of data.

A way to retain the best features of ridge regression and lasso is to *penalize both the ℓ^1-norm and the ℓ^2-norm of w*.

Elastic Net Regression

There are still undesirable features of lasso, especially when the dimension n of the data is much larger than the number m of data.

A way to retain the best features of ridge regression and lasso is to *penalize both the ℓ^1-norm and the ℓ^2-norm of w.*

Elastic net regression is the following minimization problem:

$$\begin{aligned} &\text{minimize} && \|\xi\|_2^2 + K\|w\|_2^2 + \tau\|w\|_1 \\ &\text{subject to} \\ & && y_i - x_i^\top w - b = \xi_i, \quad i = 1, \ldots, m \end{aligned}$$

where K and τ are positive constants.

Elastic Net Regression

Elastic net can also be solved using an iterative process (ADMM) which solves linear systems involving SPD matrices.

Elastic Net Regression

Elastic net can also be solved using an iterative process (ADMM) which solves linear systems involving SPD matrices.

When m is much larger than n, elastic net is much slower than lasso, especially for small K.

Elastic Net Regression

Elastic net can also be solved using an iterative process (ADMM) which solves linear systems involving SPD matrices.

When m is much larger than n, elastic net is much slower than lasso, especially for small K.

Remarkably, least squares, ridge regression, lasso, and elastic net, all rely on *solving linear systems involving SPD matrices*.

Elastic Net Regression

Elastic net can also be solved using an iterative process (ADMM) which solves linear systems involving SPD matrices.

When m is much larger than n, elastic net is much slower than lasso, especially for small K.

Remarkably, least squares, ridge regression, lasso, and elastic net, all rely on *solving linear systems involving SPD matrices*.

This is why most of this course will be devoted to these topics! The notion of *orthogonality* also play a crucial role.

2. Extrema of Real-Valued Functions

In most optimization problems we need to find necessary conditions for a function $J\colon \Omega \to \mathbb{R}$ to have a local extremum with respect to a subset U of Ω (where Ω is open). This can be done in two cases:

2. Extrema of Real-Valued Functions

In most optimization problems we need to find necessary conditions for a function $J\colon \Omega \to \mathbb{R}$ to have a local extremum with respect to a subset U of Ω (where Ω is open). This can be done in two cases:

(1) The set U is defined by a set of equations,

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous and differentiable.

2. Extrema of Real-Valued Functions

In most optimization problems we need to find necessary conditions for a function $J\colon \Omega \to \mathbb{R}$ to have a local extremum with respect to a subset U of Ω (where Ω is open). This can be done in two cases:

(1) The set U is defined by a set of equations,

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous and differentiable.

(2) The set U is defined by a set of inequalities,

$$U = \{x \in \Omega \mid \varphi_i(x) \leq 0,\ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous and differentiable.

Equality Constraints

In (1), the equations $\varphi_i(x) = 0$ are called *equality constraints*, and in (2), the inequalities $\varphi_i(x) \leq 0$ are called *inequality constraints*. The case of equality constraints is much easier to deal with and is treated next.

Equality Constraints

In (1), the equations $\varphi_i(x) = 0$ are called *equality constraints*, and in (2), the inequalities $\varphi_i(x) \leq 0$ are called *inequality constraints*. The case of equality constraints is much easier to deal with and is treated next.

In the case of equality constraints, a necessary condition for a local extremum with respect to U can be given in terms of *Lagrange multipliers*. In the case of inequality constraints, there is also a necessary condition for a local extremum with respect to U in terms of generalized Lagrange multipliers and the *Karush–Kuhn–Tucker* conditions.

Definition of a Local Minimum

Let J: $E \to \mathbb{R}$ be a real-valued function defined on a normed vector space E. Ideally we would like to find where the function J reaches a minimum or a maximum value, at least locally.

Definition of a Local Minimum

Let $J\colon E \to \mathbb{R}$ be a real-valued function defined on a normed vector space E. Ideally we would like to find where the function J reaches a minimum or a maximum value, at least locally.

Definition. If $J\colon E \to \mathbb{R}$ is a real-valued function defined on a normed vector space E, we say that J has a *local minimum* (or *relative minimum*) at the point $u \in E$ if there is some open subset $W \subseteq E$ containing u such that

$$J(u) \leq J(w) \quad \text{for all } w \in W.$$

Definition of a Local Maximum

Similarly, we say that J has a *local maximum* (or *relative maximum*) at the point $u \in E$ if there is some open subset $W \subseteq E$ containing u such that

$$J(u) \geq J(w) \quad \text{for all } w \in W.$$

Definition of a Local Maximum

Similarly, we say that J has a *local maximum* (or *relative maximum*) at the point $u \in E$ if there is some open subset $W \subseteq E$ containing u such that

$$J(u) \geq J(w) \quad \text{for all } w \in W.$$

In either case, we say that J has a *local extremum* (or *relative extremum*) at u.

Necessary Condition for Local Extrema

We begin with a *necessary condition* for a local extremum.

Necessary Condition for Local Extrema

We begin with a *necessary condition* for a local extremum.

Proposition. Let E be a normed vector space and let $J\colon \Omega \to \mathbb{R}$ be a function, with Ω some open subset of E. If the function J has a local extremum at some point $u \in \Omega$ and if J is differentiable at u, then

$$dJ_u = J'(u) = 0.$$

Critical Point

Definition. A point $u \in \Omega$ such that $J'(u) = 0$ is called a *critical point* of J.

Critical Point

Definition. A point $u \in \Omega$ such that $J'(u) = 0$ is called a *critical point* of J.

If $E = \mathbb{R}^n$, then the condition $dJ_u = 0$ is equivalent to the system

$$\begin{aligned} \frac{\partial J}{\partial x_1}(u_1, \ldots, u_n) &= 0 \\ &\vdots \\ \frac{\partial J}{\partial x_n}(u_1, \ldots, u_n) &= 0. \end{aligned}$$

Necessary Condition for Local Extrema

The condition of the preceding proposition is only a *necessary* condition for the existence of an extremum, but *not* a sufficient condition.

Necessary Condition for Local Extrema

The condition of the preceding proposition is only a *necessary* condition for the existence of an extremum, but *not* a sufficient condition.

Here are some counter-examples.

Necessary Condition for Local Extrema

The condition of the preceding proposition is only a *necessary* condition for the existence of an extremum, but *not* a sufficient condition.

Here are some counter-examples.
If f: $\mathbb{R} \to \mathbb{R}$ is the function given by $f(x) = x^3$, since $f'(x) = 3x^2$, we have $f'(0) = 0$, but 0 is neither a minimum nor a maximum of f as evidenced by the graph shown in Figure 3.

Illustration of a Cubic Curve

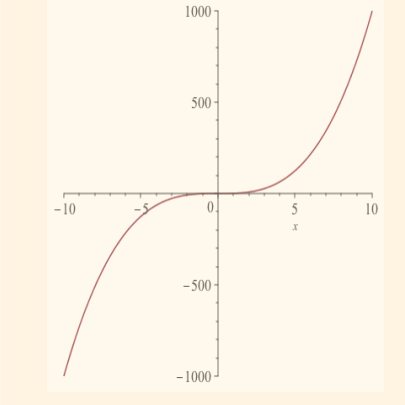

Figure 3: The graph of $f(x) = x^3$. Note that $x = 0$ is a saddle point and not a local extremum.

Necessary Condition for Local Extrema

If $g\colon \mathbb{R}^2 \to \mathbb{R}$ is the function given by $g(x, y) = x^2 - y^2$, then $g'_{(x,y)} = (2x \;\; -2y)$, so $g'_{(0,0)} = (0 \;\; 0)$, yet near $(0, 0)$ the function g takes negative and positive values. See Figure 4.

Illustration of a Hyperbolic Paraboloid

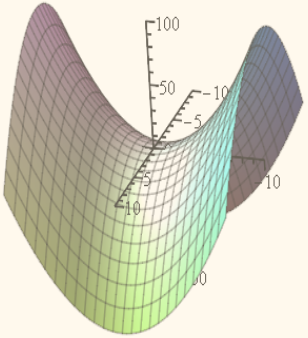

Figure 4: The graph of $g(x, y) = x^2 - y^2$. Note that $(0, 0)$ is a saddle point and not a local extremum.

Necessary Condition for Local Extrema

It is very important to note that the hypothesis that Ω *is open* is crucial for the validity of the preceding proposition.

Necessary Condition for Local Extrema

It is very important to note that the hypothesis that Ω *is open* is crucial for the validity of the preceding proposition.

For example, if J is the identity function on $\mathbb{R}$ and $U = [0, 1]$, a closed subset, then $J'(x) = 1$ for all $x \in [0, 1]$, even though J has a minimum at $x = 0$ and a maximum at $x = 1$.

3. Constrained Optimization

In many practical situations, we need to look for local extrema of a function J *under additional constraints*. This situation can be formalized conveniently as follows. We have a function $J\colon \Omega \to \mathbb{R}$ defined on some open subset Ω of a normed vector space, but we also have some subset U of Ω, and we are looking for the local extrema of J *with respect to the set* U.

3. Constrained Optimization

In many practical situations, we need to look for local extrema of a function J *under additional constraints*. This situation can be formalized conveniently as follows. We have a function $J\colon \Omega \to \mathbb{R}$ defined on some open subset Ω of a normed vector space, but we also have some subset U of Ω, and we are looking for the local extrema of J *with respect to the set* U.

The elements $u \in U$ are often called *feasible solutions* of the optimization problem consisting in finding the local extrema of some objective function J with respect to some subset U of Ω defined by a set of constraints. Note that in most cases, U is *not* open. In fact, U is usually closed.

Equality Constraints

In order to find necessary conditions for a function $J\colon \Omega \to \mathbb{R}$ to have a local extremum with respect to a subset U of Ω (where Ω is open), we need to *incorporate* the definition of U into these conditions. This can be done when the set U is defined by a set of equations,

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ \ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable).

Equality Constraints

In order to find necessary conditions for a function $J\colon \Omega \to \mathbb{R}$ to have a local extremum with respect to a subset U of Ω (where Ω is open), we need to *incorporate* the definition of U into these conditions. This can be done when the set U is defined by a set of equations,

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ \ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable).

The equations $\varphi_i(x) = 0$ are called *equality constraints*.

Necessary Condition for Constrained Extrema

In the case of equality constraints, a *necessary condition* for a local extremum with respect to U can be given in terms of *Lagrange multipliers*.

Necessary Condition for Constrained Extrema

Theorem (*Necessary condition for a constrained extremum in terms of Lagrange multipliers*). Let Ω be an open subset of $\mathbb{R}^n$, consider m C^1-functions $\varphi_i \colon \Omega \to \mathbb{R}$ (with $1 \leq m < n$), let

$$U = \{v \in \Omega \mid \varphi_i(v) = 0,\ 1 \leq i \leq m\},$$

and let $u \in U$ be a point such that the derivatives $d\varphi_i(u) \in \mathcal{L}(\mathbb{R}^n; \mathbb{R})$ are *linearly independent*; equivalently, assume that the $m \times n$ matrix $\big((\partial\varphi_i/\partial x_j)(u)\big)$ has rank m.

Necessary Condition for Constrained Extrema

If $J\colon \Omega \to \mathbb{R}$ is a function which is differentiable at $u \in U$ and if J has a local constrained extremum at u, then there exist m numbers $\lambda_i(u) \in \mathbb{R}$, uniquely defined, such that

$$dJ(u) + \lambda_1(u)d\varphi_1(u) + \cdots + \lambda_m(u)d\varphi_m(u) = 0;$$

Necessary Condition for Constrained Extrema

If $J\colon \Omega \to \mathbb{R}$ is a function which is differentiable at $u \in U$ and if J has a local constrained extremum at u, then there exist m numbers $\lambda_i(u) \in \mathbb{R}$, uniquely defined, such that

$$dJ(u) + \lambda_1(u)d\varphi_1(u) + \cdots + \lambda_m(u)d\varphi_m(u) = 0;$$

or equivalently,

$$\nabla J(u) + \lambda_1(u)\nabla\varphi_1(u) + \cdots + \lambda_m(u)\nabla\varphi_m(u) = 0.$$

Lagrange Multipliers

Definition. The numbers $\lambda_i(u)$ involved in the preceding theorem are called the *Lagrange multipliers* associated with the constrained extremum u.

Lagrange Multipliers

Definition. The numbers $\lambda_i(u)$ involved in the preceding theorem are called the *Lagrange multipliers* associated with the constrained extremum u.

The linear independence of the linear forms $d\varphi_i(u)$ is equivalent to the fact that the Jacobian matrix $\big((\partial\varphi_i/\partial x_j)(u)\big)$ of $\varphi = (\varphi_1, \ldots, \varphi_m)$ at u has rank m. If $m = 1$, the linear independence of the $d\varphi_i(u)$ reduces to the condition $\nabla\varphi_1(u) \neq 0$.

The Lagrangian

A fruitful way to reformulate the use of Lagrange multipliers is to introduce the notion of the Lagrangian associated with our constrained extremum problem.

The Lagrangian

A fruitful way to reformulate the use of Lagrange multipliers is to introduce the notion of the Lagrangian associated with our constrained extremum problem.

