Mathematics of Machine Learning

Version 0.5

Christopher Griffin

Contents

List of Figures

About This Document

This is a set of lectures notes about the Mathematics of Machine Learning.

CHAPTER 1

Perceptron and Elementary Linear Classifiers

1. Hyperplanes and Half-Spaces

DEFINITION 1.1 (Hyperplane). Let $\mathbf{w} \in \mathbb{R}^n$ be a constant vector in n-dimensional space and let $b \in \mathbb{R}$ be a constant scalar. The set of points

$$H = \left\{\mathbf{x} \in \mathbb{R}^n | \mathbf{w}^T\mathbf{x} = b\right\} \tag{1.1}$$

is a *hyperplane* in n-dimensional space. *Note the use of column vectors for* $\mathbf{w}$ *and* $\mathbf{x}$ *in this definition.*

EXAMPLE 1.2. Consider the hyper-plane $2x_1 + 3x_2 + x_3 = 5$. This is shown in Figure 1.1. This hyperplane is composed of the set of points $(x_1, x_2, x_3) \in \mathbb{R}^3$ satisfying $2x_1+3x_2+x_3 = 5$.

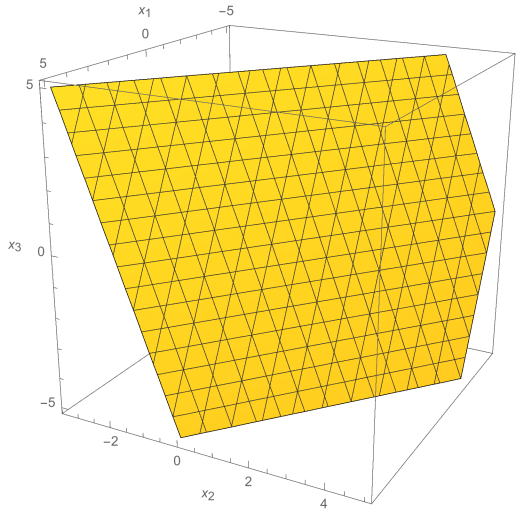


Figure 1.1. A hyperplane in 3 dimensional space: A hyperplane is the set of points satisfying an equation $\mathbf{w}^T\mathbf{x} = b$, where k is a constant in $\mathbb{R}$ and $\mathbf{w}$ is a constant vector in $\mathbb{R}^n$ and $\mathbf{x}$ is a variable vector in $\mathbb{R}^n$. The equation is written as a matrix multiplication using our assumption that all vectors are column vectors.

This can be plotted implicitly or explicitly by solving for one of the variables, say x_3. We can write x_3 as a function of the other two variables as:

$$x_3 = 5 - 2x_1 - 3x_2 \tag{1.2}$$

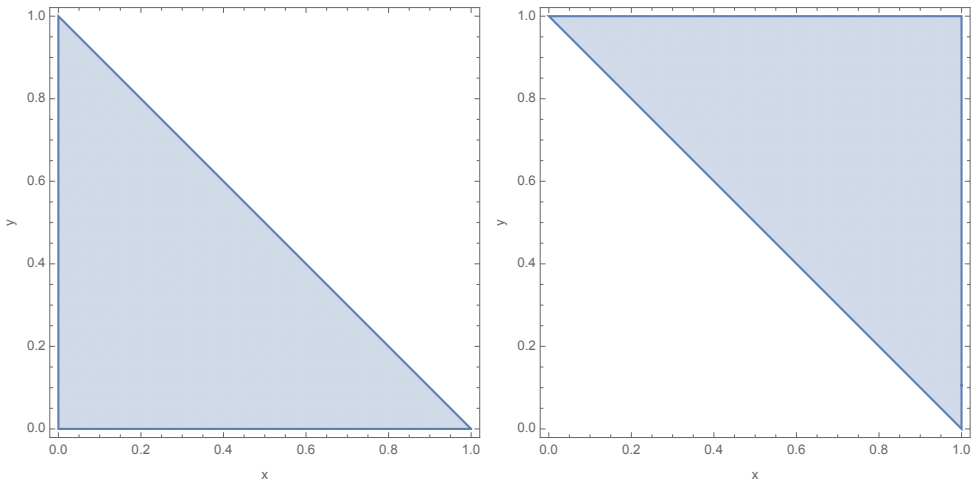
DEFINITION 1.3 (Half-Space). Let $\mathbf{w} \in \mathbb{R}^n$ be a constant vector in n-dimensional space and let $b \in \mathbb{R}$ be a constant scalar. The sets of points

$$H_l(\mathbf{w}, b) = \left\{\mathbf{x} \in \mathbb{R}^n | \mathbf{w}^T\mathbf{x} \leq b\right\} \tag{1.3}$$

$$H_u(\mathbf{w}, b) = \left\{\mathbf{x} \in \mathbb{R}^n | \mathbf{w}^T\mathbf{x} \geq b\right\} \tag{1.4}$$

are the half-spaces defined by the hyperplane $\mathbf{w}^T\mathbf{x} = b$. If $b = 0$, it is sufficient to write $H_l(\mathbf{w})$ or $H_u(\mathbf{w})$.

EXAMPLE 1.4. Consider the two dimensional hyperplane (line) $x_1 + x_2 = 1$. Then the two half-spaces associated with this hyper-plane are shown in Figure 1.2. A half-space is



(a) H_l (b) H_u

Figure 1.2. Two half-spaces defined by a hyper-plane: A half-space is so named because any hyper-plane divides $\mathbb{R}^n$ (the space in which it resides) into two halves, the side "on top" and the side "on the bottom."

so named because the hyperplane $\mathbf{w}^T\mathbf{x} = b$ literally separates $\mathbb{R}^n$ into two halves: the half above the hyperplane and the half below the hyperplane.

DEFINITION 1.5 (Polyhedral Set). If $P \subseteq \mathbb{R}^n$ is the intersection of a finite number of half-spaces, then P is a *polyhedral set*. Formally, let $\mathbf{w}_1, \ldots, \mathbf{w}_m \in \mathbb{R}^n$ be a finite set of constant vectors and let $b_1, \ldots, b_m \in \mathbb{R}$ be constants. Consider the set of half-spaces:

$$H_i = \{\mathbf{x} | \mathbf{w}_i^T\mathbf{x} \leq b_i\}$$

Then the set:

$$P = \bigcap_{i=1}^{m} H_i \tag{1.5}$$

is a *polyhedral set*.

EXERCISE 1.1. Show that the familiar triangle (polygon) with points $(0, 0)$, $(1, 0)$ and $(0, 1)$ is a polyhedral set by finding three half-spaces whose intersection defines it.

REMARK 1.6. Note that a half-space is tautologically a polyhedral set. A hyperplane defined by $\mathbf{w}^T\mathbf{x} = b$ is a polyhedral set because it consists of all the points in the half-spaces defined by the inequalities $\mathbf{w}^T\mathbf{x} \geq b$ and $\mathbf{w}^T\mathbf{x} \leq b$. One can use this fact to show the following theorem, whose proof is outside the scope of this course.

THEOREM 1.7. *Every polyhedral set is convex.*

2. Linear Separability and Binary Linear Classifiers

DEFINITION 1.8 (Linear Separability). Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ be a set of (n-dimensional) vectors (features) with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$. Then $(\mathbf{X}, \mathbf{y})$ is *linearly separable* if there is a constant $\mathbf{w} \in \mathbb{R}^n$ so that $y_i = 1$ if and only if $\mathbf{w}^T\mathbf{x}_i > 0$. That is, all vectors in Class 1 lie in $H_u(\mathbf{w})$.

EXAMPLE 1.9. An example of a separating hyperplane using the hyperplane $2x_1 + 3x_2 + x_3 = 5$ is shown in Figure 1.3. For example, if $x_1 = x_2 = x_3$ where given, then $2(1)+3(1)+1 = 6 > 5$. Thus, this would be a blue point, assuming this is a separating hyperplane for the data sample.

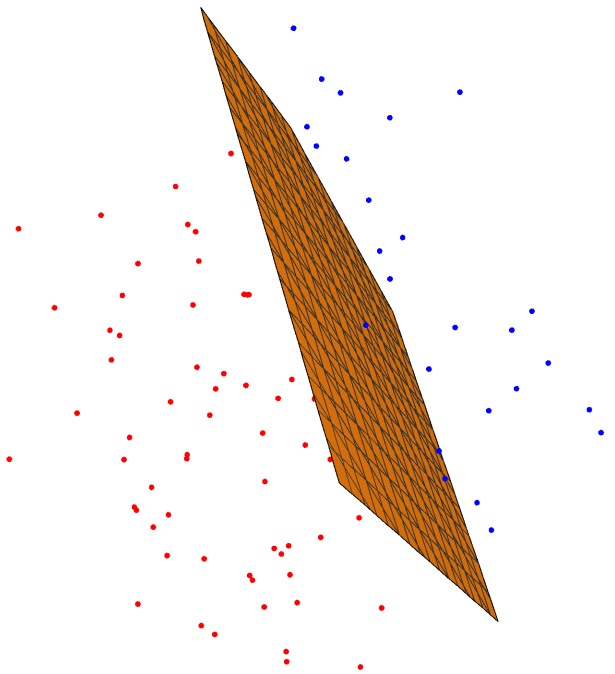


Figure 1.3. A separating hyperplane in $\mathbb{R}^3$. Notice all points above the hyperplane are colored blue (in Class 1). All points below are colored red (in Class 0).

DEFINITION 1.10 (Binary Linear Classifier). Given a set of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$, a *binary linear classifier* defined by the hyperplane $\mathbf{w}^T\mathbf{x} = 0$ is the function:

$$f(\mathbf{x}; \mathbf{w}) = \begin{cases} 1 & \text{if } \mathbf{w}^T\mathbf{x} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.6}$$

REMARK 1.11 (Affine Classifiers). It may be the case that the desired classifier is given by the hyperplane $\mathbf{w}^T\mathbf{x} = b$ for some $b \neq 0$. In this case, each vector $\mathbf{x}_i = \langle x_1, \ldots, x_n \rangle$ may be transformed into a new vector $\tilde{\mathbf{x}}_i = \langle 1, x_1, \ldots, x_n \rangle \in \mathbb{R}^{n+1}$. The new vector $\tilde{\mathbf{w}} = \langle -b | \mathbf{w} \rangle$ will separate the data since:

$$\tilde{\mathbf{w}}^T\tilde{\mathbf{x}} = 0 \iff -b + \mathbf{w}^T\mathbf{x} = 0 \iff \mathbf{w}^T\mathbf{x} = b$$

PROPOSITION 1.12. *Given a set of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$, there is a (correct) binary linear classifier for $\mathbf{X}$ if and only if $(\mathbf{X}, \mathbf{y})$ is linearly separable.*

EXERCISE 1.2. Prove the Proposition 1.12.

3. Separation as an Optimization Problem

REMARK 1.13. Finding a linear classifier (assuming one exists) can be phrased as an optimization problem with a straight forward set of constraints. Given a set of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$, suppose that:

$$C_0 = \{i \in \{1, \ldots, N\} : y_i = 0\}$$
$$C_1 = \{i \in \{1, \ldots, N\} : y_i = 1\}$$

then constraints for a linear classifier problem are:

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i \leq 0 & \forall i \in C_0 \\ \mathbf{w}^T\mathbf{x}_i \geq 0 & \forall i \in C_1 \end{cases} \tag{1.7}$$

In this problem, the *decision variables* are contained in the vector $\mathbf{w}$, while the parameters are $(\mathbf{X}, \mathbf{y})$. Notice, we relax the strict equality so that we can use max and min rather than inf and sup. Let W_{LC} be the feasible region defined by the inequalities in Expression 1.7. This feasible region is a polyhedral set.

PROPOSITION 1.14. *If W_{LC} is non-empty and consists of more than a single point, then there are an infinite number of vectors $\mathbf{w}$ satisfying the inequalities in Expression 1.7.*

PROOF. The result follows from the convexity of W_{LC}. □

REMARK 1.15. To pick a specific $\mathbf{w} \in W_{LC}$ requires the imposition of an objective function and (presumably) an algorithm for optimizing that objective function. The remainder of this chapter introduces two such objective functions.

4. Perceptron

REMARK 1.16. The *Perceptron* is (perhaps) the oldest machine learning algorithm (approach) and was originally developed by the Cornell Aeronautical Laboratory and funded by the Office of Naval Research. The algorithm is also the first *feed forward neural network*, which we will discuss in later chapters. At its core, the Perceptron is an algorithm for learning $\mathbf{w}$ to build a linear classifier given data and binary classifiers $(\mathbf{X}, \mathbf{y})$.

REMARK 1.17. For the Perceptron algorithm presented below, we assume that Remark 1.11 is used. We will use the binary linear classifier function from Definition 1.10, but in an algorithmic way. To facilitate this, define:

$$f_P(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.8}$$

Students who have taken differential equations may note this is a form of the *Heaviside step function*. Note further that:

$$f_P(\mathbf{w}^T\mathbf{x}) = f(\mathbf{x})$$

where f is given in Definition 1.10.

Perceptron
Input: A finite set of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$,
Input: A binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$ of the data.
Input: A *learning rate* $\alpha \ll 1$.
Input: A maximum iteration number $K \geq 1$.
Input: A convergence threshold $\epsilon \geq 0$.

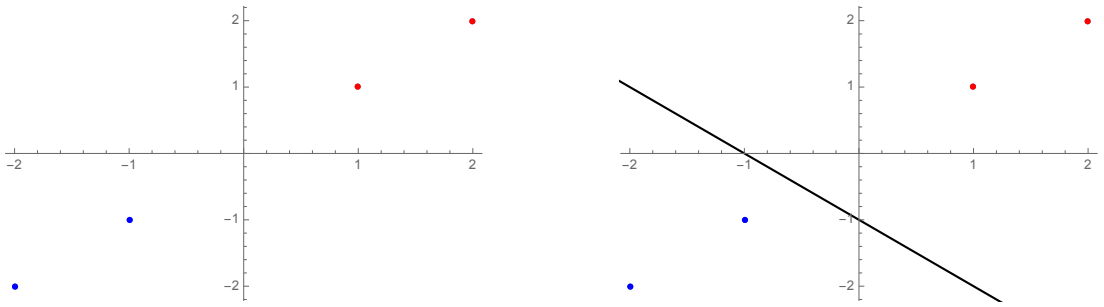
(1) $k := 0$
(2) $\mathbf{w}_k := \mathbf{0}$ {Random initialization is also acceptable.}
(3) **do**
(4) **for each** $j \in \{1, \ldots, N\}$ **do**
(5) $z := f_P(\mathbf{w}_{k+j-1}^T \mathbf{x}_j)$
(6) $\mathbf{w}_{k+j} := \mathbf{w}_{k+j-1} + \alpha\,(y_j - z)\,\mathbf{x}_j$
(7) **end for**
(8) $k := k + N$
(9) **while** $k/N \leq K$ and $|\mathbf{w}_k - \mathbf{w}_{k-N}| > \epsilon$

Algorithm 1. Perceptron algorithm for computing $\mathbf{w}$.

EXAMPLE 1.18. Consider the exceptionally simple data set of row-vectors written in matrix form:

$$\mathbf{X} = \begin{bmatrix} -2 & -2 \\ -1 & -1 \\ 1 & 1 \\ 2 & 2 \end{bmatrix}$$

and suppose that $\mathbf{y} = \langle 0, 0, 1, 1 \rangle$. The data is illustrated in Figure 4(a). Red data points corresponding $y_i = 1$. We modify $\mathbf{X}$ so that each data element begins with a 1, thus



(a) Simple Data (b) Perceptron Output

Figure 1.4. An example of the operation of the perceptron algorithm on a simple data set.

obtaining:

$$\tilde{\mathbf{X}} = \begin{bmatrix} 1 & -2 & -2 \\ 1 & -1 & -1 \\ 1 & 1 & 1 \\ 1 & 2 & 2 \end{bmatrix}$$

Beginning with $\mathbf{w}_1 = \langle 0, 0, 0 \rangle$, we compute:

$$\mathbf{w}^T \mathbf{x}_1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ -2 \end{bmatrix} = 0$$

and:

$$z = f_P(0) = 0$$

Since $y_1 = 0$, we know that $\alpha(y_1 - z) = 0$ and thus $\mathbf{w}_1 = \mathbf{w}_0$. The same is true for the next point $\mathbf{x}_2 = \langle 1, -1, -1 \rangle$. For the third point however, now that:

$$\mathbf{w}^T \mathbf{x}_3 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = 0,$$

so $z = f_P(0) = 0$ but, $y_3 = 1$. Thus:

$$\mathbf{w}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \alpha(1 - 0) \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \alpha \\ \alpha \end{bmatrix}$$

On $\mathbf{x}_4$, we compute:

$$\mathbf{w}^T \mathbf{x}_3 = \begin{bmatrix} \alpha & \alpha & \alpha \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} = 5\alpha,$$

so $z = f_P(5\alpha) = 1$ and $y_4 = 1$. Thus $\alpha(y_1 - z) = 0$ and $\mathbf{w}_4 = \mathbf{w}_3$. This completes the inner for-loop (Lines 4 - 7). We now test whether At this point $k = 4$ and we can use this to determine if we've completed enough iterations. Now, that $|\mathbf{w}_4 - \mathbf{w}_0|^2 = 3\alpha^2$. If this value

is less than ϵ, stop. Otherwise, the outer do-loop (Lines 3 - 9) are executed again. On any subsequent execution, $\mathbf{w}_k$ will *not* change because the $\mathbf{w} = \langle \alpha, \alpha, \alpha \rangle$ produces a separating hyperplane for the data (we assumed $\alpha \ll 1$). Thus, the perceptron algorithm converges in this case. The resulting separating hyperplane when $\alpha = 0.05$ is shown in Figure 4(b).

5. Derivation of the Perceptron Algorithm

DEFINITION 1.19 (Sigmoid Function). The *sigmoid function or logistic function* (with parameter β) is the function:

$$S(x;\beta) = \frac{1}{1+e^{-\beta x}} \tag{1.9}$$

A plot when $\beta = 1$ is shown in Figure 1.5.

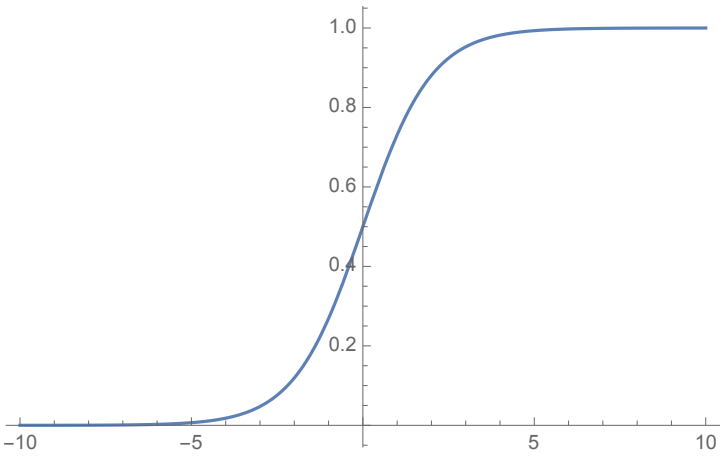


Figure 1.5. The sigmoid or logistic function is an "S"-shaped function that is a *relaxation* of the step-function.

DEFINITION 1.20 (Step Function with Half-Maximum Convention). The *Heaviside step function with half-maximum convention* is the function:

$$H(x) = \begin{cases} 0 & \text{if } x < 0 \\ \frac{1}{2} & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases} \tag{1.10}$$

REMARK 1.21. The next theorem relies on results on function sequence convergence from real and functional analysis, which are outside the scope of this course. However, we provide a reasonable justification.

THEOREM 1.22. *For all ϵ there exists an β so that:*

$$\int_{-\infty}^{\infty} |S(x;\beta) - H(x)|^2 \, dx < \epsilon \tag{1.11}$$

Thus, $\lim_{\beta\to\infty} S(x;\beta) = H(x)$.

REMARK 1.23. Given two functions $f, g : \mathbb{R} \to \mathbb{R}$, the $\mathcal{L}^2$-norm is the expression:

$$\left(\int_{-\infty}^{\infty} |f(x) - g(x)|^2 dx\right)^{\frac{1}{2}} \tag{1.12}$$

The integral is usually taken to be a *Lebesgue integral* but that is *way* outside of the scope of these notes. It suffices to say that the integral is a generalization of the Riemann integral that one learns about in Calculus. Generally speaking, we might think of a sequence of *functions*: $f_1, f_2, \ldots, f_k, \ldots$ and talk about this sequence of functions *converging to another function* g in the $\mathcal{L}^2$-norm. This is exactly what we mean when we write: $\lim_{\beta\to\infty} S(x;\beta) = H(x)$.

SKETCH OF PROOF. Ignore the jump discontinuity in $H(x)$. We may compute:

$$\int_{-\infty}^{\infty} |S(x;\beta) - H(x)|^2 dx = \int_{-\infty}^{0} |S(x;\beta)|^2 dx + \int_{0}^{\infty} |S(x;\beta) - 1|^2 dx$$

and show this converges. Consider:

$$\int_{-\infty}^{0} |S(x;\beta)|^2 dx = \int_{-\infty}^{0} \frac{1}{(1+e^{-\beta x})^2} dx$$

Let $u = e^{-\beta x}$, then $du = -\beta e^{-\beta x} dx = -\beta u dx$. Substitution yields:

$$\int_{-\infty}^{0} \frac{1}{(1+e^{-\beta x})^2} dx = \int_{\infty}^{1} \frac{-1}{\beta u(1+u)^2} du = \int_{1}^{\infty} \frac{1}{\beta u(1+u)^2} du$$

Using partial fraction expansion we can write:

$$\frac{1}{\beta u(1+u)^2} = \frac{1}{\beta}\left(\frac{1}{u} - \frac{2}{(1+u)^2} - \frac{u}{(1+u)^2}\right)$$

Each piece can then be integrated in-turn:

$$\int \frac{1}{u} du = \log(u)$$
$$\int \frac{-2}{(1+u)^2} = \frac{2}{1+u}$$
$$\int \frac{-u}{(1+u)^2} du = -\frac{1}{1+u} - \log(1+u)$$

Here the first integral is fundamental; the second integral is a simply polynomial integration and the third integral is accomplished by a second substitution (let $v = (1+u)^2$). Combining we obtain:

$$\int_{1}^{\infty} \frac{1}{\beta u(1+u)^2} du = \frac{1}{\beta}\left(\frac{2}{1+u} - \frac{1}{1+u} + \log(u) - \log(1+u)\right)\Bigg|_{1}^{\infty} =$$
$$\frac{1}{\beta}\left(\frac{1}{1+u} + \log\left(\frac{u}{1+u}\right)\right)\Bigg|_{1}^{\infty} = \frac{1}{\beta}\left(-\frac{1}{2} - \log\left(\frac{1}{2}\right)\right) = \frac{1}{\beta}\left(-\frac{1}{2} + \log(2)\right)$$

By symmetry:

$$\int_{0}^{\infty} |S(x;\beta) - 1|^2 \, dx = \frac{1}{\beta}\left(-\frac{1}{2} + \log(2)\right)$$

as well. Thus:

$$\int_{-\infty}^{\infty} |S(x;\beta) - H(x)|^2 dx = \frac{2}{\beta}\left(-\frac{1}{2} + \log(2)\right)$$

Thus we can see that:

$$\lim_{\beta\to\infty} \int_{-\infty}^{\infty} |S(x;\beta) - H(x)|^2 dx = \lim_{\beta\to\infty} \frac{2}{\beta}\left(-\frac{1}{2} + \log(2)\right) = 0$$

Consequently, for all ϵ there exists an β satisfying Inequality 1.11 and we have shown that $S(x;\beta)$ converges to $H(x)$ in the $\mathcal{L}^2$ norm. This completes the proof. □

EXERCISE 1.3. Fill in the missing steps in the proof by showing the three integral components are computed correctly. Also, compute the positive half of the integral directly without using a symmetry argument.

REMARK 1.24 (Pricing Out a Constraint). In optimization, a constraint can be moved to the objective function by supplying a function that is large (assuming a minimization problem) when the constraint is not satisfied. This process is called *pricing out the constraint*.

Consider the linear classifier constraints (Inequalities 1.7). Another way to write these constraints is:

$$\begin{cases} H(\mathbf{w}^T\mathbf{x}_i) = 0 & \forall i \in C_0 \\ H(\mathbf{w}^T\mathbf{x}_i) = 1 & \forall i \in C_1 \end{cases} \tag{1.13}$$

Where H is the step-function with half-maximum convention. Given an equality constraint: $h(x) = r$, the standard way to *price out the constraint*[1] by using the function $(h(x) - r)^2$. Thus we can price out the linear classifier constraints as:

$$J_{LC}(\mathbf{w};\mathbf{x}) = \sum_{i=1}^{N} \frac{1}{2}\left(y_i - H(\mathbf{w}^T\mathbf{x}_i)\right)^2 \tag{1.14}$$

In a perfect world, the data separate so that none of them lie on the hyperplane $\mathbf{w}^T\mathbf{x} = 0$. Using the step-function with half-maximum convention added benefit that it penalizes data lying on the hyperplane. We can now re-write the linear classifier problem as the unconstrained optimization problem:

$$\begin{cases} \min \;\; J_{LC}(\mathbf{w};\mathbf{x}) = \sum_{i=1}^{N} \frac{1}{2}\left(y_i - H(\mathbf{w}^T\mathbf{x}_i)\right)^2 \\ s.t. \;\; \mathbf{w} \in \mathbb{R}^n \end{cases} \tag{1.15}$$

Notice the vector of parameters $\mathbf{w}$ *contain our decision variables.* This can present a dangerous mental bias because in most optimization problems $\mathbf{x}$ is the decision vector; here it plays the role of parameters.

REMARK 1.25. A major problem with using J_{LC} in the objective is that it is non-differentiable. Certainly, one could use a derivative-free optimization method (see [**Gri12b**],

[1]Recall, pricing out a constraint means turning it into a part of the objective function by using a penalty function method.

Chapter 8). However, this gives us *no* insight into the Perceptron. An alternate approach is to study the *relaxed problem*:

$$\begin{cases} \min \;\; J_{LC}(\mathbf{w};\mathbf{x},\beta) = \displaystyle\sum_{j=1}^{N} \frac{1}{2}\left(y_j - S(\mathbf{w}^T\mathbf{x}_j;\beta)\right)^2 \\ s.t. \;\; \mathbf{w} \in \mathbb{R}^n \end{cases} \tag{1.16}$$

The resulting objective function is differentiable and can be analyzed in the limiting case when $\beta \to \infty$ to obtain results on the Perceptron. Before proceeding, we remark that:

$$S'(x;\beta) = \frac{\beta e^{-\beta x}}{\left(1+e^{-\beta x}\right)^2} = \beta S(x;\beta)\left(1 - S(x;\beta)\right) \tag{1.17}$$

THEOREM 1.26. *Let ϵ be a fixed step-size in Gradient descent. Then the gradient descent update for the unconstrained optimization problem 1.16 is:*

$$\mathbf{w}_{k+1} := \mathbf{w}_k + \epsilon \cdot \sum_{j=1}^{N} \left[y_j - S\left(\mathbf{w}_k^T\mathbf{x}_j;\beta\right)\right] \cdot S'\left(\mathbf{w}_k^T\mathbf{x}_j;\beta\right) \cdot \mathbf{x}_j \tag{1.18}$$

PROOF. Note $\mathbf{w}^T\mathbf{x}_j = w_1 x_{j_1} + w_2 x_{j_2} + \cdots + w_n x_{j_n}$. Thus:

$$\frac{\partial \mathbf{w}^T\mathbf{x}_j}{\partial w_i} = x_{j_i}$$

For simplicity, let:

$$J_{LC}^j = \frac{1}{2}\left(y_j - S(\mathbf{w}^T\mathbf{x}_j;\beta)\right)^2$$

Then by the chain rule:

$$\frac{\partial J_{LC}^j}{\partial w_i} = \frac{\partial J_{LC}^j}{\partial S} \cdot \left(-\frac{\partial S}{\partial \mathbf{w}^T\mathbf{x}_j}\right) \cdot \frac{\partial \mathbf{w}^T\mathbf{x}_j}{\partial w_i}$$

We have:

$$\frac{\partial J_{LC}^j}{\partial S} = \left(y_j - S(\mathbf{w}^T\mathbf{x}_j;\beta)\right)$$

$$\frac{\partial S}{\partial \mathbf{w}^T\mathbf{x}_j} = S'(\mathbf{w}^T\mathbf{x}_j;\beta) = \frac{\beta e^{-\beta \mathbf{w}^T\mathbf{x}_j}}{1+e^{-\beta \mathbf{w}^T\mathbf{x}_j}}$$

Thus the gradient of J_{LC}^j with respect to $\mathbf{w}$ is a vector of n elements with i^{th} element:

$$-\left(y_j - S(\mathbf{w}^T\mathbf{x}_j;\beta)\right) S'(\mathbf{w}^T\mathbf{x}_j;\beta) x_{j_i}$$

Summing over the J_{LC}^j we obtain:

$$\nabla_{\mathbf{w}} J_{LC}(\mathbf{w};\mathbf{x},\beta) = -\sum_{j=1}^{N} \left(y_j - S(\mathbf{w}^T\mathbf{x}_j;\beta)\right)\left(\frac{\beta e^{-\beta \mathbf{w}^T\mathbf{x}_j}}{\left(1+e^{-\beta \mathbf{w}^T\mathbf{x}_j}\right)^2}\right)\mathbf{x}_j \tag{1.19}$$

Assuming a step-length of ϵ we replace $\mathbf{w}$ with:

$$\mathbf{w} \leftarrow \mathbf{w} - \epsilon \cdot \nabla_{\mathbf{w}} J_{LC}(\mathbf{w};\mathbf{x},\beta)$$

in gradient descent. The gradient descent update rule follows immediately. □

REMARK 1.27. Notice that:

$$\lim_{\beta\to\infty} S'(x;\beta) = \delta(x) = \begin{cases} +\infty & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

Thus, except when $\mathbf{w}^T\mathbf{x} = 0$, $S'(\mathbf{w}^T\mathbf{x};\beta)$ approaches 0 as $S(\mathbf{w}^T\mathbf{x};\beta)$ approaches $H(\mathbf{w}^T\mathbf{x})$. This means, that gradient descent *cannot function properly* using the step function. However, if we approximate $\epsilon \cdot \delta(x)$ with α, the learning rate and α is *small*, then the *approximate gradient descent* is:

$$\mathbf{w}_{k+1} := \mathbf{w}_k + \sum_{j=1}^{N} \alpha \cdot \left[y_j - H\left(\mathbf{w}_k^T\mathbf{x}_j;\beta\right)\right] \cdot \mathbf{x}_j \tag{1.20}$$

This is the *batch update Perceptron rule.*

REMARK 1.28 (Streaming Data). If data streams in a piece at a time and we wish to update $\mathbf{w}$ each time, then at j (for each piece of data) we can update $\mathbf{w}$ by taking a gradient descent step using $\nabla_{\mathbf{w}} J_{LC}^j$. This removes the summation from the update step and yields:

$$\mathbf{w}_{k+1} := \mathbf{w}_k + \alpha \cdot \left[y_j - H\left(\mathbf{w}_k^T\mathbf{x}_j;\beta\right)\right] \cdot \mathbf{x}_j, \tag{1.21}$$

which is exactly the Perceptron update rule, when the step function f_P is replaced by the Heaviside step function with half-maximum convention.

