Chapter 2

Linear functions

In this chapter we introduce linear and affine functions, and describe some common settings where they arise, including regression models.

2.1 Linear functions

Function notation. The notation $f : \mathbf{R}^n \to \mathbf{R}$ means that f is a *function* that maps real n-vectors to real numbers, *i.e.*, it is a scalar-valued function of n-vectors. If x is an n-vector, then $f(x)$, which is a scalar, denotes the *value* of the function f at x. (In the notation $f(x)$, x is referred to as the *argument* of the function.) We can also interpret f as a function of n scalar arguments, the entries of the vector argument, in which case we write $f(x)$ as

$$f(x) = f(x_1, x_2, \ldots, x_n).$$

Here we refer to $x_1, \ldots, x_n$ as the arguments of f. We sometimes say that f is real-valued, or scalar-valued, to emphasize that $f(x)$ is a real number or scalar.

To describe a function $f : \mathbf{R}^n \to \mathbf{R}$, we have to specify what its value is for any possible argument $x \in \mathbf{R}^n$. For example, we can define a function $f : \mathbf{R}^4 \to \mathbf{R}$ by

$$f(x) = x_1 + x_2 - x_4^2$$

for any 4-vector x. In words, we might describe f as the sum of the first two elements of its argument, minus the square of the last entry of the argument. (This particular function does not depend on the third element of its argument.)

Sometimes we introduce a function without formally assigning a symbol for it, by directly giving a formula for its value in terms of its arguments, or describing how to find its value from its arguments. An example is the *sum function*, whose value is $x_1 + \cdots + x_n$. We can give a name to the value of the function, as in $y = x_1 + \cdots + x_n$, and say that y is a function of x, in this case, the sum of its entries.

Many functions are not given by formulas or equations. As an example, suppose $f : \mathbf{R}^3 \to \mathbf{R}$ is the function that gives the lift (vertical upward force) on a particular

airplane, as a function of the 3-vector x, where x_1 is the angle of attack of the airplane (*i.e.*, the angle between the airplane body and its direction of motion), x_2 is its air speed, and x_3 is the air density.

The inner product function. Suppose a is an n-vector. We can define a scalar-valued function f of n-vectors, given by

$$f(x) = a^T x = a_1x_1 + a_2x_2 + \cdots + a_nx_n \tag{2.1}$$

for any n-vector x. This function gives the inner product of its n-vector argument x with some (fixed) n-vector a. We can also think of f as forming a weighted sum of the elements of x; the elements of a give the weights used in forming the weighted sum.

Superposition and linearity. The inner product function f defined in (2.1) satisfies the property

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) \\ &= a^T(\alpha x) + a^T(\beta y) \\ &= \alpha(a^T x) + \beta(a^T y) \\ &= \alpha f(x) + \beta f(y) \end{aligned}$$

for all n-vectors x, y, and all scalars α, β. This property is called *superposition*. A function that satisfies the superposition property is called *linear*. We have just shown that the inner product with a fixed vector is a linear function.

The superposition equality

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y) \tag{2.2}$$

looks deceptively simple; it is easy to read it as just a re-arrangement of the parentheses and the order of a few terms. But in fact it says a lot. On the left-hand side, the term $\alpha x + \beta y$ involves *scalar-vector* multiplication and *vector addition*. On the right-hand side, $\alpha f(x) + \beta f(y)$ involves ordinary *scalar multiplication* and *scalar addition*.

If a function f is linear, superposition extends to linear combinations of any number of vectors, and not just linear combinations of two vectors: We have

$$f(\alpha_1x_1 + \cdots + \alpha_kx_k) = \alpha_1f(x_1) + \cdots + \alpha_kf(x_k),$$

for any n vectors $x_1, \ldots, x_k$, and any scalars $\alpha_1, \ldots, \alpha_k$. (This more general k-term form of superposition reduces to the two-term form given above when $k = 2$.) To see this, we note that

$$\begin{aligned} f(\alpha_1x_1 + \cdots + \alpha_kx_k) &= \alpha_1f(x_1) + f(\alpha_2x_2 + \cdots + \alpha_kx_k) \\ &= \alpha_1f(x_1) + \alpha_2f(x_2) + f(\alpha_3x_3 + \cdots + \alpha_kx_k) \\ &\vdots \\ &= \alpha_1f(x_1) + \cdots + \alpha_kf(x_k). \end{aligned}$$

In the first line here, we apply (two-term) superposition to the argument

$$\alpha_1 x_1 + (1)(\alpha_2 x_2 + \cdots + \alpha_k x_k),$$

and in the other lines we apply this recursively.