Definition. The *Lagrangian* associated with our constrained extremum problem is the function $L\colon \Omega \times \mathbb{R}^m \to \mathbb{R}$ given by

$$L(v, \lambda) = J(v) + \lambda_1 \varphi_1(v) + \cdots + \lambda_m \varphi_m(v),$$

with $\lambda = (\lambda_1, \ldots, \lambda_m)$.

Critical Point of the Lagrangian

Proposition. There exists some $\mu = (\mu_1, \ldots, \mu_m)$ and some $u \in U$ such that

$$dJ(u) + \mu_1 d\varphi_1(u) + \cdots + \mu_m d\varphi_m(u) = 0$$

if and only if

$$dL(u, \mu) = 0,$$

or equivalently

$$\nabla L(u, \mu) = 0;$$

that is, iff (u, μ) is a *critical point* of the Lagrangian L.

The Lagrangian Technique

The beauty of the Lagrangian is that the constraints $\{\varphi_i(v) = 0\}$ have been incorporated into the function $L(v, \lambda)$, and that the necessary condition for the existence of a constrained local extremum of J is reduced to the necessary condition for the existence of a local extremum of the *unconstrained* L.

Lagrangian Technique Example

Example. Let us apply the above method to the following example in which $E_1 = \mathbb{R}$, $E_2 = \mathbb{R}$, $\Omega = \mathbb{R}^2$, and

$$J(x_1, x_2) = -x_2$$
$$\varphi(x_1, x_2) = x_1^2 + x_2^2 - 1.$$

Lagrangian Techique Example

Observe that

$$U = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 = 1\}$$

is the unit circle, and since

$$\nabla\varphi(x_1, x_2) = \begin{pmatrix} 2x_1 \\ 2x_2 \end{pmatrix},$$

Lagrangian Techique Example

Observe that

$$U = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 = 1\}$$

is the unit circle, and since

$$\nabla\varphi(x_1, x_2) = \begin{pmatrix} 2x_1 \\ 2x_2 \end{pmatrix},$$

it is clear that $\nabla\varphi(x_1, x_2) \neq 0$ for every point $= (x_1, x_2)$ on the unit circle.

Lagrangian Technique Example

If we form the Lagrangian

$$L(x_1, x_2, \lambda) = -x_2 + \lambda(x_1^2 + x_2^2 - 1),$$

a necessary condition for J to have a constrained local extremum is that $\nabla L(x_1, x_2, \lambda) = 0$,

Lagrangian Technique Example

If we form the Lagrangian

$$L(x_1, x_2, \lambda) = -x_2 + \lambda(x_1^2 + x_2^2 - 1),$$

a necessary condition for J to have a constrained local extremum is that $\nabla L(x_1, x_2, \lambda) = 0$, so the following equations must hold:

$$\begin{aligned} 2\lambda x_1 &= 0 \\ -1 + 2\lambda x_2 &= 0 \\ x_1^2 + x_2^2 &= 1. \end{aligned}$$

Lagrangian Technique Example

The second equation implies that $\lambda \neq 0$, and then the first yields $x_1 = 0$, so the third yields $x_2 = \pm 1$, and we get two solutions:

$$\lambda = \frac{1}{2}, \qquad (x_1, x_2) = (0, 1)$$

$$\lambda = -\frac{1}{2}, \qquad (x_1', x_2') = (0, -1).$$

Lagrangian Technique Example

The second equation implies that $\lambda \neq 0$, and then the first yields $x_1 = 0$, so the third yields $x_2 = \pm 1$, and we get two solutions:

$$\lambda = \frac{1}{2}, \qquad (x_1, x_2) = (0, 1)$$

$$\lambda = -\frac{1}{2}, \qquad (x_1', x_2') = (0, -1).$$

We can check immediately that the first solution is a minimum and the second is a maximum.

4. The Objective of Optimization Theory

The main goal of *optimization theory* is to construct *algorithms* to find solutions (often approximate) of problems of the form

$$\begin{aligned} &\text{find} \quad u \\ &\text{such that} \quad u \in U \text{ and } J(u) = \inf_{v \in U} J(v), \end{aligned}$$

where U is a given subset of a (real) vector space V (possibly infinite dimensional) and $J\colon \Omega \to \mathbb{R}$ is a function defined on some open subset Ω of V such that $U \subseteq \Omega$.

Restated Objective of Optimization Theory

The minimization problem

$$\begin{aligned} &\text{find} \quad u \\ &\text{such that} \quad u \in U \text{ and } J(u) = \inf_{v \in U} J(v) \end{aligned}$$

is often presented in the following more informal way:

Restated Objective of Optimization Theory

The minimization problem

$$\begin{aligned} &\text{find} \quad u \\ &\text{such that} \quad u \in U \text{ and } J(u) = \inf_{v \in U} J(v) \end{aligned}$$

is often presented in the following more informal way:

$$\begin{aligned} &\text{minimize} \quad J(v) \\ &\text{subject to} \quad v \in U. \end{aligned}$$

(Problem M)

Minimizer of the Optimization Problem

A vector $u \in U$ such that $J(u) = \inf_{v \in U} J(v)$ is often called a *minimizer* of J over U.

Minimizer of the Optimization Problem

A vector $u \in U$ such that $J(u) = \inf_{v \in U} J(v)$ is often called a *minimizer* of J over U.

Some authors denote the set of minimizers of J over U by $\operatorname{argmin}_{v \in U} J(v)$ and write

$$u \in \operatorname{argmin}_{v \in U} J(v)$$

to express that u is such a minimizer.

Constraints and Functional of the Optimization Problem

In most cases, U is defined as the set of solutions of a finite sets of *constraints*, either equality constraints $\varphi_i(v) = 0$, or inequality constraints $\varphi_i(v) \leq 0$, where the $\varphi_i \colon \Omega \to \mathbb{R}$ are some given functions.

Constraints and Functional of the Optimization Problem

In most cases, U is defined as the set of solutions of a finite sets of *constraints*, either equality constraints $\varphi_i(v) = 0$, or inequality constraints $\varphi_i(v) \leq 0$, where the $\varphi_i \colon \Omega \to \mathbb{R}$ are some given functions.

The function J is often called the *functional* of the optimization problem.

Optimization Problems: Equality Constraints

Earlier we investigated the problem of determining when a function $J\colon \Omega \to \mathbb{R}$ defined on some open subset Ω of a normed vector space E has a local extremum in a subset U of Ω defined by equational constraints, namely

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ \ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable).

Optimization Problems: Equality Constraints

Earlier we investigated the problem of determining when a function $J\colon \Omega \to \mathbb{R}$ defined on some open subset Ω of a normed vector space E has a local extremum in a subset U of Ω defined by equational constraints, namely

$$U = \{x \in \Omega \mid \varphi_i(x) = 0,\ \ 1 \leq i \leq m\},$$

where the functions $\varphi_i\colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable).

We gave a *necessary condition* in terms of the *Lagrange multipliers*.

Optimization Problems: Inequality Constraints

Our goal is to find a necessary criterion for a function $J\colon \Omega \to \mathbb{R}$ to have a minimum on a subset U defined by *inequality* constraints $\varphi_i(x) \leq 0$, where the functions φ_i are convex.

Optimization Problems: Inequality Constraints

Our goal is to find a necessary criterion for a function $J\colon \Omega \to \mathbb{R}$ to have a minimum on a subset U defined by *inequality* constraints $\varphi_i(x) \leq 0$, where the functions φ_i are convex.

There is a *necessary condition* for a function J to have a minimum on a subset U defined by qualified inequality constraints in terms of the *Karush–Kuhn–Tucker conditions* (for short KKT conditions), which involve *nonnegative Lagrange multipliers*.

Optimization Problems: Inequality Constraints

In general, the KKT conditions are useless unless the constraints are convex.

Optimization Problems: Inequality Constraints

In general, the KKT conditions are useless unless the constraints are convex.

Furthermore, if J is also convex and if the KKT conditions hold, then J has a global minimum.

Equality Constraints as Inequalities

From now on we assume that U is defined by a set of inequalities, that is

$$U = \{x \in \Omega \mid \varphi_i(x) \leq 0,\ 1 \leq i \leq m\},$$

where the functions $\varphi_i \colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable).

Equality Constraints as Inequalities

From now on we assume that U is defined by a set of inequalities, that is

$$U = \{x \in \Omega \mid \varphi_i(x) \leq 0,\ 1 \leq i \leq m\},$$

where the functions $\varphi_i \colon \Omega \to \mathbb{R}$ are continuous (and usually differentiable). An equality constraint $\varphi_i(x) = 0$ is treated as the conjunction of the two inequalities $\varphi_i(x) \leq 0$ and $-\varphi_i(x) \leq 0$.

5. *Role of Convexity in Optimization*

Since the astute reader will notice the word *convex* has appeared numerous times we need to first define the notion of a convex function.

Definition of a Convex Set

Definition. Given any real vector space E, we say that a subset C of E is *convex* if either $C = \emptyset$ or if for every pair of points $u, v \in C$, the line segment connecting u and v is contained in C, i.e.,

$$(1 - \lambda)u + \lambda v \in C \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 \leq \lambda \leq 1.$$

Definition of a Convex Set

Definition. Given any real vector space E, we say that a subset C of E is *convex* if either $C = \emptyset$ or if for every pair of points $u, v \in C$, the line segment connecting u and v is contained in C, i.e.,

$$(1 - \lambda)u + \lambda v \in C \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 \leq \lambda \leq 1.$$

Given any two points $u, v \in E$, the *line segment* $[u, v]$ is the set

$$[u, v] = \{(1 - \lambda)u + \lambda v \in E \mid \lambda \in \mathbb{R},\ 0 \leq \lambda \leq 1\}.$$

Definition of a Convex Set

Definition. Given any real vector space E, we say that a subset C of E is *convex* if either $C = \emptyset$ or if for every pair of points $u, v \in C$, the line segment connecting u and v is contained in C, i.e.,

$$(1 - \lambda)u + \lambda v \in C \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 \leq \lambda \leq 1.$$

Given any two points $u, v \in E$, the *line segment* $[u, v]$ is the set

$$[u, v] = \{(1 - \lambda)u + \lambda v \in E \mid \lambda \in \mathbb{R},\ 0 \leq \lambda \leq 1\}.$$

Clearly, a nonempty set C is convex iff $[u, v] \subseteq C$ whenever $u, v \in C$.

Illustration of a Convex Set

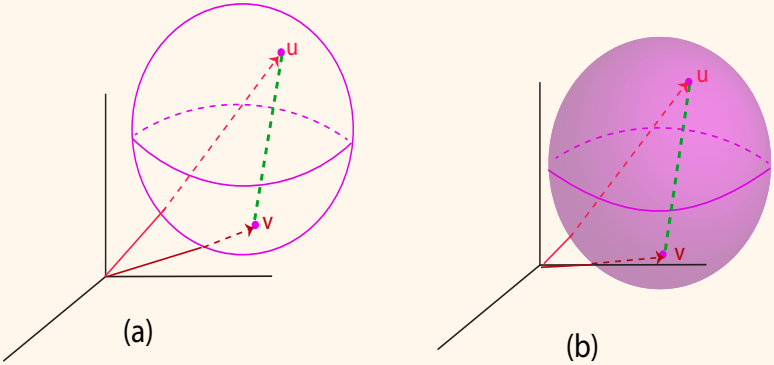

Figure 5: Figure (a) shows that a sphere is not convex in $\mathbb{R}^3$ since the dashed green line does not lie on its surface. Figure (b) shows that a solid ball is convex in $\mathbb{R}^3$.

Definition of a Convex Function

Definition. If C is a nonempty convex subset of E, a function $f\colon C \to \mathbb{R}$ is *convex* (on C) if for every pair of points $u, v \in C$,

$$f((1-\lambda)u + \lambda v) \leq (1-\lambda)f(u) + \lambda f(v) \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 \leq \lambda \leq 1;$$

Definition of a Convex Function

Definition. If C is a nonempty convex subset of E, a function $f\colon C \to \mathbb{R}$ is *convex* (on C) if for every pair of points $u, v \in C$,

$$f((1-\lambda)u + \lambda v) \leq (1-\lambda)f(u) + \lambda f(v) \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 \leq \lambda \leq 1;$$

the function f is *strictly convex* (on C) if for every pair of distinct points $u, v \in C$ $(u \neq v)$,

$$f((1-\lambda)u + \lambda v) < (1-\lambda)f(u) + \lambda f(v) \quad \text{for all } \lambda \in \mathbb{R} \text{ such that } 0 < \lambda < 1.$$

Illustration of a Convex Function

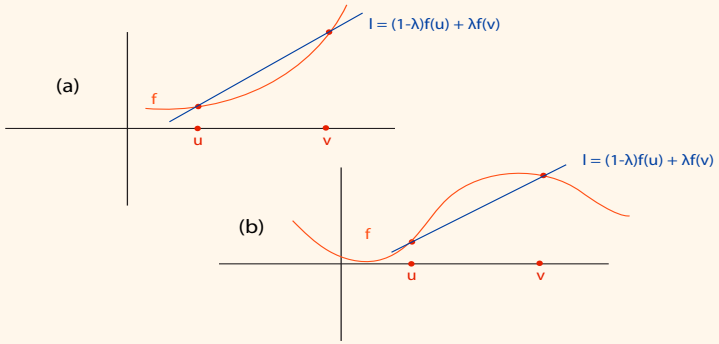

Figure 6: Figures (a) and (b) are the graphs of real valued functions. Figure (a) is the graph of convex function since the blue line lies above the graph of f. Figure (b) shows the graph of a function which is not convex.