6. Convergence of the Perceptron Algorithm

LEMMA 1.29. *Suppose* $\mathbf{x}, \mathbf{y} \in \mathbf{R}^n$, *then:*

$$\|\mathbf{x}+\mathbf{y}\|^2 = \|\mathbf{x}\|^2 + 2\mathbf{x}^T\mathbf{y} + \|\mathbf{y}\|^2$$

□

LEMMA 1.30. *Let* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ *and let* θ *be the angle between* $\mathbf{x}$ *and* $\mathbf{y}$, *then*

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T\mathbf{y} = \|\mathbf{x}\| \, \|\mathbf{y}\| \cos\theta \tag{1.22}$$

□

REMARK 1.31. The preceding lemma can be proved using the *law of cosines* from trigonometry. The following small lemma follows and is is proved as Theorem 1 of [**MT03**]:

EXERCISE 1.4. Prove Lemma 1.29. [Hint: Use the fact that $\|\mathbf{z}\|^2 = \left|\mathbf{z}^T\mathbf{z}\right|$. Set $\mathbf{z} = \mathbf{x}+\mathbf{y}$. Matrix (vector) multiplication distributes.]

THEOREM 1.32. *Let* $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ *be a set of (n-dimensional) vectors (features) with an associated binary classification* $y_i \in \{0,1\}$ $(i = 1,\ldots,N)$. *We make the following assumptions:*

1. *The vectors are bounded in norm so that* $\|\mathbf{x}_j\| \le R$ *for all* $j = 1,\ldots,N$ *where* R *is a real number with* $R > 0$.
2. *The data* $(\mathbf{X},\mathbf{y})$ *are linearly separable.*
3. *There is some* $\tilde{\mathbf{w}}$ *so that* $\tilde{\mathbf{w}}^T\mathbf{x}_j > \gamma$ *for all* $\mathbf{x}_j \in C_1$ *and* $\tilde{\mathbf{w}}^T\mathbf{x}_j < -\gamma$ *for all* $\mathbf{x}_j \in C_0$

Then the Perceptron algorithm converges to a $\mathbf{w}$ *that generates a separating hyperplane* $\mathbf{w}^T\mathbf{x} = 0$.

REMARK 1.33. We will execute the proof using two lemmas. In the first, we show that the dot product between the current guess for $\mathbf{w}_{k+j}$ (in Algorithm 1) and $\tilde{\mathbf{w}}$ is increasing. We then show that the size of $\mathbf{w}_{k+j}$ is bounded. We conclude that it is the angle between $\tilde{\mathbf{w}}$ and $\mathbf{w}_{k+j}$ that is decreasing and thus making this dot product grow. Thus, $\mathbf{w}_{k+j}$ becomes parallel to $\tilde{\mathbf{w}}$

LEMMA 1.34. *Given the assumptions of Theorem 1.32,* $\tilde{\mathbf{w}}^T\mathbf{w}_k$ *monotonically increases on each iteration of the algorithm.*

PROOF. For simplicity, assume we have a single counter k so that:

$$\mathbf{w}_k := \mathbf{w}_{k-1} + \alpha\left(y_k - z\right)\mathbf{x}_k$$

rather than doubly indexing the counter as we did in Algorithm 1. Here, as before:

$$z := f_P(\mathbf{w}_{k-1}^T\mathbf{x}_j)$$

Then:

$$\tilde{\mathbf{w}}^T\mathbf{w}_k = \tilde{\mathbf{w}}^T\mathbf{w}_{k-1} + \alpha\left(y_k - z\right)\tilde{\mathbf{w}}^T\mathbf{x}_k$$

If $y_k \neq z$, then:

(1) If $y_k = 1$ and $z = 0$, then:

$$\tilde{\mathbf{w}}^T\mathbf{w}_k - \tilde{\mathbf{w}}^T\mathbf{w}_{k-1} = \alpha\tilde{\mathbf{w}}^T\mathbf{x}_k > \alpha\gamma$$

(2) If $y_k = 0$ and $z = 1$, then:

$$\tilde{\mathbf{w}}^T\mathbf{w}_k - \tilde{\mathbf{w}}^T\mathbf{w}_{k-1} = \alpha(-1)\tilde{\mathbf{w}}^T\mathbf{x}_k < -\alpha\gamma$$

Dividing the second inequality through by -1 implies (again):

$$\tilde{\mathbf{w}}^T\mathbf{w}_k - \tilde{\mathbf{w}}^T\mathbf{w}_{k-1} > \alpha\gamma$$

If $\mathbf{w}_0 = \mathbf{0}$ (as we assumed in Algorithm 1), then after k iterations, we have:

$$\tilde{\mathbf{w}}^T\mathbf{w}_k > k\alpha\gamma$$

Consequently, the dot product of $\mathbf{w}_k$ and $\tilde{\mathbf{w}}$ monotonically increases on each iteration. □

LEMMA 1.35. *Given the assumptions of Theorem 1.32,* $||\mathbf{w}_k||^2$ *is bounded.*

PROOF. Consider the square-norm of $\mathbf{w}_k$:

$$\left\|\mathbf{w}_k\right\|^2 = \left\|\mathbf{w}_{k-1} + \alpha\left(y_k - z\right)\mathbf{x}_k\right\|^2 = \left\|\mathbf{w}_{k-1}\right\|^2 + 2\alpha\left(y_k - z\right)\mathbf{w}_{k-1}^T\mathbf{x}_k + \alpha^2(y_k - z)^2\left\|\mathbf{x}_k\right\|^2$$

by Lemma 1.29. If $y_k \neq z$, then:

(1) If $y_k = 1$ and $z = 0$, then $(y_k - z) > 0$ and $\mathbf{w}_{k-1}^T\mathbf{x}_k < 0$ and consequently:

$$\left\|\mathbf{w}_{k-1}\right\|^2 + 2\alpha\left(y_k - z\right)\mathbf{w}_{k-1}^T\alpha^2(y_k - z)^2\mathbf{x}_k + \left\|\mathbf{x}_k\right\|^2 < \left\|\mathbf{w}_{k-1}\right\|^2 + \alpha^2(y_k - z)^2\left\|\mathbf{x}_k\right\|^2.$$

(2) If $y_k = 0$ and $z = 1$, then $(y_k - z) < 0$ and $\mathbf{w}_{k-1}^T\mathbf{x}_k > 0$ and the previous inequality still holds.

We assumed $\|\mathbf{x}_k\|^2 < R^2$ and thus:

$$\|\mathbf{w}_k\|^2 < \|\mathbf{w}_{k-1}\|^2 + \alpha^2 R^2$$

because $(y_k - z)^2 \leq 1$. If $\mathbf{w}_0 = \mathbf{0}$, then after k iterations we have:

$$\|\mathbf{w}_k\|^2 < k(\alpha R)^2$$

□

Proof of Theorem 1.32. From Lemma 1.30 we have:

$$\cos\theta_{\mathbf{w}_k,\tilde{\mathbf{w}}} = \frac{\tilde{\mathbf{w}}^T \mathbf{w}_k}{\|\tilde{\mathbf{w}}\| \, \|\mathbf{w}_k\|},$$

where $\theta_{\mathbf{w}_k,\tilde{\mathbf{w}}}$ is the angle between $\mathbf{w}_k$ and $\tilde{\mathbf{w}}$. From the proof of Lemma 1.34, we have:

$$\frac{\tilde{\mathbf{w}}^T \mathbf{w}_k}{\|\tilde{\mathbf{w}}\| \, \|\mathbf{w}_k\|} \geq \frac{k\alpha\gamma}{\|\tilde{\mathbf{w}}\| \, \|\mathbf{w}_k\|}$$

From the proof of Lemma 1.35, we have $\|\mathbf{w}_k\| < \alpha R\sqrt{k}$ and thus:

$$\frac{k\alpha\gamma}{\|\tilde{\mathbf{w}}\| \, \|\mathbf{w}_k\|} > \frac{k\alpha\gamma}{\alpha R\sqrt{k} \, \|\tilde{\mathbf{w}}\|}$$

We conclude that:

$$\frac{k\alpha\gamma}{\alpha R\sqrt{k} \, \|\tilde{\mathbf{w}}\|} < \cos\theta_{\mathbf{w}_k,\tilde{\mathbf{w}}}$$

Thus cosine function is bounded above by 1 and will achieve this value when $\mathbf{w}_k$ is parallel to $\tilde{\mathbf{w}}$; i.e., $\theta_{\mathbf{w}_k,\tilde{\mathbf{w}}} = 0$. The left-hand-side is increasing and thus the angle $\theta_{\mathbf{w}_k,\tilde{\mathbf{w}}}$ must decrease on each iteration. Furthermore, the bound on cosine implies that:

$$\frac{k\alpha\gamma}{\alpha R\sqrt{k} \, \|\tilde{\mathbf{w}}\|} \leq 1 \implies k \leq \left(\frac{R \, \|\tilde{\mathbf{w}}\|}{\gamma}\right)^2$$

Thus, the algorithm must converge after a finite number of iterations. This completes the proof. □

Remark 1.36. Another way to prove convergence under the given assumptions is to find a $\beta \gg 0$ so that $S(x;\beta)$ is a good approximation of $H(x)$ assuming that $|\tilde{\mathbf{w}}^T \mathbf{x}| > \gamma$; i.e., the margin of separation is greater than γ. In this case, convergence is ensured by the convergence of gradient descent to a fixed point. This proof is more complex, but yields more insight into the fact that elementary machine learning is simply an application of optimization.

CHAPTER 2

Support Vector Machines and Advanced Linear Classifiers

REMARK 2.1. In this chapter we discuss a special kind of linear classifiers called *Support Vector Machines* (SVM). SVM solve many of the problems we observed with the Perceptron, in particular the fact that Perceptron algorithm could return any separating hyperplane, rather than a specific one.

1. Some Preliminaries

LEMMA 2.2. *Let:*

$$H^+ = \left\{\mathbf{x} \in \mathbb{R}^n : \mathbf{w}^T\mathbf{x} - b = k\right\}$$
$$H^- = \left\{\mathbf{x} \in \mathbb{R}^n : \mathbf{w}^T\mathbf{x} - b = -k\right\}$$

for some vector $\mathbf{w} \in \mathbb{R}^n$*, and scalars* $b, k \in \mathbb{R}$*,* $k > 0$*. Then the Euclidean distance from* H^+ *to* H^- *is* $2k/\left\|\mathbf{w}\right\|$*.*

PROOF. From Exercise A.2 (see Appendix A) the vector $\mathbf{w}/\left\|\mathbf{w}\right\|$ is a unit vector normal to both H^+ and H^-. If $k \neq 0$, clearly these hyperplanes are parallel and therefore if $\mathbf{x}_0 \in H^+$, then the line $\mathbf{x_0} + \alpha\mathbf{w}/\left\|\mathbf{w}\right\|$ will intersect H^- for some value of α and the distance between H^+ and H^- will be exactly $|\alpha|$ (because $\mathbf{w}/\left\|\mathbf{w}\right\|$ is a unit vector). We can compute α. The fact that $\mathbf{x_0} \in H^+$ implies that:

$$\mathbf{w}^T\mathbf{x}_0 - b = k \implies \mathbf{w}^T\mathbf{x}_0 = b + k$$

If $\mathbf{x_0} + \alpha\mathbf{w}/\left\|\mathbf{w}\right\| \in H^-$, then:

$$\mathbf{w}^T\left(\mathbf{x_0} + \alpha\mathbf{w}/\left\|\mathbf{w}\right\|\right) - b = -k \implies$$
$$\mathbf{w}^T\mathbf{x}_0 + \alpha\frac{\left\|\mathbf{w}\right\|^2}{\left\|\mathbf{w}\right\|} - b + k = 0 \implies$$
$$b + k + \alpha\left\|w\right\| - b + k = 0 \implies$$
$$\alpha = \frac{-2k}{\left\|\mathbf{w}\right\|}$$

Thus, computing:

$$\left\|\alpha\frac{\mathbf{w}}{\left\|\mathbf{w}\right\|}\right\| = |\alpha|\left\|\frac{\mathbf{w}}{\left\|\mathbf{w}\right\|}\right\| = |\alpha| = \frac{2k}{\left\|\mathbf{w}\right\|}$$

This is the distance from H^+ to H^-. This completes the proof. □

REMARK 2.3. This fact is used in most SVM documentation for when $k = 1$.

PROPOSITION 2.4. *Suppose that* $\mathbf{x}_0, \mathbf{w} \in \mathbb{R}^n$ *and* $b \in \mathbb{R}$. *The distance from* $\mathbf{x}_0$ *to the hyperplane defined by the equation* $\mathbf{w}^T\mathbf{x}_0 - b = 0$ *is:*

$$\frac{\left|b - \mathbf{w}^T\mathbf{x}_0\right|}{\|\mathbf{w}\|} \tag{2.1}$$

EXERCISE 2.1. Prove Proposition 2.4. [Hint: Find $|\alpha|$ so that $\mathbf{x}_0 + \alpha\mathbf{w}/\|w\|$ lies on the hyperplane, just as we did in the proof of Lemma 2.2.]

LEMMA 2.5. *Let* $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ *be a set of (n-dimensional) vectors (features) with an associated binary classification* $y_i \in \{0, 1\}$ *(*$i = 1, \ldots, N$*). Suppose* $(\mathbf{X}, \mathbf{y})$ *is linearly separable. Then there is a vector* $\mathbf{w} \in \mathbb{R}^n$ *and a scalar* $b \in \mathbb{R}$ *such that* $\mathbf{w}^T\mathbf{x}_i - b > 0$ *if* $y_i = 1$ *and* $\mathbf{w}^T\mathbf{x}_i - b < 0$ *if* $y_i = 0$.

PROOF. The fact that there is a vector $\mathbf{w} \in \mathbb{R}^n$ so that $\mathbf{w}^T\mathbf{x}_i - b > 0$ if $y_i = 1$ is clear from the definition of linear separability (Definition 1.8). It follows at least that $\mathbf{w}^T\mathbf{x}_i - b \leq 0$ for all i such that $y_i = 0$.

If the inequality is strict for $\mathbf{X}$, then we are done. Suppose there is some i with $y_i = 0$ so that $\mathbf{w}^T\mathbf{x}_i - b = 0$. Let H be the hyperplane defined by the equation $\mathbf{w}^T\mathbf{x}_i - b = 0$ and let $\mathbf{x}_j$ be the data point with $y_j = 1$ so that the distance from $\mathbf{x}_j$ to H is as small as possible. In particular, this distance is:

$$\epsilon = \frac{\left|\mathbf{w}^T\mathbf{x}_j - b\right|}{\|\mathbf{w}\|}$$

by Proposition 2.4. (Note we have reversed the sign in the numerator's absolute value for simplicity.) The fact that $\mathbf{w}^T\mathbf{x}_k > b$ for all k so that $\mathbf{y}_k = 1$ implies that $\mathbf{x}_j$ is simply the data point that makes $\mathbf{w}^T\mathbf{x}_j$ as close to b as possible. That is, $\mathbf{w}^T\mathbf{x}_k \geq \mathbf{w}^T\mathbf{x}_j > b$ for all k so that $y_k = 1$.

Define the new hyperplane:

$$H' = \left\{\mathbf{z} \in \mathbb{R}^n : \mathbf{z} = \mathbf{x} + \frac{1}{2}\frac{\mathbf{w}^T\mathbf{x}_j - b}{\|\mathbf{w}\|}\frac{\mathbf{w}}{\|\mathbf{w}\|},\ \mathbf{x} \in H\right\} \tag{2.2}$$

This new hyperplane moves each point in H toward $\mathbf{x}_j$ (see Exercise 2.2). This new hyperplane is defined by the equation:

$$\mathbf{w}^T\mathbf{z} - \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right) = 0 \tag{2.3}$$

To see this, let $\mathbf{z} \in H'$ and note that:

$$\mathbf{w}^T\mathbf{z} = \mathbf{w}^T\left(\mathbf{x} + \frac{1}{2}\frac{\mathbf{w}^T\mathbf{x}_j - b}{\|w\|}\frac{\mathbf{w}}{\|\mathbf{w}\|}\right)$$

for some $\mathbf{x} \in H$. Expanding we obtain:

$$\mathbf{w}^T\mathbf{z} = \mathbf{w}^T\mathbf{x} + \frac{1}{2}\frac{\mathbf{w}^T\mathbf{x}_j - b}{\|\mathbf{w}\|}\frac{\|\mathbf{w}\|^2}{\|\mathbf{w}\|} = b + \frac{1}{2}\mathbf{w}^T\mathbf{x}_j - \frac{1}{2}b = \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right)$$

This establishes Equation 2.3. We can now show that this is a separating hyperplane with the desired properties. Consider $\mathbf{x}_k$ with $y_k = 1$. Then:

$$\mathbf{w}^T\mathbf{x}_k - \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right) > \mathbf{w}^T\mathbf{x}_j - \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right) = \frac{1}{2}\left(\mathbf{w}^T\mathbf{x}_j - b\right) > 0$$

On the other hand, consider any $\mathbf{x}_k$ so that $y_k = 0$. Then:

$$\mathbf{w}^T\mathbf{x}_k - \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right) \leq b - \left(\frac{\mathbf{w}^T\mathbf{x}_j + b}{2}\right) < b - \frac{2b}{2} = 0$$

because $\mathbf{w}^T\mathbf{x}_j > b$ by assumption. This completes the proof. □

EXERCISE 2.2. Let $\mathbf{w}, \mathbf{x}_0 \in \mathbb{R}^n$ be vectors and suppose $b \in \mathbb{R}$ is a scalar. Let H be the hyperplane defined by the equation $\mathbf{w}^T\mathbf{x} - b = 0$. Show that moving each point in H by adding the (vector) quantity:

$$\frac{\mathbf{w}^T\mathbf{x}_0 - b}{\|w\|}\frac{\mathbf{w}}{\|w\|}$$

yields a new hyperplane H' containing $\mathbf{x}_0$.

2. Hard Margin Support Vector Machines

REMARK 2.6. As in Remark 1.13, we will assume that we are given a set of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$, and that:

$$C_0 = \{i \in \{1, \ldots, N\} : y_i = 0\}$$
$$C_1 = \{i \in \{1, \ldots, N\} : y_i = 1\}$$

In Chapter 1, we absorbed the constant b from the expression $\mathbf{w}^T\mathbf{x} - b = 0$ into the coefficient vector because it too is an unknown quantity. In this chapter, we will *not* do this, but instead will re-write b as w_0, to emphasize that it too is an unknown quantity.

REMARK 2.7. It is worth noting that mosts texts on the subject explicitly replace the Class 0 by the Class -1. We will not do that to maintain consistency with the previous chapter. However, we will note when -1 is used in place of 0 explicitly.

DEFINITION 2.8 (Margin Distance). Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ be a set of (n-dimensional) vectors. Let H be the hyperplane defined by the equation $\mathbf{w}^T\mathbf{x} - w_0 = 0$. The margin distance from H to $\mathbf{X}$ is:

$$d(\mathbf{X}, H) = \min_{i \in \{1,\ldots,N\}} \min_{\mathbf{x} \in H} \|\mathbf{x} - \mathbf{x}_i\| \tag{2.4}$$

That is, it is the smallest distance between any point in $\mathbf{X}$ and a point on the hyperplane H.

DEFINITION 2.9 (Maximum-Margin Separating Hyperplane). Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ be a set of (n-dimensional) vectors (features) with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$ and suppose the data $(\mathbf{x}, \mathbf{y})$ are linearly separable. Let:

$$\mathbf{X}_0 = \{\mathbf{x}_i\}_{i \in C_0}$$
$$\mathbf{X}_1 = \{\mathbf{x}_i\}_{i \in C_1}$$

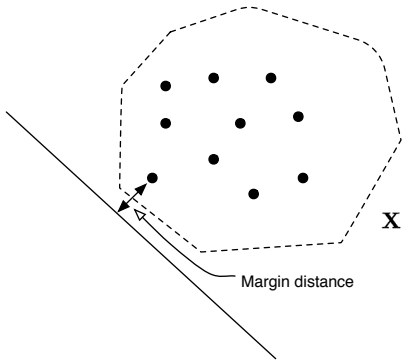


Figure 2.1. An illustration of the margin distance. The "illustrated distance line" from the point to the hyperplane (line) is perpendicular to the hyperplane.

The *maximum-margin separating hyperplane* is any hyperplane H defined by equation $\mathbf{w}^T\mathbf{x} - w_0 = 0$ such that $d(\mathbf{X}_0, H) = d(\mathbf{X}_1, H)$ and this distance is *maximized*.

THEOREM 2.10. *Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ be a set of (n-dimensional) vectors (features) with an associated binary classification $y_i \in \{0, 1\}$ $(i = 1, \ldots, N)$ and suppose the data $(\mathbf{x}, \mathbf{y})$ are linearly separable. If $(w_0^*, \mathbf{w}^*)$ solves the problem:*

$$\text{Hard} - \text{SVM} \begin{cases} \min \ \frac{1}{2}\|\mathbf{w}\|^2 \\ s.t. \ \mathbf{w}^T\mathbf{x}_i - w_0 \geq 1 \quad \forall i \in C_1 \\ \qquad \mathbf{w}^T\mathbf{x}_i - w_0 \leq -1 \quad \forall i \in C_0 \end{cases} \tag{2.5}$$

then:

1. *The hyperplane H defined by the equation $\mathbf{w}^{*T}\mathbf{x} - w_0^*$ is a separating hyperplane for the data and*
2. *This hyperplane is the maximum-margin separating hyperplane.*

PROOF. First, we show the Problem 2.5 has a feasible solution. By Lemma 2.5, there is some $\mathbf{v} \in \mathbb{R}^n$ and $v_0 \in \mathbb{R}$ so that $\mathbf{v}^T\mathbf{x}_i - v_0 > 0$ if $y_i = 1$ and $\mathbf{v}^T\mathbf{x}_i - v_0 < 0$ if $y_i = 0$¿ Let $\alpha \in \mathbb{R}$ so that $\mathbf{v}^T\mathbf{x}_i - v_0 \geq \alpha$ if $y_i = 1$ and $\mathbf{v}^T\mathbf{x}_i - v_0 \leq \alpha$ if $y_i = 0$. Such an α must exist. Then setting $\mathbf{w} = \mathbf{v}/\alpha$ and $w_0 = v_0/\alpha$ establishes the existence of a feasible solution to Problem 2.5. The objective function of Problem 2.5 is bounded below by 0 and therefore, the problem must have at least one globally optimal solution. It is clear any such solution must yield a separating hyperplane.

We now show this optimal solution is the maximum margin separating hyperplane. Any minimizer of the function $z = \frac{1}{2}\|\mathbf{w}\|^2$ must also minimize $\|\mathbf{w}\|$ because $\|\mathbf{w}\| \geq 0$ and therefore must maximize $1/\|\mathbf{w}\|$. Thus, minimizing $\|\mathbf{w}\|^2$ is equivalent to maximizing $1/\|\mathbf{w}\|$. We will call this the alternate objective function.

Let H^+ be the hyperplane defined by the equation $\mathbf{w}^{*T}\mathbf{x}_i - w_0^* = 1$ and H^- be the hyperplane defined by the equation $\mathbf{w}^{*T}\mathbf{x}_i - w_0 = -1$. By Lemma 2.2 the distance from H is $2/\|\mathbf{w}^*\|$ and therefore the distance from either H^+ or H^- to H is $1/\|\mathbf{w}^*\|$. Therefore we know that $d(\mathbf{X}_0, H), d(\mathbf{X}_1, H) \geq 1/\|\mathbf{w}^*\|$. Consequently the margin distance is maximized, since the optimal alternate objective function value (that we are maximizing) is a lower bound on

the margin distance. The fact that this is the maximum margin distance follows from the that fact necessarily some of the constraints in Problem 2.5 must be binding (otherwise we could find alternate hyperplanes H^+ or H^- closer to one of the sets and thus a better H). Thus $d(\mathbf{X}_0, H), d(\mathbf{X}_1, H) = 1/\|\mathbf{w}^*\|$. This completes the proof. □

REMARK 2.11. It is worth noting that Problem 2.5 is nothing more than a special kind of quadratic programming problem where:

$$\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{I}_n \mathbf{w}$$

with $\mathbf{I}_n$ the $n \times n$ identity matrix. Since $\mathbf{I}_n$ is positive definite, it is in fact a *convex* quadratic programming problem and thus is amenable to several methods of solution.

EXERCISE 2.3. Given the data $(\mathbf{X}, \mathbf{y})$ find matrices $\mathbf{Q}$ and $\mathbf{A}$ and vectors $\mathbf{c}$ and $\mathbf{b}$ and write Problem 2.5 as a classical quadratic programming problem with form:

$$\begin{aligned} \min \ & \frac{1}{2}\mathbf{w}^T \mathbf{Q} \mathbf{w} + \mathbf{c}^T \mathbf{w} \\ s.t. \ & \mathbf{A}\mathbf{w} \leq \mathbf{b} \end{aligned} \tag{2.6}$$

Conclude that the problem is convex.

EXAMPLE 2.12. Consider the data set from Example 1.18.

$$\mathbf{X} = \begin{bmatrix} -2 & -2 \\ -1 & -1 \\ 1 & 1 \\ 2 & 2 \end{bmatrix}$$

and suppose that $\mathbf{y} = \langle 0, 0, 1, 1 \rangle$. The data is illustrated in Figure 4(a). In this example, $N = 4$ (there are 4 data points) and $n = 2$, each data point has two *features*; i.e., $\mathbf{x}_i \in \mathbb{R}^2$ for $i = 1, \ldots, 4$. The vector $\mathbf{w} = \langle w_1, w_2 \rangle$ and we seek to solve the following quadratic programming problem:

$$\left\{ \begin{array}{rll} \min & \frac{1}{2}w_1^2 + \frac{1}{2}w_2^2 & \underline{\text{Dual Var.}} \\ s.t. & 2w_1 + 2w_2 - w_0 \geq 1 & (\lambda_1) \\ & w_1 + w_2 - w_0 \geq 1 & (\lambda_2) \\ & -w_1 - w_2 - w_0 \leq -1 & (\lambda_3) \\ & -2w_1 - 2w_2 - w_0 \leq -1 & (\lambda_4) \end{array} \right. \tag{2.7}$$

This problem can be solved numerically using any number of solvers, but it is instructive to analyze the Karush-Kuhn-Tucker conditions for the problem (see Section 5 of Appendix C). Assuming gradients are computed with the decision variable vector $\langle w_0, w_1, w_2 \rangle$ we obtain the following Kuhn-Tucker Dual Feasibility condition:

$$\begin{bmatrix} 0 \\ w_1 \\ w_2 \end{bmatrix} + \lambda_1 \begin{bmatrix} 1 \\ -2 \\ -2 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + \lambda_3 \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \lambda_4 \begin{bmatrix} -1 \\ -2 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.8}$$

The complementary slackness conditions are:

$$\begin{aligned}
\lambda_1 (2w_1 + 2w_2 - w_0 - 1) &= 0 \\
\lambda_2 (w_1 + w_2 - w_0 - 1) &= 0 \\
\lambda_3 (-w_1 + -w_2 - w_0 + 1) &= 0 \\
\lambda_4 (-2w_1 - 2w_2 - w_0 + 1) &= 0
\end{aligned}$$

We require $\lambda_i \geq 0$ for all i to ensure dual feasibility. If we assume that $\lambda_1 = \lambda_4 = 0$ and $\lambda_2 = \lambda_3 \neq 0$, then we must solve the system of equations:

$$\begin{aligned}
\lambda_2 - \lambda_3 &= 0 \\
w_1 - \lambda_2 - \lambda_3 &= 0 \\
w_2 - \lambda_2 - \lambda_3 &= 0 \\
w_1 + w_2 - w_0 &= 1 \\
-w_1 - w_2 - w_0 &= -1
\end{aligned}$$

The first three equations arise from the Kuhn-Tucker equality and our assumptions on λ_1 and λ_4. The second two equations arise from primal feasibility and complementary slackness. Clearly, the second two equations have only one solution: $w_0 = 0$ and $w_1 = w_2 = \frac{1}{2}$. Furthermore, $\lambda_2 = \lambda_3$ and clearly $\lambda_2 = \lambda_3 = \frac{1}{4}$ solves the complete system. Thus, $w_0 = \lambda_1 = \lambda_4 = 0$ and $w_1 = w_2 = \frac{1}{2}$ and $\lambda_2 = \lambda_3 = \frac{1}{4}$ satisfies the Karush-Kuhn-Tucker conditions and thus must be an optimal solution. Therefore, $\mathbf{w} = \langle \frac{1}{2}, \frac{1}{2} \rangle$ defines a maximum margin separating hyperplane for the data. The resulting maximum-margin hyperplane is illustrated in Figure 2(a) while the KKT conditions assuming $w_0 = 0$ are illustrated in Figure 2(b).

