Matrix Calculus for CS181 (adapted from Math 22b)

Skyler Wu (and Angela Li)

Spring 2022

**Also intended for the brothers and sisters of the Stanford Statistics First-Year Cohort. This is the more rudimentary version. Written by Skyler Wu '24.*

1 Introduction

Broadly, matrix calculus is the study of multivariable calculus over the spaces of matrices, which allows for much cleaner calculations, especially in applications to statistics, machine learning, and econometrics, among others. As we saw in vector calculus, where finding entities such as the gradient or Jacobian allowed us to perform and execute very elegant calculations and proofs, extending this idea to matrix calculus allows us to carry this out to an even greater extent.

The crux of matrix calculus involves compiling the many derivatives of either (1) a multivariate function with respect to one variable or (2) a single function with respect to multiple variables, into a single matrix or vector, analogous to the ideas regarding derivatives that we have already learned in vector calculus. Then, in accordance with the many properties of matrices and matrix operations that we learned in 22a, matrix calculus allows for the execution of elegant calculus via these matrices, bringing together many of the concepts we have touched upon this year.

In this paper, we will first mathematically introduce the idea of matrix calculus, including fundamental properties and notation, and the six main types of matrix derivatives, four of which we have already seen in vector calculus, and two of which are unique to our matrix calculations carried out in this paper. In this section, we will also be making explicit connections back to both 22a and 22b to emphasize the elegant intersections of the concepts. Next, we will move to proving various theorems in the form of matrix derivatives and identities, and will conclude with a brief application of these concepts to machine learning and provide a derivation via matrix calculus for the well-known least squares regression.

2 Fundamentals, Notation, and Basic Properties

2.1 Notation

We will first review some matrix notation conventions from 22a that are relevant to our upcoming proofs.

2.1.1 Scalars, Vectors, and Matrices

Scalars are denoted by non-bolded, lowercase letters, e.g. a, x, y.

Vectors are defined here as matrices with one column, so all vectors are single-column matrices, i.e. column vectors or matrices of dimension $n \times 1$ for $n \in \mathbb{Z}^+$. We denote vectors here with bolded, lowercase letters, e.g. $\mathbf{a}$, $\mathbf{x}$, $\mathbf{y}$.

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

Matrices will be notated throughout this paper with bolded, uppercase letters, e.g. $\mathbf{A}$, $\mathbf{X}$, $\mathbf{Y}$. Let $\mathbf{A}$ be a $m \times n$ matrix, then we have

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

We can also use the following notation to denote the matrix $\mathbf{A}$, where the first subscript, i, refers to the corresponding row in the matrix of that element, and the second subscript, j, refers to its corresponding column in the matrix.

$$\mathbf{A} = [a_{ij}],$$

for $i = 1, 2, \cdots, m$ and $j = 1, 2, \cdots, n$ [9, p. 4].

Furthermore, we will explicitly define a_{ij} to be the entry in the i^{th} row and j^{th} column of $\mathbf{A}$. Note that we explicitly use the lowercase a in order to avoid confusion with some other mathematical objects to be described later.

The **identity matrix**, denoted $\mathbf{I}_n$, where n is the dimension of this square matrix, is given as follows:

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

The **inverse** of the matrix $\mathbf{A}$, if it exists (based on the Inverse Matrix Theorem from 22a), is denoted $\mathbf{A}^{-1}$, where

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

The **transpose** of the matrix $\mathbf{A}$, is the $n \times m$ matrix, denoted $\mathbf{A}^T$ as follows:

$$\mathbf{A}^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Finally, the **determinant** of the matrix $\mathbf{A}$ is denoted by either of the following:

$$|\mathbf{A}| = \det(\mathbf{A}).$$

Now, with the relevant notation in place, we move to a brief discussion of an important distinction in layout of our matrix derivatives.

2.1.2 Numerator vs. Denominator Layout [10, p. 1]

While learning about matrix calculus, we learned that there are a few different ways, notationally, to layout these derivatives, and during our research, found that sticking to one notation throughout a series of calculations is actually imperative to its accuracy. In this paper, we will be using **numerator layout**, as we have been subordinately using throughout 22b, but will take a moment here to make this distinction clear before proceeding.