Examples of Convex Sets

Example. Here are some common examples of convex sets.

Examples of Convex Sets

Example. Here are some common examples of convex sets.

- Subspaces $V \subseteq E$ of a vector space E are convex.
- *Affine subspaces*, that is, sets of the form $u + V$, where V is a subspace of E and $u \in E$, are convex.

Examples of Convex Sets

Example. Here are some common examples of convex sets.

- Subspaces $V \subseteq E$ of a vector space E are convex.
- *Affine subspaces*, that is, sets of the form $u + V$, where V is a subspace of E and $u \in E$, are convex.
- Balls (open or closed) are convex. Given any linear form $\varphi\colon E \to \mathbb{R}$, for any scalar $c \in \mathbb{R}$, the *closed half–spaces*

$$H^+_{\varphi,c} = \{u \in E \mid \varphi(u) \geq c\}, \qquad H^-_{\varphi,c} = \{u \in E \mid \varphi(u) \leq c\},$$

are convex.

Examples of Convex Sets

Example. Here are some common examples of convex sets.

- Subspaces $V \subseteq E$ of a vector space E are convex.
- *Affine subspaces*, that is, sets of the form $u + V$, where V is a subspace of E and $u \in E$, are convex.
- Balls (open or closed) are convex. Given any linear form $\varphi\colon E \to \mathbb{R}$, for any scalar $c \in \mathbb{R}$, the *closed half–spaces*

$$H^+_{\varphi,c} = \{u \in E \mid \varphi(u) \geq c\}, \qquad H^-_{\varphi,c} = \{u \in E \mid \varphi(u) \leq c\},$$

  are convex.
- Any intersection of half–spaces is convex.
- More generally, any intersection of convex sets is convex.

Examples of Convex Functions

Example. Here are some common examples of convex and concave functions.

Examples of Convex Functions

Example. Here are some common examples of convex and concave functions.

- Linear forms are convex functions (but not strictly convex).
- Any norm $\| \| : E \to \mathbb{R}_+$ is a convex function.

Examples of Convex Functions

Example. Here are some common examples of convex and concave functions.

- Linear forms are convex functions (but not strictly convex).
- Any norm $\| \| : E \to \mathbb{R}_+$ is a convex function.
- The max function,

$$\max(x_1, \ldots, x_n) = \max\{x_1, \ldots, x_n\}$$

  is convex on $\mathbb{R}^n$.
- The exponential $x \mapsto e^{cx}$ is strictly convex for any $c \neq 0$ ($c \in \mathbb{R}$).

Examples of Convex Functions

Example. Here are some common examples of convex and concave functions.

- Linear forms are convex functions (but not strictly convex).
- Any norm $\| \| : E \to \mathbb{R}_+$ is a convex function.
- The max function,

$$\max(x_1, \ldots, x_n) = \max\{x_1, \ldots, x_n\}$$

  is convex on $\mathbb{R}^n$.
- The exponential $x \mapsto e^{cx}$ is strictly convex for any $c \neq 0$ ($c \in \mathbb{R}$).
- The logarithm function is concave on $\mathbb{R}_+ - \{0\}$.
- The *log-determinant function* $\log \det$ is concave on the set of symmetric positive definite matrices. This function plays an important role in convex optimization.

6. Optimization with Convex Constraints

If the domain U is defined by *convex* inequality constraints satisfying mild differentiability conditions and if the constraints at u are qualified, then there is a *necessary* condition for the function J to have a local minimum at $u \in U$ involving *generalized Lagrange multipliers*.

Optimization with Convex Constraints

We have the following theorem.

Optimization with Convex Constraints

We have the following theorem.

Theorem. Let $\varphi_i \colon \Omega \to \mathbb{R}$ be m *convex constraints* defined on some *open convex* subset Ω of a finite-dimensional Euclidean vector space V (more generally, a real Hilbert space V), let $J \colon \Omega \to \mathbb{R}$ be some function, let U be given by

$$U = \{x \in \Omega \mid \varphi_i(x) \leq 0,\ 1 \leq i \leq m\},$$

and let $u \in U$ be any point such that the functions φ_i and J are differentiable at u.

Necessary Condition for Minimization with Convex Constraints

(1) If J has a local minimum at u with respect to U, and if the constraints are *qualified*, then there exist some scalars $\lambda_i(u) \in \mathbb{R}$, such that the *KKT condition* hold:

$$J_u{}' + \sum_{i=1}^{m} \lambda_i(u)(\varphi_i')_u = 0$$

and

$$\sum_{i=1}^{m} \lambda_i(u)\varphi_i(u) = 0, \quad \lambda_i(u) \geq 0, \quad i = 1, \ldots, m.$$

Necessary Condition for Minimization with Convex Constraints

Equivalently, in terms of gradients, the above conditions are expressed as

$$\nabla J_u + \sum_{i=1}^{m} \lambda_i(u) \nabla(\varphi_i)_u = 0,$$

and

$$\sum_{i=1}^{m} \lambda_i(u)\varphi_i(u) = 0, \quad \lambda_i(u) \geq 0, \quad i = 1, \ldots, m.$$

Sufficient Condition for Minimization with Convex Constraints

(2) Conversely, if the restriction of J to U is *convex* and if there exist scalars $(\lambda_1, \ldots, \lambda_m) \in \mathbb{R}^m_+$ such that the KKT conditions hold, then the function J has a (global) minimum at u with respect to U.

Sufficient Condition for Minimization with Convex Constraints

(2) Conversely, if the restriction of J to U is *convex* and if there exist scalars $(\lambda_1, \ldots, \lambda_m) \in \mathbb{R}^m_+$ such that the KKT conditions hold, then the function J has a (global) minimum at u with respect to U.

The scalars $\lambda_i(u)$ are often called *generalized Lagrange multipliers*.

The Lagrangian

The above theorem suggests introducing the function $L\colon \Omega \times \mathbb{R}^m_+ \to \mathbb{R}$ given by

$$L(v, \lambda) = J(v) + \sum_{i=1}^{m} \lambda_i \varphi_i(v),$$

with $\lambda = (\lambda_1, \ldots, \lambda_m)$.

The Lagrangian

The above theorem suggests introducing the function $L\colon \Omega \times \mathbb{R}^m_+ \to \mathbb{R}$ given by

$$L(v, \lambda) = J(v) + \sum_{i=1}^{m} \lambda_i \varphi_i(v),$$

with $\lambda = (\lambda_1, \ldots, \lambda_m)$.

The function L is called the *Lagrangian* of the *Minimization Problem* (P):

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m. \end{aligned}$$

The Lagrangian and the KKT Conditions

The KKT conditions of the preceding theorem imply that for any $u \in U$, if the vector $\lambda = (\lambda_1, \ldots, \lambda_m)$ is known and if u is a minimum of J on U, then

$$\frac{\partial L}{\partial u}(u) = 0$$
$$J(u) = L(u, \lambda).$$

The Lagrangian and the KKT Conditions

The KKT conditions of the preceding theorem imply that for any $u \in U$, if the vector $\lambda = (\lambda_1, \ldots, \lambda_m)$ is known and if u is a minimum of J on U, then

$$\frac{\partial L}{\partial u}(u) = 0$$
$$J(u) = L(u, \lambda).$$

The Lagrangian technique "absorbs" the constraints into the new objective function L and reduces the problem of finding a constrained minimum of the function J, to the problem of finding an *unconstrained* minimum of the function $L(v, \lambda)$.

The Lagrangian and the KKT Conditions

The KKT conditions of the preceding theorem imply that for any $u \in U$, if the vector $\lambda = (\lambda_1, \ldots, \lambda_m)$ is known and if u is a minimum of J on U, then

$$\frac{\partial L}{\partial u}(u) = 0$$
$$J(u) = L(u, \lambda).$$

The Lagrangian technique "absorbs" the constraints into the new objective function L and reduces the problem of finding a constrained minimum of the function J, to the problem of finding an *unconstrained* minimum of the function $L(v, \lambda)$.

This is the *main point* of Lagrangian duality which will be treated next.

7. Primal and Dual Minimization Problems

In this section we investigate methods to solve the *Primal Minimization Problem* (P):

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m. \end{aligned}$$

7. Primal and Dual Minimization Problems

In this section we investigate methods to solve the *Primal Minimization Problem* (P)*:*

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m. \end{aligned}$$

It turns out that under certain conditions the original Problem (P), called *primal problem*, can be solved in two stages with the help another Problem (D), called the *dual problem*.

Dual Problem

The Dual Problem (D) is a *maximization problem* involving a function G, called the *Lagrangian dual*, and it is obtained by *minimizing* the *Lagrangian* $L(v, \mu)$ of Problem (P) over the variable $v \in \mathbb{R}^n$, holding μ fixed, where $L \colon \Omega \times \mathbb{R}^m_+ \to \mathbb{R}$ is given by

$$L(v, \mu) = J(v) + \sum_{i=1}^{m} \mu_i \varphi_i(v),$$

with $\mu \in \mathbb{R}^m_+$.

Duality Method for Solving Problem (P)

The two steps of the method are:

Duality Method for Solving Problem (P)

The two steps of the method are:

(1) Find the dual function $\mu \mapsto G(\mu)$ explicitly by solving the *minimization problem* of finding the minimum of $L(v, \mu)$ with respect to $v \in \Omega$, holding μ fixed.

Duality Method for Solving Problem (P)

The two steps of the method are:

(1) Find the dual function $\mu \mapsto G(\mu)$ explicitly by solving the *minimization problem* of finding the minimum of $L(v, \mu)$ with respect to $v \in \Omega$, holding μ fixed. This is an *unconstrained* minimization problem (with $v \in \Omega$). If we are lucky, a unique minimizer u_μ such that $G(\mu) = L(u_\mu, \mu)$ can be found.

Duality Method for Solving Problem (P)

The two steps of the method are:

(1) Find the dual function $\mu \mapsto G(\mu)$ explicitly by solving the *minimization problem* of finding the minimum of $L(v, \mu)$ with respect to $v \in \Omega$, holding μ fixed. This is an *unconstrained* minimization problem (with $v \in \Omega$). If we are lucky, a unique minimizer u_μ such that $G(\mu) = L(u_\mu, \mu)$ can be found.

(2) Solve the *maximization problem* of finding the maximum of the function $\mu \mapsto G(\mu)$ over all $\mu \in \mathbb{R}^m_+$.

Duality Method for Solving Problem (P)

The two steps of the method are:

(1) Find the dual function $\mu \mapsto G(\mu)$ explicitly by solving the *minimization problem* of finding the minimum of $L(v, \mu)$ with respect to $v \in \Omega$, holding μ fixed. This is an *unconstrained* minimization problem (with $v \in \Omega$). If we are lucky, a unique minimizer u_μ such that $G(\mu) = L(u_\mu, \mu)$ can be found.

(2) Solve the *maximization problem* of finding the maximum of the function $\mu \mapsto G(\mu)$ over all $\mu \in \mathbb{R}^m_+$. This is basically an unconstrained problem, except for the fact that $\mu \in \mathbb{R}^m_+$.

Duality Method for Solving Problem (P)

If Steps (1) and (2) are successful, under some suitable conditions on the function J and the constraints φ_i (for example, if they are convex), for any solution $\lambda \in \mathbb{R}_+^m$ obtained in Step (2), the vector u_λ obtained in Step (1) is an optimal solution of Problem (P).

Duality Method for Solving Problem (P)

If Steps (1) and (2) are successful, under some suitable conditions on the function J and the constraints φ_i (for example, if they are convex), for any solution $\lambda \in \mathbb{R}_+^m$ obtained in Step (2), the vector u_λ obtained in Step (1) is an optimal solution of Problem (P).

The local minima of a function $J\colon \Omega \to \mathbb{R}$ over a domain U defined by inequality constraints are *saddle points* of the Lagrangian $L(v, \mu)$ associated with J and the constraints φ_i.

Duality Method for Solving Problem (P)

If Steps (1) and (2) are successful, under some suitable conditions on the function J and the constraints φ_i (for example, if they are convex), for any solution $\lambda \in \mathbb{R}^m_+$ obtained in Step (2), the vector u_λ obtained in Step (1) is an optimal solution of Problem (P).

The local minima of a function $J\colon \Omega \to \mathbb{R}$ over a domain U defined by inequality constraints are *saddle points* of the Lagrangian $L(v, \mu)$ associated with J and the constraints φ_i.

In this presentation we do not discuss saddle points since this would take too much time.

Primal Minimization Problem

We now return to our main Minimization Problem (P):

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m, \end{aligned}$$

where $J\colon \Omega \to \mathbb{R}$ and the constraints $\varphi_i\colon \Omega \to \mathbb{R}$ are some functions defined on some open subset Ω of some finite-dimensional Euclidean vector space V (more generally, a real Hilbert space V).