The superposition equality (2.2) is sometimes broken down into two properties, one involving the scalar-vector product and one involving vector addition in the argument. A function $f : \mathbf{R}^n \to \mathbf{R}$ is linear if it satisfies the following two properties.

- *Homogeneity.* For any n-vector x and any scalar α, $f(\alpha x) = \alpha f(x)$.
- *Additivity.* For any n-vectors x and y, $f(x + y) = f(x) + f(y)$.

Homogeneity states that scaling the (vector) argument is the same as scaling the function value; additivity says that adding (vector) arguments is the same as adding the function values.

Inner product representation of a linear function. We saw above that a function defined as the inner product of its argument with some fixed vector is linear. The converse is also true: If a function is linear, then it can be expressed as the inner product of its argument with some fixed vector.

Suppose f is a scalar-valued function of n-vectors, and is linear, *i.e.*, (2.2) holds for all n-vectors x, y, and all scalars α, β. Then there is an n-vector a such that $f(x) = a^T x$ for all x. We call $a^T x$ the *inner product representation* of f.

To see this, we use the identity (1.1) to express an arbitrary n-vector x as $x = x_1 e_1 + \cdots + x_n e_n$. If f is linear, then by multi-term superposition we have

$$\begin{aligned} f(x) &= f(x_1 e_1 + \cdots + x_n e_n) \\ &= x_1 f(e_1) + \cdots + x_n f(e_n) \\ &= a^T x, \end{aligned}$$

with $a = (f(e_1), f(e_2), \ldots, f(e_n))$. The formula just derived,

$$f(x) = x_1 f(e_1) + x_2 f(e_2) + \cdots + x_n f(e_n) \tag{2.3}$$

which holds for any linear scalar-valued function f, has several interesting implications. Suppose, for example, that the linear function f is given as a subroutine (or a physical system) that computes (or results in the output) $f(x)$ when we give the argument (or input) x. Once we have found $f(e_1)$, $\ldots$, $f(e_n)$, by n calls to the subroutine (or n experiments), we can predict (or simulate) what $f(x)$ will be, for *any* vector x, using the formula (2.3).

The representation of a linear function f as $f(x) = a^T x$ is *unique*, which means that there is only one vector a for which $f(x) = a^T x$ holds for all x. To see this, suppose that we have $f(x) = a^T x$ for all x, and also $f(x) = b^T x$ for all x. Taking $x = e_i$, we have $f(e_i) = a^T e_i = a_i$, using the formula $f(x) = a^T x$. Using the formula $f(x) = b^T x$, we have $f(e_i) = b^T e_i = b_i$. These two numbers must be the same, so we have $a_i = b_i$. Repeating this argument for $i = 1, \ldots, n$, we conclude that the corresponding elements in a and b are the same, so $a = b$.

Examples.

- *Average.* The *mean* or *average* value of an n-vector is defined as

$$f(x) = (x_1 + x_2 + \cdots + x_n)/n,$$

  and is denoted $\mathbf{avg}(x)$ (and sometimes $\overline{x}$). The average of a vector is a linear function. It can be expressed as $\mathbf{avg}(x) = a^T x$ with

$$a = (1/n, \ldots, 1/n) = \mathbf{1}/n.$$

- *Maximum.* The maximum element of an n-vector x, $f(x) = \max\{x_1, \ldots, x_n\}$, is not a linear function (except when $n = 1$). We can show this by a counterexample for $n = 2$. Take $x = (1, -1)$, $y = (-1, 1)$, $\alpha = 1/2$, $\beta = 1/2$. Then

$$f(\alpha x + \beta y) = 0 \neq \alpha f(x) + \beta f(y) = 1.$$

Affine functions. A linear function plus a constant is called an *affine* function. A function $f : \mathbf{R}^n \to \mathbf{R}$ is affine if and only if it can be expressed as $f(x) = a^T x + b$ for some n-vector a and scalar b, which is sometimes called the *offset*. For example, the function on 3-vectors defined by

$$f(x) = 2.3 - 2x_1 + 1.3x_2 - x_3,$$

is affine, with $b = 2.3$, $a = (-2, 1.3, -1)$.