When taking the derivative of a vector with respect to another vector, i.e. $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$, which we have seen in 22b (with the Jacobian) and will detail further in the next section, there are often two ways people consider laying out the resulting derivative matrix. Assuming $\mathbf{y}$ is an $m \times 1$ column vector and $\mathbf{x}$ is an $n \times 1$ column vector, then this resulting derivative matrix (i.e., the Jacobian matrix), $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$, can either be of size $m \times n$ or of size $n \times m$. This is the primary difference between numerator and denominator layout, which we will now detail more explicitly.

Numerator layout involves laying out the derivative matrix based on the dimensions of $\mathbf{y}$ and $\mathbf{x}^T$: in other words, the resultant $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ matrix should have the same dimensions as $\mathbf{y}\mathbf{x}^T$. Thus, under numerator layout, $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ should be an $m \times n$ matrix, i.e., the Jacobian notation we are familiar with from 22b.

Denominator layout involves laying out the derivative matrix based on $\mathbf{x}$ and $\mathbf{y}^T$: in other words, the resultant $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ should have the same dimensions as $\mathbf{x}\mathbf{y}^T$. Thus, under denominator layout, $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ should be an $n \times m$ matrix.

The implications and structure of numerator layout will be more clear once we introduce the matrix derivatives explicitly in the next section.

Note: Although we have introduced two main layouts here, there are more notational choices than the aforementioned numerator and denominator layouts. In some cases, mathematicians or authors may choose a certain notation independently for one type of derivative, depending on what calculations they are working with. There are a variety of benefits and drawbacks to each notation, so while we are choosing to remain consistent with numerator layout throughout this paper, neither is absolutely better than any other; rather, we simply want to maintain consistency and clarity.

2.2 Six Main Types of Matrix Derivatives

Now that we have established the necessary matrix layout, we will proceed with matrix differentiation. There are six main types of derivatives expressible as matrices, detailed in the table below [10, p. 1].

	Scalar	Vector	Matrix
Scalar	$\frac{\partial y}{\partial x}$	$\frac{\partial \mathbf{y}}{\partial x}$	$\frac{\partial \mathbf{Y}}{\partial x}$
Vector	$\frac{\partial y}{\partial \mathbf{x}}$	$\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$	
Matrix	$\frac{\partial y}{\partial \mathbf{X}}$		

Table 1: The above table summarizes the six main matrix derivatives, four of which we have seen before in 22b (3.3.1-3.3.4) and two of which are new (3.3.5 and 3.3.6), that we will be explaining in the next six subsections. For each derivative in the table, the top row explains what we are taking the derivative of, and the leftmost column is what entity we are taking this derivative with respect to.

In section 2.2.2, we introduced the ideas of numerator and denominator layout, including the key fundamental difference between the two and the importance of sticking with one notation. Again, we will be using numerator notation here, so our partial derivatives with respect to the numerator will be laid out according to the shape of the numerator, and the partial derivatives with respect to the denominator will be laid out according to the shape of the transpose of the denominator.

2.2.1 Scalar by Scalar Derivative

Definition 2.1 (Scalar by Scalar Derivative)**.** From single-variable calculus, the scalar by scalar derivative, denoted

$$\frac{\partial y}{\partial x},$$

is a scalar found by taking the derivative of the scalar y with respect to x. [8, p. 106]

2.2.2 Scalar by Vector Derivative

Definition 2.2 (Scalar by Vector Derivative)**.** From 22b, the scalar by vector derivative, denoted

$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial y}{\partial x_j} \end{bmatrix}$$

is a <u>$1 \times n$ row vector</u>, in which each element is found by taking the derivative of y with respect to that corresponding component of $\mathbf{x}$, assuming y is a scalar and $\mathbf{x}$ is a $n \times 1$ column vector [8, p. 112].

Remark: As we defined in 22b, the gradient of a scalar function f with respect to a vector $x \in \mathbb{R}^n$, notated ∇f, is the transpose of the scalar by vector derivative [8, p. 112].

$$\nabla f = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

From this definition, we saw useful applications to the directional derivative, tangent planes, etc.

2.2.3 Vector by Scalar Derivative

Definition 2.3 (Vector by Scalar Derivative)**.** From 22b, the vector by scalar derivative, denoted

$$\frac{\partial \mathbf{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \frac{\partial y_2}{\partial x} \\ \vdots \\ \frac{\partial y_m}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_i}{\partial x} \end{bmatrix}$$

is a <u>$m \times 1$ column vector</u> (maintaining numerator layout), in which each element is found by taking the derivative of the i^{th} component of a $m \times 1$ column vector $\mathbf{y}$, $i = 1, \cdots, m$ with respect to a scalar x. [8, p. 120]

Remark: As we defined in 22b, the **tangent vector**, whether that be velocity function or another tangent to a curve, is a vector by scalar derivative.