EXERCISE 2.4. Suppose $(\mathbf{w}^*, w_0^*)$ is the optimal solution to Problem 2.5 and let $\lambda_1, \ldots, \lambda_N$ be the Lagrange multipliers (dual variables). Let:

$$\tilde{y}_i = \begin{cases} 1 & \text{if } y_i = 1 \\ -1 & \text{if } y_i = 0 \end{cases} \tag{2.9}$$

Show that:

$$\mathbf{w}^* = \sum_{i=1}^{N} \tilde{y}_i \lambda_i \mathbf{x}_i \tag{2.10}$$

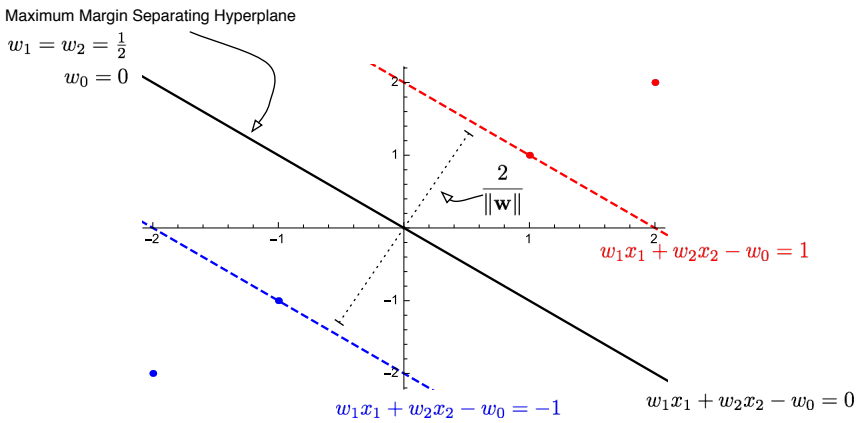
and

$$\sum_{i=1}^{N} \lambda_i \tilde{y}_i = 0 \tag{2.11}$$

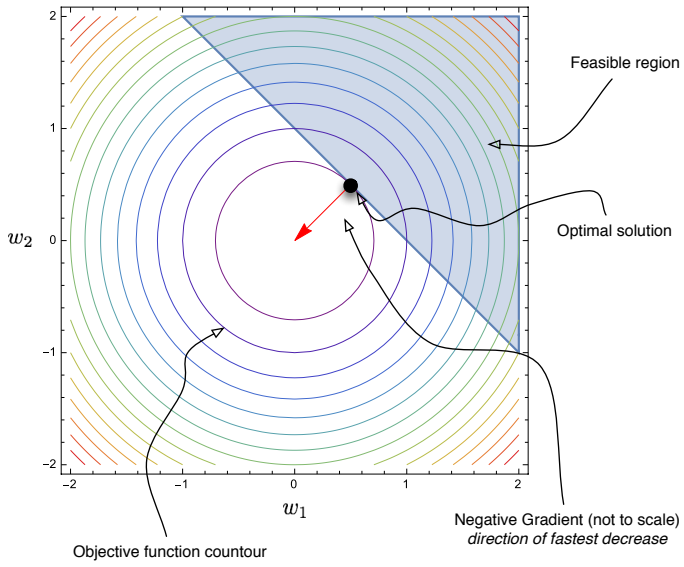
and if $\lambda_i \neq 0$, then:

$$w_0 = \tilde{y}_i - \mathbf{w}^{*T} \mathbf{x}_i \tag{2.12}$$

[Hint: This is tricky: Start by writing the inequality constraints in Problem 2.5 as $\tilde{y}_i \left(\mathbf{w}^T \mathbf{x}_i - w_0 \right) \geq 1$. When written in less-than-or-equal to form you obtain:$g_i(\mathbf{w}, w_0) = -\tilde{y}_i \left(\mathbf{w}^T \mathbf{x}_i - w_0 \right) + 1 \leq$



(a) Support Vector Classifier



(b) SVM KKT Conditions

Figure 2.2. (a) An illustration of a simple support vector machine with the margin illustrated. (b) The KKT conditions for the simple support vector machine.

0. Assume the decision variables are ordered (in the vector) $\langle w_0, \mathbf{w}\rangle$. Then compute the gradient of g_i and you get: $\langle -\tilde{y}_i, -\tilde{y}_i\mathbf{x}_i\rangle$. If $z(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$, then $\nabla z = \langle 0, \mathbf{w}\rangle$. Write the Kuhn-Tucker equality (dual constraint) and solve for $\mathbf{w}$ and you obtain Equation 2.10. Look at the first elements to obtain Equation 2.11. Now reason about Equation 2.12 using complementary slackness.]

3. Soft Margin Support Vector Machines

REMARK 2.13. It is still possible to use SVM methods even if the underlying assumption of linear separability of the data is not satisfied. The simplest approach is to price out the constraints and solve the resulting unconstrained optimization problem.

DEFINITION 2.14 (Soft Margin Support Vector Machine). The *soft margin support vector machine* is the linear classifier that results from solving the unconstrained optimization problem:

$$\min \alpha \|\mathbf{w}\|^2 + \sum_{i \in C_0} \left(\max\left\{0, \mathbf{w}^T\mathbf{x}_i - w_0 + 1\right\}\right)^2 + \sum_{i \in C_1} \left(\max\left\{0, 1 - \mathbf{w}^T\mathbf{x}_i + w_0\right\}\right)^2 \tag{2.13}$$

REMARK 2.15. There are a few things worth noting about Expression 2.13. The first is that most texts set the soft margin objective function as:

$$\min \alpha \|\mathbf{w}\|^2 + \frac{1}{N}\sum_{i \in C_0} \max\left\{0, \mathbf{w}^T\mathbf{x}_i - w_0 + 1\right\} + \frac{1}{N}\sum_{i \in C_1} \max\left\{0, 1 - \mathbf{w}^T\mathbf{x}_i + w_0\right\} \tag{2.14}$$

The two differences are:

(1) The priced-out constraints all have a coefficient of $\frac{1}{N}$. This is largely irrelevant as it can be absorbed into the weighting constant α as needed.
(2) The second difference is the use of $(\max\{0, \cdot\})$ rather than $(\max\{0, \cdot\})^2$. While both are convex functions, meaning the objective functions in either case are convex, the form given in Expression 2.13 has the benefit that $(\max\{0, \cdot\})^2$ is *differentiable*, meaning gradient descent based techniques can be applied directly. This is illustrated in Figure 2.3.

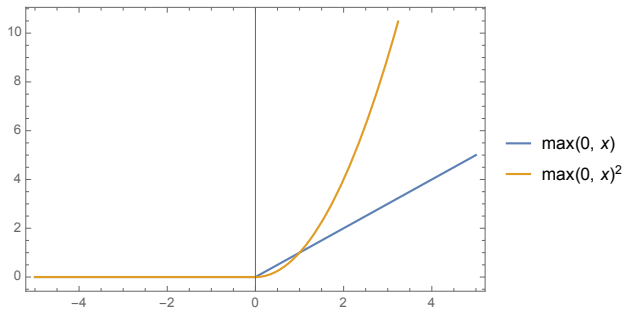


Figure 2.3. The function $\max\{0, \cdot\}^2$ is differentiable everywhere, while $\max\{0, \cdot\}$ is not.

EXERCISE 2.5. Show that if:

$$g(\mathbf{w}) = \left(\max\left\{0, \mathbf{w}^T\mathbf{x}_i - w_0 + 1\right\}\right)^2$$

then:

$$\frac{\partial g}{\partial w_j} = \begin{cases} 2\left(\mathbf{w}^T\mathbf{x}_i - w_0 + 1\right)\mathbf{x}_{i_j} & \text{if } \mathbf{w}^T\mathbf{x}_i - w_0 + 1 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial g}{\partial w_0} = \begin{cases} -2\left(\mathbf{w}^T\mathbf{x}_i - w_0 + 1\right) & \text{if } \mathbf{w}^T\mathbf{x}_i - w_0 + 1 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

EXERCISE 2.6. Analyze Problem 1.16 with $S(x; \beta)$ when $\beta = 0$. Relate Problem 1.16 to the Soft Margin SVM.

REMARK 2.16. The following proposition, which we state but do not prove asserts that if the data set $(\mathbf{X}, \mathbf{y})$ is linearly separable, then for *small* values of α, the resulting solution to Problem 2.13 is a maximum-margin separating hyperplane. To see this, note that when $\alpha = 0$, then any separating hyperplane is an optimal solution to Problem 2.13. As α is increased away from 0, the hard margin constraints still dominate the objective function, but the importance of the margin distance grows. At some point, the pieces of the objective function balance and the result is an maximum margin separating hyperplane. Formalizing this argument is tricky, which is why we do not prove the result formally.

PROPOSITION 2.17. *If* $(\mathbf{X}, \mathbf{y})$ *is linearly separable, then there is some* $\alpha > 0$ *so that:*

$$(\mathbf{w}^*, w_0^*) = \arg\min \alpha \|\mathbf{w}\|^2 + \sum_{i \in C_0} \left(\max\left\{0, \mathbf{w}^T \mathbf{x}_i - w_0 + 1\right\}\right)^2 + \sum_{i \in C_1} \left(\max\left\{0, 1 - \mathbf{w}^T \mathbf{x}_i + w_0\right\}\right)^2$$

defines a maximum margin separating hyperplane. □

EXAMPLE 2.18. We illustrate the difference between soft and hard SVM's using a randomly generated data set of 100 points. The resulting hard and soft computed maximum-margin linear separators are shown in Figure 2.4. It is worth noting in Figure 2.4 that we

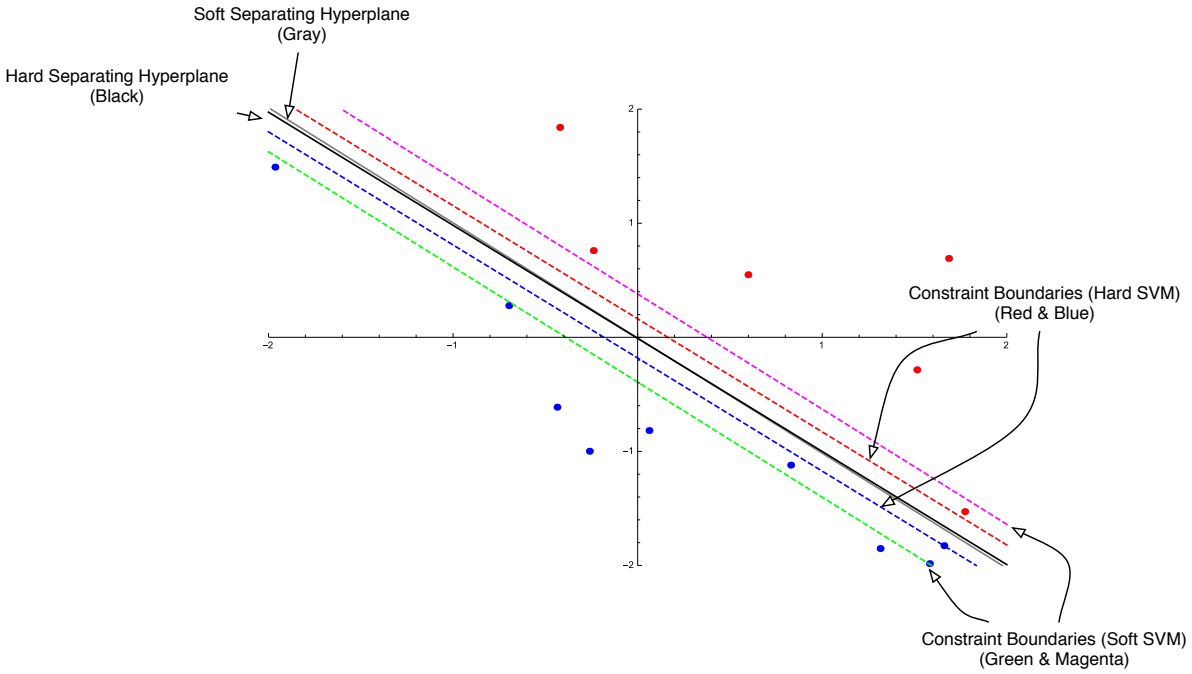


Figure 2.4. A comparison of hard and soft SVM. Notice the similarity between the two separating hyperplanes even though the margin hyperplanes are quite different. Note $\alpha = 0.1$.

set $\alpha = 0.1$ and not all data points are shown to facilitate visual comparison between the separating hyperplanes.

4. Alternate Formulation

REMARK 2.19. Recall from Remark 2.15 that an alternate formulation of the objective function in the Soft Margin SVM Expression 2.14:

$$\min \alpha \|\mathbf{w}\|^2 + \frac{1}{N}\sum_{i\in C_0} \max\left\{0, \mathbf{w}^T\mathbf{x}_i - w_0 + 1\right\} + \frac{1}{N}\sum_{i\in C_1} \max\left\{0, 1 - \mathbf{w}^T\mathbf{x}_i + w_0\right\}$$

This objective function, which is not differentiable can be replaced by a second constrained optimization problem, yielding an equivalent and useful formulation. To do this, we will use the formulation from Exercise 2.4.

DEFINITION 2.20 (Alternate Soft-Margin SVM). Suppose $\tilde{\mathbf{y}}$ is defined by Equation 2.9. Then the *constrained form of the soft margin SVM* problem is[1]:

$$\begin{cases} \min \;\; z(\mathbf{w}, \boldsymbol{\zeta}) = \dfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\zeta_i \\ s.t. \;\; \tilde{y}_i(\mathbf{w}^T\mathbf{x}_i - w_0) \geq 1 - \zeta_i \quad \forall i \in \{1,\ldots,N\} \\ \qquad \zeta_i \geq 0 \quad \forall i \in \{1,\ldots,N\} \end{cases} \tag{2.15}$$

REMARK 2.21. There are other forms of the problem, in particular where $\frac{1}{2}$ is replaced by a parameter (e.g., α) and C is replaced by $\frac{1}{N}$. However, formulations that use this tend to be sloppy in their handling of the parameters.

REMARK 2.22. If $(\mathbf{X}, \mathbf{y})$ is linearly separable, then an optimal solution has $\zeta_i^* = 0$ and the problem is equivalent to Problem 2.5. This problem is convex, differentiable and has linear constraints. Furthermore, if needed the N variables ζ_i can be replaced by a single common variable $\zeta \geq 0$. In either case, convexity implies that the problem is amenable to analysis by Lagrangian dual (see Section 6 of Appendix C).

THEOREM 2.23. *Suppose that $\lambda_1, \ldots, \lambda_N$ are the Lagrange multipliers of constraints of the form $\tilde{y}_i(\mathbf{w}^T\mathbf{x}_i - w_0) \geq 1 - \zeta_i$ and $\rho_1, \ldots, \rho_N$ are the Lagrange multipliers for the constraints $\zeta_i \geq 0$. Then:*

$$\begin{cases} \max\limits_{\boldsymbol{\lambda}} \;\; \mathcal{L}(\boldsymbol{\lambda}) = \sum\limits_i \lambda_i - \dfrac{1}{2}\sum\limits_i\sum\limits_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i \\ s.t. \;\; \sum\limits_i \lambda_i\tilde{y}_i = 0 \\ \qquad 0 \leq \lambda_i \leq C \quad \forall i \in \{1,\ldots,N\} \end{cases} \tag{2.16}$$

is the Lagrangian (Wolfe) Dual for Problem 2.15.

PROOF. Putting the constraints in standard form, we have:

$$g_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) = 1 - \zeta_i - \tilde{y}_i\left(\mathbf{w}^T\mathbf{x}_i - w_0\right) \leq 0 \quad \forall i \in \{1,\ldots,N\}$$
$$h_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) = -\zeta_i \leq 0 \quad \forall i \in \{1,\ldots,N\}$$

[1]This form is taken from Andrew Ng's lecture notes from Stanford's Machine learning class, http://cs229.stanford.edu/notes/cs229-notes3.pdf, which are an excellent alternative source.

Assuming the decision variables are ordered according to the vector $\langle w_0, \mathbf{w}, \boldsymbol{\zeta} \rangle$. The relevant gradients are:

$$\nabla z = \begin{bmatrix} 0 \\ \mathbf{w} \\ \mathbf{C} \end{bmatrix} = \qquad \nabla g_i = \begin{bmatrix} \tilde{y}_i \\ -\tilde{y}_i \mathbf{x}_i \\ -\mathbf{e}_i \end{bmatrix} \qquad \nabla h_i = \begin{bmatrix} 0 \\ \mathbf{0} \\ -\mathbf{e}_i \end{bmatrix},$$

where $\mathbf{0} \in \mathbb{R}^n$ is the n-dimensional zero vector, $\mathbf{e}_i$ is a standard basis vector in $\mathbb{R}^n$ and $\mathbf{C} \in \mathbb{R}^n$ is a vector consisting entirely of the constant C. The Kuhn-Tucker equality is:

$$\nabla z + \sum_{i=1}^{N} \lambda_i \nabla g_i + \sum_{i=1}^{N} \rho_i \nabla h_i = \mathbf{0}$$

From the first row of the Kuhn-Tucker equality, we can deduce that:

$$\sum_i \lambda_i \tilde{y}_i = 0 \tag{2.17}$$

Consider the equations derived from row 2 to $N + 1$ This yields the expression:

$$\mathbf{w} = \sum_i \lambda_i \tilde{y}_i \mathbf{x}_i$$

just as in Equation 2.10. Finally, if we consider the equations derived from rows $N + 2$ to $2N + 1$, we see:

$$C - \lambda_i - \rho_i = 0 \implies C = \lambda_i + \rho_i \tag{2.18}$$

We can now construct the Lagrangian Dual. Note, that since Problem 2.15 is convex and differentiable as we observed in Remark 2.22, we have the dual optimization problem:

$$\begin{cases} \max\limits_{\mathbf{w}, w_0, \boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\rho}} & z(\mathbf{w}, \boldsymbol{\zeta}) + \sum\limits_i \lambda_i g_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) + \sum\limits_i \rho_i h_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) \\ s.t. & \nabla z + \sum\limits_{i=1}^{N} \lambda_i \nabla g_i + \sum\limits_{i=1}^{N} \rho_i \nabla h_i = \mathbf{0} \\ & \boldsymbol{\lambda}, \boldsymbol{\rho} \geq \mathbf{0} \end{cases}$$

We'll use Equation 2.10 to re-write $\lambda_i g_i(\mathbf{w}, w_0, \boldsymbol{\zeta})$ in the objective function:

$$\sum_i \lambda_i g_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) = \sum_i \lambda_i \left(1 - \zeta_i - \tilde{y}_i \left(\sum_j \lambda_j \tilde{y}_j \mathbf{x}_j^T \right) \mathbf{x}_i - w_0 \right) =$$
$$\sum_i \lambda_i - \sum_i \lambda_i \zeta_i - \sum_i \sum_j \lambda_i \lambda_j \tilde{y}_i \tilde{y}_j \mathbf{x}_j^T \mathbf{x}_i + w_0 \sum_i \lambda_i \tilde{y}_i$$

Applying Equation 2.17 we can simplify this expression:

$$\sum_i \lambda_i g_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) = \sum_i \lambda_i - \sum_i \sum_j \lambda_i \lambda_j \tilde{y}_i \tilde{y}_j \mathbf{x}_j^T \mathbf{x}_i - \sum_i \lambda_i \zeta_i \tag{2.19}$$

We can also use Equations 2.10 and 2.18 to rewrite $z(\mathbf{w}, \boldsymbol{\zeta})$:

(2.20)
$$z(\mathbf{w}, \boldsymbol{\zeta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \zeta_i = \frac{1}{2}\mathbf{w}^T\mathbf{w}) + C\sum_i \zeta_i = \frac{1}{2}\left(\sum_j \lambda_j \tilde{y}_j \mathbf{x}_j^T\right)\left(\sum_i \lambda_i \tilde{y}_i \mathbf{x}_i\right) =$$
$$\frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i + \sum_i C\zeta_i = \frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i + \sum_i(\lambda_i + \rho_i)\zeta_i$$

Finally, note that:

(2.21)
$$\sum_i \rho_i h_i(\mathbf{w}, w_0, \boldsymbol{\zeta}) = \sum_i \rho_i(-\zeta_i) = -\sum_i \rho_i\zeta_i$$

Combing all these elements, the objective function of the Lagrangian dual problem becomes:

$$\frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i + \sum_i(\lambda_i + \rho_i)\zeta_i +$$
$$\sum_i \lambda_i - \sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i - \sum_i \lambda_i\zeta_i - \sum_i \rho_i\zeta_i$$

Combining like terms (shown in colors), this simplifies to:

(2.22)
$$\mathcal{L}(\boldsymbol{\lambda}) = \sum_i \lambda_i - \frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i$$

Using Equations 2.10 and 2.17 we have ensured the Kuhn-Tucker equality is satisfied. Thus the Lagrangian (Wolfe) Dual is:

$$\begin{cases} \max\limits_{\boldsymbol{\lambda}} \ \mathcal{L}(\boldsymbol{\lambda}) = \sum\limits_i \lambda_i - \frac{1}{2}\sum\limits_i\sum\limits_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\mathbf{x}_j^T\mathbf{x}_i \\ s.t. \ \ \sum\limits_i \lambda_i\tilde{y}_i = 0 \\ \qquad \mathbf{0} \leq \boldsymbol{\lambda} \leq \mathbf{C} \end{cases}$$

Where $\lambda_i \leq C$ for all i because $\lambda_i = C - \rho_i$ and $\rho_i \geq 0$ necessarily. This completes the proof. □

REMARK 2.24. Notice that $\mathbf{x}_j^T\mathbf{x}_i$ is nothing more than the dot-product of the sample vector $\mathbf{x}_i$ and the sample vector $\mathbf{x}_j$. Thus, the dual objective function is sometimes written:

(2.23)
$$\mathcal{L}(\boldsymbol{\lambda}) = \sum_i \lambda_i - \frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j(\mathbf{x}_i \cdot \mathbf{x}_j) = \sum_i \lambda_i \frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j\tilde{y}_i\tilde{y}_j\langle\mathbf{x}_i, \mathbf{x}_j\rangle$$

For the final section of this chapter, it helps to think of this operation as a dot product. Notice that the dot product $\mathbf{x}_i \cdot \mathbf{x}_j$ is nothing more than a scaled measure of the angle between the vector $\mathbf{x}_i$ and $\mathbf{x}_j$. In particular, the more $\mathbf{x}_j$ points in the direction of $\mathbf{x}_i$, the larger dot product is. Thus, $\mathbf{x}_i \cdot \mathbf{x}_j$ is (in some sense) a similarity measurement between the two vectors.

REMARK 2.25. It is worth noting that Problem 2.16 can be solved in any number of ways since it is a quadratic programming problem with simple constraints. In particular, the SMO Algorithm [] and sub-gradient methods [] have been very successful. We discuss the SMO algorithm in a subsequent section.

5. The Kernel Trick: Non-linearly Separable Data

THEOREM 2.26 (Cover's Theorem). *Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ be a data set with a classification $\mathbf{y} \in \{0,1\}^N$ that is not linearly separable. Then there is a (non-linear) transformation $\varphi : \mathbf{X} \to \mathbb{R}^m$ where $m \geq n$ so that $\varphi(\mathbf{X})$ is linearly separable (in $\mathbb{R}^m$).*

PROOF. The simplex Δ_N contains N vertices and every partition of the vertices (extreme points) of the simplex into two groups can be separated by a hyperplane. Thus, let φ assign each point in $\mathbf{X}$ to some extreme point of the simpled Δ_N. □

REMARK 2.27. Cover's rather trivial theorem argues that data is easier to separate using linear classifiers in higher dimensional space than it is in lower dimensional space. This fact forms the basis for deep neural networks, as we'll see later.

It stands to reason that given a set of data $\mathbf{X}$ that is not linearly separable, it would be nice to find a mapping so that $\varphi(\mathbf{X})$ is linearly separable after an appropriate non-linear transform.

EXAMPLE 2.28. Consider Figure 2.5(a), which was generated by creating 100 random points and then defining the classification:

$$y_j = \begin{cases} 1 & \text{if } x_{i_1} + x_{i_2}^2 > 0 \\ 0 & \text{otherwise} \end{cases}$$

Clearly this data set if not linearly separable. However, applying the non-linear transformation $\varphi : (x_{i_1}, x_{i_2}) \mapsto (x_{i_1}, x_{i_2}^2)$ yields a linearly separable data set as shown in Figure 2.5(b). A classifier for this data set, could then work by first applying φ and then the linear classifier on the transformed data. *Unfortunately, we rarely are given a nice linear transform that will work.*

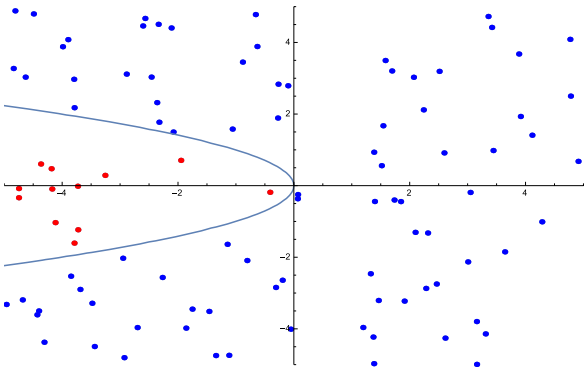
REMARK 2.29. The remainder of this chapter uses the Lagrangian dual formulation. Notice that the objective function of the Lagrangian dual in the transformed space is:

$$\mathcal{L}(\boldsymbol{\lambda}) = \sum_i \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j \tilde{y}_i \tilde{y}_j (\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)) \tag{2.24}$$

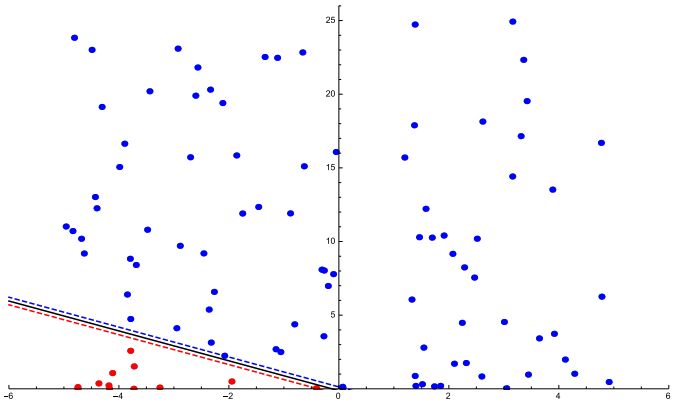
So it would seem that if we need to compute $\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$ for all pairs of data vectors $(\mathbf{x}_i, \mathbf{x}_j)$. The *kernel trick*, ideally, allows us to avoid this and do something computationally more efficient.

DEFINITION 2.30 (Kernel Trick). Suppose there is a continuous function $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ so that:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j \tag{2.25}$$



(a) Not Linearly Separable



(b) Linearly Separable

Figure 2.5. (a) Data that is not linearly separable. (b) A non-linear transform of the non-separable data leads to a new linearly separable data.

called *Kernel function*. Then, the *kernel trick* is to substitute $K(\mathbf{x}_i, \mathbf{x}_j)$ (which is assumed to be easier to compute) into Equation 2.24 to obtain:

$$\mathcal{L}(\boldsymbol{\lambda}) = \sum_i \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j \tilde{y}_i \tilde{y}_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.26}$$

Moreover, for a finite number of samples, we can create a matrix $\mathbf{K} \in \mathbb{R}^{N \times N}$ so that:

$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.27}$$

is called the *Kernel matrix* or *Gram matrix*.

PROPOSITION 2.31. *Assume there is a Kernel function K for the (non-linear) transform φ. Then K is symmetric and the Kernel matrix $\mathbf{K}$ is positive semi-definite.*

PROOF. If K is a kernel function, then:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j) = \varphi(\mathbf{x}_j) \cdot \varphi(\mathbf{x}_i) = K(\mathbf{x}_j, \mathbf{x}_i) \tag{2.28}$$

by the symmetry of the dot product. Consequently, $\mathbf{K}$ is necessarily symmetric. Let $\mathbf{z}$ be an arbitrary vector in $\mathbb{R}^N$. Then:

$$\mathbf{z}^T\mathbf{K}\mathbf{z} = \sum_i \sum_j \mathbf{K}_{ij} z_i z_j = \sum_i \sum_j \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j) z_i z_j =$$

$$\sum_i \sum_j (z_i \varphi(\mathbf{x}_i)) \cdot (z_j \varphi(\mathbf{x}_j)) = \sum_i z_i \varphi(\mathbf{x}_i) \cdot \left(\sum_j z_j \varphi(\mathbf{x}_j) \right)$$

Let $\mathbf{r} = \sum_i z_i \varphi(\mathbf{x}_i) = \sum_j z_i \varphi(\mathbf{x}_i)$ be a vector in $\mathbb{R}^m$. Then:

$$\mathbf{z}^T\mathbf{K}\mathbf{z} = \sum_i z_i \varphi(\mathbf{x}_i) \cdot \mathbf{r} = \mathbf{r} \cdot \left(\sum_i z_i \varphi(\mathbf{x}_i) \right) = \|\mathbf{r}\|^2 \geq 0 \tag{2.29}$$

by the *symmetry of the dot product*. Thus, $\mathbf{K}$ is symmetric and positive semi-definite. □

REMARK 2.32. Whenever $\mathbf{K}$ is symmetric and positive semi-definite (or more exactly, the kernel function K generates a symmetric positive semi-definite kernel matrix), then the kernel is said to be *valid*.

REMARK 2.33. Note that the identity map $\varphi(\mathbf{x}_i) = \mathbf{x}_i$ leads to a kernel function that simply returns the dot product in n dimensional space. This leads to a valid kernel and thus:

$$-\frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j \tilde{y}_i \tilde{y}_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle,$$

which appears in the objective function of the Lagrangian dual takes its maximum at $\mathbf{0}$ because the derived kernel matrix $\mathbf{K}_{ij} = \mathbf{x}_i^T \mathbf{x}_j$ is symmetric positive-definite. (Negating it implies that this quantity is at most zero.) Thus the maximization is well defined.

REMARK 2.34. When given an arbitrary data set $(\mathbf{X}, \mathbf{y})$, it is rarely obvious which kernel function to use; i.e., the situation is rarely as nice as Example 2.28. Therefore, a number of standard kernels are usually attempted:

- Polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^r$, where r is a parameter to be chosen. There is also the *inhomogenous* polynomial transform $(1 + \mathbf{x}_i \cdot \mathbf{x}_j)^r$.
- Gaussian (Radial Basis Function): $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$ for $\gamma > 0$.
- Hyperbolic Tangent: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left((\beta \mathbf{x}_i \cdot \mathbf{x}_j + c\right)$ for $\beta > 0$ and $c < 0$.

Notice, even though Kernel methods are called non-parametric, this is a complete misnomer. The parameters of the kernel function *must* be specified, even if this is done by a second level of optimization to select the parameters.

REMARK 2.35. The kernel function we have discussed in this chapter is used to obtain several deep results in statistical learning theory through Mercer's theorem and reproducing kernel Hilbert spaces, which we will not discuss at this point. However, understanding the basics of kernel functions, is a foundational element of this subject.

REMARK 2.36 (Multi-Class Classifiers). Applying linear classifiers to multi-class data (i.e., where $\mathbf{y}$ can take on more than values 0 and 1) is well studied , but consensus on the best approach has not been reached. Here are two approaches:

- If there are l classes, define l linear classifiers where each class is compared against all other classes (i.e., make $\mathbf{y}' \in \{0,1\}^N$ where 0 indicates that the sample is not in the class in question and 1 indicates the sample is in the class in question). Ties can be broken by using a data points distance from the separating hyperplane.
- Another approach is to solve for $l(l-1)/2$ classifiers one for each combination of classes and then use voting to decide a winning class [].