Any affine scalar-valued function satisfies the following variation on the superposition property:

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y),$$

for all n-vectors x, y, and all scalars α, β that satisfy $\alpha + \beta = 1$. For linear functions, superposition holds for *any* coefficients α and β; for affine functions, it holds *when the coefficients sum to one* (*i.e.*, when the argument is an affine combination).

To see that the restricted superposition property holds for an affine function $f(x) = a^T x + b$, we note that, for any vectors x, y and scalars α and β that satisfy $\alpha + \beta = 1$,

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) + b \\ &= \alpha a^T x + \beta a^T y + (\alpha + \beta)b \\ &= \alpha(a^T x + b) + \beta(a^T y + b) \\ &= \alpha f(x) + \beta f(y). \end{aligned}$$

(In the second line we use $\alpha + \beta = 1$.)

This restricted superposition property for affine functions is useful in showing that a function f is *not* affine: We find vectors x, y, and numbers α and β with $\alpha + \beta = 1$, and verify that $f(\alpha x + \beta y) \neq \alpha f(x) + \beta f(y)$. This shows that f cannot be affine. As an example, we verified above that superposition does not hold for the maximum function (with $n > 1$); the coefficients in our counterexample are $\alpha = \beta = 1/2$, which sum to one, which allows us to conclude that the maximum function is not affine.

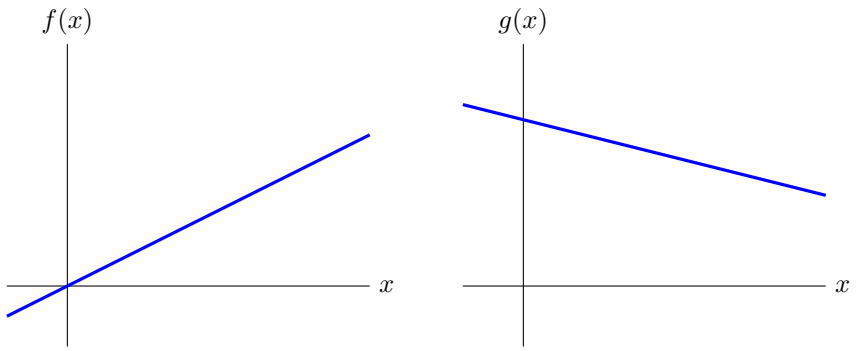

Figure 2.1 *Left.* The function f is linear. *Right.* The function g is affine, but not linear.

The converse is also true: Any scalar-valued function that satisfies the restricted superposition property is affine. An analog of the formula (2.3) is

$$f(x) = f(0) + x_1\left(f(e_1) - f(0)\right) + \cdots + x_n\left(f(e_n) - f(0)\right), \tag{2.4}$$

which holds when f is affine, and x is any n-vector. (See exercise 2.7.) This formula shows that for an affine function, once we know the $n+1$ numbers $f(0)$, $f(e_1)$, $\ldots$, $f(e_n)$, we can predict (or reconstruct or evaluate) $f(x)$ for any n-vector x. It also shows how the vector a and constant b in the representation $f(x) = a^T x + b$ can be found from the function f: $a_i = f(e_i) - f(0)$, and $b = f(0)$.

In some contexts affine functions are called linear. For example, when x is a scalar, the function f defined as $f(x) = \alpha x + \beta$ is sometimes referred to as a linear function of x, perhaps because its graph is a line. But when $\beta \neq 0$, f is not a linear function of x, in the standard mathematical sense; it *is* an affine function of x. In this book we will distinguish between linear and affine functions. Two simple examples are shown in figure 2.1.

A civil engineering example. Many scalar-valued functions that arise in science and engineering are well approximated by linear or affine functions. As a typical example, consider a steel structure like a bridge, and let w be an n-vector that gives the weight of the load on the bridge in n specific locations, in metric tons. These loads will cause the bridge to deform (move and change shape) slightly. Let s denote the distance that a specific point on the bridge sags, in millimeters, due to the load w. This is shown in figure 2.2. For weights the bridge is designed to handle, the sag is very well approximated as a linear function $s = f(x)$. This function can be expressed as an inner product, $s = c^T w$, for some n-vector c. From the equation $s = c_1 w_1 + \cdots + c_n w_n$, we see that $c_1 w_1$ is the amount of the sag that is due to the weight w_1, and similarly for the other weights. The coefficients c_i, which have units of mm/ton, are called *compliances*, and give the sensitivity of the sag with respect to loads applied at the n locations.