Lagrangian of the Minimization Problem

Definition. The *Lagrangian* of the Minimization Problem (P) defined above is the function $L\colon \Omega \times \mathbb{R}_+^m \to \mathbb{R}$ given by

$$L(v, \mu) = J(v) + \sum_{i=1}^{m} \mu_i \varphi_i(v),$$

with $\mu = (\mu_1, \ldots, \mu_m)$. The numbers μ_i are called *generalized Lagrange multipliers*.

Lagrangian Duality

Definition. Given the Minimization Problem (P)

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m, \end{aligned}$$

where $J\colon \Omega \to \mathbb{R}$ and the constraints $\varphi_i\colon \Omega \to \mathbb{R}$ are some functions defined on some open subset Ω of some finite-dimensional Euclidean vector space V (more generally, a real Hilbert space V), the function $G\colon \mathbb{R}^m_+ \to \mathbb{R}$ given by

$$G(\mu) = \inf_{v \in \Omega} L(v, \mu) \quad \mu \in \mathbb{R}^m_+,$$

is called the *Lagrange dual function* (or simply *dual function*).

Lagrange Dual Problem

Problem (D)

$$\begin{array}{ll} \text{maximize} & G(\mu) \\ \text{subject to} & \mu \in \mathbb{R}^m_+ \end{array}$$

is called the *Lagrange dual problem*.

Lagrange Dual Problem

Problem (D)

$$\begin{array}{ll} \text{maximize} & G(\mu) \\ \text{subject to} & \mu \in \mathbb{R}^m_+ \end{array}$$

is called the *Lagrange dual problem*.

Problem (P) is often called the *primal problem*, and (D) is the *dual problem*. The variable μ is called the *dual variable*.

Dual as a Convex Optimization Problem

Since

$$L(v, \mu) = J(v) + \sum_{i=1}^{m} \mu_i \varphi_i(v),$$

the function $G(\mu) = \inf_{v \in \Omega} L(v, \mu)$ is the pointwise infimum of some affine functions of μ, so it is *concave*, even if the φ_i are not convex.

Dual as a Convex Optimization Problem

Since

$$L(v, \mu) = J(v) + \sum_{i=1}^{m} \mu_i \varphi_i(v),$$

the function $G(\mu) = \inf_{v \in \Omega} L(v, \mu)$ is the pointwise infimum of some affine functions of μ, so it is *concave*, even if the φ_i are not convex.

One of the main advantages of the dual problem over the primal problem is that it is a *convex optimization problem*, since we wish to maximize a concave objective function G (thus minimize $-G$, a convex function), and the constraints $\mu \geq 0$ are convex. In a number of practical situations, the dual function G can indeed be computed.

Dual as a Partial Function

To be perfectly rigorous, we should mention that the dual function G is actually a *partial function*, because it takes the value $-\infty$ when the map $v \mapsto L(v, \mu)$ is unbounded below.

Dual Bounds Primal Problem (P)

Another important property of the dual function G is that it provides a *lower bound* on the value of the objective function J.

Dual Bounds Primal Problem (P)

Another important property of the dual function G is that it provides a *lower bound* on the value of the objective function J.

If the Primal Problem (P) has a minimum denoted p^* and the Dual Problem (D) has a maximum denoted d^*, then

$$d^* \leq p^* \qquad (\dagger_w)$$

known as *weak duality*.

Strong Duality

Definition. The difference $p^* - d^* \geq 0$ is called the *optimal duality gap*. If the duality gap is zero, that is, $p^* = d^*$, then we say that *strong duality* holds.

Strong Duality

Definition. The difference $p^* - d^* \geq 0$ is called the *optimal duality gap*. If the duality gap is zero, that is, $p^* = d^*$, then we say that *strong duality* holds.

Even when the duality gap is strictly positive, the inequality $(\dagger_w)$ can be helpful to find a lower bound on the optimal value of a primal problem that is difficult to solve, since the dual problem is *always* convex.

Strong Duality

Definition. The difference $p^* - d^* \geq 0$ is called the *optimal duality gap*. If the duality gap is zero, that is, $p^* = d^*$, then we say that *strong duality* holds.

Even when the duality gap is strictly positive, the inequality $(\dagger_w)$ can be helpful to find a lower bound on the optimal value of a primal problem that is difficult to solve, since the dual problem is *always* convex.

If the primal problem and the dual problem are feasible and if the optimal values p^* and d^* are finite and $p^* = d^*$ (no duality gap), then the complementary slackness conditions hold for the inequality constraints.

Complementary Slackness Conditions

Proposition (*complementary slackness*). Given the Minimization Problem (P)

$$\begin{aligned} &\text{minimize} && J(v) \\ &\text{subject to} && \varphi_i(v) \leq 0, \quad i = 1, \ldots, m, \end{aligned}$$

and its Dual Problem (D)

$$\begin{aligned} &\text{maximize} && G(\mu) \\ &\text{subject to} && \mu \in \mathbb{R}^m_+, \end{aligned}$$

Complementary Slackness Conditions

if both (P) and (D) are feasible, $u \in U$ is an optimal solution of (P), $\lambda \in \mathbb{R}^m_+$ is an optimal solution of (D), and $J(u) = G(\lambda)$, then

$$\sum_{i=1}^{m} \lambda_i \varphi_i(u) = 0.$$

In other words, *if the constraint φ_i is inactive at u, then $\lambda_i = 0$.*

8. Ridge Regression

Given a set of training data $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, with $x_i \in \mathbb{R}^n$ and $y_i \in \mathbb{R}$, if X is the $m \times n$ matrix

$$X = \begin{pmatrix} x_1^\top \\ \vdots \\ x_m^\top \end{pmatrix},$$

in which the row vectors $x_i^\top$ are the rows of X, then *ridge regression* is the following optimization problem.

Ridge Regression: Program (RR3)

Program $(\mathbf{RR3})$:

$$\begin{aligned} \text{minimize} \quad & \xi^\top \xi + K w^\top w \\ \text{subject to} \quad & \\ & y - Xw - b\mathbf{1} = \xi, \end{aligned}$$

with $y, \xi, \mathbf{1} \in \mathbb{R}^m$ and $w \in \mathbb{R}^n$. Note that in Program $(\mathbf{RR3})$ minimization is performed over ξ, w and b, but b is *not* penalized in the objective function.

Ridge Regression: Program (RR3) Solution

The objective function is *strictly convex*.

Ridge Regression: Program (RR3) Solution

The objective function is *strictly convex*.

The Lagrangian associated with this program is

$$L(\xi, w, b, \lambda) = \xi^\top \xi + Kw^\top w - w^\top X^\top \lambda - \xi^\top \lambda - b\mathbf{1}^\top \lambda + \lambda^\top y.$$

Ridge Regression: Program (RR3) Solution

The objective function is *strictly convex*.

The Lagrangian associated with this program is

$$L(\xi, w, b, \lambda) = \xi^\top \xi + K w^\top w - w^\top X^\top \lambda - \xi^\top \lambda - b\mathbf{1}^\top \lambda + \lambda^\top y.$$

Since L is *(strictly) convex as a function of* ξ, b, w, it has a minimum iff $\nabla L_{\xi,b,w} = 0$.

Ridge Regression: Dual Function of (RR3)

We get

$$\begin{aligned} \lambda &= 2\xi \\ \mathbf{1}^\top \lambda &= 0 \\ w &= \frac{1}{2K} X^\top \lambda = X^\top \frac{\xi}{K}. \end{aligned}$$

If we set $\xi = K\alpha$, we obtain $\lambda = 2K\alpha$, $w = X^\top \alpha$, and

$$G(\alpha) = -K\alpha^\top (XX^\top + KI_m)\alpha + 2K\alpha^\top y.$$

Ridge Regression: Dual Program of (RR3)

Since $K > 0$ and $\lambda = 2K\alpha$, the dual to ridge regression is the following program

Ridge Regression: Dual Program of (RR3)

Since $K > 0$ and $\lambda = 2K\alpha$, the dual to ridge regression is the following program

Program (DRR3):

$$\begin{aligned} &\text{minimize} && \alpha^\top (XX^\top + KI_m)\alpha - 2\alpha^\top y \\ &\text{subject to} && \\ & && \mathbf{1}^\top \alpha = 0, \end{aligned}$$

where the minimization is over α.

Ridge Regression: Solution to (DRR3)

Since $K > 0$ the matrix $XX^\top + KI_m$ is SPD.

Ridge Regression: Solution to (DRR3)

Since $K > 0$ the matrix $XX^\top + KI_m$ is SPD.

This implies that the minimization problem has a unique solution obtained by solving the KKT-equations

$$\begin{pmatrix} XX^\top + KI_m & \mathbf{1}_m \\ \mathbf{1}_m^\top & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \mu \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix}.$$

Ridge Regression: Solution to (DRR3)

We get

$$\mu = (\mathbf{1}^\top (XX^\top + KI_m)^{-1}\mathbf{1})^{-1}\mathbf{1}^\top (XX^\top + KI_m)^{-1}y$$
$$\alpha = (XX^\top + KI_m)^{-1}(y - \mu\mathbf{1}).$$

Ridge Regression: Solution to (DRR3)

We get

$$\mu = (\mathbf{1}^\top (XX^\top + KI_m)^{-1}\mathbf{1})^{-1}\mathbf{1}^\top (XX^\top + KI_m)^{-1}y$$
$$\alpha = (XX^\top + KI_m)^{-1}(y - \mu\mathbf{1}).$$

Interestingly $b = \mu$, which is not obvious a priori.

Proposition. We have $b = \mu$.

Ridge Regression: Program (RR3) Solution

In summary the KKT-equations determine both α and μ, and so $w = X^\top \alpha$ and b as well.

Ridge Regression: Averaging Formula for b

There is also a useful expression of b as an average. We have

$$b = \overline{y} - \sum_{j=1}^{n} \overline{X^j} w_j = \overline{y} - (\overline{X^1} \ \cdots \ \overline{X^n}) w,$$

where $\overline{y}$ is the mean of y and $\overline{X^j}$ is the mean of the jth column of X.

Ridge Regression: Illustrated Example

Example. Consider the data set (X, y_1) with

$$X = \begin{pmatrix} -10 & 11 \\ -6 & 5 \\ -2 & 4 \\ 0 & 0 \\ 1 & 2 \\ 2 & -5 \\ 6 & -4 \\ 10 & -6 \end{pmatrix}, \quad y_1 = \begin{pmatrix} 0 \\ -2.5 \\ 0.5 \\ -2 \\ 2.5 \\ -4.2 \\ 1 \\ 4 \end{pmatrix}$$

as illustrated in Figure 7.

Ridge Regression: Illustrated Example

We find that $\bar{y} = -0.0875$ and $(\overline{X^1}, \overline{X^2}) = (0.125, 0.875)$. For the value $K = 5$, we obtain

$$w = \begin{pmatrix} 0.9207 \\ 0.8677 \end{pmatrix}, \quad b = -0.9618,$$

for $K = 0.1$, we obtain

$$w = \begin{pmatrix} 1.1651 \\ 1.1341 \end{pmatrix}, \quad b = -1.2255,$$

and for $K = 0.01$,

$$w = \begin{pmatrix} 1.1709 \\ 1.1405 \end{pmatrix}, \quad b = -1.2318.$$

See Figure 8.

Ridge Regression: Illustrated Example

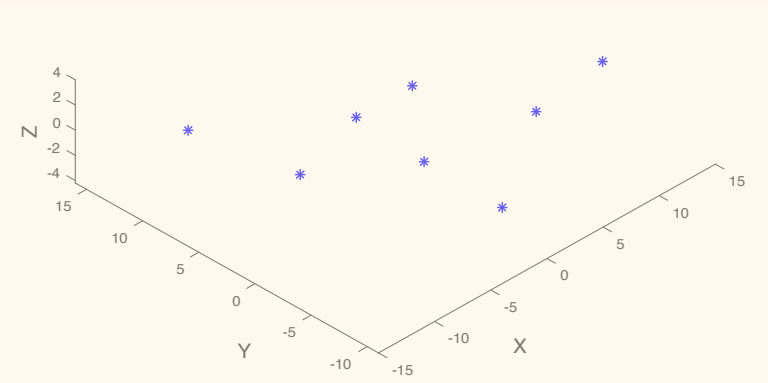

Figure 7: The data set (X, y_1).

Ridge Regression: Illustrated Example

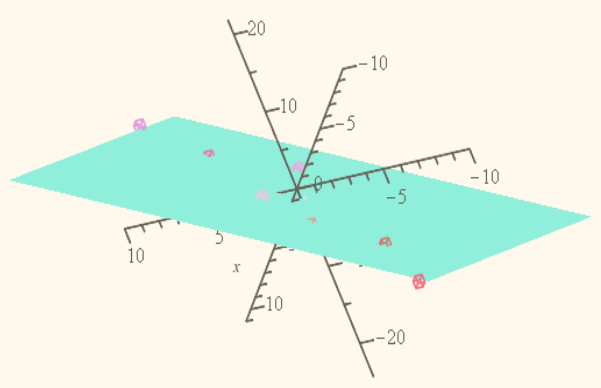

Figure 8: The graph of the plane $f(x, y) = 1.1709x + 1.1405y - 1.2318$ as an approximate fit to the data (X, y_1).