EXERCISE 2.7. Find an expression that will allow you to derive w_0 from the solution of Problem 2.16.

CHAPTER 3

Regression and Logistic Regression

1. Linear Regression

REMARK 3.1. In this section, for the first time, we assume that the values in the vector $\mathbf{y}$ may take on values outside $\{0, 1\}$.

DEFINITION 3.2 (Data Model). Suppose that $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ is a set of feature vectors with corresponding *response* values $\mathbf{y} = \langle y_1, \ldots, y_N \rangle \in \mathbb{R}^N$. A data model is any function $f : \mathbb{R}^n \to \mathbb{R}$ that maps $\mathbf{x}_j$ to $\hat{y}_j$, the model-approximated response corresponding to y_j.

REMARK 3.3. If we model the response $\mathbf{y}$ with the function $f : \mathbb{R}^n \to \mathbb{R}$, then we write $\mathbf{y} \sim f(\mathbf{X})$ or $y_i \sim f(\mathbf{x}_i)$ to indicate that f is a model with (presumably) non-zero residuals.

DEFINITION 3.4 (Residual). Given a data model $f : \mathbb{R}^n \to \mathbb{R}$, the j^{th} residual of the model given data set $(\mathbf{X}, \mathbf{y})$ is $\epsilon_j = y_j - f(\mathbf{x}_j)$.

DEFINITION 3.5 (Linear Model). Suppose that $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ is a set of feature vectors with corresponding *response* values $\mathbf{y} = \langle y_1, \ldots, y_N \rangle \in \mathbb{R}^N$. A linear model of the data set $(\mathbf{X}, \mathbf{y})$ is a function:

$$f(x_1, \ldots, x_n) = w_0 + w_1 x_1 + \cdots + w_n x_n \tag{3.1}$$

REMARK 3.6. If we assume that each feature vector $\mathbf{x}_j$ is prepended with 1, then Equation 3.1 can be rewritten as:

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x} = \mathbf{x}^T \mathbf{w}$$

where $\mathbf{x} = \langle 1, x_1, \ldots, x_n \rangle$. We will assume this convention throughout the remainder of this chapter.

DEFINITION 3.7 (Sum of Square Error). Suppose that $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \subset \mathbb{R}^n$ is a set of feature vectors with corresponding *response* values $\mathbf{y} = \langle y_1, \ldots, y_N \rangle \in \mathbb{R}^N$. The square error of a model $f : \mathbb{R}^n \to \mathbb{R}$ is given by the formula:

$$\mathcal{E}(f; \mathbf{X}, \mathbf{y}) = \sum_{j=1}^{N} (y_j - \hat{y}_j)^2 = \sum_{j=1}^{N} (y_j - f(\mathbf{x}_j))^2 \tag{3.2}$$

REMARK 3.8. It is convenient to think of the set of feature vectors $\mathbf{X}$ as a matrix (rather than a set) so that row i corresponds to vector $\mathbf{x}_i^T$. Thus:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}$$

Thus $\mathbf{X} \in \mathbb{R}^{N \times n}$.

EXAMPLE 3.9. Suppose we are given the data $\{\langle -2,-2\rangle, \langle -1,-1\rangle, \langle 1,1\rangle, \langle 2,2\rangle\}$. Then adding a 1 to these vectors and writing as a matrix, we would obtain:

$$\mathbf{X} = \begin{bmatrix} 1 & -2 & -2 \\ 1 & -1 & -1 \\ 1 & 1 & 1 \\ 1 & 2 & 2 \end{bmatrix},$$

as we did in Example 1.18.

PROPOSITION 3.10. *Let $\mathbf{w} \in \mathbb{R}^n$. Then the square-error of a linear model with coefficient vector $\mathbf{w}$ is given by:*

$$\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \left(\mathbf{y}^T - \mathbf{w}^T\mathbf{X}^T\right)(\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}\mathbf{y} + \mathbf{y}^T\mathbf{y} \tag{3.3}$$

PROOF. Row j of $\mathbf{X}$ is $\mathbf{x}_j^T$. Consequently, the j^{th} element of $\mathbf{X}\mathbf{w}$ is $\mathbf{x}_j^T\mathbf{w}$, the model-approximated response corresponding to y_j. Thus the vector of residuals is given by:

$$\boldsymbol{\epsilon} = \mathbf{y} - \mathbf{X}\mathbf{w} = \begin{bmatrix} y_1 - \mathbf{x}_1^T\mathbf{w} \\ y_2 - \mathbf{x}_2^T\mathbf{w} \\ \vdots \\ y_N - \mathbf{x}_N^T\mathbf{w} \end{bmatrix}$$

Note the sum of square error is simply $\boldsymbol{\epsilon}^T\boldsymbol{\epsilon} = \|\boldsymbol{\epsilon}\|^2$. Thus:

$$\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \left(\mathbf{y}^T - \mathbf{w}^T\mathbf{X}^T\right)(\mathbf{y} - \mathbf{X}\mathbf{w})$$

Expanding the right-hand-side, we obtain:

$$\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{y}^T\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}$$

The terms $\mathbf{w}^T\mathbf{X}^T\mathbf{y}$ and $\mathbf{y}^T\mathbf{X}\mathbf{w}$ are *both* scalars and transposes of each other, thus:

$$\mathbf{w}^T\mathbf{X}^T\mathbf{y} = \mathbf{y}^T\mathbf{X}\mathbf{w}$$

Thus, we may write:

$$\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}\mathbf{y} + \mathbf{y}^T\mathbf{y}$$

This completes the proof. □

THEOREM 3.11. *Let $(\mathbf{X},\mathbf{y})$ be a data set and suppose $\mathbf{X}^T\mathbf{X}$ is non-singular. Then:*

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

minimizes $\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y})$.

PROOF. Minimizing sum of square error is equivalent to solving the unconstrained quadratic programming problem:

$$\min \quad \mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}\mathbf{y} + \mathbf{y}^T\mathbf{y}$$

The KKT conditions are simply:

$$\nabla_{\mathbf{w}}\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = \mathbf{0}$$

where $\nabla_{\mathbf{w}}$ indicates we are differentiating with respect to the variables in $\mathbf{w}$ *only*. Taking the (partial) derivative and setting equal to zero, we obtain:

$$\nabla_{\mathbf{w}}\mathcal{E}(\mathbf{w};\mathbf{X},\mathbf{y}) = 2\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{X}^T\mathbf{y} = \mathbf{0}$$

Solving for $\mathbf{w}$ we obtain:

$$\mathbf{w} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

as required assuming $\left(\mathbf{X}^T\mathbf{X}\right)$ is non-singular. □

EXERCISE 3.1. Suppose we do not prepend a 1 to the data so that we seek $\mathbf{w}$ and w_0 to minimize the sum of squares error of $f(\mathbf{x};\mathbf{w}) = w_0 + \mathbf{w}^T\mathbf{x}$. Show that:

$$\mathcal{E}(w_0,\mathbf{w};\mathbf{X},\mathbf{y}) = \mathbf{y}^T\mathbf{y} - 2\cdot\mathbf{w}^T\mathbf{X}\mathbf{y} - 2\cdot w_0\mathbf{1}^T\mathbf{y} + 2\cdot\mathbf{1}^T\mathbf{X}\mathbf{w} + \mathbf{1}^T\mathbf{1}w_0^2 + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} \tag{3.4}$$

Here $\mathbf{1} = \langle 1,1,\ldots,1\rangle$ is a vector of 1's of length n and each row of $\mathbf{X}$ is a feature vector but *without* the 1's. Take the derivative of this expression with respect to w_0, set it equal to 0 and show that:

$$w_0^* = \frac{1}{N}\mathbf{1}^T\left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) = \frac{1}{N}\sum_{j=1}^{N}\left(y_j - \mathbf{w}^T\mathbf{x}_j\right) \tag{3.5}$$

Consequently, show that if each data vector $\mathbf{x}_j$ is *re-centered* to $\tilde{\mathbf{x}}_j = \mathbf{x}_j - \mathbf{1}w_0$, then the resulting $\mathbf{w}^* = \left(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}\right)^{-1}\tilde{\mathbf{X}}^T\mathbf{y}$, where each row $\mathbf{x}_j^T$ in $\mathbf{X}$ is replaced with $\tilde{\mathbf{x}}_j$ to obtain $\tilde{\mathbf{X}}$. [Hint: $\mathbf{1}^T\mathbf{1} = N$ and if $\mathbf{z} = \langle z_1,\ldots,z_n\rangle$ is a vector, then $\mathbf{1}^T\mathbf{z} = z_1 + z_2 + \cdots + z_n$.]

REMARK 3.12. Suppose the values in $\mathbf{y}$ fall into a discrete set of ranges that correspond to categories $\mathcal{C} = \{C_1,\ldots,C_k\}$. This could mean that the values only take

Given such an optimal $\mathbf{w}$, a sample $\mathbf{x}$ not present in the given data $\mathbf{X}$ would would be classified to

$$\arg\min_{y\in\mathcal{C}}(y - \mathbf{w}^T\mathbf{x})^2. \tag{3.6}$$

As such, linear regression is a called a *descriminative* approach to classification.

EXAMPLE 3.13. Consider the extremely simple data set:

$$\begin{bmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}$$

with corresponding values $\mathbf{y} = \langle -1,0,2,3\rangle$. Find a best fit line for this data.

2. Logistic Regression - Regression Formulation

REMARK 3.14. We now return to our classification problem. For the remainder of this section, let $(\mathbf{X},\mathbf{y})$ be a set of N feature vectors with classification $y_i \in \{0,1\}$. Without loss of generality, assume $\mathbf{X}$ is modified so there is a 1 at the front of each feature vector as we did in the previous section.

REMARK 3.15 (Proportion Data). In the simplest formulation, logistic regression does not work on classes $\{0, 1\}$ but rather probabilities that a feature vector $\mathbf{x}_i$ will map to Class 1. Thus, the data set we consider has form $(\mathbf{X}, \mathbf{p})$, where $p_i \in [0, 1]$.

REMARK 3.16. Transforming class data in the form $(\mathbf{X}, \mathbf{y})$ to proportion data $(\mathbf{X}, \mathbf{p})$ can be accomplished using multiple samples (e.g., collect more than one sample with the same feature vector). It can also be accomplished by *binning* the data. For example if we are collecting information about people and we are given only 1 sample for each individual from ages 18 - 65, breaking the groups into age ranges 18-24, 24-30 will bin the data and generate multiple samples. Notice, however, in this case, the feature vector changes.

REMARK 3.17. Recall the *sigmoid function or logistic function* (with $\beta = 1$) is the function:

$$S(z) = \frac{1}{1 + e^{-z}} \tag{3.7}$$

This is an "S" shaped curve that approaches 1 as $x \to \infty$ and 0 as $x \to -\infty$. (See Figure 1.5.)

DEFINITION 3.18 (Logistic Model). The *logistic model* assumes that p_i (the probability that $y_i = 1$), given feature vector $\mathbf{x}_i$ is given by:

$$\Pr(y_i = 1|\mathbf{x}_i) = p_i \sim S(\mathbf{w}^T\mathbf{x}_i) = \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x}_i)} \tag{3.8}$$

The parameter vector $\mathbf{w} \in \mathbb{R}^n$ must be fitted, as in regression.

DEFINITION 3.19 (Logistic Classification). Given an unknown sample $\mathbf{x}_{N+1}$, the logistic classifier is defined as:

$$y_{N+1} = \begin{cases} 1 & \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x}_{N+1})} > \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

REMARK 3.20. Note, when p_i is defined as in Equation 3.8, then $p_i > \frac{1}{2}$ exactly when $\mathbf{w}^T\mathbf{x}_i > 0$. Thus, logistic classifiers are a special form of linear classifiers.

THEOREM 3.21. *Assume $p_i \in (0, 1)$. There is a non-linear transform $\varphi(p_i)$ so that:*

$$\varphi(p_i) \sim \mathbf{w}^T\mathbf{x}$$

just in case Equation 3.8 holds.

PROOF. Assume:

$$p_i \sim \frac{1}{1 + \exp(-\mathbf{w}^T\mathbf{x}_i)}$$

Then:

$$\frac{1}{p_i} \sim 1 + \exp(-\mathbf{w}^T \mathbf{x}_i) \implies$$
$$\frac{1}{p_i} - 1 \sim \exp(-\mathbf{w}^T \mathbf{x}_i) \implies$$
$$\log\left(\frac{1-p_i}{p_i}\right) \sim -\mathbf{w}^T \mathbf{x}_i \implies$$
$$\log\left(\frac{p_i}{1-p_i}\right) \sim \mathbf{w}^T \mathbf{x}_i$$

The last transformation follows from the fact that $-\log(x) = \log(1/x)$. Let:

$$\varphi(p_i) = \log\left(\frac{p_i}{1-p_i}\right)$$

This completes the proof. □

DEFINITION 3.22. The transform φ is called the *logit transform*.

EXAMPLE 3.23. This example comes from data [McD85][1]. In [McD85], McDonald measured the allele (gene) frequencies at a certain point in the crustacean *Megalorchestia californiana* (see Figure 3.1). The (simplified) data are shown in the Table 1. We can apply

Figure 3.1. A picture of *Megalorchestia californiana* taken from http://www.biostathandbook.com/simplelogistic.html.

Latitude	Allele Proportion
48.1	0.748
45.2	0.577
44	0.521
43.7	0.483
43.5	0.628
37.8	0.259
36.6	0.304
34.3	0

Table 1. The allele proportion for *Megalorchestia californiana* at various latitudes for use in a logistic regression.

the logit transform to the data to obtain a new data set, on which we can use ordinary linear regression. This is shown in Table 2. The linear regression is shown in Figure 3.2. The fit is

[1]See http://www.biostathandbook.com/simplelogistic.html, from which this example is derived.

Latitude	Logit Transform (Allele Proportion)
48.1	1.08797389
45.2	0.310470087
44	0.084049444
43.7	-0.068026221
43.5	0.523646312
37.8	-1.051172564
36.6	-0.828321959
34.3	–

Table 2. The logit transform of the allele proportion. Notice when when the proportion is zero, we do not have a logit transform.

reasonable and provides good explanatory power for the distribution of alleles as a function of the latitude.

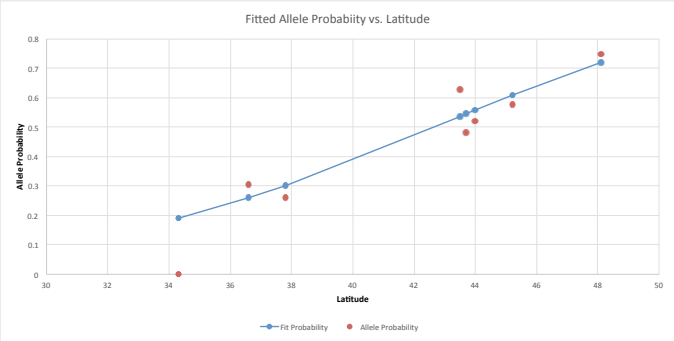


Figure 3.2. The linear regression of the logit transform of the allele data vs. the latitude is shown.

3. Logistic Regression - Likelihood Formulation

REMARK 3.24. A critical problem with executing a logistic regression by turning it into a linear regression is we cannot handle situations easily where $p_i = 0$ or $p_i = 1$. To mitigate this, we can instead approach the problem by maximizing a likelihood function. To construct the likelihood function, we again use $\tilde{y}_i \in \{-1, 1\}$ with $\tilde{y}_i = 1$ if $y_i = 1$ and $\tilde{y}_i = -1$ if $y_i = 0$.

DEFINITION 3.25 (Likelihood). Let $p(\mathbf{x}_i; \mathbf{w})$ denote the probability that $y_i = \tilde{y}_i = 1$. Then the likelihood function for the classifiers is:

$$\mathcal{L}(\mathbf{x}_i, \tilde{y}_i; \mathbf{w}) = \begin{cases} p(\mathbf{x}_i; \mathbf{w}) & \text{if } \tilde{y}_i = 1 \\ 1 - p(\mathbf{x}_i; \mathbf{w}) & \text{otherwise} \end{cases} \tag{3.10}$$

LEMMA 3.26. *If:*

$$p(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x}_i)}$$

then:

$$1 - p(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1 + \exp(\mathbf{w}^T \mathbf{x}_i)}$$

Thus:

$$\mathcal{L}(\mathbf{x}_i, \tilde{y}_i; \mathbf{w}) = \frac{1}{1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)}$$

EXERCISE 3.2. Prove Lemma 3.26.

DEFINITION 3.27 (Conditional Dataset Likelihood Function). Given a data set $(\mathbf{X}, \tilde{\mathbf{y}})$ the *logistic likelihood function* is:

$$\mathcal{L}(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w}) = \prod_{i=1}^{N} \mathcal{L}(\mathbf{x}_i, \tilde{y}_i; \mathbf{w}) = \prod_{i=1}^{N} \frac{1}{1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)}$$

REMARK 3.28. The following lemma follows at once from the laws of logarithms.

LEMMA 3.29. *The log-likelihood function is:*

$$\ell(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w}) = \log \mathcal{L}(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w}) = -\sum_{i=1}^{N} \log\left(1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)\right)$$

□

COROLLARY 3.30. *The log-likelihood function $\ell(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w})$ is concave. Thus the unconstrained optimization problem:*

$$\max_{\mathbf{w} \in \mathbb{R}^n} \ell(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w}) \tag{3.11}$$

has at least one global solution.

PROOF. The exponential function is convex and the composition of a linear function $-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i$ with a convex function is convex. Therefore, $\exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)$ is convex and by extension $1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)$ is convex. The logarithm function is non-decreasing on its domain (in fact, it's monotonically increasing). The composition of a non-decreasing function with a convex function is convex. Therefore $\log\left(1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)\right)$ is convex for each i. The sum of convex function is convex, therefore:

$$\sum_{i=1}^{N} \log\left(1 + \exp(-\tilde{y}_i \mathbf{w}^T \mathbf{x}_i)\right)$$

is convex. It follows at once that its negative is concave and therefore ℓ is concave. □

REMARK 3.31. There is no closed form solution to the Problem 3.11. However, since ℓ is concave, gradient ascent can be used to find an optimal solution, as we illustrate in the next example.

EXAMPLE 3.32. Consider the data set:

$$\mathbf{X} = \begin{bmatrix} 1 & -2 & -2 \\ 1 & -1 & -1 \\ 1 & 1 & 1 \\ 1 & 2 & 2 \end{bmatrix} \qquad \tilde{\mathbf{y}} = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

Then $\mathbf{w} = \langle w_0, w_1, w_2 \rangle$ and the log-likelihood function is:

$$\ell(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w}) = -\log\left(1 + \exp(w_0 - 2w_1 - 2w_2)\right) - \log\left(1 + \exp(w_0 - w_1 - w_2)\right) - \log\left(1 + \exp(-w_0 - w_1 - w_2)\right) - \log\left(1 + \exp(-w_0 - 2w_1 - 2w_2)\right)$$

Using a numerical solver to maximize $\ell(\mathbf{X}, \tilde{\mathbf{y}}; \mathbf{w})$, the resulting separating hyperplane (which is really used in the Logistic function) is shown in Figure 3.3. Notice, it is different than

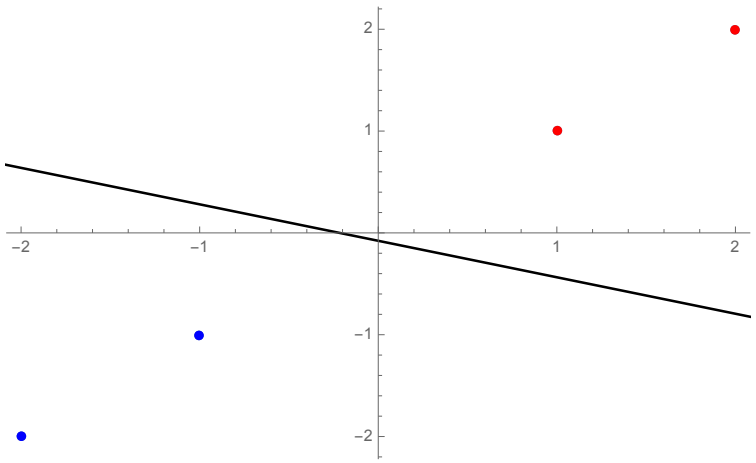


Figure 3.3. The linear separator that results from maximizing the log-likelihood function of the logistic regression.

what would be expected from a support vector machine. This difference, however, is due to the fact that ℓ is unbounded and concave, but $\nabla_{\mathbf{w}}\ell \to 0$ as $w_1, w_2 \to \infty$. Thus, numerical optimization terminates when tolerance is reached.

4. Performing Regression and Logistic Regression with Matlab

CHAPTER 4

Introduction to Neural Networks

1. Fundamental Definitions

DEFINITION 4.1 (Directed Graph). A *directed graph* $G = (V, E)$ is composed of a set of vertices V and a set of edges $E \subseteq V \times V$.

EXAMPLE 4.2. Directed graphs are usually visualized by making the vertices circles or dots and the edges arrows between the vertices. For example, consider the graph $V = \{v_1, v_2, v_3, v_4, v_5\}$ with edges $E = \{(v_1, v_3), (v_2, v_3), (v_3, v_4), (v_4, v_5)\}$. The visual representation of this directed graph is shown in Figure 4.1.

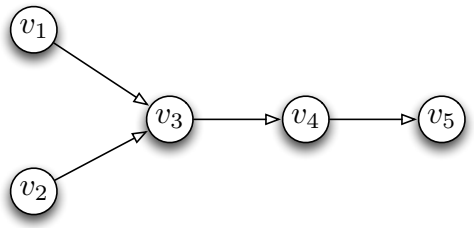


Figure 4.1. A directed graph consisting of five vertices and the edges connecting them visualized as arrows.

DEFINITION 4.3 (Self-Loop). An edge (v_1, v_2) in a directed graph $G = (V, E)$ is a *self-loop* if $v_1 = v_2$.

DEFINITION 4.4 (Simple Graph). A directed graph is simple if its edge set contains no self-loops.

EXAMPLE 4.5. The graph in Figure 4.1 is simple.

DEFINITION 4.6 (Walk/Path). A walk on a directed graph $G = (V, E)$ is a sequence $w = (v_1, e_1, v_2, \ldots, v_{n-1}, e_{n-1}, v_n)$ with $v_i \in V$ for $i = 1, \ldots, n$, $e_i \in E$ and and $e_i = (v_i, v_{i+1})$ for $i = 1, \ldots, n-1$.

DEFINITION 4.7 (Path / Cycle). Let $G = (V, E)$ be a directed graph. A walk $w = (v_1, e_1, v_2, \ldots, v_{n-1}, e_{n-1}, v_n)$ is a *path* if for each $i = 1, \ldots, n$, v_i occurs only once in the sequence w. A walk is a cycle if the walk $w' = (v_1, e_1, v_2, \ldots, v_{n-1})$ is a path and $v_1 = v_n$.

DEFINITION 4.8 (Acyclic). A directed graph $G = (V, E)$ is acyclic if there is no walk on G that is a cycle.

DEFINITION 4.9 (Walk Length). The length of a walk w in a directed graph G is the number of edges it contains.

EXAMPLE 4.10. A directed graph with a cycle is shown in Figure 4.2. On the other hand, the graph in Figure 4.1 is acyclic.

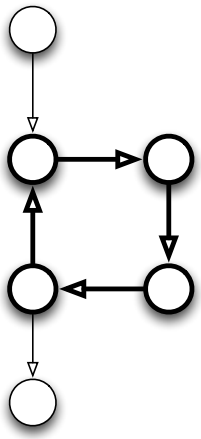


Figure 4.2. A directed graph with cycle.

DEFINITION 4.11 (In/Out Degree). Let $G = (V, E)$ be a directed graph. The *in-degree* of a vertex $v \in V$, denoted $\deg_i(v)$, is the the number of edges $e \in E$ such that $e = (u, v)$ for some $u \in V$. The *out-degree* of $v \in V$, denoted $\deg_o(v)$, is the number of edges in $e \in E$ such that $e = (v, u)$ for some $u \in V$.

EXAMPLE 4.12. In Figure 4.1, the in-degrees in order are $\{0, 0, 2, 1, 1\}$ while the out-degrees are $\{1, 1, 1, 0\}$.

DEFINITION 4.13 (k-Partite Graph). A graph $G = (V, E)$ is k-partite if:

$$V = V_1 \cup V_2 \cup \cdots \cup V_k$$

where $V_i \cap V_j = \emptyset$ for $i \neq j$ and if $e = (u, v)$, then there are i and j in $\{1, \ldots, k\}$ such that $u \in V_i$ and $v \in V_j$ and $i \neq j$.

REMARK 4.14. In a k-partite graph, edges only connect vertices in different partitions of the vertex set. They cannot connect to elements in the same partition. A 2-partite graph is usually called *bipartite*, while a 3-partite graph is called *tripartite*.

EXAMPLE 4.15. A tripartite graph is shown in Figure 4.3.

REMARK 4.16. It should be clear if $G = (V, E)$ is simple and $|V| = n$, then G is an n-partite graph. However, when discussing k-partite graphs, the smallest possible k is used that makes the definition true. Thus, we would not say that the graph in Figure 4.3 is 5-partite.

2. Neural Networks

DEFINITION 4.17. For $k \geq 1$, let $\mathcal{C}^k(\mathbb{R}^n)$ denote the set of all function with k derivatives where each derivative is *continuous*. A function that is $C^0(\mathbb{R}^n)$ is continuous, but not necessarily differentiable everywhere.

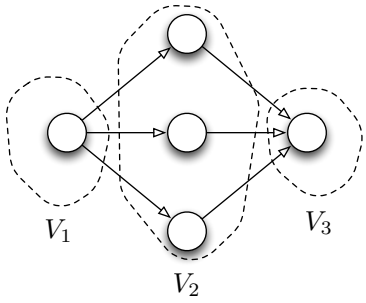


Figure 4.3. A tripartite graph whose vertex set is partitioned into 3 subsets.

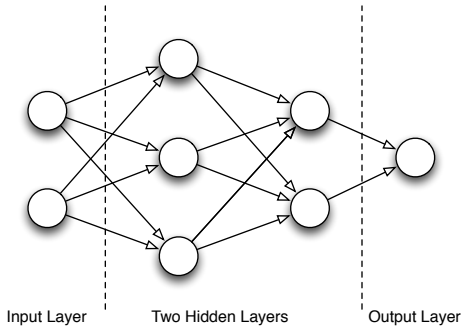


Figure 4.4. A neural network with two hidden layers and an input and output layer.

DEFINITION 4.18. A *neural network* is a tuple $\mathcal{N} = (G, F, I, O)$, where:

(1) $G = (V, E)$ is a directed graph whose vertices are called artificial neurons.
(2) The sets $I, O \subset V$ are the input and output neurons respectively and $I \cap O = \emptyset$.
(3) For each $v \in V \setminus (I \cup O)$, $F : v \mapsto f_v \in \mathcal{C}^1(\mathbb{R}^{\deg_i(v)})$. That is, to each neuron that is neither an input nor an output, we assign a differentiable function f_v that takes $\deg_i(v)$ inputs and returns an output.

REMARK 4.19. In general, the graph of a neural network is considered to be k-partite and each partition $V_1, \ldots, V_k$ is called a *layer*. As we'll see, when we discuss computation, neural networks always contain at least 3 layers:

(1) The first layer is called the *input layer*. Data from outside the network is fed into this layer.
(2) The last layer is called the *output layer*. Computed data from inside the neural network is returned at in these vertices.
(3) Other layers are called *hidden layers* because a user, when given a neural network, cannot see these layers.

Figure 4.4 illustrates a neural network with two hidden layers. As a rule, the size of the neural network is given by the number of hidden layers, rather than the number of total layers.

3. Computing with a Neural Network

DEFINITION 4.20 (Undefined). A quantity that is *undefined* is denoted by the symbol $\uparrow$.

REMARK 4.21. Some texts use the !, which we avoid to prevent confusion with factorial.

DEFINITION 4.22 (Neural Network Computation and State). Given a neural network $\mathcal{N} = (G, F, I, O)$ and a vector of data $\mathbf{x} \in \mathbb{R}^{|I|}$ the *state function* $Q : V \times \mathbb{Z}_+ \to \mathbb{R}$ of the neural network (describing the computation) is defined recursively with the following algorithm:

(1) Set $t = 0$. For all $v \in V$:

$$Q(v, 0) = \begin{cases} \mathbf{x}_j & \text{if } I = \{i_1, \ldots, i_{|I|}\} \text{ and } v = i_j \\ \uparrow & \textit{otherwise} \end{cases}$$

(2) Set $A = I$, this is the set of computing neurons. Set $B = \emptyset$.
(3) For each $v \in A$ and $u \in V$ such that $(v, u) \in E$, set $Q(u, t + 1) = f_u(Q(v, t))$. Set $B = B \cup \{u\}$.
(4) For all $v \in V \setminus B$: $Q(u, t + 1) = Q(u, t)$.
(5) Set $A = B$. Set $t = t + 1$. Goto 3.

REMARK 4.23. The algorithm defines computation by a neural network. If there is some t^* so that $Q(v, t) = Q(v, t^*)$ for all $t \geq t^*$, then the neural network computation terminates with this state. More generally, if there is some $Q^* : V \to \mathbb{R}^*$ so that:

$$\lim_{t \to \infty} Q(v, t) \to Q^*(v)$$

Then the neural network converges to this state and the outputs converge to $Q^*(o)$ for each $o \in O$.

DEFINITION 4.24 (Neural Network Function). Let $\mathcal{N} = (G, F, I, O)$ be a neural network with $O = \{o_1, \ldots, o_{|O|}\}$ and let $\mathbf{x} \in \mathbb{R}^{|I|}$ be an input data vector. The *function computed by* $\mathcal{N}$ is $f_{\mathcal{N}} : \mathbb{R}^{|I|} \times \mathbb{Z}_+ \to \mathbb{R}^{|O|}$ where if $f_{\mathcal{N}}(\mathbf{x}, t) = \mathbf{y}$, then $\mathbf{y}_i = Q(o_i, t)$.

REMARK 4.25. In this way, a recurrent neural network is a discrete time dynamical system that has an equilibrium point just in case Q^* exists for some input.

DEFINITION 4.26 (Feed Forward Neural Network). A neural network $\mathcal{N} = (G, F, I, O)$ is *feed forward* if G is acyclic.