The vector c can be computed by (numerically) solving a partial differential equation, given the detailed design of the bridge and the mechanical properties of

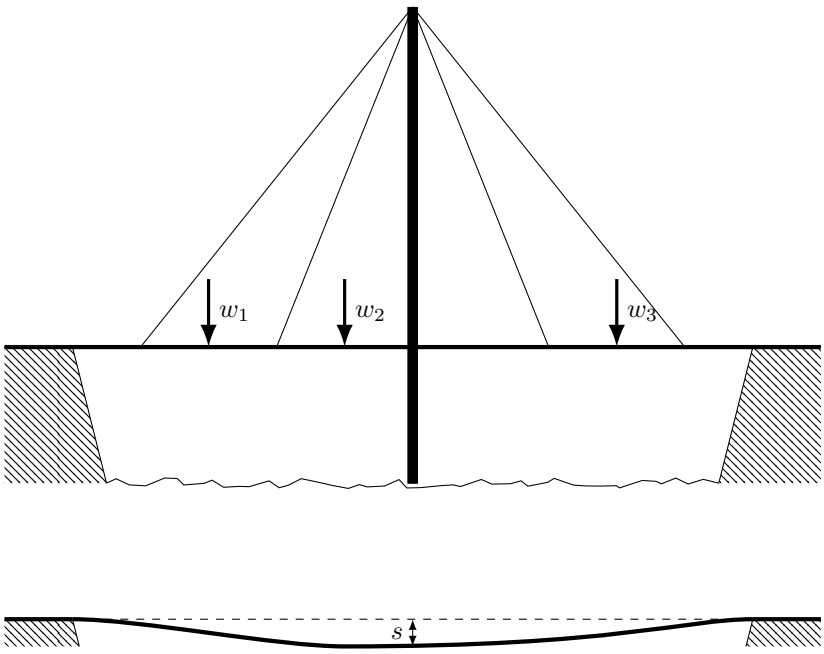

Figure 2.2 A bridge with weights w_1, w_2, w_3 applied in 3 locations. These weights cause the bridge to sag in the middle, by an amount s. (The sag is exaggerated in this diagram.)

w_1	w_2	w_3	Measured sag	Predicted sag
1	0	0	0.12	—
0	1	0	0.31	—
0	0	1	0.26	—
0.5	1.1	0.3	0.481	0.479
1.5	0.8	1.2	0.736	0.740

Table 2.1 Loadings on a bridge (first three columns), the associated measured sag at a certain point (fourth column), and the predicted sag using the linear model constructed from the first three experiments (fifth column).

the steel used to construct it. This is always done during the design of a bridge. The vector c can also be *measured* once the bridge is built, using the formula (2.3). We apply the load $w = e_1$, which means that we place a one ton load at the first load position on the bridge, with no load at the other positions. We can then measure the sag, which is c_1. We repeat this experiment, moving the one ton load to positions $2, 3, \ldots, n$, which gives us the coefficients $c_2, \ldots, c_n$. At this point we have the vector c, so we can now *predict* what the sag will be with any other loading. To check our measurements (and linearity of the sag function) we might measure the sag under other more complicated loadings, and in each case compare our prediction (*i.e.*, $c^T w$) with the actual measured sag.

Table 2.1 shows what the results of these experiments might look like, with each row representing an experiment (*i.e.*, placing the loads and measuring the sag). In the last two rows we compare the measured sag and the predicted sag, using the linear function with coefficients found in the first three experiments.

2.2 Taylor approximation

In many applications, scalar-valued functions of n variables, or relations between n variables and a scalar one, can be *approximated* as linear or affine functions. In these cases we sometimes refer to the linear or affine function relating the variables and the scalar variable as a *model*, to remind us that the relation is only an approximation, and not exact.

Differential calculus gives us an organized way to find an approximate affine model. Suppose that $f : \mathbf{R}^n \to \mathbf{R}$ is differentiable, which means that its partial derivatives exist (see §C.1). Let z be an n-vector. The (first-order) *Taylor approximation* of f near (or at) the point z is the function $\hat{f}(x)$ of x defined as

$$\hat{f}(x) = f(z) + \frac{\partial f}{\partial x_1}(z)(x_1 - z_1) + \cdots + \frac{\partial f}{\partial x_n}(z)(x_n - z_n),$$

where $\frac{\partial f}{\partial x_i}(z)$ denotes the partial derivative of f with respect to its ith argument, evaluated at the n-vector z. The hat appearing over f on the left-hand side is