Ridge Regression: Illustrated Example

We conclude that the points (X_i, y_i) (where X_i is the ith row of X) almost lie on the plane of equation

$$x + y - z - 1 = 0,$$

and that f is almost the function given by $f(x, y) = 1.1x + 1.1y - 1.2$. See Figures 9 and 10.

Ridge Regression: Illustrated Example

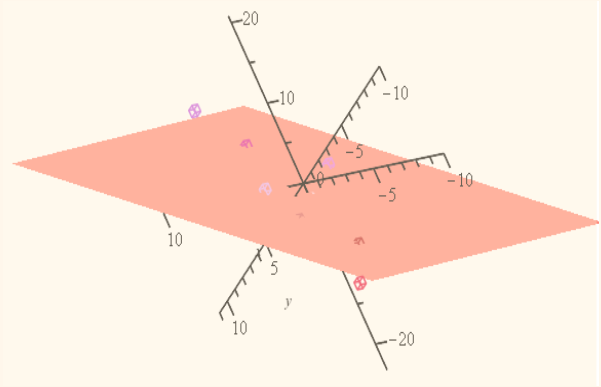

Figure 9: The graph of the plane $f(x, y) = 1.1x + 1.1y - 1.2$ as an approximate fit to the data (X, y_1).

Ridge Regression: Illustrated Example

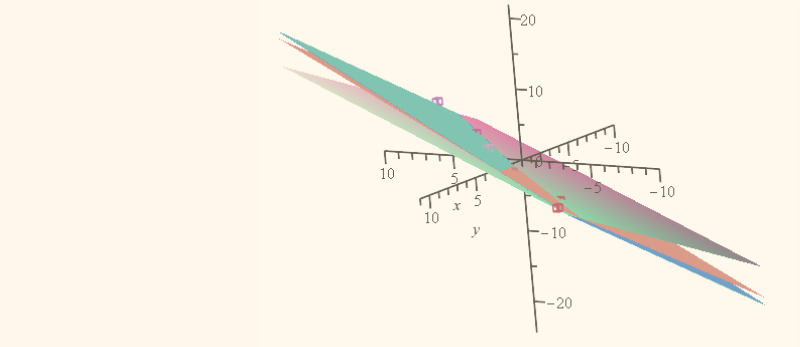

Figure 10: A comparison of how the graphs of the planes corresponding to $K = 1, 0.1, 0.01$ and the salmon plane of equation $f(x, y) = 1.1x + 1.1y - 1.2$ approximate the data (X, y_1).

Ridge Regression: Illustrated Example

If we change y_1 to

$$y_2 = \begin{pmatrix} 0 & -2 & 1 & -1 & 2 & -4 & 1 & 3 \end{pmatrix}^\top,$$

as evidenced by Figure 11, the exact solution is

$$w = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad b = -1,$$

and for $K = 0.01$, we find that

$$w = \begin{pmatrix} 0.9999 \\ 0.9999 \end{pmatrix}, \quad b = -0.9999.$$

Ridge Regression: Illustrated Example

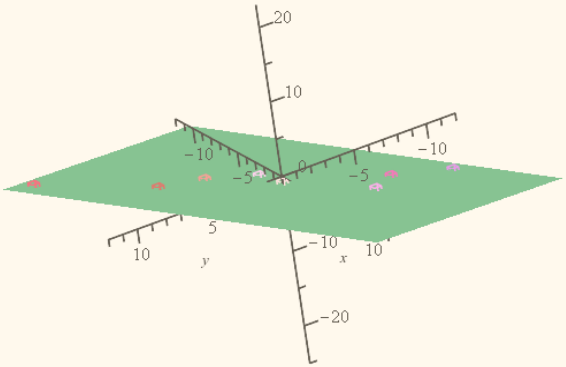

Figure 11: The data (X, y_2) is contained within the graph of the plane $f(x, y) = x + y - 1$.

Ridge Regression: Learning an Affine Function

We can see how the choice of K affects the quality of the solution (w, b) by computing the norm $\|\xi\|_2$ of the error vector. We notice that the smaller K is, the smaller is this norm.

Ridge Regression: Learning an Affine Function

We can see how the choice of K affects the quality of the solution (w, b) by computing the norm $\|\xi\|_2$ of the error vector. We notice that the smaller K is, the smaller is this norm.

As a least squares problem, the solution is given in terms of the pseudo-inverse $[X\,\mathbf{1}]^+$ of $[X\,\mathbf{1}]$ by

$$\begin{pmatrix} w \\ b \end{pmatrix} = [X\,\mathbf{1}]^+ y.$$

9. Lasso Regression

The main weakness of ridge regression is that the estimated weight vector w usually has many nonzero coefficients.

9. Lasso Regression

The main weakness of ridge regression is that the estimated weight vector w usually has many nonzero coefficients.

As a consequence, ridge regression does not scale up well.

9. Lasso Regression

The main weakness of ridge regression is that the estimated weight vector w usually has many nonzero coefficients.

As a consequence, ridge regression does not scale up well.

In practice we need methods capable of handling millions of parameters, or more.

Lasso Regression

A way to encourage sparsity of the vector w, which means that many coordinates of w are zero, is to replace the quadratic penalty function $\tau w^\top w = \tau \|w\|_2^2$ by the penalty function $\tau \|w\|_1$, with the ℓ^2-norm replaced by the ℓ^1-norm.

Lasso Regression

A way to encourage sparsity of the vector w, which means that many coordinates of w are zero, is to replace the quadratic penalty function $\tau w^\top w = \tau \|w\|_2^2$ by the penalty function $\tau \|w\|_1$, with the ℓ^2-norm replaced by the ℓ^1-norm.

This method was first proposed by Tibshirani around 1996, under the name *lasso*, which stands for "least absolute selection and shrinkage operator."

Lasso Regression

A way to encourage sparsity of the vector w, which means that many coordinates of w are zero, is to replace the quadratic penalty function $\tau w^\top w = \tau \|w\|_2^2$ by the penalty function $\tau \|w\|_1$, with the ℓ^2-norm replaced by the ℓ^1-norm.

This method was first proposed by Tibshirani around 1996, under the name *lasso*, which stands for "least absolute selection and shrinkage operator."

This method is also known as *ℓ^1-regularized regression*, but this is not as cute as "lasso," which is used predominantly.

Lasso Regression: Notational Convention

Given a set of training data $\{(x_1, y_1), \ldots, (x_m, y_m)\}$, with $x_i \in \mathbb{R}^n$ and $y_i \in \mathbb{R}$, if X is the $m \times n$ matrix

$$X = \begin{pmatrix} x_1^\top \\ \vdots \\ x_m^\top \end{pmatrix},$$

in which the row vectors $x_i^\top$ are the rows of X, then *lasso regression* is the following optimization problem.

Lasso Regression: Problem (lasso1)

Program (**lasso1**):

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\xi^\top \xi + \tau \|w\|_1 \\ \text{subject to} & \\ & y - Xw = \xi, \end{array}$$

minimizing over ξ and w, where $\tau > 0$ is some constant determining the influence of the regularizing term $\|w\|_1$.

Lasso Regression: (lasso1) Reduction

The difficulty with the regularizing term $\|w\|_1 = |w_1| + \cdots + |w_n|$ is that the map $w \mapsto \|w\|_1$ is **not** differentiable for all w.

Lasso Regression: (lasso1) Reduction

The difficulty with the regularizing term $\|w\|_1 = |w_1| + \cdots + |w_n|$ is that the map $w \mapsto \|w\|_1$ is **not** differentiable for all w.

This difficulty can be overcome by using subgradients, but the dual of the above program can also be obtained in an elementary fashion by using a trick, which is that if $x \in \mathbb{R}$, then

$$|x| = \max\{x, -x\}.$$

Lasso Regression: (lasso1) Reduction

The difficulty with the regularizing term $\|w\|_1 = |w_1| + \cdots + |w_n|$ is that the map $w \mapsto \|w\|_1$ is **not** differentiable for all w.

This difficulty can be overcome by using subgradients, but the dual of the above program can also be obtained in an elementary fashion by using a trick, which is that if $x \in \mathbb{R}$, then

$$|x| = \max\{x, -x\}.$$

Using this trick, by introducing a vector $\epsilon \in \mathbb{R}^n$ of *nonnegative* variables, we can rewrite lasso minimization as follows:

Lasso Regression: Program (lasso2)

Program lasso regularization (**lasso2**):

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\xi^\top \xi + \tau \mathbf{1}_n^\top \epsilon \\ \text{subject to} \quad & \\ & y - Xw = \xi \\ & w \leq \epsilon \\ & -w \leq \epsilon. \end{aligned}$$

minimizing over ξ, w and ϵ, with $y, \xi \in \mathbb{R}^m$, and $w, \epsilon, \mathbf{1}_n \in \mathbb{R}^n$.

Lasso Regression: Program (lasso2)

Program lasso regularization (**lasso2**):

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\xi^\top \xi + \tau \mathbf{1}_n^\top \epsilon \\ \text{subject to} \quad & \\ & y - Xw = \xi \\ & w \leq \epsilon \\ & -w \leq \epsilon. \end{aligned}$$

minimizing over ξ, w and ϵ, with $y, \xi \in \mathbb{R}^m$, and $w, \epsilon, \mathbf{1}_n \in \mathbb{R}^n$.
The constraints $w \leq \epsilon$ and $-w \leq \epsilon$ are equivalent to $|w_i| \leq \epsilon_i$ for $i = 1, \ldots, n$, so for an optimal solution we must have $\epsilon \geq 0$ and $|w_i| = \epsilon_i$, that is, $\|w\|_1 = \epsilon_1 + \cdots + \epsilon_n$.

10. Alternating Direction Method of Multipliers

The *alternating direction method of multipliers*, for short ADMM, is the best method known for solving optimization problems for which the function J to be optimized can be split into two independent parts, as $J(x, z) = f(x) + g(z)$, and to consider the Minimization Problem (P_{admm}),

$$\begin{aligned} &\text{minimize} && f(x) + g(z) \\ &\text{subject to} && Ax + Bz = c, \end{aligned}$$

for some $p \times n$ matrix A, some $p \times m$ matrix B, and with $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, and $c \in \mathbb{R}^p$. We also assume that f and g are *convex*.

Iterative Steps of ADMM

The above problem can be solved using an iterative process applying to the *augmented Lagrangian*

$$L_\rho(x, z, \lambda) = f(x) + g(z) + \lambda^\top (Ax + Bz - c) + (\rho/2) \|Ax + Bz - c\|_2^2 ,$$

with $\lambda \in \mathbb{R}^p$ and for some $\rho > 0$.

Iterative Steps of ADMM

The above problem can be solved using an iterative process applying to the *augmented Lagrangian*

$$L_\rho(x, z, \lambda) = f(x) + g(z) + \lambda^\top (Ax + Bz - c) + (\rho/2) \|Ax + Bz - c\|_2^2,$$

with $\lambda \in \mathbb{R}^p$ and for some $\rho > 0$.

Given some initial values (z^0, λ^0), the *ADMM method* consists of the following iterative steps:

$$x^{k+1} = \arg\min_x L_\rho(x, z^k, \lambda^k)$$
$$z^{k+1} = \arg\min_z L_\rho(x^{k+1}, z, \lambda^k)$$
$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c).$$

ADMM Methodology of Sequential Updates

Instead of performing a minimization step jointly over x and z, as the step

$$(x^{k+1}, z^{k+1}) = \arg\min_{x,z} L_\rho(x, z, \lambda^k),$$

ADMM first performs an x-minimization step, and then a z-minimization step. Thus x and z are updated in an alternating or sequential fashion, which accounts for the term *alternating direction*.

Specializing ADMM to Quadratic Programs

We specialize ADMM to quadratic programs of the following form:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^\top Px + q^\top x + r \\ \text{subject to} & Ax = b,\ x \geq 0, \end{array}$$

where P is an $n \times n$ *symmetric positive semidefinite* matrix, $q \in \mathbb{R}^n$, $r \in \mathbb{R}$, and A is an $m \times n$ matrix of rank m.

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{aligned} &\text{minimize} && f(x) + g(z) \\ &\text{subject to} && x - z = 0, \end{aligned}$$

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x - z = 0, \end{array}$$

with

$$f(x) = \frac{1}{2} x^\top P x + q^\top x + r, \quad \text{dom}(f) = \{x \in \mathbb{R}^n \mid Ax = b\},$$

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x - z = 0, \end{array}$$

with

$$f(x) = \frac{1}{2} x^\top P x + q^\top x + r, \quad \text{dom}(f) = \{x \in \mathbb{R}^n \mid Ax = b\},$$

and

$$g = I_{\mathbb{R}^n_+},$$

the indicator function of the positive orthant $\mathbb{R}^n_+$.

Specializing ADMM to Quadratic Programs

Then ADMM consists of the following steps:

$$
\begin{aligned}
x^{k+1} &= \arg\min_{x} \left(f(x) + (\rho/2) \left\| x - z^k + u^k \right\|_2^2 \right) \\
z^{k+1} &= (x^{k+1} + u^k)_+ \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1},
\end{aligned}
$$

where $u^k = \lambda^k/\rho$ (this is the scaled version of ADMM). Here, v_+ is the vector obtained by setting the negative components of v to zero.