THEOREM 4.27. *If $\mathcal{N} = (G, F, I, O)$ is a feed forward neural network, then $f_{\mathcal{N}}$ exists and can be computed in finite time.*

PROOF. Since there are no cycles in G, once $Q(v, t)$ is defined by the algorithm in Definition 4.22, it will never change. Furthermore, there is some h^* so that the length of any walk from a vertex in I to a vertex in O is at most H. Therefore, $Q(v, t)$ is defined for all $t \geq h^*$. Therefore Q^* exists. □

THEOREM 4.28 (Approximation by a Neural Network). *Let $f : [a, b] \subset \mathbb{R} \to \mathbb{R}$ be any continuous function with compact support; i.e., $[a, b]$ is a compact set in $\mathbb{R}$. Then for all ϵ there is feed forward neural network $\mathcal{N}$ with the property that:*

$$\int_a^b |f(x) - f_{\mathcal{N}}(x)|^2 \, dx < \epsilon \tag{4.1}$$

PROOF. Let $x_1 = a$ and $x_n = b$ and let $\Delta x = (b-a)/n$. Define:

$$\hat{f}(x) = \sum_{i=1}^{n-1} f\left(\frac{x_i + x_{i+1}}{2}\right)(H(x - x_i) - H(x - x_{i+1})) \tag{4.2}$$

where H is the Heaviside step function, which we know from Theorem 1.22 can be arbitrarily closely approximated by the differentiable function $S(x;\beta)$. For $x \in [x_i, x_{i+1}]$, we have

$$\hat{f}(x) = f\left(\frac{x_i + x_{i+1}}{2}\right)$$

Let:

$$A_i = \max_{x \in [x_i, x_{i+1}]} \left| f(x) - \hat{f}(x) \right|^2 \tag{4.3}$$

This value must exist by Weierstraß theorem. Thus:

$$\int_{x_i}^{x_{i+1}} \left| f(x) - \hat{f}(x) \right|^2 \, dx \leq A_i(x_{i+1} - x_i) \tag{4.4}$$

As $n \to \infty$, $A_i \to 0$ by the continuity of f. Thus, choose n so that $x_{i+1} - x_i < \sqrt{\epsilon/n}$ and $A_i < \sqrt{\epsilon/n}$ for all $i = 1, \ldots, n$. Then:

$$\int_a^b \left| f(x) - \hat{f}(x) \right|^2 dx < n\left(\frac{\epsilon}{n}\right) = \epsilon. \tag{4.5}$$

From Equation 4.2, let ρ_i be the coefficient of $H(x - x_i)$, which is:

$$\rho_i = \begin{cases} f\left(\frac{x_i + x_{i+1}}{2}\right) & \text{if } i = 1 \\ -f\left(\frac{x_{i-1} + x_i}{2}\right) & \text{if } i = n \\ f\left(\frac{x_i + x_{i+1}}{2}\right) - f\left(\frac{x_{i-1} + x_i}{2}\right) & \text{otherwise} \end{cases} \tag{4.6}$$

Define a neural network as follows: Let $I = \{v_0\}$ and $O = \{v_{n+2}\}$ be singleton input and output neurons. Let v_i be assigned the function $H(x - x_i)$ (or its differentiable variation $S(x;\beta)$). To v_{n+1} assign the function $\sum_{i=1}^{n} \rho_i y_i$, where y_i are the inputs. Assume the graph structure is the edge set containts all edges of the form (v_0, v_i), (v_i, v_{n+1}) for $i = 1, \ldots, n$ and $(v_{n+1}, v_{n+2}$. The resulting neural network is illustrated in Figure 4.5. This neural network computes the function $\hat{f}(x)$, which we have already shown can be defined to be arbitrarily close to the function $f(x)$. Thus we have identified a neural network whose about is arbitrarily close to the given function $f(x)$. This completes the proof. □

REMARK 4.29. The previous universal approximation theorem can be generalized to arbitrary, but bounded functions with compact support on $\mathbb{R}^n$. Thus, neural networks are *universal approximators*. It is worth noting that a feed forward neural network was sufficient.

4. Fitting a Neural Network from Data

REMARK 4.30. In the previous section, we showed that it was sufficient to assign exactly two kinds of differentiable functions to a neuron:

(1) The linear function $l : \mathbb{R}^n \to \mathbb{R}$ with:

$$l(\mathbf{x}) = w_0 + w_1\mathbf{x}_1 + \cdots + w_n\mathbf{x}_n = \mathbf{w}^T\mathbf{x} + w_0$$

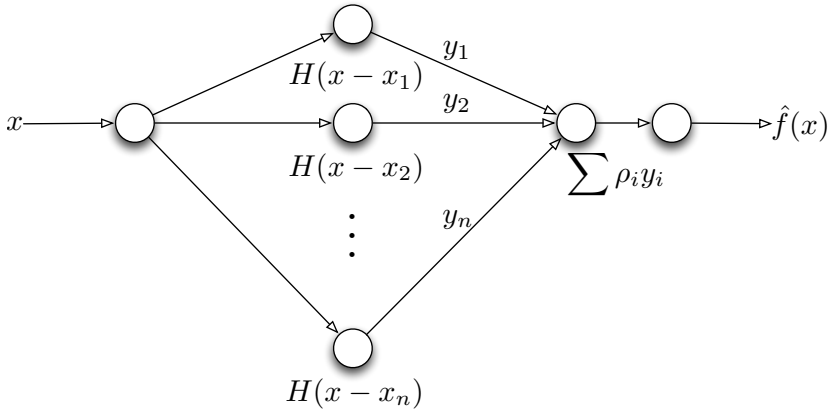


Figure 4.5. A simple feed forward neural network can approximate an arbitrary function.

(2) The sigmoid function of a linear functionation:

$$S(\mathbf{x}) = \frac{1}{1 + \exp\left(-\left(\mathbf{w}^T\mathbf{x} + w_0\right)\right)}$$

The function S is called an *activation function*. It can be some other function like $\tanh(\mathbf{w}^T\mathbf{x} + w_0)$ or it can be more exotic like a Gaussian function, which is common in convolutional neural networks. Thus fitting data to a neural network involves finding values for the parameters $\mathbf{w}$ and w_0 for each neuron in which they occur.

REMARK 4.31. To see how to fit a neural network, we first show that we have already seen at least two simple neural networks.

PROPOSITION 4.32. *Perceptron is a neural network.*

PROOF. The proof is by picture. Consider the neural network shown in Figure 4.6. Clearly, if we replace S with H (the Heaviside step function) or introduce a very high β

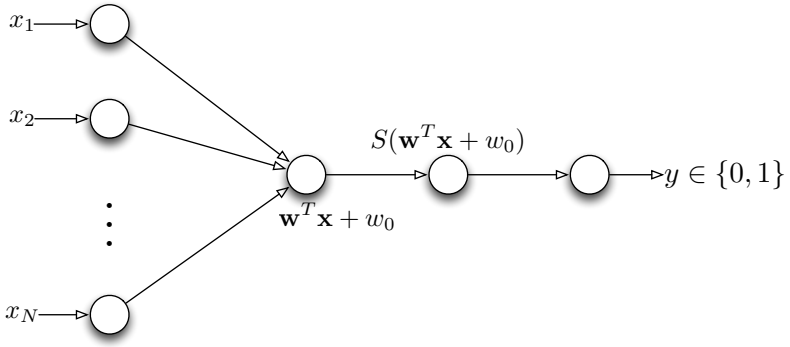


Figure 4.6. This simple neural network computes the perceptron.

multiplier as in $S(x, \beta)$, this neural network computes the perceptron function. □

COROLLARY 4.33. *The logistic regression is also computed by a neural network.* □

REMARK 4.34. Training a neural network is like performing non-linear regression. In particular, the neural network function computes $f_{\mathcal{N}}(\mathbf{x}_i; \mathbf{w}) \in \mathbb{R}^m$ when given input $\mathbf{x}_i \in \mathbb{R}^n$ from data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$. Here $\mathbf{w}$ is the set of *all* unknown parameters in the functions computed by the artificial neurons. Assuming there is a preferred output $\mathbf{y}_i \in \mathbb{R}^m$. The objective is then to solve:

$$\min_{\mathbf{w}} \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y}) = \frac{1}{2} \sum_{i=1}^{N} \|\mathbf{y}_i - f_{\mathcal{N}}(\mathbf{x}_i; \mathbf{w})\|^2 \tag{4.7}$$

This minimizes the square error on the output. Alternative *error* functions are also possible.

REMARK 4.35. Assume each neuron is assigned the function $S(\mathbf{w}^T\mathbf{x})$, where for simplicity, we assume that $\mathbf{x}$ has a 1 prepended so that w_0 does not need to be added. This can be accomplished by defining a set of neurons with constant output of 1. This is illustrated in Figure 4.7.

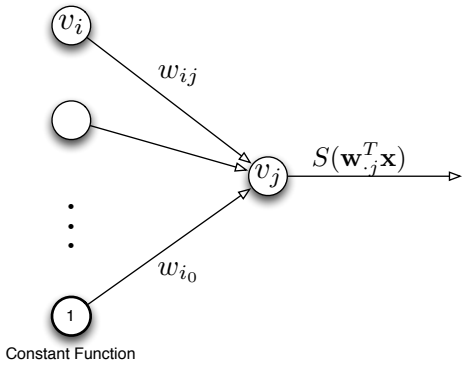


Figure 4.7. A neural network can be analyzed more efficiently by assigning the weights to the edges and assuming a subset of the neurons produce a constant function 1 that is multiplied by a w_{i_0} term to produce the constant offset.

THEOREM 4.36. *The gradient descent update minimizing $\mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})$ can be computed backwards from the output neurons, assuming a constant learning rate α.*

PROOF. Let $\mathcal{N}$ be a feed forward neural network and let $f_{\mathcal{N}}$ be the function is computes. Gradient descent applied to the error function $\mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})$ yields an update to w_{ij} of:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} - \alpha \frac{\partial \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})}{\partial w_{ij}} \tag{4.8}$$

The function $\mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})$ is layered, consequently the chain rule is required. Let $x_j = S(\mathbf{w}_{\cdot j}\mathbf{x})$ be the output from v_j (see Figure 4.7). Then:

$$\frac{\partial \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})}{\partial w_{ij}} = \frac{\partial \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})}{\partial x_j} \frac{\partial x_j}{\partial (\mathbf{w}_{\cdot j}^T \mathbf{x})} \frac{\partial \mathbf{w}_{\cdot j}^T \mathbf{x}}{\partial w_{ij}}$$

Computing from right to left:

$$\frac{\partial \mathbf{w}_{\cdot j}^T \mathbf{x}}{\partial w_{ij}} = x_i \tag{4.9}$$

the output of neuron v_i.

$$\frac{\partial x_j}{\partial (\mathbf{w}_{\cdot j}^T \mathbf{x})} = S'(\mathbf{w_j^T x}) = S(\mathbf{w_j^T x}) \left(1 - S(\mathbf{w_j^T x})\right) = x_j(1 - x_j) \tag{4.10}$$

We showed this in Equation 1.17. The last derivative must be computed recursively and *backwards* from the output neurons toward the input neurons. If v_j is an output neuron with output $\hat{y}_j$, then:

$$\frac{\partial \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})}{\partial x_j} = (\hat{y}_j - y_j), \tag{4.11}$$

because we are essentially just taking the derivative of $\frac{1}{2}(\hat{y}_j - y_j)^2$ with respect to $x_j = \hat{y}_j$. On the other hand, if v_j is a hidden layer before the output, then let $N_j = N_o(v_j)$ be the index set of neurons (vertices) that receive output from v_j. These are the out-neighbors of v_j. Then:

$$\frac{\partial \mathcal{E}(f_{\mathcal{N}}, \mathbf{X}, \mathbf{Y})}{\partial x_j} = \sum_{k \in N_j} \frac{\partial \mathcal{E}}{\partial x_k} \frac{\partial x_k}{\partial x_j} = \sum_{k \in N_j} \frac{\partial \mathcal{E}}{\partial x_k} \frac{\partial x_k}{\partial \mathbf{w}_{\cdot k}^T \mathbf{x}} \frac{\partial \mathbf{w}_{\cdot k}^T \mathbf{x}}{\partial x_j} = \sum_{k \in N_j} \frac{\partial \mathcal{E}}{\partial x_k} \frac{\partial x_k}{\partial \mathbf{w}_{\cdot k}^T \mathbf{x}} w_{jk} \tag{4.12}$$

Thus, we have shown the derivatives of the error function for variables in Layer l must be computed with derivates from Layer $l + 1$. Thus, if we denote:

$$\frac{\partial \mathcal{E}}{\partial w_{ij}} = \delta_j x_i \tag{4.13}$$

with:

$$\delta_j = \frac{\partial \mathcal{E}}{\partial x_j} \frac{\partial x_j}{\partial (\mathbf{w}_{\cdot j}^T \mathbf{x})} = \begin{cases} (x_j - y_j) x_j (1 - x_j) & \text{if } v_j \in O \\ \left(\sum_{k \in N_j} \delta_k w_{jk} \right) x_j (1 - x_j) \end{cases} \tag{4.14}$$

Thus Equation 4.8 can be rewritten as:

$$\Delta w_{ij}^{(t)} = w_{ij}^{(t+1)} - w_{ij}^{(t)} = -\alpha \frac{\partial \mathcal{E}}{\partial w_{ij}} = \begin{cases} -\alpha (x_j - y_j) x_j (1 - x_j) x_i & \text{if } v_j \in O \\ -\alpha \left(\sum_{k \in N_j} \delta_k w_{jk} \right) x_j (1 - x_j) x_i \end{cases} \tag{4.15}$$

This completes the proof. □

REMARK 4.37. This update rule is called *back propagation*. It is a generalization of the perceptron update rule. In fact, this update rule is exactly the perceptron update rule when restricted to the perceptron neural network.

APPENDIX A

Review of Matrix Properties

1. Fields and Matrices

DEFINITION A.1 (Group). A *group* is a pair $(S, \circ)$ where S is a set and $\circ : S \times S \to S$ is a binary operation so that:

(1) The binary operation $\circ$ is associative; that is, if s_1, s_2 and s_3 are in S, then $(s_1 \circ s_2) \circ s_3 = s_1 \circ (s_2 \circ s_3)$.
(2) There is a unique identity element $e \in S$ so that for all $s \in S$, $e \circ s = s \circ e = s$.
(3) For every element $s \in S$ there is an inverse element $s^{-1} \in S$ so that $s \circ s^{-1} = s^{-1} \circ s = e$.

If $\circ$ is commutative, that is for all $s_1, s_2 \in S$ we have $s_1 \circ s_2 = s_2 \circ s_1$, then $(S, \circ)$ is called a *commutative group* (or *abelian group*).

EXAMPLE A.2. This course is *not* about group theory. If you're interested in groups in the more abstract sense, it's worth considering taking Math 435, which is all about abstract algebra. One of the simplest examples of a group is the set of integers $\mathbb{Z}$ under the binary operation of addition.

DEFINITION A.3 (Sub-Group). Let $(S, \circ)$ be a group. A *subgroup* of $(S, \circ)$ is a group $(T, \circ)$ so that $T \subseteq S$. The subgroup $(T, \circ)$ shares the identify of the group $(S, \circ)$.

EXAMPLE A.4. Consider the group $(\mathbb{Z}, +)$. If $2\mathbb{Z}$ is the set of even integers, then $(2\mathbb{Z}, +)$ is a subgroup of $(\mathbb{Z}, +)$ because that even integers are closed under addition.

DEFINITION A.5 (Number Field). A *field* (or number field) is a tuple $(S, +, \cdot, 0, 1)$ where:

(1) $(S, +)$ is a commutative group with unit 0,
(2) $(S \setminus \{0\}, \cdot)$ is a commutative group with unit 1
(3) The operation $\cdot$ *distributes* over the operation $+$ so that if a_1, a_2, and a_3 are elements of F, then $a_1 \cdot (a_2 + a_3) = a_1 \cdot a_2 + a_1 \cdot a_3$.

EXAMPLE A.6. The archetypal example of a field is the field of real numbers $\mathbb{R}$ with addition and multiplication playing the expected roles. Another common field is the field of complex numbers $\mathbb{C}$ (numbers of the form $a + bi$ with $i = \sqrt{-1}$ the imaginary unit) with their addition and multiplication rules defined as expected.

DEFINITION A.7 (Matrix). An $m \times n$ matrix is a rectangular array of values (*scalars*), drawn from a field. If F is the field, we write $F^{m \times n}$ to denote the set of $m \times n$ matrices with entries drawn from F.

REMARK A.8. For the remainder of this appendix, we will use the field $\mathbb{R}$.

2. Basic Matrix Operations

DEFINITION A.9 (Matrix Addition). If $\mathbf{A}$ and $\mathbf{B}$ are both in $\mathbb{R}^{m\times n}$, then $\mathbf{C} = \mathbf{A} + \mathbf{B}$ is the matrix sum of $\mathbf{A}$ and $\mathbf{B}$ and

$$\mathbf{C}_{ij} = \mathbf{A}_{ij} + \mathbf{B}_{ij} \text{ for } i = 1, \ldots, m \text{ and } j = 1, \ldots, n \tag{A.1}$$

EXAMPLE A.10.

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix} \tag{A.2}$$

DEFINITION A.11 (Row/Column Vector). A $1 \times n$ matrix is called a *row vector*, and a $m \times 1$ matrix is called a *column vector*. For the remainder of these notes, every vector will be thought of **column vector** unless otherwise noted. A column vector $\mathbf{x}$ in $\mathbb{R}^{n\times 1}$ (or $\mathbb{R}^n$) is: $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$.

It should be clear that any row of matrix $\mathbf{A}$ could be considered a row vector in $\mathbb{R}^n$ and any column of $\mathbf{A}$ could be considered a column vector in $\mathbb{R}^m$.

DEFINITION A.12 (Standard Euclidean Norm). Let $\mathbf{x} \in \mathbb{R}^n$ be a vector. The standard Euclidean norm is:

$$\|\mathbf{x}\| = \sqrt{x_1^2 + \cdots + x_n^2} \tag{A.3}$$

DEFINITION A.13 (Unit Vector). A vector $\mathbf{x}$ is a unit vector if $\|\mathbf{x}\| = 1$, where 1 is the unit in the field.

DEFINITION A.14 (Dot Product). Recall that if $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ are two n-dimensional vectors, then the *dot product* (*scalar product*) is:

$$\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} x_i y_i \tag{A.4}$$

where x_i is the i^{th} component of the vector $\mathbf{x}$. Clearly if $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n\times 1}$, then

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y} \tag{A.5}$$

where $\mathbf{x}^T \in \mathbb{R}^{1\times n}$ is the transpose of $\mathbf{x}$ when treated as a matrix.

EXERCISE A.1. Show that $\|\mathbf{x}\|^2 = \mathbf{x}^T\mathbf{x} = \mathbf{x}^T\mathbf{I}_n\mathbf{x}$ for $\mathbf{x} \in \mathbb{R}^n$.

LEMMA A.15. *Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and let θ be the angle between $\mathbf{x}$ and $\mathbf{y}$, then*

$$\mathbf{x} \cdot \mathbf{y} = ||\mathbf{x}||||\mathbf{y}|| \cos\theta \tag{A.6}$$

□

REMARK A.16. The preceding lemma can be proved using the *law of cosines* from trigonometry. The following small lemma follows and is is proved as Theorem 1 of [MT03]:

LEMMA A.17. *Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. Then the following hold:*

1. *The angle between $\mathbf{x}$ and $\mathbf{y}$ is less than $\pi/2$ (i.e., acute) iff $\mathbf{x} \cdot \mathbf{y} > 0$.*
2. *The angle between $\mathbf{x}$ and $\mathbf{y}$ is exactly $\pi/2$ (i.e., the vectors are orthogonal) iff $\mathbf{x} \cdot \mathbf{y} = 0$.*
3. *The angle between $\mathbf{x}$ and $\mathbf{y}$ is greater than $\pi/2$ (i.e., obtuse) iff $\mathbf{x} \cdot \mathbf{y} < 0$.*

□

DEFINITION A.18. Two vectors $\mathbf{x}$ and $\mathbf{y}$ are *orthogonal* if $\mathbf{x} \cdot \mathbf{y} = 0$. (Here 0 is the zero in the field over which the vectors are defined.)

DEFINITION A.19 (Orthonormal Vectors). If two vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal and both vectors have norm equal to 1 (the unit in the field), then they are said to be *orthonormal*.

EXERCISE A.2. Show that if $\mathbf{x} \in \mathbb{R}^n$, then $\mathbf{x}/\left\|\mathbf{x}\right\|$ is a unit vector.

LEMMA A.20 (Schwartz Inequality). *Let* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. *Then:*

$$(\mathbf{x}^T\mathbf{y})^2 \leq (\mathbf{x}^T\mathbf{x}) \cdot (\mathbf{y}^T\mathbf{y}) \tag{A.7}$$

This is equivalent to:

$$(\mathbf{x}^T\mathbf{y})^2 \leq ||\mathbf{x}||^2||\mathbf{y}||^2 \tag{A.8}$$

DEFINITION A.21 (Matrix Multiplication). If $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times p}$, then $\mathbf{C} = \mathbf{AB}$ is the *matrix product* of $\mathbf{A}$ and $\mathbf{B}$ and

$$\mathbf{C}_{ij} = \mathbf{A}_{i\cdot} \cdot \mathbf{B}_{\cdot j} \tag{A.9}$$

Note, $\mathbf{A}_{i\cdot} \in \mathbb{R}^{1\times n}$ (an n-dimensional vector) and $\mathbf{B}_{\cdot j} \in \mathbb{R}^{n\times 1}$ (another n-dimensional vector), thus making the dot product meaningful.

EXAMPLE A.22.

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1(5)+2(7) & 1(6)+2(8) \\ 3(5)+4(7) & 3(6)+4(8) \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix} \tag{A.10}$$

DEFINITION A.23 (Matrix Transpose). If $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a $m \times n$ matrix, then the *transpose* of $\mathbf{A}$ dented $\mathbf{A}^T$ is an $m \times n$ matrix defined as:

$$\mathbf{A}^T_{ij} = \mathbf{A}_{ji} \tag{A.11}$$

EXAMPLE A.24.

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^T = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \tag{A.12}$$

The matrix transpose is a particularly useful operation and makes it easy to transform column vectors into row vectors, which enables multiplication. For example, suppose $\mathbf{x}$ is an $n \times 1$ column vector (i.e., $\mathbf{x}$ is a vector in $\mathbb{R}^n$) and suppose $\mathbf{y}$ is an $n \times 1$ column vector. Then:

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T\mathbf{y} \tag{A.13}$$

EXERCISE A.3. Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$. Use the definitions of matrix addition and transpose to prove that:

$$(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T \tag{A.14}$$

[Hint: If $\mathbf{C} = \mathbf{A} + \mathbf{B}$, then $\mathbf{C}_{ij} = \mathbf{A}_{ij} + \mathbf{B}_{ij}$, the element in the (i, j) position of matrix $\mathbf{C}$. This element moves to the (j, i) position in the transpose. The (j, i) position of $\mathbf{A}^T + \mathbf{B}^T$ is $\mathbf{A}^T_{ji} + \mathbf{B}^T_{ji}$, but $\mathbf{A}^T_{ji} = \mathbf{A}_{ij}$. Reason from this point.]

EXERCISE A.4. Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$. Prove by example that $\mathbf{AB} \neq \mathbf{BA}$; that is, matrix multiplication is *not commutative*. [Hint: Almost any pair of matrices you pick (that can be multiplied) will not commute.]

EXERCISE A.5. Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ and let, $\mathbf{B} \in \mathbb{R}^{n\times p}$. Use the definitions of matrix multiplication and transpose to prove that:

$$(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T \tag{A.15}$$

[Hint: Use similar reasoning to the hint in Exercise A.3. But this time, note that $\mathbf{C}_{ij} = \mathbf{A}_{i\cdot} \cdot \mathbf{B}_{\cdot j}$, which moves to the (j, i) position. Now figure out what is in the (j, i) position of $\mathbf{B}^T\mathbf{A}^T$.]

Let $\mathbf{A}$ and $\mathbf{B}$ be two matrices with the same number of rows (so $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{B} \in \mathbb{R}^{m\times p}$). Then the augmented matrix $[\mathbf{A}|\mathbf{B}]$ is:

$$\left[\begin{array}{cccc|cccc} a_{11} & a_{12} & \dots & a_{1n} & b_{11} & b_{12} & \dots & b_{1p} \\ a_{21} & a_{22} & \dots & a_{2n} & b_{21} & b_{22} & \dots & b_{2p} \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_{m1} & b_{m2} & \dots & b_{mp} \end{array}\right] \tag{A.16}$$

Thus, $[\mathbf{A}|\mathbf{B}]$ is a matrix in $\mathbb{R}^{m\times(n+p)}$.

EXAMPLE A.25. Consider the following matrices:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

Then $[\mathbf{A}|\mathbf{B}]$ is:

$$[\mathbf{A}|\mathbf{B}] = \left[\begin{array}{cc|c} 1 & 2 & 7 \\ 3 & 4 & 8 \end{array}\right]$$

EXERCISE A.6. By analogy define the augmented matrix $\left[\frac{\mathbf{A}}{\mathbf{B}}\right]$. Note, this is **not** a fraction. In your definition, identify the appropriate requirements on the relationship between the number of rows and columns that the matrices must have. [Hint: Unlike $[\mathbf{A}|\mathbf{B}]$, the number of rows don't have to be the same, since your concatenating on the rows, not columns. There should be a relation between the numbers of columns though.]

3. Special Matrices and Vectors

DEFINITION A.26 (Identify Matrix). The $n \times n$ *identify matrix* is:

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \tag{A.17}$$

When it is clear from context, we may simply write $\mathbf{I}$ and omit the subscript n.

EXERCISE A.7. Let $\mathbf{A} \in \mathbb{R}^{n\times n}$. Show that $\mathbf{AI}_n = \mathbf{I}_n\mathbf{A} = \mathbf{A}$. Hence, $\mathbf{I}$ is an identify for the matrix multiplication operation on square matrices. [Hint: Do the multiplication out long hand.]

DEFINITION A.27 (Standard Basis Vector). The standard basis vector $\mathbf{e}_i \in \mathbb{R}^n$ is:

$$\mathbf{e}_i = \left\langle \underbrace{0, 0, \ldots,}_{i-1}, 1, \underbrace{0, \ldots, 0}_{n-i-1} \right\rangle$$

Note, this definition is only valid for $n \geq i$. Further the standard basis vector $\mathbf{e}_i$ is also the i^{th} row or column of $\mathbf{I}_n$.

DEFINITION A.28 (Unit and Zero Vectors). The vector $\mathbf{1} \in \mathbb{R}^n$ is the *one vector* $\mathbf{1} = \langle 1, 1, \ldots, 1 \rangle$. Similarly, the *zero vector* $\mathbf{0} = \langle 0, 0, \ldots, 0 \rangle \in \mathbb{R}^n$. We assume that the length of $\mathbf{e}$ and $\mathbf{0}$ will be determined from context.

DEFINITION A.29 (Symmetric Matrix). Let $\mathbf{M} \in \mathbb{R}^{n \times n}$ be a matrix. The matrix $\mathbf{M}$ is symmetric if $\mathbf{M} = \mathbf{M}^T$.

DEFINITION A.30 (Invertible Matrix). Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a square matrix. If there is a matrix $\mathbf{A}^{-1}$ such that

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_n \tag{A.18}$$

then matrix $\mathbf{A}$ is said to be *invertible* (or *nonsingular*) and $\mathbf{A}^{-1}$ is called its *inverse*. If $\mathbf{A}$ is not invertible, it is called a *singular* matrix.

DEFINITION A.31 (Diagonal Matrix). A *diagonal matrix* is a (square) matrix with the property that $\mathbf{D}_{ij} = 0$ for $i \neq j$ and $\mathbf{D}_{ii}$ may take any value in the field on which $\mathbf{D}$ is defined.

REMARK A.32. Thus, a diagonal matrix has (usually) non-zero entries only on its main diagonal. These matrices will play a critical roll in our analysis.

4. Matrix Definiteness

DEFINITION A.33 (Definiteness).
(1) A matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is *positive semi-definite* if for all $\mathbf{x} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^T\mathbf{M}\mathbf{x} \geq 0$.
(2) The matrix $\mathbf{M}$ is *positive definite* if for all $\mathbf{x} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^T\mathbf{M}\mathbf{x} > 0$.
(3) The matrix $\mathbf{M}$ is *negative definite* if $-M$ is positive definite. That is, if $\mathbf{x}^T\mathbf{M}\mathbf{x} < 0$ for all $\mathbf{x} \in \mathbb{R}^n$.
(4) The matrix $\mathbf{M}$ *negative semi-definite* if $-M$ is positive semi-definite.

If $\mathbf{M}$ satisfies none of these properties, then $\mathbf{M}$ is *indefinite*.

REMARK A.34. We note that this is not the most general definition of matrix definiteness. In general, matrix definiteness can be defined for complex matrices and has specialization to Hermitian matrices.

REMARK A.35. Positive semi-definiteness is also called non-negative definiteness and negative semi-definiteness is also called non-positive definiteness.

EXAMPLE A.36. Consider the matrix:

$$\mathbf{M} = \begin{bmatrix} \alpha, 0 \\ 0, \beta \end{bmatrix}$$

where $\alpha, \beta > 0$. This matrix is positive definite. To see this, choose any vector $\mathbf{x} = \langle x_1, x_2 \rangle$ and compute:

$$\mathbf{x}^T \mathbf{M} \mathbf{x} = \alpha x_1^2 + \beta x_2^2$$

This quantity is always positive, thus $\mathbf{M}$ must be positive definite.

LEMMA A.37. *A matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is positive definite if and only if for every vector $\mathbf{x} \in \mathbb{R}^n$, there is an $\alpha \in \mathbb{R}_+$ (that is $\alpha > 0$) such that $\mathbf{x}^T \mathbf{M} \mathbf{x} > \alpha \mathbf{x}^T \mathbf{x}$.*

EXERCISE A.8. Prove Lemma A.37.

5. Permutations[1]

DEFINITION A.38 (Permutation / Permutation Group). A *permutation* on a set $V = \{1, \ldots, n\}$ of n elements is a bijective mapping f from V to itself. A *permutation group* on a set V is a set of permutations with the binary operation of functional composition.

EXAMPLE A.39. Consider the set $V = \{1, 2, 3, 4\}$. A permutation on this set that maps 1 to 2 and 2 to 3 and 3 to 1 can be written as: $(1, 2, 3)(4)$ indicating the cyclic behavior that $1 \to 2 \to 3 \to 1$ and 4 is fixed. In general, we write $(1, 2, 3)$ instead of $(1, 2, 3)(4)$ and suppress any elements that do not move under the permutation.