Note on notation: We emphasize again the importance of maintaining numerator notation throughout this paper in order to execute sound calculations. For example, our vector by scalar derivative from this section, $\frac{\partial \mathbf{y}}{\partial x}$, is a $n \times 1$ *column vector* while the scalar by vector derivative from the section before, $\frac{\partial y}{\partial \mathbf{x}}$, is a $1 \times n$ *row vector* assuming $\mathbf{x}$ and $\mathbf{y}$ are both $n \times 1$ column vectors.

2.2.4 Vector by Vector Derivative

Definition 2.4 (Vector by Vector Derivative)**.** From 22b, the vector by vector derivative, more familiarly known to us as the Jacobian matrix, is denoted

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} = \left[\frac{\partial y_i}{\partial x_j}\right]$$

and is a $\underline{m \times n \text{ matrix}}$, in which the i, jth element is found by taking the derivative of the the i^{th} component of a $m \times 1$ column vector $\mathbf{y}$, $i = 1, \cdots, m$ with respect to the j^{th} component of a $n \times 1$ column vector $\mathbf{x}$, $j = 1, \cdots, n$ [8, p. 209].

2.2.5 Scalar by Matrix Derivative

Definition 2.5 (Scalar by Matrix Derivative)**.** The scalar by matrix derivative of a scalar function y with respect to an $m \times n$ matrix $\mathbf{X}$ is an $\underline{n \times m \text{ matrix}}$ in which the i, jth element is found by taking the derivative of y with respect to the j, ith element of $\mathbf{X}$ [essentially, with respect to $\mathbf{X}^{\mathrm{T}}$ [10, p. 1], notated as follows:

$$\frac{\partial y}{\partial \mathbf{X}} = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{21}} & \cdots & \frac{\partial y}{\partial x_{m1}} \\ \frac{\partial y}{\partial x_{12}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{m2}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{1n}} & \frac{\partial y}{\partial x_{2n}} & \cdots & \frac{\partial y}{\partial x_{mn}} \end{bmatrix} = \left[\frac{\partial y}{\partial x_{ji}}\right].$$

Example 2.1. Let $y = \sin(a)b^2e^cd$ and

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

so

$$\mathbf{X}^{\mathrm{T}} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}.$$

Then,

$$\frac{\partial y}{\partial \mathbf{X}} = \begin{bmatrix} \frac{\partial y}{\partial a} & \frac{\partial y}{\partial c} \\ \frac{\partial y}{\partial b} & \frac{\partial y}{\partial d} \end{bmatrix} = \begin{bmatrix} \cos(a)b^2e^cd & \sin(a)b^2e^cd \\ 2\sin(a)be^cd & \sin(a)b^2e^c \end{bmatrix}.$$

2.2.6 Matrix by Scalar Derivative

Definition 2.6 (Matrix by Scalar Derivative)**.** The matrix by scalar derivative of an $m \times n$ matrix $\mathbf{Y}$ with respect to a scalar function x is an $\underline{m \times n \text{ matrix}}$ in which the i, jth element is found by taking the derivative of the i, jth element of $\mathbf{Y}$ with respect to x [10, p. 1], notated as follows:

$$\frac{\partial \mathbf{Y}}{\partial x} = \begin{bmatrix} \frac{\partial y_{11}}{\partial x} & \frac{\partial y_{12}}{\partial x} & \cdots & \frac{\partial y_{1n}}{\partial x} \\ \frac{\partial y_{21}}{\partial x} & \frac{\partial y_{22}}{\partial x} & \cdots & \frac{\partial y_{2n}}{\partial x} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_{m1}}{\partial x} & \frac{\partial y_{m2}}{\partial x} & \cdots & \frac{\partial y_{mn}}{\partial x} \end{bmatrix} = \left[\frac{\partial y_{ij}}{\partial x}\right].$$

Example 2.2. Let

$$\mathbf{Y} = \begin{bmatrix} x & \cos(x) \\ e^x & x^2 \end{bmatrix},$$

then

$$\frac{\partial \mathbf{Y}}{\partial x} = \begin{bmatrix} 1 & -\sin(x) \\ e^x & 2x \end{bmatrix}.$$

2.3 Matrix Differentiation Rules

Thankfully, the differentiation rules we know from Calc I through Math 22b almost directly carry over into matrix differentiation.