Specializing ADMM to Quadratic Programs

Then ADMM consists of the following steps:

$$
\begin{aligned}
x^{k+1} &= \arg\min_x \left(f(x) + (\rho/2) \left\| x - z^k + u^k \right\|_2^2 \right) \\
z^{k+1} &= (x^{k+1} + u^k)_+ \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1},
\end{aligned}
$$

where $u^k = \lambda^k/\rho$ (this is the scaled version of ADMM). Here, v_+ is the vector obtained by setting the negative components of v to zero. The x-update involves solving the KKT equations

$$
\begin{pmatrix} P + \rho I & A^\top \\ A & 0 \end{pmatrix} \begin{pmatrix} x^{k+1} \\ y \end{pmatrix} = \begin{pmatrix} -q + \rho(z^k - u^k) \\ b \end{pmatrix}.
$$

Lasso Regression: Program (lasso1) Solution

The best way to solve lasso minimization is to use ADMM.

Lasso Regression: Program (lasso1) Solution

The best way to solve lasso minimization is to use ADMM.

Lasso minimization can be stated as the following optimization problem:

$$\text{minimize} \quad (1/2)\,\|Ax - b\|_2^2 + \tau\,\|x\|_1\,,$$

with $A = X$, $b = y$ and $x = w$, to conform with our original formulation.

Lasso Regression: Program (lasso1) Solution

The best way to solve lasso minimization is to use ADMM.

Lasso minimization can be stated as the following optimization problem:

$$\text{minimize} \quad (1/2)\,\|Ax-b\|_2^2+\tau\,\|x\|_1\,,$$

with $A=X$, $b=y$ and $x=w$, to conform with our original formulation.

The *lasso minimization* is converted to the following problem in ADMM form:

$$\begin{array}{ll}\text{minimize} & \|Ax-b\|_2^2+\tau\,\|z\|_1\\ \text{subject to} & x-z=0.\end{array}$$

Lasso Regression: ADMM Solution

Then the ADMM procedure is

$$
\begin{aligned}
x^{k+1} &= (A^\top A + \rho I)^{-1}(A^\top b + \rho(z^k - u^k)) \\
z^{k+1} &= S_{\tau/\rho}(x^{k+1} + u^k) \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1}
\end{aligned}
$$

where $\rho > 0$ is some given constant.

Lasso Regression: ADMM Solution

Then the ADMM procedure is

$$
\begin{aligned}
x^{k+1} &= (A^\top A + \rho I)^{-1}(A^\top b + \rho(z^k - u^k)) \\
z^{k+1} &= S_{\tau/\rho}(x^{k+1} + u^k) \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1}
\end{aligned}
$$

where $\rho > 0$ is some given constant.

Since $\rho > 0$, the matrix $A^\top A + \rho I$ is symmetric positive definite. Note that the x-update looks like a *ridge regression step*.

Soft Thresholding Operator

In the above procedure, the function S_c known as a *soft thresholding operator*. If $v \in \mathbb{R}$ it is given by

$$S_c(v) = \begin{cases} v - c & \text{if } v > c \\ 0 & \text{if } |v| \leq c \\ v + c & \text{if } v < -c. \end{cases}$$

Soft Thresholding Operator

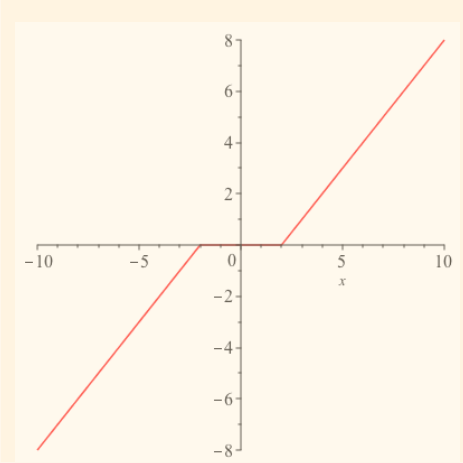

Figure 12: The graph of S_c (when $c = 2$).

Soft Thresholding Operator

The operator S_c is extended to vectors in $\mathbb{R}^n$ component wise, that is, if $x = (x_1, \ldots, x_n)$, then

$$S_c(x) = (S_c(x_1), \ldots, S_c(x_n)).$$

Soft Thresholding Operator

The operator S_c is extended to vectors in $\mathbb{R}^n$ component wise, that is, if $x = (x_1, \ldots, x_n)$, then

$$S_c(x) = (S_c(x_1), \ldots, S_c(x_n)).$$

The soft thresholding operator is one of the built-in functions in `Matlab`.

11. Classification/Separation Problem

In this section we describe the following *classification problem*, or perhaps more accurately, *separation problem* (into two classes).

11. Classification/Separation Problem

In this section we describe the following *classification problem*, or perhaps more accurately, *separation problem* (into two classes).

Suppose we have two nonempty disjoint finite sets of p *blue* points $\{u_i\}_{i=1}^{p}$ and q *red* points $\{v_j\}_{j=1}^{q}$ in $\mathbb{R}^n$

11. Classification/Separation Problem

In this section we describe the following *classification problem*, or perhaps more accurately, *separation problem* (into two classes).

Suppose we have two nonempty disjoint finite sets of p *blue* points $\{u_i\}_{i=1}^p$ and q *red* points $\{v_j\}_{j=1}^q$ in $\mathbb{R}^n$

Our goal is to find a hyperplane H of equation $w^\top x - b = 0$ (where $w \in \mathbb{R}^n$ is a nonzero vector and $b \in \mathbb{R}$), such that all the blue points u_i are in one of the two open half-spaces determined by H, and all the red points v_j are in the other open half-space determined by H.

Classification/Separation Problem

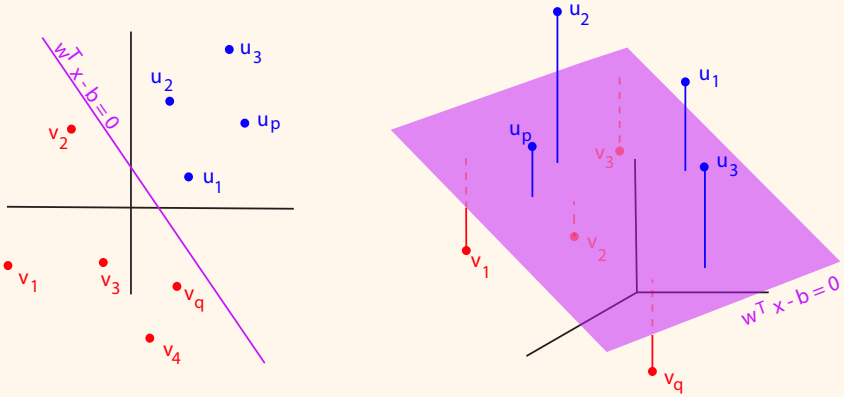

Figure 13: Two examples of the SVM separation problem. The left figure is SVM in $\mathbb{R}^2$, while the right figure is SVM in $\mathbb{R}^3$.

Classification/Separation Problem

Without loss of generality, we may assume that

$$w^\top u_i - b > 0 \qquad \text{for } i = 1, \ldots, p$$
$$w^\top v_j - b < 0 \qquad \text{for } j = 1, \ldots, q.$$

Classification/Separation Problem

Of course, separating the blue and the red points may be impossible, as we will see in the next figure for four points where the line segments (u_1, u_2) and (v_1, v_2) intersect.

Classification/Separation Problem

Of course, separating the blue and the red points may be impossible, as we will see in the next figure for four points where the line segments (u_1, u_2) and (v_1, v_2) intersect.

If a hyperplane separating the two subsets of blue and red points exists, we say that they are *linearly separable*.

Example of An Inseparable Problem

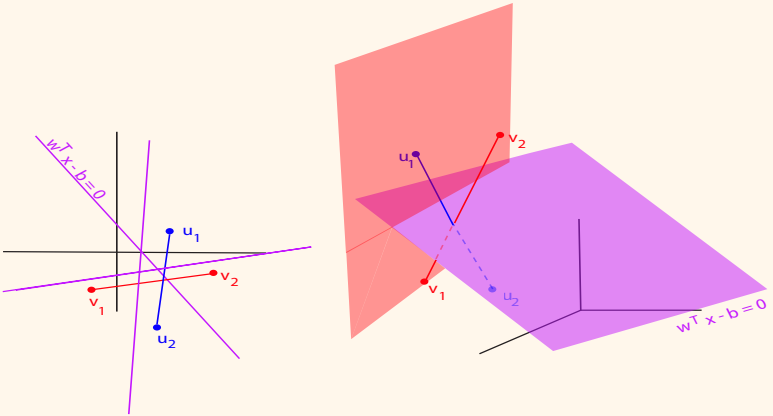

Figure 14: Two examples in which it is impossible to find purple hyperplanes which separate the red and blue points.

Choosing the Hyperplane

Consider our two subsets of p *blue* points $\{u_i\}_{i=1}^{p}$ and q *red* points $\{v_j\}_{j=1}^{q}$.

Choosing the Hyperplane

Consider our two subsets of p *blue* points $\{u_i\}_{i=1}^{p}$ and q *red* points $\{v_j\}_{j=1}^{q}$.

Since there are infinitely many hyperplanes separating the two subsets (if indeed the two subsets are linearly separable), we would like to come up with a "good" criterion for choosing such a hyperplane.

Hard Margin Support Vector Machine

The idea that was advocated by Vapnik is to consider the distances $d(u_i, H)$ and $d(v_j, H)$ from *all* the points to the hyperplane H, and to pick a hyperplane H that *maximizes the smallest of these distances*.

Hard Margin Support Vector Machine

The idea that was advocated by Vapnik is to consider the distances $d(u_i, H)$ and $d(v_j, H)$ from *all* the points to the hyperplane H, and to pick a hyperplane H that *maximizes the smallest of these distances*.

In machine learning this strategy is called finding a *maximal margin hyperplane*, or *hard margin support vector machine*.

Distance from Point to Hyperplane

Since the distance from a point x to the hyperplane H of equation $w^\top x - b = 0$ is

$$d(x, H) = \frac{|w^\top x - b|}{\|w\|},$$

(where $\|w\| = \sqrt{w^\top w}$ is the Euclidean norm of w), it is convenient to temporarily assume that $\|w\| = 1$, so that

$$d(x, H) = |w^\top x - b|.$$

See the following figure.

Distance from Point to Hyperplane

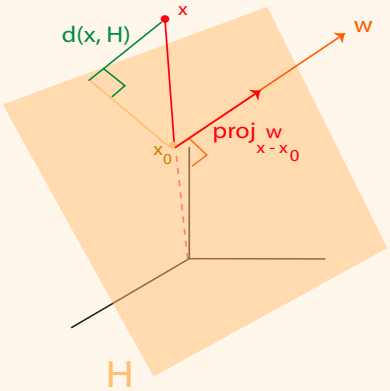

Figure 15: In $\mathbb{R}^3$, the distance from a point to the plane $w^\top x - b = 0$ is given by the projection onto the normal w.

Hard Margin Support Vector Machine

Then with our sign convention, we have

$$d(u_i, H) = w^\top u_i - b \qquad i = 1, \ldots, p$$
$$d(v_j, H) = -w^\top v_j + b \qquad j = 1, \ldots, q.$$

Hard Margin Support Vector Machine

If we let

$$\delta = \min\{d(u_i, H),\ d(v_j, H) \mid 1 \leq i \leq p,\ 1 \leq j \leq q\},$$

then the hyperplane H should be chosen so that

$$\begin{aligned} w^\top u_i - b &\geq \delta & i &= 1, \ldots, p \\ -w^\top v_j + b &\geq \delta & j &= 1, \ldots, q, \end{aligned}$$

and such that $\delta > 0$ is *maximal*.

Hard Margin Support Vector Machine

If we let

$$\delta = \min\{d(u_i, H),\ d(v_j, H) \mid 1 \leq i \leq p,\ 1 \leq j \leq q\},$$

then the hyperplane H should be chosen so that

$$\begin{aligned} w^\top u_i - b &\geq \delta & i &= 1, \ldots, p \\ -w^\top v_j + b &\geq \delta & j &= 1, \ldots, q, \end{aligned}$$

and such that $\delta > 0$ is *maximal*.

The distance δ is called the *margin* associated with the hyperplane H.

Formulating the Separation Problem SVM_{h1}

This is indeed one way of formulating the two-class separation problem as an optimization problem with a linear objective function $J(\delta, w, b) = \delta$, and affine and quadratic constraints (SVM_{h1}):

$$
\begin{aligned}
&\text{maximize} \quad \delta \\
&\text{subject to} \\
&\qquad w^\top u_i - b \geq \delta \qquad i = 1, \ldots, p \\
&\qquad -w^\top v_j + b \geq \delta \qquad j = 1, \ldots, q \\
&\qquad \|w\| \leq 1.
\end{aligned}
$$

Formulating the Separation Problem SVM_{h1}

This is indeed one way of formulating the two-class separation problem as an optimization problem with a linear objective function $J(\delta, w, b) = \delta$, and affine and quadratic constraints (SVM_{h1}):

$$
\begin{aligned}
&\text{maximize} \quad \delta \\
&\text{subject to} \\
&\qquad w^\top u_i - b \geq \delta \qquad i = 1, \ldots, p \\
&\qquad -w^\top v_j + b \geq \delta \qquad j = 1, \ldots, q \\
&\qquad \|w\| \leq 1.
\end{aligned}
$$

This problem has an optimal solution $\delta > 0$ iff the two subsets are linearly separable.