For the permutation taking 1 to 3 and 3 to 1 and 2 to 4 and 4 to 2 we write $(1, 3)(2, 4)$ and say that this is the *product* of $(1, 3)$ and $(2, 4)$. When determining the impact of a permutation on a number, we read the permutation from right to left. Thus, if we want to determine the impact on 2, we read from right to left and see that 2 goes to 4. By contrast, if we had the permutation: $(1, 3)(1, 2)$ then this permutation would take 2 to 1 first and then 1 to 3 thus 2 would be mapped to 3. The number 1 would be first mapped to 2 and then stop. The number 3 would be mapped to 1. Thus we can see that $(1, 3)(1, 2)$ has the same action as the permutation $(1, 2, 3)$.

DEFINITION A.40 (Symmetric Group). Consider a set V with n elements in it. The permutation group S_n contains every possible permutation of the set with n elements.

EXAMPLE A.41. Consider the set $V = \{1, 2, 3\}$. The symmetric group on V is the set S_3 and it contains the permutations:

(1) The identity: (1)(2)(3)
(2) (12)(3)
(3) (13)(2)
(4) (23)(1)
(5) (123)
(6) (132)

PROPOSITION A.42. *For each n, $|S_n| = n!$.*

EXERCISE A.9. Prove Proposition A.42

DEFINITION A.43 (Transposition). A permutation of the form (a_1, a_2) is called a transposition.

[1]This section is used purely to understand the general definition of the determinant of a matrix.

THEOREM A.44. *Every permutation can be expressed as the product of transpositions.*

PROOF. Consider the permutation $(a_1, a_2, \ldots, a_n)$. We may write:

$$(a_1, a_2, \ldots, a_n) = (a_1, a_n)(a_1, a_{n-1}) \cdots (a_1, a_2) \tag{A.19}$$

Observe the effect of these two permutations on a_i. For $i \neq 1$ and $i \neq n$, then reading from right to left (as the permutation is applied) we see that a_i maps to a_1, which reading further right to left is mapped to a_{i+1} as we expect. If $i = 1$, then a_1 maps to a_2 and there is no further mapping. Finally, if $i = n$, then we read left to right to the only transposition containing a_n and see that a_n maps to a_1. Thus Equation A.19 holds. This completes the proof. □

REMARK A.45. The following theorem is useful for our work on matrices in the second part of this chapter, but its proof is outside the scope of these notes. The interested reader can see Chapter 2.2 of [**Fra99**].

THEOREM A.46. *No permutation can be expressed as both a product of an even and an odd number of transpositions.* □

DEFINITION A.47 (Even/Odd Permutation). Let $\sigma \in S_n$ be a permutation. If σ can be expressed as an *even* number of transpositions, then it is *even*, otherwise σ is *odd*. The *signature* of the permutation is:

$$\text{sgn}(\sigma) = \begin{cases} -1 & \sigma \text{ is odd} \\ 1 & \sigma \text{ is even} \end{cases} \tag{A.20}$$

6. Eigenvalues and Eigenvectors

DEFINITION A.48 (Determinant). Let $\mathbf{M} \in \mathbb{R}^{n \times n}$. The *determinant* of $\mathbf{M}$ is:

$$\det(\mathbf{A}) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \prod_{i=1}^{n} \mathbf{A}_{i\sigma(i)} \tag{A.21}$$

Here $\sigma \in S_n$ represents a permutation over the set $\{1, \ldots, n\}$ and $\sigma(i)$ represents the value to which i is mapped under σ.

EXAMPLE A.49. Consider an arbitrary 2×2 matrix:

$$\mathbf{M} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

There are only two permutations in the set S_2: the identity permutation (which is even) and the transposition $(1, 2)$ which is odd. Thus, we have:

$$\det(\mathbf{M}) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = \mathbf{M}_{11}\mathbf{M}_{22} - \mathbf{M}_{12}\mathbf{M}_{21} = ad - bc$$

This is the formula that one would expect from a course in matrices (like Math 220).

DEFINITION A.50 (Eigenvalue and (Right) Eigenvector). Let $\mathbf{M} \in \mathbb{R}^{n \times n}$. An eigenvalue, eigenvector pair $(\lambda, \mathbf{x})$ is a scalar and $n \times 1$ vector such that:

$$\mathbf{M}\mathbf{x} = \lambda\mathbf{x} \tag{A.22}$$

REMARK A.51. A *left eigenvector* is defined analogously with $\mathbf{x}^T\mathbf{M} = \lambda\mathbf{x}^T$, when $\mathbf{x}$ is considered a column vector. We will deal exclusively with right eigenvectors and hence when we say "eigenvector" we mean a right eigenvector.

DEFINITION A.52 (Characteristic Polynomial). If $\mathbf{M} \in \mathbb{R}^{n\times n}$ then its *characteristic polynomial* is:

$$\det(\lambda\mathbf{I}_n - \mathbf{M}) \tag{A.23}$$

REMARK A.53. The following theorem is useful for computing eigenvalues of small matrices and defines the characteristic polynomial for a matrix. Its proof is outside the scope of these notes, but would occur in a Math 436 class. (See Chapter 8.2 of [**Lan87**].)

THEOREM A.54. *A value λ is an eigenvalue for $\mathbf{M} \in \mathbb{R}^{n\times n}$ if and only if it satisfies the characteristic equation:*

$$\det(\lambda\mathbf{I}_n - \mathbf{M}) = 0$$

Furthermore, $\mathbf{M}$ and $\mathbf{M}^T$ share eigenvalues. □

EXAMPLE A.55. Consider the matrix:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

The characteristic polynomial is computed as:

$$\det(\lambda\mathbf{I}_n - \mathbf{M}) = \begin{vmatrix} \lambda - 1 & 0 \\ 0 & \lambda - 2 \end{vmatrix} = (\lambda - 1)(\lambda - 2) - 0 = 0$$

Thus the characteristic polynomial for this matrix is:

$$\lambda^2 - 3\lambda + 2 \tag{A.24}$$

The roots of this polynomial are $\lambda_1 = 1$ and $\lambda_2 = 2$. Using these eigenvalues, we can compute eigenvectors:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{A.25}$$

$$\mathbf{x}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{A.26}$$

and observe that:

$$\mathbf{M}\mathbf{x}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = 1\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \lambda_1\mathbf{x}_1 \tag{A.27}$$

and

$$\mathbf{M}\mathbf{x}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = 2\begin{bmatrix} 0 \\ 1 \end{bmatrix}\lambda_2\mathbf{x}_2 \tag{A.28}$$

as required. Computation of eigenvalues and eigenvectors is usually accomplished by computer and several algorithms have been developed. Those interested readers should consult (e.g.) [**Dat95**].

REMARK A.56. You can use your calculator to return the eigenvalues and eigenvectors of a matrix, as well as several software packages, like Matlab and Mathematica.

REMARK A.57. It is important to remember that eigenvectors are unique *up to scale*. That is, if $\mathbf{M}$ is a square matrix and $(\lambda, \mathbf{x})$ is an eigenvalue eigenvector pair for $\mathbf{M}$, then so is $(\lambda, \alpha\mathbf{x})$ for $\alpha \neq 0$. This is because:

$$\mathbf{M}\mathbf{x} = \lambda\mathbf{x} \implies \mathbf{M}(\alpha\mathbf{x}) = \lambda(\alpha\mathbf{x}) \tag{A.29}$$

DEFINITION A.58 (Degenerate Eigenvalue). An eigenvalue is *degenerate* if it is a *multiple root* of the characteristic polynomial. The multiplicity of the root is the multiplicity of the eigenvalue.

EXAMPLE A.59. Consider the identify matrix $\mathbf{I}_2$. It has characteristic polynomial $(\lambda-1)^2$, which has one multiple root 1. Thus $\lambda = 1$ is a degenerate eigenvalue for this matrix. However, this matrix does have two eigenvectors $[1\ \ 0]^T$ and $[0\ \ 1]^T$.

7. Linear Combinations, Span, Linear Independence

DEFINITION A.60. Let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be vectors in $\in \mathbb{R}^n$ and let $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$ be scalars. Then

$$\alpha_1\mathbf{x}_1 + \cdots + \alpha_m\mathbf{x}_m \tag{A.30}$$

is a *linear combination* of the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$.

Clearly, any linear combination of vectors in $\mathbb{R}^n$ is also a vector in $\mathbb{R}^n$.

DEFINITION A.61 (Span). Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ be a set of vectors in $\in \mathbb{R}^n$, then the span of $\mathcal{X}$ is the set:

$$\text{span}(\mathcal{X}) = \{\mathbf{y} \in \mathbb{R}^n | \mathbf{y} \text{ is a linear combination of vectors in } \mathcal{X}\} \tag{A.31}$$

DEFINITION A.62 (Linear Independence). Let $\mathbf{x}_1, \ldots, \mathbf{x}_m$ be vectors in $\in \mathbb{R}^n$. The vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$ are *linearly dependent* if there exists $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$, not all zero, such that

$$\alpha_1\mathbf{x}_1 + \cdots + \alpha_m\mathbf{x}_m = \mathbf{0} \tag{A.32}$$

If the set of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$ is not linearly dependent, then they are *linearly independent* and Equation A.32 holds just in case $\alpha_i = 0$ for all $i = 1, \ldots, n$.

EXERCISE A.10. Consider the vectors $\mathbf{x}_1 = \langle 0, 0\rangle$ and $\mathbf{x}_2 = \langle 1, 0\rangle$. Are these vectors linearly independent? Explain why or why not.

EXAMPLE A.63. In $\mathbb{R}^3$, consider the vectors:

$$\mathbf{x}_1 = \begin{bmatrix}1\\1\\0\end{bmatrix},\ \mathbf{x}_2 = \begin{bmatrix}1\\0\\1\end{bmatrix},\ \mathbf{x}_3 = \begin{bmatrix}0\\1\\1\end{bmatrix}$$

We can show these vectors are linearly independent: Suppose there are values $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}$ such that

$$\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \alpha_3\mathbf{x}_3 = 0$$

Then:

$$\begin{bmatrix}\alpha_1\\\alpha_1\\0\end{bmatrix} + \begin{bmatrix}\alpha_2\\0\\\alpha_2\end{bmatrix}\ \begin{bmatrix}0\\\alpha_3\\\alpha_3\end{bmatrix} = \begin{bmatrix}\alpha_1+\alpha_2\\\alpha_1+\alpha_3\\\alpha_2+\alpha_3\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}$$

Thus we have the system of linear equations:

$$\begin{aligned} \alpha_1 \quad +\alpha_2 \qquad &= 0 \\ \alpha_1 \qquad + \alpha_3 \quad &= 0 \\ \alpha_2 + \alpha_3 \quad &= 0 \end{aligned}$$

which can be written as the matrix expression:

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

This is just a simple matrix equation, but note that the three vectors we are focused on: $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$, have become the *columns* of the matrix on the left-hand-side. We can use Gauss-Jordan elimination to solve this matrix equation yielding: $\alpha_1 = \alpha_2 = \alpha_3 = 0$. Thus these vectors are linearly independent.

REMARK A.64. It is worthwhile to note that the zero vector $\mathbf{0}$ makes any set of vectors a linearly dependent set.

EXERCISE A.11. Prove the remark above.

EXERCISE A.12. Show that the vectors

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \ \mathbf{x}_2 = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}, \ \mathbf{x}_3 = \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix}$$

are *not* linearly independent. [Hint: Following the example, create a matrix whose columns are the vectors in question and solve a matrix equation with right-hand-side equal to zero. Using Gauss-Jordan elimination, show that a zero row results and thus find the infinite set of values solving the system.]

REMARK A.65. So far we have only given examples and exercises in which the number of vectors was equal to the dimension of the space they occupied. Clearly, we could have, for example, 3 linearly independent vectors in 4 dimensional space. We illustrate this case in the following example.

EXAMPLE A.66. Consider the vectors:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \ \mathbf{x}_2 = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

Determining linear independence requires us to solve the matrix equation:

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

The augmented matrix:

$$\left[\begin{array}{cc|c} 1 & 4 & 0 \\ 2 & 5 & 0 \\ 4 & 6 & 0 \end{array}\right]$$

represents the matrix equation. Using Gauss-Jordan elimination yields:

$$\left[\begin{array}{cc|c} 1 & 4 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{array}\right]$$

This implies that the following system of equations:

$$\begin{aligned} \alpha_1 + 4\alpha_2 &= 0 \\ \alpha_2 &= 0 \\ 0\alpha_1 + 0\alpha_2 &= 0 \end{aligned}$$

The last equation is tautological (true regardless of the values of α_1 and α_2). The second equation implies $\alpha_2 = 0$. Using this value in first equation implies that $\alpha_1 = 0$. This is the unique solution to the problem and thus the vectors are linearly independent.

The following theorem is related to the example above. It's proof is outside the scope of the course. It should be taught in a Linear Algebra course (Math 436). Proofs can be found in most Linear Algebra textbooks. Again, see [**Lan87**] (Theorem 3.1) for a proof using vector spaces.

THEOREM A.67. *Let $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$. If $m > n$, then the vectors are linearly dependent.*

8. Basis

DEFINITION A.68 (Basis). Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ be a set of vectors in $\mathbb{R}^n$. The set $\mathcal{X}$ is called a *basis* of $\mathbb{R}^n$ if $\mathcal{X}$ is a linearly independent set of vectors and every vector in $\mathbb{R}^n$ is in the span of $\mathcal{X}$. That is, for any vector $\mathbf{w} \in \mathbb{R}^n$ we can find scalar values $\alpha_1, \ldots, \alpha_m$ such that

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i \tag{A.33}$$

EXAMPLE A.69. We can show that the vectors:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \ \mathbf{x}_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \ \mathbf{x}_3 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

form a basis of $\mathbb{R}^3$. We already know that the vectors are linearly independent. To show that $\mathbb{R}^3$ is in their span, chose an arbitrary vector in $\mathbb{R}^m$: $\langle a, b, c \rangle$. Then we hope to find coefficients α_1, α_2 and α_3 so that:

$$\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \alpha_3\mathbf{x}_3 = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Expanding this, we must find α_1, α_2 and α_3 so that:

$$\begin{bmatrix} \alpha_1 \\ \alpha_1 \\ 0 \end{bmatrix} + \begin{bmatrix} \alpha_2 \\ 0 \\ \alpha_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \alpha_3 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Just as in Example A.63, this can be written as an augmented matrix representing a set of linear equations:

$$\left[\begin{array}{ccc|c} 1 & 1 & 0 & a \\ 1 & 0 & 1 & b \\ 0 & 1 & 1 & c \end{array}\right] \tag{A.34}$$

Applying Gauss-Jordan elimination to the augmented matrix yields:

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 1/2\,a + 1/2\,b - 1/2\,c \\ 0 & 1 & 0 & -1/2\,b + 1/2\,a + 1/2\,c \\ 0 & 0 & 1 & 1/2\,c + 1/2\,b - 1/2\,a \end{array}\right] \tag{A.35}$$

which clearly has a solution for all a, b, and c. Another way of seeing this is to note that the matrix:

$$\mathbf{A} = \left[\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{array}\right] \tag{A.36}$$

is invertible.

The following theorem on the size of a basis in $\mathbb{R}^n$ is outside the scope of this course. A proof can be found in [Lan87].

THEOREM A.70. *If $\mathcal{X}$ is a basis of $\mathbb{R}^n$, then $\mathcal{X}$ contains precisely n vectors.*

EXERCISE A.13. Show that the vectors

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \ \mathbf{x}_2 = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}, \ \mathbf{x}_3 = \begin{bmatrix} 7 \\ 8 \\ 9 \end{bmatrix}$$

are not a basis for $\mathbb{R}^3$.

9. Diagonalization and Jordan's Decomposition Theorem

DEFINITION A.71 (Diagonalization). Let $\mathbf{A}$ be an $n \times n$ matrix (with entries drawn from any field[2], but for the time being we will assume $\mathbb{R}$). The matrix $\mathbf{A}$ can be diagonalized if there exists an $n \times n$ diagonal matrix $\mathbf{D}$ and another $n \times n$ matrix $\mathbf{P}$ so that:

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{P} = \mathbf{D} \tag{A.37}$$

In this case, $\mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ is the *diagonalization* of $\mathbf{A}$.

REMARK A.72. Clearly if $\mathbf{A}$ is diagonalizable, then:

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1} \tag{A.38}$$

The following theorem is proven in [].

[2]A field is a mathematical structure that generalizes the basic notions of arithmetic we are familiar with over $\mathbb{R}$. We will not cover them in detail in this course, but they should have been discussed a little in Linear Algebra (Math 220).

THEOREM A.73. *If* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *is diagonalizable, then the entries of* $\mathbf{D}$ *are the eigenvalues of* $\mathbf{A}$ *while the columns of* $\mathbf{P}$ *are the eigenvectors of* $\mathbf{A}$*. Moreover,* $\mathbf{A}$ *is diagonalizable if and only if the eigenvectors span an* n*-dimensional linear space (i.e.,* $\mathbf{P}$ *is inverible).*

EXAMPLE A.74. Consider the following matrix:

$$\mathbf{A} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{A.39}$$

To diagonalize $\mathbf{A}$, we compute its eigenvalues and eigenvectors yielding:

$$\lambda_1 = i$$
$$\lambda_2 = -i$$

for the eigenvalues and:

$$\mathbf{v}_1 = \begin{bmatrix} i \\ 1 \end{bmatrix} \quad \mathbf{v}_2 = \begin{bmatrix} -i \\ 1 \end{bmatrix}$$

where $i = \sqrt{-1}$ is the imaginary number. We can now compute $\mathbf{P}$ and $\mathbf{D}$ as:

$$\mathbf{D} = \begin{bmatrix} -i & 0 \\ 0 & i \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} -i & i \\ 1 & 1 \end{bmatrix}$$

It is helpful to note that:

$$\mathbf{P}^{-1} = \begin{bmatrix} \frac{i}{2} & \frac{1}{2} \\ \frac{-i}{2} & \frac{1}{2} \end{bmatrix}$$

Arithmetic manipulation shows us that:

$$\mathbf{PD} = \begin{bmatrix} -1 & -1 \\ -i & i \end{bmatrix}$$

Thus:

$$\mathbf{PDP}^{-1} = \begin{bmatrix} -1 & -1 \\ -i & i \end{bmatrix} \begin{bmatrix} \frac{i}{2} & \frac{1}{2} \\ \frac{-i}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} = \mathbf{A}$$

as required. (Remember that $i^2 = -1$.)

DEFINITION A.75 (Nilpotent Matrix). A matrix $\mathbf{N}$ is *nilpotent* if there is some integer $k > 0$ so that $\mathbf{N}^k = \mathbf{0}$

REMARK A.76. We generalize the notion of *diagonalization* in a concept called the *Jordan Normal Form*. Jordan Normal Form is well outside the scope of the class, but it can be summarized in the following theorem.

THEOREM A.77. *Let* $\mathbf{A}$ *be a square matrix with complex entries (i.e.,* $\mathbf{A} \in \mathbb{C}^{n\times n}$*). Then there exists matrices* $\mathbf{P}$*,* $\mathbf{\Lambda}$ *and* $\mathbf{N}$ *so that: (1)* $\mathbf{\Lambda}$ *is a diagonal matrix with the eigenvalues of* $\mathbf{M}$ *appearing on the diagonal. (2)* $\mathbf{N}$ *is a nilpotent matrix and (3)* $\mathbf{P}$ *is a matrix whose columns are composed of pseudo-eigenvectors and (4):*

$$\mathbf{A} = \mathbf{P}(\mathbf{\Lambda} + \mathbf{N})\mathbf{P}^{-1}, \tag{A.40}$$

When $\mathbf{A}$ *is diagonalizable, then* $\mathbf{N} = \mathbf{0}$ *and* $\mathbf{P}$ *is a matrix whose columns are composed of eigenvectors.*

APPENDIX B

Calculus and Analytical Geometry

REMARK B.1. This appendix is about the Calculus and Analytical Geometry that is used in Optimization, which is necessary for machine learning. The appendix, ideally, is self-contained. Basic definitions like *continuity* and *differentiability* are assumed. The following notation is worth mentioning: a function f that is differentiable k times and all those derivatives are continuous is said to be C^k and we may write $f \in C^k$.

1. Some Geometry for Optimization

REMARK B.2. We'll denote vectors in $\mathbb{R}^n$ in boldface. So $\mathbf{x} \in \mathbb{R}^n$ is an n-dimensional vector and we have $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$. We'll always associate an n-dimensional vectors with a $n \times 1$ matrix (column vector) unless otherwise noted. Thus, when we write $\mathbf{x} \in \mathbb{R}^n$ we also mean $\mathbf{x} \in \mathbb{R}^{n \times 1}$ (the set of $n \times 1$ matrices with entries from $\mathbb{R}$). See Appendix A for a complete review of vector operations, like dot products.

DEFINITION B.3 (Graph). Let $z : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be function, then the *graph* of z is the set of $n + 1$ tuples:

$$\{(\mathbf{x}, z(\mathbf{x})) \in \mathbb{R}^{n+1} | \mathbf{x} \in D\} \tag{B.1}$$

REMARK B.4. When $z : D \subseteq \mathbb{R} \to \mathbb{R}$, the graph is precisely what you'd expect. It's the set of pairs $(x, y) \in \mathbb{R}^2$ so that $y = z(x)$. This is the graph that you learned about back in Algebra 1.

DEFINITION B.5 (Level Set). Let $z : \mathbb{R}^n \to \mathbb{R}$ be a function and let $c \in \mathbb{R}$. Then the *level set of value c for function z* is the set:

$$\{\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n | z(\mathbf{x}) = c\} \subseteq \mathbb{R}^n \tag{B.2}$$

EXAMPLE B.6. Consider the function $z = x^2 + y^2$. The level set of z at 4 is the set of points $(x, y) \in \mathbb{R}^2$ such that:

$$x^2 + y^2 = 4 \tag{B.3}$$

You will recognize this as the equation for a circle with radius 4. We illustrate this in the following two figures. Figure B.1 shows the level sets of z as they sit on the 3D plot of the function, while Figure B.2 shows the level sets of z in $\mathbb{R}^2$. The plot in Figure B.2 is called a *contour plot*.

DEFINITION B.7. (Line) Let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then the *line* defined by vectors $\mathbf{x}_0$ and $\mathbf{h}$ is the function $\mathbf{l}(t) = \mathbf{x}_0 + t\mathbf{h}$. Clearly $l : \mathbb{R} \to \mathbb{R}^n$. The vector $\mathbf{h}$ is called the direction of the line.

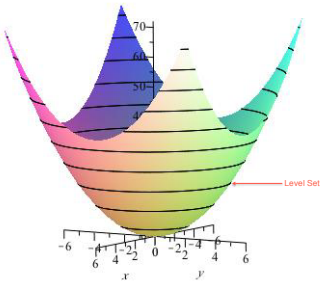


Figure B.1. Plot with Level Sets Projected on the Graph of z. The level sets existing in $\mathbb{R}^2$ while the graph of z existing $\mathbb{R}^3$. The level sets have been projected onto their appropriate heights on the graph.

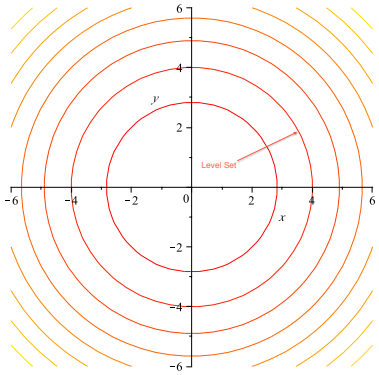


Figure B.2. Contour Plot of $z = x^2 + y^2$. The circles in $\mathbb{R}^2$ are the level sets of the function. The lighter the circle hue, the higher the value of c that defines the level set.

EXAMPLE B.8. Let $\mathbf{x_0} = (2, 1)$ and let $\mathbf{h} = (2, 2)$. Then the line defined by $\mathbf{x}_0$ and $\mathbf{h}$ is shown in Figure B.3. The set of points on this line is the set $L = \{(x, y) \in \mathbb{R}^2 : x = 2 + 2t, y = 1 + 2t, t \in \mathbb{R}\}$.

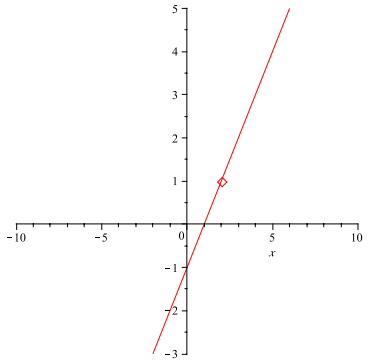


Figure B.3. A Line Function: The points in the graph shown in this figure are in the set produced using the expression $\mathbf{x}_0 + \mathbf{h}t$ where $\mathbf{x_0} = (2, 1)$ and let $\mathbf{h} = (2, 2)$.

DEFINITION B.9 (Directional Derivative). Let $z : \mathbb{R}^n \to \mathbb{R}$ and let $\mathbf{h} \in \mathbb{R}^n$ be a vector (direction) in n-dimensional space. Then the directional derivative of z at point $\mathbf{x}_0 \in \mathbb{R}^n$ in the direction of $\mathbf{h}$ is

$$\frac{d}{dt} z(\mathbf{x}_0 + t\mathbf{h})\bigg|_{t=0} \tag{B.4}$$

when this derivative exists.

DEFINITION B.10 (Gradient). Let $z : \mathbb{R}^n \to \mathbb{R}$ be function and let $\mathbf{x}_0 \in \mathbb{R}^n$. Then the *gradient* of z at $\mathbf{x}_0$ is the vector in $\mathbb{R}^n$ given by:

$$\nabla z(\mathbf{x}_0) = \left(\frac{\partial z}{\partial x_1}(\mathbf{x}_0), \ldots, \frac{\partial z}{\partial x_n}(\mathbf{x}_0) \right) \tag{B.5}$$

THEOREM B.11. *If $z : \mathbb{R}^n \to \mathbb{R}$ is differentiable, then all directional derivatives exist. Furthermore, the directional derivative of z at $\mathbf{x}_0$ in the direction of $\mathbf{h}$ is given by:*

$$\nabla z(\mathbf{x}_0) \cdot \mathbf{h} \tag{B.6}$$

where $\cdot$ denotes the dot product of two vectors.

PROOF. Let $\mathbf{l}(t) = \mathbf{x}_0 + \mathbf{h}t$. Then $\mathbf{l}(t) = (l_1(t), \ldots, l_n(t))$; that is, $\mathbf{l}(t)$ is a vector function whose i^{th} component is given by $l_i(t) = \mathbf{x}_{0_i} + \mathbf{h}_i t$.

Apply the chain rule:

$$\frac{dz(\mathbf{l}(t))}{dt} = \frac{\partial z}{\partial l_1}\frac{dl_1}{dt} + \cdots + \frac{\partial z}{\partial l_n}\frac{dl_n}{dt} \tag{B.7}$$

Thus:

$$\frac{d}{dt} z(\mathbf{l}(t)) = \nabla z \cdot \frac{d\mathbf{l}}{dt} \tag{B.8}$$

Clearly $d\mathbf{l}/dt = \mathbf{h}$. We have $\mathbf{l}(0) = \mathbf{x}_0$. Thus:

$$\frac{d}{dt} z(\mathbf{x}_0 + t\mathbf{h})\bigg|_{t=0} = \nabla z(\mathbf{x}_0) \cdot \mathbf{h} \tag{B.9}$$

□

THEOREM B.12. *Let $z : \mathbb{R}^n \to \mathbb{R}$ be differentiable, $\mathbf{x}_0 \in \mathbb{R}^n$. If $\nabla z(\mathbf{x}_0) \neq 0$, then $\nabla z(\mathbf{x}_0)$ points in the direction in which z is increasing fastest.*

PROOF. Recall $\nabla z(\mathbf{x}_0) \cdot \mathbf{h}$ is the directional derivative of z in direction $\mathbf{h}$ at $\mathbf{x}_0$. Assume that $\mathbf{h}$ is a unit vector. We know that:

$$\nabla z(\mathbf{x}_0) \cdot \mathbf{h} = ||\nabla z(\mathbf{x}_0)|| \cos\theta \tag{B.10}$$

(because we assumed $\mathbf{h}$ was a unit vector) where θ is the angle between the vectors $\nabla z(\mathbf{x}_0)$ and $\mathbf{h}$. The function $\cos\theta$ is largest when $\theta = 0$, that is when $\mathbf{h}$ and $\nabla z(\mathbf{x}_0)$ are parallel vectors. (If $\nabla z(\mathbf{x}_0) = 0$, then the directional derivative is zero in all directions.) □

THEOREM B.13. *Let $z : \mathbb{R}^n \to \mathbb{R}$ be differentiable and let $\mathbf{x}_0$ lie in the level set S defined by $z(\mathbf{x}) = k$ for fixed $k \in \mathbb{R}$. Then $\nabla z(\mathbf{x}_0)$ is normal to the set S in the sense that if $\mathbf{h}$ is a tangent vector at $t = 0$ of a path $\mathbf{c}(t)$ contained entirely in S with $\mathbf{c}(0) = \mathbf{x}_0$, then $\nabla z(\mathbf{x}_0) \cdot \mathbf{h} = 0$.*

REMARK B.14. Before giving the proof, we illustrate this theorem in Figure B.4. The function is $z(x, y) = x^4 + y^2 + 2xy$ and $\mathbf{x}_0 = (1, 1)$. At this point $\nabla z(\mathbf{x}_0) = (6, 4)$. We include the tangent line to the level set at the point (1,1) to illustrate the normality of the gradient to the level curve at the point.

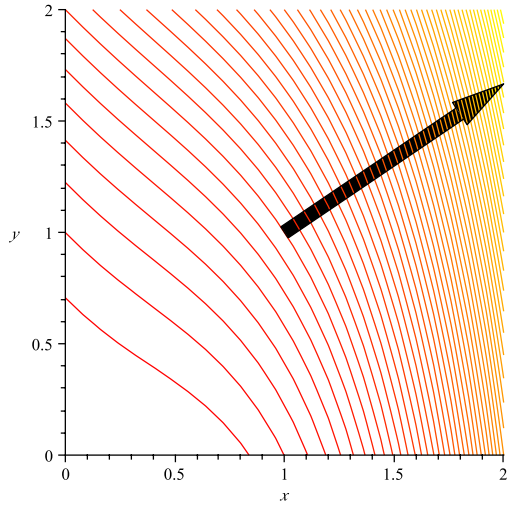


Figure B.4. A Level Curve Plot with Gradient Vector: We've scaled the gradient vector in this case to make the picture understandable. Note that the gradient is perpendicular to the level set curve at the point $(1, 1)$, where the gradient was evaluated. You can also note that the gradient is pointing in the direction of steepest ascent of $z(x, y)$.