2.3.1 Sum Rule

By linearity of differentiation, as with single-variable calculus and vector calculus, the derivative of the sum is the sum of the derivatives [7, p. 8]. Let u and v be scalars, and $\mathbf{X}$ be an $m \times n$ matrix, then we have

$$\frac{\partial(u+v)}{\partial \mathbf{X}} = \frac{\partial u}{\partial \mathbf{X}} + \frac{\partial v}{\partial \mathbf{X}}.$$

Similarly, if $\mathbf{U}$ and $\mathbf{V}$ are $m \times n$ matrices and x is a scalar, we have

$$\frac{\partial(\mathbf{U}+\mathbf{V})}{\partial x} = \frac{\partial \mathbf{U}}{\partial x} + \frac{\partial \mathbf{V}}{\partial x}.$$

2.3.2 Product Rule

Our product rule for matrix derivatives is analogous to those for vectors and scalars [7, p. 8]. Let u and v be scalars, and $\mathbf{X}$ be an $m \times n$ matrix, then we have

$$\frac{\partial(uv)}{\partial \mathbf{X}} = u\frac{\partial v}{\partial \mathbf{X}} + v\frac{\partial u}{\partial \mathbf{X}}.$$

Similarly, if $\mathbf{U}$ and $\mathbf{V}$ are $n \times n$ matrices and x is a scalar, we have

$$\frac{\partial(\mathbf{U}\mathbf{V})}{\partial x} = \mathbf{U}\frac{\partial \mathbf{V}}{\partial x} + \mathbf{V}\frac{\partial \mathbf{U}}{\partial x}.$$

2.3.3 Chain Rule

Finally, our chain rule is also analogous [7, p. 15]. Let u and y be scalars, where u is in terms of $\mathbf{X}$ and y is in terms of u and $\mathbf{X}$ be an $m \times n$ matrix, then we can take the derivative of y with respect to $\mathbf{X}$ as follows:

$$\frac{\partial y}{\partial \mathbf{X}} = \frac{\partial y}{\partial u}\frac{\partial u}{\partial \mathbf{X}}.$$

Similarly, if $\mathbf{Y}$ is a $m \times n$ matrix in terms of u, and u and x are scalars with u in terms of x, then

$$\frac{\partial \mathbf{Y}}{\partial x} = \frac{\partial \mathbf{Y}}{\partial u}\frac{\partial u}{\partial x}.$$

3 Select Matrix Derivatives and Identities

3.1 Thinking Element-wise

The main strategy for computing matrix derivatives is to first find the partial derivative with respect to an arbitrary element, and then generalize our results to the entire matrix/vector as appropriate. To demonstrate the concept of breaking down complex vector and matrix derivatives into element-sized pieces, we will proceed to prove a few theorems/identities that are commonly used in machine learning and statistics.

Note that in this paper, all vectors should be interpreted as column vectors. Out of consideration for paper length, in this paper, we will focus on examples of taking the derivative of a *scalar* function with respect to a vector or matrix.

Theorem 1. $\frac{\partial \mathbf{w}^T\mathbf{x}}{\partial \mathbf{w}} = \frac{\partial \mathbf{x}^T\mathbf{w}}{\partial \mathbf{w}} = \mathbf{x}^T$, for $\mathbf{x}, \mathbf{w} \in \mathbb{R}^n$ [4, p. 4-6].

Proof. Let $f(\mathbf{w}) = \mathbf{w}^T\mathbf{x} = \mathbf{x}^T\mathbf{w}$ (as dot products are commutative). Note that f is a scalar function, so we are taking the derivative of a scalar function with respect to a vector – our answer, per our notational norms, should be a *row* vector. Then, expanding element-wise, we have the following, where x_i is the i^{th} element of $\mathbf{x}$ and w_i is the i^{th} element of $\mathbf{w}$:

$$f(\mathbf{w}) = x_1w_1 + x_2w_2 + \cdots + x_nw_n.$$

We can rewrite our derivative as:

$$\frac{\partial \mathbf{w}^T\mathbf{x}}{\partial \mathbf{w}} = \frac{\partial f}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial f}{\partial w_1} & \frac{\partial f}{\partial w_2} & \cdots & \frac{\partial f}{\partial w_n} \end{bmatrix}.$$