The Optimal Solution for SVM_{h1}

We used the constraint $\|w\| \leq 1$ rather than $\|w\| = 1$ because the former is qualified, whereas the latter is not. But if (w, b, δ) is an optimal solution, then $\|w\| = 1$, as shown in the following proposition.

The Optimal Solution for SVM_{h1}

We used the constraint $\|w\| \leq 1$ rather than $\|w\| = 1$ because the former is qualified, whereas the latter is not. But if (w, b, δ) is an optimal solution, then $\|w\| = 1$, as shown in the following proposition.

Proposition. If (w, b, δ) is an optimal solution of Problem (SVM_{h1}), so in particular $\delta > 0$, then we must have $\|w\| = 1$.

The Optimal Solution for SVM_{h1}

Vapnik proved that if the two subsets are linearly separable, then Problem (SVM_{h1}) has a *unique* optimal solution.

The Optimal Solution for SVM_{h1}

Vapnik proved that if the two subsets are linearly separable, then Problem (SVM_{h1}) has a *unique* optimal solution.

Theorem. If two disjoint subsets of p *blue* points $\{u_i\}_{i=1}^{p}$ and q *red* points $\{v_j\}_{j=1}^{q}$ are linearly separable, then Problem (SVM_{h1}) has a *unique* optimal solution consisting of a hyperplane of equation $w^\top x - b = 0$ separating the two subsets with maximum margin δ.

12. Converting from Affine to Quadratic Functional

Since $\delta > 0$ (otherwise the data would not be separable into two disjoint sets), we can divide the affine constraints by δ to obtain

$$\begin{aligned} w'^{\top} u_i - b' &\geq 1 && i = 1, \ldots, p \\ -w'^{\top} v_j + b' &\geq 1 && j = 1, \ldots, q, \end{aligned}$$

except that now, w' is not necessarily a unit vector.

Converting from Affine to Quadratic Functional

To obtain the distances to the hyperplane H, we need to divide by $\|w'\|$ and then we have

$$\frac{w'^{\top} u_i - b'}{\|w'\|} \geq \frac{1}{\|w'\|} \qquad i = 1, \ldots, p$$

$$\frac{-w'^{\top} v_j + b'}{\|w'\|} \geq \frac{1}{\|w'\|} \qquad j = 1, \ldots, q,$$

Converting from Affine to Quadratic Functional

To obtain the distances to the hyperplane H, we need to divide by $\|w'\|$ and then we have

$$\frac{w'^{\top} u_i - b'}{\|w'\|} \geq \frac{1}{\|w'\|} \qquad i = 1, \ldots, p$$

$$\frac{-w'^{\top} v_j + b'}{\|w'\|} \geq \frac{1}{\|w'\|} \qquad j = 1, \ldots, q,$$

which means that the shortest distance from the data points to the hyperplane is $\delta = 1/\|w'\|$.

The Optimization Problem (SVM_{h2})

Therefore, we wish to *maximize* $1/\|w\|$, that is, to *minimize* $\|w\|$, so we obtain the following optimization Problem (SVM_{h2}):

The Optimization Problem (SVM_{h2})

Therefore, we wish to *maximize* $1/\|w\|$, that is, to *minimize* $\|w\|$, so we obtain the following optimization Problem (SVM_{h2}):

Hard margin SVM (SVM_{h2}):

$$\begin{aligned}
&\text{minimize} \quad \frac{1}{2}\|w\|^2 \\
&\text{subject to} \\
&\qquad w^\top u_i - b \geq 1 \qquad i = 1, \ldots, p \\
&\qquad -w^\top v_j + b \geq 1 \qquad j = 1, \ldots, q.
\end{aligned}$$

Solving (SVM_{h2}) *Via the KKT Conditions*

The objective function $J(w) = 1/2 \|w\|^2$ is *convex*, so the last proposition of the KKT lesson applies and gives us a necessary and sufficient condition for having a minimum in terms of the KKT conditions.

Solving (SVM_{h2}) *Via the KKT Conditions*

The objective function $J(w) = 1/2\,\|w\|^2$ is *convex*, so the last proposition of the KKT lesson applies and gives us a necessary and sufficient condition for having a minimum in terms of the KKT conditions.

Observe that the trivial solution $w = 0$ is impossible, because the blue constraints would be

$$-b \geq 1,$$

Solving (SVM_{h2}) *Via the KKT Conditions*

The objective function $J(w) = 1/2\,\|w\|^2$ is *convex*, so the last proposition of the KKT lesson applies and gives us a necessary and sufficient condition for having a minimum in terms of the KKT conditions.

Observe that the trivial solution $w = 0$ is impossible, because the blue constraints would be

$$-b \geq 1,$$

that is $b \leq -1$, and the red constraints would be

$$b \geq 1,$$

but these are contradictory.

Solving (SVM_{h2}) *via the Lagrangian*

Our goal is to find w and b, and optionally, $\delta = 1/\|w\|$. In theory this can be done using the KKT conditions but in the present case it is much more efficient to solve the dual.

13. Solving Hard Margin SVM Problem (SVM_{h2})

Recall the **Hard margin SVM** problem (SVM_{h2}):

$$
\begin{aligned}
&\text{minimize} \quad \frac{1}{2}\|w\|^2, && w \in \mathbb{R}^n \\
&\text{subject to} \\
&\qquad w^\top u_i - b \geq 1 && i = 1, \ldots, p \\
&\qquad -w^\top v_j + b \geq 1 && j = 1, \ldots, q.
\end{aligned}
$$

13. Solving Hard Margin SVM Problem (SVM_{h2})

Recall the **Hard margin SVM** problem (SVM_{h2}):

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|w\|^2, && w \in \mathbb{R}^n \\ &\text{subject to} \\ &&& w^\top u_i - b \geq 1 && i = 1, \ldots, p \\ &&& -w^\top v_j + b \geq 1 && j = 1, \ldots, q. \end{aligned}$$

We need to find the dual program.

Lagrangian of Hard Margin (SVM_{h2})

Step 1. Write the Lagrangian in matrix form.

Lagrangian of Hard Margin (SVM_{h2})

Step 1. Write the Lagrangian in matrix form.

Let X be the $n \times (p+q)$ matrix given by

$$X = \begin{pmatrix} -u_1 & \cdots & -u_p & v_1 & \cdots & v_q \end{pmatrix}.$$

Lagrangian of Hard Margin (SVM_{h2})

Step 1. Write the Lagrangian in matrix form.

Let X be the $n \times (p+q)$ matrix given by

$$X = \begin{pmatrix} -u_1 & \cdots & -u_p & v_1 & \cdots & v_q \end{pmatrix}.$$

We obtain the Lagrangian

$$\begin{aligned} L(w, b, \lambda, \mu) = &\frac{1}{2} \begin{pmatrix} w^\top & b \end{pmatrix} \begin{pmatrix} I_n & 0_n \\ 0_n^\top & 0 \end{pmatrix} \begin{pmatrix} w \\ b \end{pmatrix} + \\ &\begin{pmatrix} w^\top & b \end{pmatrix} \begin{pmatrix} X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} \\ \mathbf{1}_p^\top \lambda & -\mathbf{1}_q^\top \mu \end{pmatrix} + \begin{pmatrix} \lambda^\top & \mu^\top \end{pmatrix} \mathbf{1}_{p+q}. \end{aligned}$$

Dual Function of Hard Margin (SVM_{h2})

Step 2. Find the dual function $G(\lambda, \mu)$.

Dual Function of Hard Margin (SVM_{h2})

Step 2. Find the dual function $G(\lambda, \mu)$.

In order to find the dual function $G(\lambda, \mu)$, we need to minimize $L(w, b, \lambda, \mu)$ with respect to w and b and for this, since the objective function J is convex and since $\mathbb{R}^{n+1}$ is convex and open, a necessary and sufficient condition for a minimum is that $\nabla L_{w,b} = 0$, where $\nabla L_{w,b}$ is the gradient of $L(w, b, \lambda, \mu)$.

Dual Function of Hard Margin (SVM_{h2})

Step 2. Find the dual function $G(\lambda, \mu)$.

In order to find the dual function $G(\lambda, \mu)$, we need to minimize $L(w, b, \lambda, \mu)$ with respect to w and b and for this, since the objective function J is convex and since $\mathbb{R}^{n+1}$ is convex and open, a necessary and sufficient condition for a minimum is that $\nabla L_{w,b} = 0$, where $\nabla L_{w,b}$ is the gradient of $L(w, b, \lambda, \mu)$.

We have

$$\nabla L_{w,b} = \begin{pmatrix} w + X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} \\ \mathbf{1}_p^\top \lambda \quad -\mathbf{1}_q^\top \mu \end{pmatrix}.$$

Dual Function of Hard Margin (SVM_{h2})

The necessary and sufficient condition for a minimum is

$$\nabla L_{w,b} = 0,$$

Dual Function of Hard Margin (SVM_{h2})

The necessary and sufficient condition for a minimum is

$$\nabla L_{w,b} = 0,$$

which yields

$$w = -X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} \qquad (*_1)$$

Dual Function of Hard Margin (SVM_{h2})

The necessary and sufficient condition for a minimum is

$$\nabla L_{w,b} = 0,$$

which yields

$$w = -X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} \qquad (*_1)$$

and

$$\mathbf{1}_p^\top \lambda - \mathbf{1}_q^\top \mu = 0. \qquad (*_2)$$

Dual Function of Hard Margin (SVM_{h2})

Step 3. Write the dual as a minimization problem.

Dual Function of Hard Margin (SVM_{h2})

Step 3. Write the dual as a minimization problem.

Maximizing the dual function $G(\lambda, \mu)$ over its domain of definition is equivalent to maximizing

$$\widehat{G}(\lambda, \mu) = -\frac{1}{2} \begin{pmatrix} \lambda^\top & \mu^\top \end{pmatrix} X^\top X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} + \begin{pmatrix} \lambda^\top & \mu^\top \end{pmatrix} \mathbf{1}_{p+q}$$

Dual Function of Hard Margin (SVM_{h2})

Step 3. Write the dual as a minimization problem.

Maximizing the dual function $G(\lambda, \mu)$ over its domain of definition is equivalent to maximizing

$$\widehat{G}(\lambda, \mu) = -\frac{1}{2} \begin{pmatrix} \lambda^\top & \mu^\top \end{pmatrix} X^\top X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} + \begin{pmatrix} \lambda^\top & \mu^\top \end{pmatrix} \mathbf{1}_{p+q}$$

subject to the constraint

$$\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0,$$

Convert Dual to a Minimization Problem

so we formulate the dual program as,

$$\begin{aligned}
&\text{maximize} \quad -\frac{1}{2}\begin{pmatrix}\lambda^\top & \mu^\top\end{pmatrix} X^\top X \begin{pmatrix}\lambda \\ \mu\end{pmatrix} + \begin{pmatrix}\lambda^\top & \mu^\top\end{pmatrix} \mathbf{1}_{p+q} \\
&\text{subject to} \\
&\qquad \sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0 \\
&\qquad \lambda \geq 0,\ \mu \geq 0,
\end{aligned}$$

Dual Function of Hard Margin (SVM_{h2})

or equivalently, **Dual of the Hard margin SVM** (SVM_{h2}):

$$\begin{aligned}
&\text{minimize} \quad \frac{1}{2}\begin{pmatrix}\lambda^\top & \mu^\top\end{pmatrix} X^\top X \begin{pmatrix}\lambda \\ \mu\end{pmatrix} - \begin{pmatrix}\lambda^\top & \mu^\top\end{pmatrix} \mathbf{1}_{p+q} \\
&\text{subject to} \\
&\qquad \sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0 \\
&\qquad \lambda \geq 0,\ \mu \geq 0.
\end{aligned}$$

Solving the Dual Program of (SVM_{h2})

Step 4. Solve the dual program.

Solving the Dual Program of (SVM_{h2})

Step 4. Solve the dual program.

This step involves using numerical procedures typically based on gradient descent to find λ and μ, for example, ADMM.

Solving the Dual Program of (SVM_{h2})

Step 4. Solve the dual program.

This step involves using numerical procedures typically based on gradient descent to find λ and μ, for example, ADMM.

Once λ and μ are determined, w is determined by $(*_1)$, namely

$$w = -X\begin{pmatrix}\lambda\\ \mu\end{pmatrix}.$$

Solving the Dual Program of (SVM_{h2})

Step 4. Solve the dual program.

This step involves using numerical procedures typically based on gradient descent to find λ and μ, for example, ADMM.

Once λ and μ are determined, w is determined by $(*_1)$, namely

$$w = -X\begin{pmatrix}\lambda\\ \mu\end{pmatrix}.$$

To determine b we use the KKT conditions.

Using the KKT Conditions of (SVM_{h2})

Because the primal always has a solution, so does the dual, which implies that there is at least some i_0 such that $\lambda_{i_0} > 0$. But then the constraint $\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$ implies that there is also some j_0 such that $\mu_{j_0} > 0$.