PROOF. As stated, let $\mathbf{c}(t)$ be a curve in S. Then $\mathbf{c} : \mathbb{R} \to \mathbb{R}^n$ and $z(\mathbf{c}(t)) = k$ for all $t \in \mathbb{R}$. Let $\mathbf{h}$ be the tangent vector to $\mathbf{c}$ at $t = 0$; that is:

$$\left.\frac{d\mathbf{c}(t)}{dt}\right|_{t=0} = \mathbf{h} \tag{B.11}$$

Differentiating $z(\mathbf{c}(t))$ with respect to t using the chain rule and evaluating at $t = 0$ yields:

$$\left.\frac{d}{dt} z(\mathbf{c}(t))\right|_{t=0} = \nabla z(\mathbf{c}(0)) \cdot \mathbf{h} = \nabla z(\mathbf{x}_0) \cdot \mathbf{h} = 0 \tag{B.12}$$

Thus $\nabla z(\mathbf{x}_0)$ is perpendicular to $\mathbf{h}$ and thus normal to the set S as required. □

2. Concave/Convex Functions and Convex Sets

DEFINITION B.15 (Convex Set). Let $X \subseteq \mathbb{R}^n$. Then the set X is convex if and only if for all pairs $\mathbf{x}_1, \mathbf{x}_2 \in X$ we have $\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2 \in X$ for all $\lambda \in [0, 1]$.

THEOREM B.16. *The intersection of a finite number of convex sets in $\mathbb{R}^n$ is convex.*

PROOF. Let $C_1, \ldots, C_n \subseteq \mathbb{R}^n$ be a finite collection of convex sets. Let

$$C = \bigcap_{i=1}^{n} C_i \tag{B.13}$$

be the set formed from the intersection of these sets. Choose $\mathbf{x}_1, \mathbf{x}_2 \in C$ and $\lambda \in [0,1]$. Consider $\mathbf{x} = \lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2$. We know that $\mathbf{x}_1, \mathbf{x}_2 \in C_1, \ldots, C_n$ by definition of C. By convexity, we know that $\mathbf{x} \in C_1, \ldots, C_n$ by convexity of each set. Therefore, $\mathbf{x} \in C$. Thus C is a convex set. □

DEFINITION B.17 (Convex Function). A function $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function if it satisfies:

$$f(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1-\lambda)f(\mathbf{x}_2) \tag{B.14}$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and for all $\lambda \in [0,1]$. When the inequality is strict, for $\lambda \in (0,1)$, the function is a strictly convex function.

EXAMPLE B.18. This definition is illustrated in Figure B.5. When f is a univariate

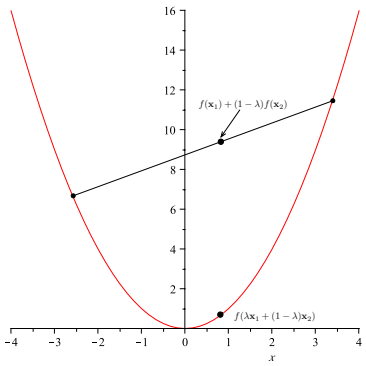


Figure B.5. A convex function: A convex function satisfies the expression $f(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1-\lambda)f(\mathbf{x}_2)$ for all $\mathbf{x}_1$ and $\mathbf{x}_2$ and $\lambda \in [0,1]$.

function, this definition can be shown to be equivalent to the definition you learned in Calculus I (Math 140) using first and second derivatives.

DEFINITION B.19 (Concave Function). A function $f : \mathbb{R}^n \to \mathbb{R}$ is a concave function if it satisfies:

$$f(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \geq \lambda f(\mathbf{x}_1) + (1-\lambda)f(\mathbf{x}_2) \tag{B.15}$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and for all $\lambda \in [0,1]$. When the inequality is strict, for $\lambda \in (0,1)$, the function is a strictly concave function.

To visualize this definition, simply flip Figure B.5 upside down. The following theorem is a powerful tool that can be used to show sets are convex. It's proof is outside the scope of the class, but relatively easy.

THEOREM B.20. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. Then the set $C = \{\mathbf{x} \in \mathbb{R}^n : f(x) \leq c\}$, where $c \in \mathbb{R}$, is a convex set.* □

EXERCISE B.1. Prove the Theorem B.20.

DEFINITION B.21 (Linear Function). A function $z : \mathbb{R}^n \to \mathbb{R}$ is *linear* if there are constants $c_1, \ldots, c_n \in \mathbb{R}$ so that:

$$z(x_1, \ldots, x_n) = c_1x_1 + \cdots + c_nx_n \tag{B.16}$$

DEFINITION B.22 (Affine Function). A function $z : \mathbb{R}^n \to \mathbb{R}$ is *affine* if $z(\mathbf{x}) = l(\mathbf{x}) + b$ where $l : \mathbb{R}^n \to \mathbb{R}$ is a linear function and $b \in \mathbb{R}$.

EXERCISE B.2. Prove that every affine function is both convex and concave.

THEOREM B.23. *Suppose that $g_1, \ldots, g_m : \mathbb{R}^n \to \mathbb{R}$ are convex functions and $h_1, \ldots, h_l : \mathbb{R}^n \to \mathbb{R}$ are affine functions. Then the set:*

$$\Omega = \{\mathbf{x} \in \mathbb{R}^n : g_i(\mathbf{x}) \leq 0, (i = 1, \ldots, m) \text{ and } h_j(\mathbf{x}) = 0, (j = 1, \ldots, l)\} \tag{B.17}$$

is convex.

EXERCISE B.3. Prove Theorem B.23

3. Concave Functions and Differentiability

REMARK B.24. The following theorem is interesting, but its proof is outside the scope of the course. There is a proof for the one-dimensional case in Rudin [**Rud76**]. The proof for the general case can be derived from this. The general proof can also be found in Appendix B of [**Ber99**].

THEOREM B.25. *Every concave (convex) function is continuous on the interior of its domain.* □

REMARK B.26. The proofs of the next two theorems are variations on those found in [**Ber99**], with some details added for clarity.

THEOREM B.27. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is concave if and only if for all $\mathbf{x}_0, \mathbf{x} \in \mathbb{R}^n$*

$$f(\mathbf{x}) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0) \tag{B.18}$$

PROOF. ($\Leftarrow$) Suppose Inequality B.18 holds. Let $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ and let $\lambda \in (0, 1)$ and let $\mathbf{x}_0 = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$. We may write:

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x}_1 - \mathbf{x}_0)$$
$$f(\mathbf{x}_2) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x}_2 - \mathbf{x}_0)$$

Multiplying the first equation by λ and the second by $1 - \lambda$ and adding yields:

$$\lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2) \leq$$
$$\lambda f(\mathbf{x}_0) + (1 - \lambda)f(\mathbf{x}_0) + \lambda\nabla f(\mathbf{x}_0)^T(\mathbf{x}_1 - \mathbf{x}_0) + (1 - \lambda)\nabla f(\mathbf{x}_0)^T(\mathbf{x}_2 - \mathbf{x}_0)$$

Simplifying the inequality, we have:

$$\lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \left(\lambda(\mathbf{x}_1 - \mathbf{x}_0) + (1 - \lambda)(\mathbf{x}_2 - \mathbf{x}_0)\right)$$

Which simplifies further to:

$$\lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \left(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2 - \lambda\mathbf{x}_0 - (1 - \lambda)\mathbf{x}_0\right)$$

Or:

$$\lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \left(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2 - \mathbf{x}_0\right) = f(\mathbf{x}_0)$$

because we assumed that:$\mathbf{x}_0 = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$.

($\Rightarrow$) Now let $\mathbf{x}, \mathbf{x}_0 \in \mathbb{R}^n$ and let $\lambda \in (0,1)$. Define $\mathbf{h} = \mathbf{x} - \mathbf{x}_0$. Let:

$$g(\lambda) = \frac{f(\mathbf{x}_0 + \lambda\mathbf{h}) - f(\mathbf{x}_0)}{\lambda} \tag{B.19}$$

Clearly, as λ approaches 0 from the right, $g(\alpha)$ approaches the directional derivative of $f(\mathbf{x})$ at $\mathbf{x}_0$ in the direction $\mathbf{h}$.

Claim 1. *The function $g(\lambda)$ is monotonically decreasing.*

Proof. Consider $\lambda_1, \lambda_2 \in (0,1)$ with $\lambda_1 < \lambda_2$ and let $\alpha = \lambda_1/\lambda_2$. Define $\mathbf{z} = \mathbf{x}_0 + \lambda_2\mathbf{h}$. Note:

$$\alpha\mathbf{z} + (1-\alpha)\mathbf{x}_0 = \mathbf{x}_0 + \alpha(\mathbf{z} - \mathbf{x}_0) = \mathbf{x}_0 + \frac{\lambda_1}{\lambda_2}(\mathbf{x}_0 + \lambda_2(\mathbf{z} - \mathbf{x}_0) - \mathbf{x}_0) = \mathbf{x}_0 + \lambda_1(\mathbf{z} - \mathbf{x}_0)$$

Thus:

$$f(\mathbf{x}_0 + \alpha(\mathbf{z} - \mathbf{x}_0)) \geq \alpha f(\mathbf{z}) + (1-\alpha)f(\mathbf{x}_0) = f(\mathbf{x}_0) + \alpha f(\mathbf{z}) - \alpha f(\mathbf{x}_0) \tag{B.20}$$

Simplifying, we obtain:

$$\frac{f(\mathbf{x}_0 + \alpha(\mathbf{z} - \mathbf{x}_0)) - f(\mathbf{x}_0)}{\alpha} \geq f(\mathbf{z}) - f(\mathbf{x}_0) \tag{B.21}$$

Since $\mathbf{z} = \mathbf{x}_0 + \lambda_2\mathbf{h}$, we have:

$$\frac{f(\mathbf{x}_0 + \alpha(\mathbf{z} - \mathbf{x}_0)) - f(\mathbf{x}_0)}{\alpha} \geq f(\mathbf{z}) - f(\mathbf{x}_0) \tag{B.22}$$

Recall $\mathbf{z} = \mathbf{x}_0 + \lambda_2\mathbf{h}$, thus $\mathbf{z} - \mathbf{x}_0 = \lambda_2\mathbf{h}$. Thus the left hand side simplifies to:

$$\frac{f(\mathbf{x}_0 + (\lambda_1/\lambda_2)(\lambda_2\mathbf{h})) - f(\mathbf{x}_0)}{\alpha} = \frac{f(\mathbf{x}_0 + \lambda_1\mathbf{h}) - f(\mathbf{x}_0)}{\lambda_1/\lambda_2} \geq f(\mathbf{x}_0 + \lambda_2\mathbf{h}) - f(\mathbf{x}_0) \tag{B.23}$$

Lastly, dividing both sides by λ_2 yields:

$$\frac{f(\mathbf{x}_0 + \lambda_1\mathbf{h}) - f(\mathbf{x}_0)}{\lambda_1} \geq \frac{f(\mathbf{x}_0 + \lambda_2\mathbf{h}) - f(\mathbf{x}_0)}{\lambda_2} \tag{B.24}$$

Thus $g(\lambda)$ is monotonically decreasing. This completes the proof of the claim. □

Since $g(\lambda)$ is monotonically decreasing, we must have:

$$\lim_{\lambda \to 0^+} g(\lambda) \geq g(1) \tag{B.25}$$

But this implies that:

$$\lim_{\lambda \to 0^+} \frac{f(\mathbf{x}_0 + \lambda\mathbf{h}) - f(\mathbf{x}_0)}{\lambda} \geq f(\mathbf{x}_0 + \mathbf{h}) - f(\mathbf{x}_0) = f(\mathbf{x}_0 + \mathbf{x} - \mathbf{x}_0) - f(\mathbf{x}_0) = f(\mathbf{x}) - f(\mathbf{x}_0) \tag{B.26}$$

since $\mathbf{h} = \mathbf{x} - \mathbf{x}_0$. Applying Theorem B.11, the inequality becomes:

$$\nabla f(\mathbf{x}_0)^T\mathbf{h} \geq f(\mathbf{x}) - f(\mathbf{x}_0) \tag{B.27}$$

which can be rewritten as:

$$f(\mathbf{x}) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T\mathbf{h} = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0) \tag{B.28}$$

This completes the proof. □

Exercise B.4. Argue that for strictly concave functions, Inequality B.18 is strict and the theorem still holds.

EXERCISE B.5. State and prove a similar theorem for convex functions.

4. Hessian Matrices and Jacobian Operators

DEFINITION B.28 (Hessian Matrix). Let $f : \mathbb{R}^n \to \mathbb{R}$ and assume f is twice differentiable. The *Hessian matrix* is given by:

$$\mathbf{H}(\mathbf{x}) = \nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(\mathbf{x}) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(\mathbf{x}) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_2^2}(\mathbf{x}) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(\mathbf{x}) & \frac{\partial^2 f}{\partial x_n \partial x_2}(\mathbf{x}) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(\mathbf{x}) \end{bmatrix} \tag{B.29}$$

EXERCISE B.6. Prove that the Hessian matrix of a twice differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is always symmetric.

EXAMPLE B.29. The Hessian matrices of quadratic functions are particularly nice. Consider the (convex) function $f(x_1, x_2) = x^2 + y^2$. It's second-order partial derivatives are:

$$\frac{\partial^2 f}{\partial x^2} = 2 \quad \frac{\partial^2 f}{\partial x \partial y} = 0$$
$$\frac{\partial^2 f}{\partial y \partial x} = 0 \quad \frac{\partial^2 f}{\partial y^2} = 2$$

Therefore, the Hessian matrix is constant:

$$\mathbf{H}(\mathbf{x}) = \nabla^2 f(\mathbf{x}) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

DEFINITION B.30 (Jacobian Operator). Let $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ be differentiable; i.e., $\mathbf{f}$ is a vector-valued function that takes vectors in $\mathbb{R}^n$. Let $f(\mathbf{x}) = \langle f_1(\mathbf{x}), \ldots, f_n(\mathbf{x}) \rangle$. The *Jacobian Operator* $\mathbf{D}$ applied to the function f at $\mathbf{x}$ yields the matrix:

$$\mathbf{Df}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{B.30}$$

5. Mean Value and Taylor's Theorem(s)

LEMMA B.31 (Mean Value Theorem). *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. (That is $f \in C^1$.) Let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then there is a $t \in (0, 1)$ such that:*

$$f(\mathbf{x}_0 + \mathbf{h}) - f(\mathbf{x}_0) = \nabla f(\mathbf{x}_0 + t\mathbf{h})^T \mathbf{h} \tag{B.31}$$

□

REMARK B.32. This is the natural generalization of the one-variable mean value theorem from calculus (Math 140 at Penn State). The expression $\mathbf{x}_0 + t\mathbf{h}$ is simply the line segment connecting $\mathbf{x}_0$ and $\mathbf{x}_0 + \mathbf{h}$. If we imagine this being the "x-axis" and the corresponding slice of $f(\cdot)$, then we can see that we're just applying the single variable mean value theorem to this slice.

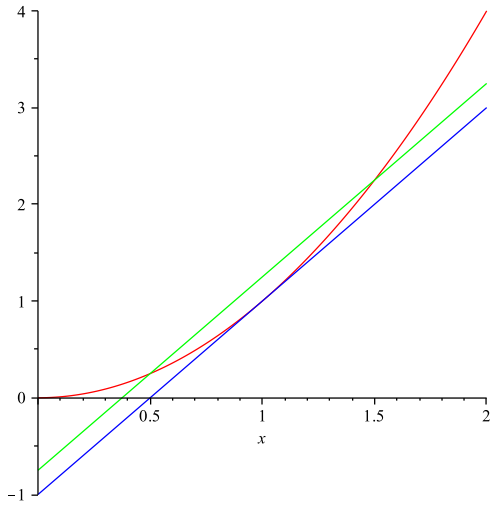


Figure B.6. An illustration of the mean value theorem in one variable. The multivariable mean value theorem is simply an application of the single variable mean value theorem applied to a slice of a function.

LEMMA B.33 (Second Order Mean Value Theorem). *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. (That is $f \in C^2$.) Let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then there is a $t \in (0, 1)$ such that:*

$$f(\mathbf{x}_0 + \mathbf{h}) - f(\mathbf{x}_0) = \nabla f(\mathbf{x}_0)^T \mathbf{h} + \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h} \tag{B.32}$$

□

LEMMA B.34 (Mean Value Theorem – Vector Valued Function). *Let $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ be a differentiable function and let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then:*

$$\mathbf{f}(\mathbf{x}_0 + \mathbf{h}) - \mathbf{f}(\mathbf{x}_0) = \int_0^1 \mathbf{D}\mathbf{f}(\mathbf{x}_0 + t\mathbf{h})\mathbf{h}dt \tag{B.33}$$

where $\mathbf{D}$ is the Jacobian operator. □

REMARK B.35. It should be noted that there is no exact analog of the mean value theorem for vector valued functions. The previous lemma is the closes thing to such an analog and it is generally referred to as such.

COROLLARY B.36. *Let $f : \mathbb{R}^n \to \mathbb{R}$ with $f \in C^2$ and let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then:*

$$\nabla f(\mathbf{x}_0 + \mathbf{h}) - \nabla f(\mathbf{x}_0) = \int_0^1 \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h}dt \tag{B.34}$$

□

REMARK B.37. The proof of this lemma rests on the single variable mean value theorem and the mean value theorem for integrals in single variable calculus.

LEMMA B.38 (Taylor's Theorem – Second Order). *Let $f : \mathbb{R}^n \to \mathbb{R}$ with $f \in C^2$ and let $\mathbf{x}_0, \mathbf{h} \in \mathbb{R}^n$. Then:*

$$f(\mathbf{x}_0 + \mathbf{h}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} + R_2(\mathbf{x}_0, \mathbf{h}) \tag{B.35}$$

where:

$$R_2(\mathbf{x}_0, \mathbf{h}) = \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h} \tag{B.36}$$

for some $t \in (0,1)$ *or:*

$$R_2(\mathbf{x}_0, \mathbf{h}) = \int_0^1 (1-t)\nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h}dt \tag{B.37}$$

□

REMARK B.39. Taylor's Theorem in the general case considers functions in C^k. One can convert between the two forms of the remainder using the mean value theorem, though this is not immediately obvious without some concentration. Most of the proofs of the remainder term use the mean value theorems. There is a very nice proof, very readable proof of Taylor's theorem in [**MT03**] (Chapter 3.2).

REMARK B.40. From Taylor's theorem, we obtain first and second order approximations for functions. That is, the first order approximation for $f(\mathbf{x}_0 + \mathbf{h})$ is:

$$f(\mathbf{x}_0 + \mathbf{h}) \sim f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} \tag{B.38}$$

while the second order approximation is:

$$f(\mathbf{x}_0 + \mathbf{h}) \sim f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} + \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0)\mathbf{h} \tag{B.39}$$

6. Hessian Definiteness and Concavity

THEOREM B.41. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be a twice differentiable function. If* f *is concave, then* $\nabla^2 f(\mathbf{x})$ *is negative semidefinite.*

PROOF. Suppose that there is a point $\mathbf{x}_0 \in \mathbb{R}^n$ and $\mathbf{h} \in \mathbb{R}^n$ such that $\mathbf{h}^T \nabla^2 f(\mathbf{x})\mathbf{h} > 0$. We may chose an $\mathbf{h}$ with a small norm so that for every $t \in [0,1]$, $\mathbf{h}^T \nabla^2 f(\mathbf{x} + t\mathbf{h})\mathbf{h} > 0$ as well. By Lemma B.38 for any $\mathbf{x} = \mathbf{x}_0 + \mathbf{h}$ we have some $t \in (0,1)$ so that:

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} + \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h} \tag{B.40}$$

But since $\mathbf{h}^T \nabla^2 f(\mathbf{x} + t\mathbf{h})\mathbf{h} > 0$, we know that

$$f(\mathbf{x}) > f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} \tag{B.41}$$

and thus by Theorem B.18, $f(\mathbf{x})$ cannot be concave, a contradiction. This completes the proof. □

THEOREM B.42. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be a twice differentiable function. If* $\nabla^2 f(\mathbf{x})$ *is negative semidefinite, then* $f(\mathbf{x})$ *is concave.*

PROOF. From Lemma B.38, we know that for every $\mathbf{x}, \mathbf{x}_0 \in \mathbb{R}^n$ there is a $t \in (0,1)$ such that when $\mathbf{h} = \mathbf{x} - \mathbf{x}_0$:

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T \mathbf{h} + \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h} \tag{B.42}$$

Since $\nabla^2 f(\mathbf{x})$ is negative semidefinite, it follows that: $\frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}_0 + t\mathbf{h})\mathbf{h} \leq 0$ and thus:

$$f(\mathbf{x}) \leq f(\mathbf{x}_0) + \nabla f(\mathbf{x}_0)^T (\mathbf{x} - \mathbf{x}_0) \tag{B.43}$$

Thus by Theorem B.18, $f(\mathbf{x})$ is concave. $\square$

EXERCISE B.7. Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ with $f(\mathbf{x}) = \mathbf{x}^T\mathbf{H}\mathbf{x}$, where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{H} \in \mathbb{R}^{n\times n}$. Show that $f(\mathbf{x})$ is convex if and only if $\mathbf{H}$ is positive semidefinite.

APPENDIX C

Optimization

1. Optimization Problems

DEFINITION C.1. Let $z : D \subseteq \mathbb{R}^n \to \mathbb{R}$. The point $\mathbf{x}^*$ is a *global maximum* for z if for all $\mathbf{x} \in D$, $z(\mathbf{x}^*) \geq z(\mathbf{x})$. A point $\mathbf{x}^* \in D$ is a *local maximum* for z if there is a neighborhood $S \subseteq D$ of $\mathbf{x}^*$ (i.e., $\mathbf{x}^* \in S$) so that for all $\mathbf{x} \in S$, $z(\mathbf{x}^*) \geq z(\mathbf{x})$. When the foregoing inequalities are strict, $\mathbf{x}^*$ is called a strict global or local maximum.

REMARK C.2. Clearly Definition C.1 is valid only for domains and functions where the concept of a neighborhood is defined and understood. In general, S must be a topologically connected set (as it is in a neighborhood in $\mathbb{R}^n$) in order for this definition to be used or at least we must be able to define the concept of *neighborhood* on the set.

DEFINITION C.3 (Maximization Problem). Let $z : D \subseteq \mathbb{R}^n \to \mathbb{R}$; for $i = 1, \ldots, m$, $g_i : D \subseteq \mathbb{R}^n \to \mathbb{R}$; and for $j = 1, \ldots, l$ $h_j : D \subseteq \mathbb{R}^n \to \mathbb{R}$ be functions. Then the general maximization problem with objective function $z(x_1, \ldots, x_n)$ and *inequality constraints* $g_i(x_1, \ldots, x_n) \leq b_i$ $(i = 1, \ldots, m)$ and *equality constraints* $h_j(x_1, \ldots, x_n) = r_j$ is written as:

$$\text{(C.1)} \qquad \begin{cases} \max & z(x_1, \ldots, x_n) \\ s.t. & g_1(x_1, \ldots, x_n) \leq 0 \\ & \vdots \\ & g_m(x_1, \ldots, x_n) \leq 0 \\ & h_1(x_1, \ldots, x_n) = 0 \\ & \vdots \\ & h_l(x_1, \ldots, x_n) = 0 \end{cases}$$

REMARK C.4. Expression C.1 is also called a *mathematical programming problem*. Naturally when constraints are involved we define the global and local maxima for the objective function $z(x_1, \ldots, x_n)$ in terms of the feasible region instead of the entire domain of z, since we are only concerned with values of $x_1, \ldots, x_n$ that satisfy our constraints.

REMARK C.5. When there are no constraints (or the only constraint is that $(x_1, \ldots, x_n) \in \mathbb{R}^n$), the problem is called an unconstrained maximization problem.

EXAMPLE C.6. Let's recall a simple optimization problem from differential calculus: Suppose I wish to build a pen to keep some goats. I have 100 meters of fencing and I wish to build the pen in a rectangle with the largest possible area. How long should the sides of the rectangle be? In this case, making the pen *better* means making it have the largest possible area.

We can write this problem as:

$$\text{(C.2)} \qquad \begin{cases} \max & A(x,y) = xy \\ s.t. & 2x + 2y - 100 = 0 \\ & x \geq 0 \\ & y \geq 0 \end{cases}$$

2. Unconstrained Optimization

REMARK C.7. If there are no constraints in an optimization problem, then the problem is called an unconstrained optimization problem. The following theorems relate optimal solutions to unconstrained problems and convex/concave functions.

2.1. Concavity and Optimization.

THEOREM C.8. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is concave and that $\mathbf{x}^*$ is a local maximizer of f. Then $\mathbf{x}^*$ is a global maximizer of f.*

PROOF. Suppose $\mathbf{x}^+ \in \mathbb{R}^n$ has the property that $f(\mathbf{x}^+) > f(\mathbf{x}^*)$. For any $\lambda \in (0,1)$ we know that:

$$f(\lambda \mathbf{x}^* + (1-\lambda)\mathbf{x}^+) \geq \lambda f(\mathbf{x}^*) + (1-\lambda) f(\mathbf{x}^+)$$

Since $\mathbf{x}^*$ is a local maximum there is an $\epsilon > 0$ so that for all $\mathbf{x} \in B_\epsilon(\mathbf{x}^*)$, $f(\mathbf{x}^*) \geq f(\mathbf{x})$. Choose λ so that $\lambda \mathbf{x}^* + (1-\lambda)\mathbf{x}^+$ is in $B_\epsilon(\mathbf{x}^*)$ and let $\mathbf{x} = \lambda \mathbf{x}^* + (1-\lambda)\mathbf{x}^+$. Let $r = f(\mathbf{x}^+) - f(\mathbf{x}^*)$. By assumption $r > 0$. Then we have:

$$f(\mathbf{x}) \geq \lambda f(\mathbf{x}^*) + (1-\lambda)(f(\mathbf{x}^*) + r)$$

But this implies that:

$$f(\mathbf{x}) \geq f(\mathbf{x}^*) + (1-\lambda) r$$

But $\mathbf{x} \in B_\epsilon(\mathbf{x}^*)$ by choice of λ, which contradicts our assumption that $\mathbf{x}^*$ is a local maximum. Thus, $\mathbf{x}^*$ must be a global maximum. □

THEOREM C.9. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is strictly concave and that $\mathbf{x}^*$ is a global maximizer of f. Then $\mathbf{x}^*$ is the unique global maximizer of f.*

EXERCISE C.1. Prove Theorem C.9. [Hint: Proceed by contradiction as in the proof of Theorem C.8.]

2.2. Necessary and Sufficient Conditions for Optimality.

THEOREM C.10. *Suppose that $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ is differentiable in an open neighborhood of a local maximizer $\mathbf{x}^* \in D$, then $\nabla f(\mathbf{x}^*) = \mathbf{0}$.*

PROOF. By way of contradiction, suppose that $\nabla f(\mathbf{x}^*) \neq \mathbf{0}$. The differentiability of f implies that $\nabla f(\mathbf{x})$ is continuous in the open neighborhood of $\mathbf{x}^*$. Thus, for all ϵ, there is a δ so that if $||\mathbf{x}^* - \mathbf{x}|| < \delta$, then $||\nabla f(\mathbf{x}^*) - \nabla f(\mathbf{x})|| < \epsilon$. Let $\mathbf{h} = \nabla f(\mathbf{x}^*)$. Trivially, $\mathbf{h}^T \mathbf{h} > 0$ and (by continuity), for some $t \in (0,1)$ (perhaps very small), we know that:

$$\text{(C.3)} \qquad \mathbf{h}^T \nabla f(\mathbf{x}^* + t\mathbf{h}) > 0$$

Let $\mathbf{p} = t\mathbf{h}$. From the mean value theorem, we know there is an $s \in (0,1)$ such that:

$$f(\mathbf{x}_0 + \mathbf{p}) - f(\mathbf{x}_0) = \nabla f(\mathbf{x}_0 + s\mathbf{p})^T \mathbf{p} > 0$$

by our previous argument. Thus, $\mathbf{x}^*$ is not a (local) maximum. □

EXERCISE C.2. In the last step of the previous proof, we assert the existence of a $t \in (0,1)$ so that Equation C.3 holds. Explicitly prove such a t must exist. [Hint: Use a component by component argument with the continuity of ∇f.]

EXERCISE C.3. Construct an analagous proof for the statement: *Suppose that $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ is differentiable in an open neighborhood of a local minimizer $\mathbf{x}^* \in D$, then $\nabla f(\mathbf{x}^*) = \mathbf{0}$.*

EXERCISE C.4. Construct an alternate proof of Theorem C.10 by studying the single-variable function $g(t) = f(\mathbf{x} + t\mathbf{h})$. You may use the fact from Math 140 that $g'(t^*) = 0$ is a necessary condition for a local maxima in the one dimensional case. (Extra credit if you prove that fact as well.)

THEOREM C.11. *Suppose that $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ is twice differentiable in an open neighborhood of a local maximizer $\mathbf{x}^* \in D$, then $\nabla f(\mathbf{x}^*) = \mathbf{0}$ and $\mathbf{H}(\mathbf{x}^*) = \nabla^2 f(\mathbf{x}^*)$ is negative-semidefinite.*

PROOF. From Theorem C.10 $\nabla f(\mathbf{x}^*) = \mathbf{0}$. Our assumption that f is twice differentiable implies that $\mathbf{H}(\mathbf{x})$ is continuous and therefore for all ϵ there exists a δ so that $||\mathbf{x}^* - \mathbf{x}|| < \delta$ then $|\mathbf{H}_{ij}(\mathbf{x}^*) - \mathbf{H}_{ij}(\mathbf{x})| < \epsilon$ for all $i = 1, \ldots, n$ and $j = 1, \ldots n$. We are just asserting pointwise continuity for the elements of the matrix $\mathbf{H}(\mathbf{x})$.