Using the KKT Conditions of (SVM_{h2})

Because the primal always has a solution, so does the dual, which implies that there is at least some i_0 such that $\lambda_{i_0} > 0$. But then the constraint $\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$ implies that there is also some j_0 such that $\mu_{j_0} > 0$.

By the KKT conditions, since the corresponding constraints are active, we have

$$w^\top u_{i_0} - b = 1, \quad -w^\top v_{j_0} + b = 1,$$

Using the KKT Conditions of (SVM_{h2})

Because the primal always has a solution, so does the dual, which implies that there is at least some i_0 such that $\lambda_{i_0} > 0$. But then the constraint $\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$ implies that there is also some j_0 such that $\mu_{j_0} > 0$.

By the KKT conditions, since the corresponding constraints are active, we have

$$w^\top u_{i_0} - b = 1, \quad -w^\top v_{j_0} + b = 1,$$

so we obtain

$$b = w^\top (u_{i_0} + v_{j_0})/2.$$

Using the KKT Conditions of (SVM_{h2})

Because the primal always has a solution, so does the dual, which implies that there is at least some i_0 such that $\lambda_{i_0} > 0$. But then the constraint $\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$ implies that there is also some j_0 such that $\mu_{j_0} > 0$.

By the KKT conditions, since the corresponding constraints are active, we have

$$w^\top u_{i_0} - b = 1, \quad -w^\top v_{j_0} + b = 1,$$

so we obtain

$$b = w^\top (u_{i_0} + v_{j_0})/2.$$

The *support vectors* are those for which the constraints are active.

14. SVM Soft Margin Problem

SVM picks a hyperplane which *maximizes the minimum distance* from these points to the separating hyperplane.

14. SVM Soft Margin Problem

SVM picks a hyperplane which *maximizes the minimum distance* from these points to the separating hyperplane.

We now consider the problem of separating two disjoint sets of points, $\{u_i\}_{i=1}^{p}$ and $\{v_j\}_{j=1}^{q}$, but this time *we do not assume that these two sets are separable*.

14. SVM Soft Margin Problem

SVM picks a hyperplane which *maximizes the minimum distance* from these points to the separating hyperplane.

We now consider the problem of separating two disjoint sets of points, $\{u_i\}_{i=1}^{p}$ and $\{v_j\}_{j=1}^{q}$, but this time *we do not assume that these two sets are separable*.

To cope with nonseparability, we allow points to invade the safety zone around the separating hyperplane, and even points on the wrong side of the hyperplane. Such a method is called *soft margin support vector machine*.

Example of a soft SVM problem

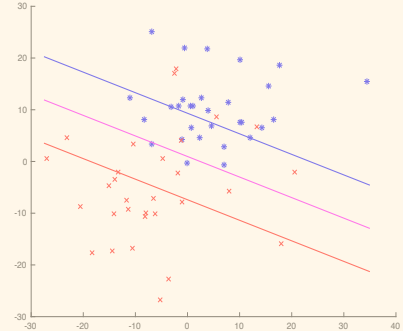

Figure 16: A soft SVM problem for two sets of 30 points.

Soft Margin Support Vector Machines

It turns out that the soft margin SVM arising from Problem (SVM_{h1}) has some problem (a potential division by 0). Soft margin SVMs arising from Problem (SVM_{h2}) do not suffer from this problem.

Soft Margin Support Vector Machines

If the sets of points $\{u_1, \ldots, u_p\}$ and $\{v_1, \ldots, v_q\}$ are not linearly separable (with $u_i, v_j \in \mathbb{R}^n$), we can use a trick from linear programming which is to introduce nonnegative "slack variables" $\epsilon = (\epsilon_1, \ldots, \epsilon_p) \in \mathbb{R}^p$ and $\xi = (\xi_1, \ldots, \xi_q) \in \mathbb{R}^q$ to relax the "hard" constraints

Soft Margin Support Vector Machines

If the sets of points $\{u_1, \ldots, u_p\}$ and $\{v_1, \ldots, v_q\}$ are not linearly separable (with $u_i, v_j \in \mathbb{R}^n$), we can use a trick from linear programming which is to introduce nonnegative "slack variables" $\epsilon = (\epsilon_1, \ldots, \epsilon_p) \in \mathbb{R}^p$ and $\xi = (\xi_1, \ldots, \xi_q) \in \mathbb{R}^q$ to relax the "hard" constraints

$$\begin{aligned} w^\top u_i - b &\geq 1 && i = 1, \ldots, p \\ -w^\top v_j + b &\geq 1 && j = 1, \ldots, q \end{aligned}$$

of Problem (SVM_{h2}) to the "soft" constraints

$$\begin{aligned} w^\top u_i - b &\geq 1 - \epsilon_i, \quad \epsilon_i \geq 0 \quad i = 1, \ldots, p \\ -w^\top v_j + b &\geq 1 - \xi_j, \quad \xi_j \geq 0 \quad j = 1, \ldots, q. \end{aligned}$$

Soft Margin Support Vector Machines

If $\epsilon_i > 0$, the point u_i may be misclassified, in the sense that it can belong to the margin (the slab), or even to the wrong half-space classifying the negative (red) points.

Soft Margin Support Vector Machines

If $\epsilon_i > 0$, the point u_i may be misclassified, in the sense that it can belong to the margin (the slab), or even to the wrong half-space classifying the negative (red) points.

If $\epsilon = 0$ and $\xi = 0$, then we recover the original constraints. By making ϵ and ξ large enough, these constraints can always be satisfied.

Soft Margin Support Vector Machines

Ideally we would like to find a separating hyperplane that *minimizes the number of misclassified points*, which means that the variables ϵ_i and ξ_j should be as small as possible, but there is a trade-off in maximizing the margin (the thickness of the slab), and minimizing the number of misclassified points.

Soft Margin Support Vector Machines

This is reflected in the choice of the objective function, and there are several options, depending on whether we minimize a linear function of the variables ϵ_i and ξ_j, or a quadratic functions of these variables, or whether we include the term $(1/2)b^2$ in the objective function.

Soft Margin Support Vector Machines

This is reflected in the choice of the objective function, and there are several options, depending on whether we minimize a linear function of the variables ϵ_i and ξ_j, or a quadratic functions of these variables, or whether we include the term $(1/2)b^2$ in the objective function.

These methods are known as *support vector classification* algorithms (for short *SVC algorithms*).

Soft Margin Support Vector Machines

A more flexible problem is obtained by using the margin $\delta = \eta / \|w\|$, where η is some positive constant that we wish to maximize.

Soft Margin Support Vector Machines

A more flexible problem is obtained by using the margin $\delta = \eta / \|w\|$, where η is some positive constant that we wish to maximize.

To do so, we add a term $-K_m\eta$ to the objective function $(1/2)w^\top w$, as well as the "regularizing term"

$$K_s\left(\sum_{i=1}^{p}\epsilon_i + \sum_{j=1}^{q}\xi_j\right)$$

whose purpose is to make ϵ and ξ sparse.

Soft Margin Support Vector Machine ($\mathrm{SVM}_{s2'}$)

We consider the following version of the soft margin support vector machine:

Soft Margin Support Vector Machine ($\mathrm{SVM}_{s2'}$)

We consider the following version of the soft margin support vector machine:

Soft margin SVM ($\mathrm{SVM}_{s2'}$):

$$
\begin{aligned}
&\text{minimize} \quad \frac{1}{2} w^\top w - K_m \eta + K_s \begin{pmatrix} \epsilon^\top & \xi^\top \end{pmatrix} \mathbf{1}_{p+q} \\
&\text{subject to} \\
&\qquad w^\top u_i - b \geq \eta - \epsilon_i, \quad \epsilon_i \geq 0 \qquad i = 1, \ldots, p \\
&\qquad -w^\top v_j + b \geq \eta - \xi_j, \quad \xi_j \geq 0 \qquad j = 1, \ldots, q \\
&\qquad\qquad \eta \geq 0.
\end{aligned}
$$

Soft Margin Support Vector Machine ($\mathrm{SVM}_{s2'}$)

This version of the SVM problem was first discussed in Schölkopf, Smola, Williamson, and Bartlett under the name of *ν-SVC* (or *ν-SVM*), and also used in Schölkopf, Platt, Shawe–Taylor, and Smola.

Soft Margin Support Vector Machine ($\mathrm{SVM}_{s2'}$)

This version of the SVM problem was first discussed in Schölkopf, Smola, Williamson, and Bartlett under the name of *ν-SVC* (or *ν-SVM*), and also used in Schölkopf, Platt, Shawe–Taylor, and Smola.

For this problem it is no longer clear that if $(w, \eta, b, \epsilon, \xi)$ is an optimal solution, then $w \neq 0$ and $\eta > 0$.

Soft Margin Support Vector Machine ($\mathrm{SVM}_{s2'}$)

This version of the SVM problem was first discussed in Schölkopf, Smola, Williamson, and Bartlett under the name of *ν-SVC* (or *ν-SVM*), and also used in Schölkopf, Platt, Shawe–Taylor, and Smola.

For this problem it is no longer clear that if $(w, \eta, b, \epsilon, \xi)$ is an optimal solution, then $w \neq 0$ and $\eta > 0$.

In fact, if the sets of points are not linearly separable and if K_s is chosen too big, Problem $(\mathrm{SVM}_{s2'})$ *may fail to have an optimal solution*.

Dual Program of ($\mathrm{SVM}_{s2'}$)

Dual of Soft margin SVM ($\mathrm{SVM}_{s2'}$):

$$\text{minimize} \quad \frac{1}{2}\begin{pmatrix}\lambda^\top & \mu^\top\end{pmatrix} X^\top X \begin{pmatrix}\lambda \\ \mu\end{pmatrix}$$

subject to

$$\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$$

$$\sum_{i=1}^{p} \lambda_i + \sum_{j=1}^{q} \mu_j \geq K_m$$

$$0 \leq \lambda_i \leq K_s, \quad i = 1, \ldots, p$$

$$0 \leq \mu_j \leq K_s, \quad j = 1, \ldots, q.$$

Solving the Dual and Computing $\mathbf{w}$

The dual program is solved using ADMM.

Solving the Dual and Computing w

The dual program is solved using ADMM.

If the primal problem is solvable, this yields solutions for λ and μ. Once a solution for λ and μ is obtained, we have

$$w = -X \begin{pmatrix} \lambda \\ \mu \end{pmatrix} = \sum_{i=1}^{p} \lambda_i u_i - \sum_{j=1}^{q} \mu_j v_j.$$

Solving the Dual and Computing w

The dual program is solved using ADMM.

If the primal problem is solvable, this yields solutions for λ and μ. Once a solution for λ and μ is obtained, we have

$$w = -X\begin{pmatrix}\lambda\\ \mu\end{pmatrix} = \sum_{i=1}^{p}\lambda_i u_i - \sum_{j=1}^{q}\mu_j v_j.$$

It remains to determine b, η, ϵ and ξ. The solution of the dual *does not* determine b, η, ϵ, ξ directly, and we are not aware of necessary and sufficient conditions that ensure that they can be determined.

Computing Remaining ($\mathrm{SVM}_{s2'}$) *Parameters*

The best we can do is to use the KKT conditions.

Computing Remaining $(\mathrm{SVM}_{s2'})$ *Parameters*

The best we can do is to use the KKT conditions.

In order to determine b and η we assume the following condition:

Standard Margin Hypothesis for $(\mathrm{SVM}_{s2'})$: There is some i_0 such that $0 < \lambda_{i_0} < K_s$, and there is some μ_{j_0} such that $0 < \mu_{j_0} < K_s$.

Computing Remaining $(\mathrm{SVM}_{s2'})$ *Parameters*

Under the **Standard Margin Hypothesis** for $(\mathrm{SVM}_{s2'})$, there is some i_0 such that $0 < \lambda_{i_0} < K_s$ and some j_0 such that $0 < \mu_{j_0} < K_s$, and by the complementary slackness conditions $\epsilon_{i_0} = 0$ and $\xi_{j_0} = 0$, so we have the *two active constraints*

$$w^\top u_{i_0} - b = \eta, \quad -w^\top v_{j_0} + b = \eta,$$

Standard Margin Hypothesis

and we can solve for b and η and we get

$$
\begin{aligned}
b &= \frac{w^\top u_{i_0} + w^\top v_{j_0}}{2} \\
\eta &= \frac{w^\top u_{i_0} - w^\top v_{j_0}}{2} \\
\delta &= \frac{\eta}{\|w\|}.
\end{aligned}
$$

Conclusion

Standard problems in machine learning, such as

(1) Learning a function.

(2) Separation, Classification of Data.

Conclusion

Standard problems in machine learning, such as

(1) Learning a function.

(2) Separation, Classification of Data.

Can be solved effectively using methods from linear algebra and (convex) optimization.

Conclusion

Standard problems in machine learning, such as

(1) Learning a function.

(2) Separation, Classification of Data.

Can be solved effectively using methods from linear algebra and (convex) optimization.

Details can be found in
Linear Algebra and Optimization with Applications to Machine Learning, Vol I and II, by Jean Gallier and Jocelyn Quaintance, World Scientific (2020).