Suppose we may choose a vector $\mathbf{h}$ so that $\mathbf{h}^T \mathbf{H}(\mathbf{x}^*)\mathbf{h} > 0$ and thus $\mathbf{H}(\mathbf{x}^*)$ is not negative semidefinite. Then by our continuity argument, we can choose an $\mathbf{h}$ with norm small enough, so we can assure that $\mathbf{h}^T \mathbf{H}(\mathbf{x}^* + \mathbf{h})\mathbf{h} > 0$. From Lemma B.38 we have for some $t \in (0,1)$:

$$f(\mathbf{x}^* + \mathbf{h}) - f(\mathbf{x}^*) = \frac{1}{2}\mathbf{h}^T \mathbf{H}(\mathbf{x}^* + t\mathbf{h})\mathbf{h} \tag{C.4}$$

since $\nabla f(\mathbf{x}^*) = \mathbf{0}$. Then it follows that $f(\mathbf{x}_0 + \mathbf{h}) - f(\mathbf{x}_0) > 0$ and thus we have found a direction in which the value of f increases and $\mathbf{x}^*$ cannot be a local maximum. □

THEOREM C.12. *Suppose that $f : D \subset \mathbb{R}^n \to \mathbb{R}$ is twice differentiable. If $\nabla f(\mathbf{x}^*) = 0$ and $\mathbf{H}(\mathbf{x}^*) = \nabla^2 f(\mathbf{x}^*)$ is negative definite, then $\mathbf{x}^*$ is a local maximum.*

PROOF. Applying Lemma B.38, we know for any $h \in \mathbb{R}^n$, there is a $t \in (0,1)$ so that:

$$f(\mathbf{x}^* + \mathbf{h}) = f(\mathbf{x}^*) + \frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x}^* + t\mathbf{h})\mathbf{h} \tag{C.5}$$

By the same argument as in the proof of Theorem C.11, we know that there is an $\epsilon > 0$ so that if $|\mathbf{h}| < \epsilon$ then for all $t \in (0,1)$, $\nabla^2 f(\mathbf{x}^* + t\mathbf{h})$ is negative definite if $\nabla^2 f(\mathbf{x}^*)$ is negative definite. Let $B_\epsilon(\mathbf{x}^*)$ the open ball centered at $\mathbf{x}^*$ with radius ϵ.

Thus we can see that for all $\mathbf{x} \in B_\epsilon(\mathbf{x}^*)$:

$$\frac{1}{2}\mathbf{h}^T \nabla^2 f(\mathbf{x})\mathbf{h} < 0$$

where $\mathbf{x} = \mathbf{x}^* + t\mathbf{h}$ for some appropriately chosen $\mathbf{h}$ and $t \in (0,1)$. Equation C.5 combined with the previous observation shows that for all $\mathbf{x} \in B_\epsilon(\mathbf{x}^*)$, $f(\mathbf{x}) < f(\mathbf{x}^*)$ and thus $\mathbf{x}^*$ is a local maximum. This completes the proof. □

EXERCISE C.5. Theorem C.12 provides sufficient conditions for $\mathbf{x}^*$ to be a *strict* local minimum. Give an example showing that the conditions are not necessary.

3. Gradient Ascent and Descent

REMARK C.13. In Theorem B.12, we proved that the gradient points in the direction of fastest increase for a function $f : \mathbb{R}^n \to \mathbb{R}$. If we are trying to identify a point $\mathbf{x}^* \in \mathbb{R}^n$ that is a local or global maximum for f, then a reasonable approach is to walk along some direction $\mathbf{p}$ so that $\nabla f(\mathbf{x})^T \mathbf{p} > 0$.

DEFINITION C.14. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous and differentiable function and let $\mathbf{p} \in \mathbb{R}^n$. If $\nabla f(\mathbf{x})^T \mathbf{p} > 0$, then $\mathbf{p}$ is called an *ascent direction*.

REMARK C.15. Care must be taken, however, since the $\nabla f(\mathbf{x})$ represents only the direction of fastest increase at $\mathbf{x}$. As a result, we only want to take a small step in the direction of $\mathbf{p}$, then re-evaluate the gradient and continue until a stopping condition is reached.

Basic Ascent Algorithm
Input: $f : \mathbb{R}^n \to \mathbb{R}$ a function to maximize, $\mathbf{x}_0 \in \mathbb{R}^n$, a starting position
Initialize: $k = 0$

(1) **do**
(2) Choose $\mathbf{p}_k \in \mathbb{R}^{n \times 1}$ and $\delta_k \in \mathbb{R}_+$ so that $\nabla f(\mathbf{x})^T \mathbf{p}_k > 0$.
(3) $\mathbf{x}_{k+1} := \mathbf{x}_k + \delta_k \mathbf{p}_k$
(4) **while** some stopping criteria are not met.

Output: $\mathbf{x}_{k+1}$

Algorithm 2. Basic Ascent Algorithm

REMARK C.16. There are some obvious ambiguities with Algorithm 2. We have neither specified how to choose $\mathbf{p}_k$ nor δ_k in Line (2) of Algorithm 2, nor have we defined specific stopping criteria for the while loop. More importantly, we'd like to prove that there is a way of choosing $\mathbf{p}_k$ so that when we use this method at Line (2), the algorithm both converges i.e., at some point we exit the loop in Lines (1)-(4) and when we exit, we have identified a local maximum, or at least a point that satisfies the necessary conditions for a local maximum (see Theorem C.10).

REMARK C.17. For the remainder of these notes, we will assume that:

$$\mathbf{p}_k = \mathbf{B}_k^{-1} \nabla f(\mathbf{x}_k)$$

where $\mathbf{B}_k \in \mathbb{R}^{n \times n}$ is some appropriately chosen symmetric and non-singular matrix.

DEFINITION C.18 (Gradient Ascent). When $\mathbf{B}_k = \mathbf{I}_n$, for some n, then Algorithm 2 is called *Gradient Ascent*. When $\mathbf{B}_k = -\mathbf{I}_n$, then Algorithm 2 is called *Gradient Descent* (and is used to find a *minimum*).

REMARK C.19. As we've noted, we cannot simply choose δ_k arbitrarily in Line (2) of Algorithm 2, since $\nabla f(\mathbf{x})$ is only the direction of greatest ascent at $\mathbf{x}$ and thus, $\mathbf{p}_k = \mathbf{B}_k^{-1} \nabla f(\mathbf{x}_k)$ is only a ascent direction in a neighborhood about $\mathbf{x}$. If we define:

$$\phi(\delta_k) = f(\mathbf{x}_k + \delta_k \mathbf{p}_k) \tag{C.6}$$

then our problem is to solve:

$$\text{(C.7)} \quad \begin{cases} \max \ \phi(\delta_k) \\ s.t. \ \ \delta_k \geq 0 \end{cases}$$

This is an optimization problem in a single variable, δ_k and assuming its solution is δ_k^* and we compute $\mathbf{x}_{k+1} := \mathbf{x}_k + \delta_k^* \mathbf{p}_k$, then we will assuredly have increased (or at least not decreased) the value of $\mathbf{f}(\mathbf{x}_{k+1})$ compared to $f(\mathbf{x}_k)$. This problem is called *line search* and is discussed in [Ber99] and [Gri12a]. It is outside the scope of this course.

REMARK C.20. It's not enough to just increase the value of $f(\mathbf{x}_k)$ at each iteration k, we must increase it by a sufficient amount. One of the easiest, though not necessarily the most computationally efficient, ways to make sure this happens is to ensure that we identify a solution to the problem in Expression C.7. In the following sections, we discuss several methods for determining a solution.

4. Convergence of Gradient Ascent

DEFINITION C.21 (Armijo Rule). Given a function $f : \mathbb{R}^n \to \mathbb{R}$ and an ascent direction $\mathbf{p}_k$ with constant $\sigma_1 \in (0, 1)$, the Armijo rule is satisfied if:

$$\text{(C.8)} \quad f(\mathbf{x}_k + \delta_k \mathbf{p}_k) - f(\mathbf{x}_k) \geq \sigma_1 \delta_k \nabla f(\mathbf{x}_k)^T \mathbf{p}_k$$

REMARK C.22. Recall, $\phi(\delta_k) = f(\mathbf{x}_k + \delta_k \mathbf{p}_k)$ consequently, Equation C.8 simply states that:

$$\text{(C.9)} \quad \phi(\delta_k) - \phi(0) \geq \sigma_1 \delta_k \nabla f(\mathbf{x}_k)^T \mathbf{p}_k = \sigma_1 \delta_k \phi'(0)$$

which simply means there is a sufficient increase in the value of the function.

DEFINITION C.23 (Gradient Related). Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function. A sequence of ascent directions $\{\mathbf{p}_k\}$ is *gradient related* if for any subsequence $K = \{k_1, k_2, \dots\}$ of $\{\mathbf{x}_k\}$ that converges to a non-stationary point of f the corresponding subsequence $\{\mathbf{p}_k\}_{k \in K}$ is bounded and has the property that:

$$\text{(C.10)} \quad \limsup_{k \to \infty, k \in K} \nabla f(\mathbf{x}_k)^T \mathbf{p}_k > 0$$

REMARK C.24. We state the following theorem without proof. The proof can be found in [Ber99] or in [Gri12a].

THEOREM C.25. *Assume that δ_k is chosen to ensure the Armijo rule holds at each iteration and the ascent directions $\mathbf{p}_k$ are chosen so they are gradient related. Then if Algorithm 2 converges, it converges to a point $\mathbf{x}^*$ so that $\nabla f(\mathbf{x}^*) = \mathbf{0}$ and the stopping criteria:*

$$\text{(C.11)} \quad ||\nabla f(\mathbf{x}_k)|| < \epsilon$$

for some small $\epsilon > 0$ may be used.

COROLLARY C.26. *Assume that δ_k is chosen by maximizing $\phi(\delta_k)$ at each iteration and the ascent directions $\mathbf{p}_k$ are chosen so they are gradient related. Then Algorithm 2 converges to a point $\mathbf{x}^*$ so that $\nabla f(\mathbf{x}^*) = \mathbf{0}$.*

COROLLARY C.27. *Suppose that* $\mathbf{p}_k = \mathbf{B}_k^{-1}\nabla f(\mathbf{x}_k)$ *in each iteration of Algorithm 2 and every matrix* $\mathbf{B}_k$ *is symmetric, positive definite. Then, Algorithm 2 converges to a stationary point of* f*.*

EXAMPLE C.28. Consider the function $F(x, y) = -2\,x^2 - 10\,y^2$. If we initialize our gradient ascent at $x = 15$, $y = 5$ and set $\epsilon = 0.001$, then we obtain the following output, which converges near the optimal point $x^* = 0$, $y^* = 0$. The zig-zagging motion is typical of the gradient ascent algorithm in cases where the largest and smallest eigenvalues for the matrix $\mathbf{Q}$ (in a pure quadratic function) are very different (see Figure C.1). In this case:

$$\mathbf{Q} = \begin{bmatrix} -2 & 0 \\ 0 & -10 \end{bmatrix}$$

Here:

$$F(x, y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} -2 & 0 \\ 0 & -10 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

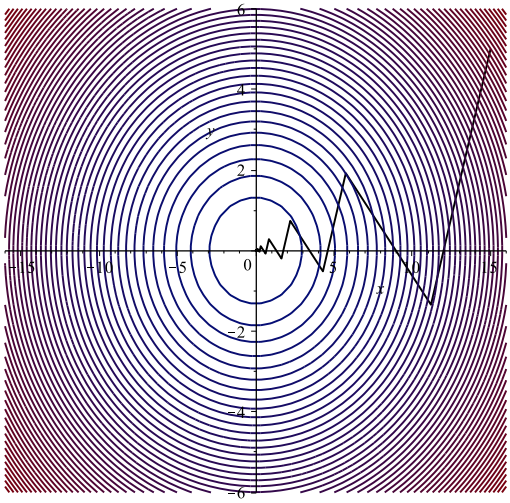


Figure C.1. Gradient ascent is illustrated on the function $F(x, y) = -2\,x^2 - 10\,y^2$ starting at $x = 15$, $y = 5$. The zig-zagging motion is typical of the gradient ascent algorithm in certain cases.

5. Karush-Kuhn-Tucker Conditions

REMARK C.29. We now turn our attention to constrained optimization. It turns out there is a very powerful theorem that discusses when a point $\mathbf{x}^* \in \mathbb{R}^n$ will maximize a function assuming some constraints. The following is the Karush-Kuhn-Tucker theorem, which we will state, but not prove.

THEOREM C.30. *Let* $z : \mathbb{R}^n \to \mathbb{R}$ *be a differentiable objective function,* $g_i : \mathbb{R}^n \to \mathbb{R}$ *be differentiable constraint functions for* $i = 1, \ldots, m$ *and* $h_j : \mathbb{R}^n \to \mathbb{R}$ *be differentiable*

constraint functions for $j = 1, \ldots, l$. If $\mathbf{x}^* \in \mathbb{R}^n$ is an optimal point satisfying an appropriate regularity condition for the following optimization problem:

$$P \begin{cases} \max & z(x_1, \ldots, x_n) \\ s.t. & g_1(x_1, \ldots, x_n) \leq 0 \\ & \vdots \\ & g_m(x_1, \ldots, x_n) \leq 0 \\ & h_1(x_1, \ldots, x_n) = 0 \\ & \vdots \\ & h_l(x_1, \ldots, x_n) = 0 \end{cases}$$

then there exists $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ and $\mu_1, \ldots \mu_l \in \mathbb{R}$ so that:

$$\textit{Primal Feasibility}: \begin{cases} g_i(\mathbf{x}^*) \leq 0 \quad \textit{for } i = 1, \ldots, m \\ h_j(\mathbf{x}^*) = 0 \quad \textit{for } j = 1, \ldots, l \end{cases}$$

$$\textit{Dual Feasibility}: \begin{cases} \nabla z(\mathbf{x}^*) - \sum_{i=1}^{m} \lambda_i \nabla g_i(\mathbf{x}^*) - \sum_{j=1}^{l} \mu_j \nabla h_j(\mathbf{x}^*) = \mathbf{0} \\ \lambda_i \geq 0 \quad \textit{for } i = 1, \ldots, m \\ \mu_j \in \mathbb{R} \quad \textit{for } j = 1, \ldots, l \end{cases}$$

$$\textit{Complementary Slackness}: \left\{ \lambda_i g_i(\mathbf{x}^*) = 0 \quad \textit{for } i = 1, \ldots, m \right.$$

THEOREM C.31. *Let $z : \mathbb{R}^n \to \mathbb{R}$ be a differentiable concave function, $g_i : \mathbb{R}^n \to \mathbb{R}$ be differentiable convex functions for $i = 1, \ldots, m$ and $h_j : \mathbb{R}^n \to \mathbb{R}$ be affine functions for $j = 1, \ldots, l$. Suppose there are $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ and $\mu_1, \ldots \mu_l \in \mathbb{R}$ so that:*

$$\textit{Primal Feasibility}: \begin{cases} g_i(\mathbf{x}^*) \leq 0 \quad \textit{for } i = 1, \ldots, m \\ h_j(\mathbf{x}^*) = 0 \quad \textit{for } j = 1, \ldots, l \end{cases}$$

$$\textit{Dual Feasibility}: \begin{cases} \nabla z(\mathbf{x}^*) - \sum_{i=1}^{m} \lambda_i \nabla g_i(\mathbf{x}^*) - \sum_{j=1}^{l} \mu_j \nabla h_j(\mathbf{x}^*) = \mathbf{0} \\ \lambda_i \geq 0 \quad \textit{for } i = 1, \ldots, m \\ \mu_j \in \mathbb{R} \quad \textit{for } j = 1, \ldots, l \end{cases}$$

$$\textit{Complementary Slackness}: \left\{ \lambda_i g_i(\mathbf{x}^*) = 0 \quad \textit{for } i = 1, \ldots, m \right.$$

then $\mathbf{x}^*$ *is a global maximizer for*

$$P\begin{cases}\max & z(x_1,\ldots,x_n)\\ s.t. & g_1(x_1,\ldots,x_n)\leq 0\\ & \vdots\\ & g_m(x_1,\ldots,x_n)\leq 0\\ & h_1(x_1,\ldots,x_n)=0\\ & \vdots\\ & h_l(x_1,\ldots,x_n)=0\end{cases}$$

REMARK C.32. The values $\lambda_1,\ldots,\lambda_m$ and $\mu_1,\ldots,\mu_l$ are sometimes called *Lagrange multipliers* and sometimes called *dual variables*. Primal Feasibility, Dual Feasibility and Complementary Slackness are called the *Karush-Kuhn-Tucker* (KKT) conditions.

REMARK C.33. The expression:

$$\nabla z(\mathbf{x}^*)-\sum_{i=1}^{m}\lambda_i\nabla g_i(\mathbf{x}^*)-\sum_{j=1}^{l}\mu_j\nabla h_j(\mathbf{x}^*)=\mathbf{0}$$

is sometimes called the Kuhn-Tucker equality.

REMARK C.34. The regularity condition mentioned in Theorem C.30 is sometimes called a constraint qualification. A common one is that the gradients of the binding constraints are all linearly independent at $\mathbf{x}^*$. There are many variations of constraint qualifications. We will not deal with these in these notes. Suffice it to say, all the problems we consider will automatically satisfy a constraint qualification, meaning the KKT theorem holds.

REMARK C.35. This theorem holds as a necessary condition even if $z(\mathbf{x})$ is not concave or the functions $g_i(\mathbf{x})$ $(i=1,\ldots,m)$ are not convex or the functions $h_j(\mathbf{x})$ $(j=1,\ldots,l)$ are not linear. In this case though, the fact that a triple: $(\mathbf{x},\boldsymbol{\lambda},\boldsymbol{\mu})\in\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^l$ does not ensure that this is an optimal solution for Problem P.

REMARK C.36. Looking more closely at the dual feasibility conditions, we see something interesting. Suppose that there are *no* equality constraints (i.e., not constraints of the form $h_j(\mathbf{x})=0$). Then the statements:

$$\nabla z(\mathbf{x}^*)-\sum_{i=1}^{m}\lambda_i\nabla g_i(\mathbf{x}^*)-\sum_{j=1}^{l}\mu_j\nabla h_j(\mathbf{x}^*)=\mathbf{0}$$
$$\lambda_i\geq 0\quad\text{for } i=1,\ldots,m$$

imply that:

$$\nabla z(\mathbf{x}^*)=\sum_{i=1}^{m}\lambda_i\nabla g_i(\mathbf{x}^*)$$
$$\lambda_i\geq 0\quad\text{for } i=1,\ldots,m$$

Specifically, this says that the *gradient of* z *at* $\mathbf{x}^*$ *is a positive combination of the gradients of the constraints at* $\mathbf{x}^*$. But more importantly, since we also have *complementary slackness*,

we know that if $\mathbf{g}_i(\mathbf{x}^*) \neq 0$, then $\lambda_i = 0$ because $\lambda_i g_i(\mathbf{x}^*) = 0$ for $i = 1, \ldots, m$. Thus, what dual feasibility is really saying is that *gradient of z at $\mathbf{x}^*$ is a positive combination of the gradients of the **binding** constraints at $\mathbf{x}^*$*. Remember, a constraint is binding if $g_i(\mathbf{x}^*) = 0$, in which case $\lambda_i \geq 0$.

REMARK C.37. Continuing from the previous remark, in the general case when we have some equality constraints, then dual feasibility says:

$$\begin{aligned} \nabla z(\mathbf{x}^*) &= \sum_{i=1}^{m} \lambda_i \nabla g_i(\mathbf{x}^*) + \sum_{j=1}^{l} \mu_j \nabla h_j(\mathbf{x}^*) \\ \lambda_i &\geq 0 \quad \text{for } i = 1, \ldots, m \\ \mu_j &\in \mathbb{R} \quad \text{for } j = 1, \ldots, l \end{aligned}$$

Since equality constraints are *always binding* this says that the *gradient of z at $\mathbf{x}^*$ is a linear combination of the gradients of the **binding** constraints at $\mathbf{x}^*$.*

EXAMPLE C.38. We'll finish the example we started with Example C.6. Let's rephrase this optimization problem in the form we saw in the theorem: We'll have:

$$\text{(C.12)} \quad \begin{cases} \max \ A(x, y) = xy \\ \ s.t. \ 2x + 2y - 100 = 0 \\ \qquad -x \leq 0 \\ \qquad -y \leq 0 \end{cases}$$

Note that the greater-than inequalities $x \geq 0$ and $y \geq 0$ in Expression C.2 have been changes to less-than inequalities by multiplying by -1. The constraints $2x + 2y = 100$ has simply been transformed to $2x + 2y - 100 = 0$. Thus, if $h(x, y) = 2x + 2y - 100$, we can see $h(x, y) = 0$ is our constraint. We can let $g_1(x, y) = -x$ and $g_2(x, y) = -y$. Then we have $g_1(x, y) \leq 0$ and $g_2(x, y) \leq 0$ as our inequality constraints. We already know that $x = y = 25$ is our optimal solution. Thus we know that there must be Lagrange multipliers μ, λ_1 and λ_2 corresponding to the constraints $h(x, y) =$, $g_1(x, y) \leq 0$ and $g_2(x, y) \leq 0$ that satisfy the KKT conditions.

Let's investigate the three components of the KKT conditions.

Primal Feasibility: If $x = y = 25$, then $h(x, y) = 2x + 2y - 100$ and clearly $h(25, 25) = 0$. Further $g_1(x, y) = -x$ and $g_2(x, y) = -y$ then $g_1(25, 25) = -25 \leq 0$ and $g_2(25, 25) = -25 \leq 0$. So primal feasibility is satisfied.

Complementary Slackness: We know that $g_1(x, y) = g_2(x, y) = -25$. Since neither of these functions is 0, we know that $\lambda_1 = \lambda_2 = 0$. This will force complementary slackness, namely:

$$\lambda_1 g_1(25, 25) = 0$$
$$\lambda_2 g_2(25, 25) = 0$$

Dual Feasibility: We already know that $\lambda_1 = \lambda_2 = 0$. That means we need to find $\mu \in \mathbb{R}$ so that:

$$\nabla A(25, 25) - \mu \nabla h(25, 25) = \mathbf{0}$$

We know that:

$$\nabla A(x,y) = \nabla xy = \begin{bmatrix} y \\ x \end{bmatrix}$$

$$\nabla h(x,y) = \nabla(2x + 2y - 100) = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

Evaluating $\nabla A(25,25)$ yields:

$$\begin{bmatrix} 25 \\ 25 \end{bmatrix} - \mu \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Thus setting $\mu = 25/2$ will accomplish our goal.

6. Lagrangian and Wolfe Dual

REMARK C.39. For this section only, we will consider the general minimization problem:

$$\text{(C.13)} \quad \begin{cases} \min \ z(x_1, \ldots, x_n) \\ s.t. \ g_1(x_1, \ldots, x_n) \leq 0 \\ \qquad \vdots \\ \qquad g_m(x_1, \ldots, x_n) \leq 0 \\ \qquad h_1(x_1, \ldots, x_n) = 0 \\ \qquad \vdots \\ \qquad h_l(x_1, \ldots, x_n) = 0 \end{cases}$$

In Chapter 2 we require the minimization form, and thus we present results on this. All results on minimization can be converted to results for a maximization problem by using negative signs appropriately in the objective.

DEFINITION C.40 (Lagrangian). Given the general minimization problem of Definition **??**, the *Lagrangian* is the function:

$$\text{(C.14)} \quad \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = z(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}) + \sum_{j=1}^{l} \mu_j h_j(\mathbf{x})$$

DEFINITION C.41 (Lagrangian Dual). Given a Lagrangian, the *Lagrangian Dual function* function is:

$$\text{(C.15)} \quad \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \inf_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$$

REMARK C.42. The Lagrangian function and dual for the general maximization problem is:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = z(\mathbf{x}) - \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}) - \sum_{j=1}^{l} \mu_j h_j(\mathbf{x})$$

while the Lagrangian Dual function is:

$$\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \sup_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$$

Notice the Kuhn-Tucker equality can be derived by taking the gradient of $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ with respect to $\mathbf{x}$.

Definition C.43 (Lagrangian Dual Problem). Given a Lagrangian dual function, the *Lagrangian Dual Problem* is:

$$\text{(C.16)} \quad \begin{cases} \max & \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{\mu}) \\ s.t. & \boldsymbol{\lambda} \geq \mathbf{0} \\ & \boldsymbol{\mu} \in \mathbb{R}^l \end{cases}$$

Theorem C.44 (Weak Duality). *The objective function of the Lagrangian Dual Problem at optimality is always a lower bound on the objective function of the original problem at optimality.* □

Remark C.45. When the objective function of the Lagrangian Dual Problem at optimality is lower than the objective function of the original problem at optimality, we have a *duality gap*. Otherwise, the two values must be identical.

Remark C.46. Suppose $z, g_1, \ldots, g_m$ are convex and differentiable and there are no inequality constraints. Then: $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ is convex and its infimum must occur when

$$\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{0}$$

by Theorems C.8 and C.10. Thus, the Lagrangian dual problem is equivalent to the constrained optimization problem:

$$\text{(C.17)} \quad \begin{cases} \max & \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \\ s.t. & \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{0} \\ & \boldsymbol{\lambda} \geq \mathbf{0} \end{cases}$$

This is called the *Wolfe Dual*.

Example C.47. Consider the optimization problem:

$$\begin{aligned} \min \quad & \frac{1}{2}x^2 + \frac{1}{2}y^2 \\ & x + y \geq 4 \end{aligned}$$

The constraint can be re-written as $-x - y + 4 \leq 0$ and the Lagrangian is then:

$$\mathcal{L}(x, y, \lambda) = \frac{1}{2}x^2 + \frac{1}{2}y^2 + \lambda(-x - y + 4)$$

The Lagrangian dual function is then:

$$\mathcal{L}(\lambda) = \inf_{x,y} \frac{1}{2}x^2 + \frac{1}{2}y^2 + \lambda(-x - y + 4)$$

The resulting Lagrangian dual problem is:

$$\begin{aligned} \max \ & \inf_{x,y} \frac{1}{2}x^2 + \frac{1}{2}y^2 + \lambda(-x - y + 4) \\ s.t. \ & \lambda \geq 0 \end{aligned}$$

Notice no matter what value of λ we choose, the value $\mathcal{L}(\lambda)$ can always be computed by simple calculus, i.e. solving for (x, y) when $\nabla_{x,y}\mathcal{L}(x, y, \lambda) = \mathbf{0}$. Or specifically:

$$\begin{aligned} x - \lambda &= 0 \\ y - \lambda &= 0 \end{aligned}$$

Thus, the Wolfe Dual is:

$$\begin{aligned} \max \ & \frac{1}{2}x^2 + \frac{1}{2}y^2 + \lambda(-x - y + 4) \\ s.t. \ & x - \lambda = 0 \\ & y - \lambda = 0 \\ & \lambda \geq 0 \end{aligned}$$

It is worth noting this problem is not convex as it is stated. However, we can simplify it further. Substitute $x = y = \lambda$ into the objective to obtain:

$$\begin{aligned} \max \ & \lambda^2 + \lambda(-2\lambda + 4) = -\lambda^2 + 4\lambda \\ s.t. \ & \lambda \geq 0 \end{aligned}$$

This optimization problem is convex (the objective function $4 - \lambda^2$ is concave). Clearly the only solution to this is $\lambda^* = x^* = y^* = 2$, as expected.

7. Quadratic Programs

Definition C.48 (Quadratic Programming Problem). Let

(1) $\mathbf{Q} \in \mathbb{R}^{n \times n}$,
(2) $\mathbf{A} \in \mathbb{R}^{m \times n}$,
(3) $\mathbf{H} \in \mathbb{R}^{l \times n}$,
(4) $\mathbf{b} \in \mathbb{R}^{m \times 1}$,
(5) $\mathbf{r} \in \mathbb{R}^{l \times 1}$ and
(6) $\mathbf{c} \in \mathbb{R}^{n \times 1}$.

Then a quadratic (maximization) programming problem is:

$$QP \begin{cases} \max \ \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ s.t. \ \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \quad \mathbf{H}\mathbf{x} = \mathbf{r} \end{cases} \tag{C.18}$$

EXAMPLE C.49. Example C.6 is an instance of a quadratic programming problem. Recall we had:

$$\begin{cases} \max & A(x,y) = xy \\ s.t. & 2x + 2y = 100 \\ & x \geq 0 \\ & y \geq 0 \end{cases}$$

We can write this as:

$$\begin{cases} \max & \frac{1}{2}\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \\ s.t. & \begin{bmatrix} 2 & 2 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = 100 \\ & \begin{bmatrix} x \\ y \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \end{cases}$$

Obviously, we can put this problem in precisely the format given in Expression C.18, if so desired.

REMARK C.50. There are several specialized algorithms for solving quadratic programs and they emerge naturally in many areas of machine learning (and statistics).

EXERCISE C.6. Suppose $\mathbf{Q}$ is positive-definite. Show that the Wolfe Dual of the quadratic programming problem:

$$QP \begin{cases} \min & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} \\ s.t. & \mathbf{A}\mathbf{x} \leq \mathbf{b} \end{cases}$$

is itself a quadratic programming problem that can be expressed entirely in the dual variables. [Hint: Let $z(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}$. Then $\nabla z = \mathbf{Q}\mathbf{x}$. The constraints can be written as a vector valued function $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ and $\mathbf{g}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$. Let $\boldsymbol{\lambda}$ be the vector of dual variables. Then the Lagrangian function is:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \tfrac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \boldsymbol{\lambda}^T(\mathbf{A}\mathbf{x} - \mathbf{b})$$

The gradient of the Laplacian is then: $\mathbf{Q}\mathbf{x} + \mathbf{A}^T\boldsymbol{\lambda}$ – the transpose is taken in order to make the dimension work in column vectors. Now, just as we did in Example C.47, write: $\mathbf{x} = -\mathbf{Q}^{-1}\mathbf{A}^T\boldsymbol{\lambda}$. Substitute this back into the objective of the Lagrangian dual and evaluate.]

Bibliography

[Ber99] D. P. Bertsekas, *Nonlinear Programming*, 2 ed., Athena Scientific, 1999.

[Dat95] B. N. Datta, *Numerical linear algebra*, Brooks/Cole, 1995.

[Fra99] J. B. Fraleigh, *A First Course in Abstract Algebra*, 6 ed., Addison-Wesley, 1999.

[Gri12a] C. Griffin, *Graph Theory: Penn State Math 485 Lecture Notes (v 1.4.1)*, http://www.personal.psu.edu/cxg286/Math485.pdf, 2011-2012.

[Gri12b] Christopher Griffin, *Numerical Optimization: Penn State Math 555 Lecture Notes*, (CC-BY-NC-SA) Online: https://sites.google.com/site/cgriffin229/home/Math555.pdf, 2012.

[Lan87] S. Lang, *Linear Algebra*, Springer-Verlag, 1987.

[McD85] J. H. McDonald, *Size-related and geographic variation at two enzyme loci in Megalorchestia californiana (amphipoda: Talitridae)*, Heredity **54** (1985), 359–366.

[MT03] J. E. Marsden and A. Tromba, *Vector calculus*, 5 ed., W. H. Freeman, 2003.

[Rud76] W. Rudin, *Principles of mathematical analysis*, McGraw-Hill, 1976.