For an arbitary w_i, using our partial derivative knowledge from Math 22b, we know that there is only one relevant term, and thus we have:

$$\frac{\partial f}{\partial w_i} = \frac{\partial(x_1w_1 + x_2w_2 + \cdots + x_nw_n)}{\partial w_i} = x_i.$$

Generalizing this result for all partial derivatives with respect to w_i, for all $i \in 1, 2, \dots n$, we have:

$$\frac{\partial \mathbf{w}^T\mathbf{x}}{\partial \mathbf{w}} = \frac{\partial f}{\partial \mathbf{w}} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix} = \mathbf{x}^T, \text{ as } \mathbf{x} \text{ itself is defined as a column vector.}$$

*note: in some references, the derivative may appear as just $\mathbf{x}$, but for notational consistency, we proceed with the transpose [11, p. 10].

Intuitively, we can think of this result as the vector/matrix analog of $\frac{d}{dx}cx = c$. This result will be crucial in our matrix calculus proof of the least-squares regression solution.

□

Let us try a slightly more involved example.

Theorem 2. $\frac{\partial \mathbf{x}^T\mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{x}^T$, with $\mathbf{x} \in \mathbb{R}^n$ [7, p. 9].

*note: [7, p. 9] reports $2\mathbf{x}$, but we include the transpose for notational consistency.

Proof. We proceed similarly as the example above. Let $f(\mathbf{x}) = \mathbf{x}^T\mathbf{x} = {x_1}^2 + {x_2}^2 + \cdots + {x_n}^2$. Because we are taking the derivative of a scalar function with respect to a vector, by our notational norms, our derivative should turn out to be a *row* vector. We can rewrite our derivative as follows:

$$\frac{\partial \mathbf{x}^T\mathbf{x}}{\partial \mathbf{x}} = \frac{\partial f}{\partial \mathbf{x}} = \frac{\partial({x_1}^2 + {x_2}^2 + \cdots + {x_n}^2)}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}.$$

Heeding the paradigm of proceeding element-wise, let us find the partial derivative with respect to an arbitrary x_i: We can see from the definition of f that there is only one relevant term, and thus, we have:

$$\frac{\partial f}{\partial x_i} = \frac{\partial({x_1}^2 + {x_2}^2 + \cdots + {x_n}^2)}{\partial x_i} = 2x_i.$$

Generalizing this result to all partial derivatives with respect to x_i, for all $i \in 1, 2, \ldots n$, we have:

$$\frac{\partial \mathbf{x}^T\mathbf{x}}{\partial \mathbf{x}} = \frac{\partial f}{\partial \mathbf{x}} = \frac{\partial({x_1}^2 + \cdots + {x_n}^2)}{\partial \mathbf{x}} = \begin{bmatrix} 2x_1 & 2x_2 & \ldots & 2x_n \end{bmatrix}.$$

$$\frac{\partial \mathbf{x}^T\mathbf{x}}{\partial \mathbf{x}} = 2\begin{bmatrix} x_1 & x_2 & \ldots & x_n \end{bmatrix} = 2\mathbf{x}^T, \text{ as } \mathbf{x} \text{ is itself a column vector.}$$

This result is the vector/matrix analog of $\frac{d}{dx}x^2 = 2x$.

□

The next derivative identity is a generalization of the derivative identity we just proved above. In the previous identity, we simply set $\mathbf{A} = \mathbf{I}$, the identity matrix.

Theorem 3. $\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^T(\mathbf{A} + \mathbf{A}^T)$, where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{A}$ is a constant $n \times n$ square matrix [4, p. 6].

Proof. Before we start, we need to remember that $\mathbf{x}^T\mathbf{A}\mathbf{x}$ is scalar. As such, we are still taking the derivative of a scalar with respect to a vector: by the rules of *numerator layout*, our answer should be a *row* vector.

As usual, we will proceed with our element-wise paradigm. To reiterate, we will treat $\mathbf{x}$ as a column vector. Because this derivative is a bit more complicated than the previous two, let us draw out $\mathbf{x}$, $\mathbf{x}^T$, and $\mathbf{A}$ for intuition:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \ \mathbf{x}^T = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \end{bmatrix}, \ \mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}.$$

Let $f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x}$. We approach piece-by-piece. By definition of matrix multiplication, and looking at the shapes of $\mathbf{x}^T$ and $\mathbf{A}$, we see:

$$\mathbf{x}^T\mathbf{A} = \begin{bmatrix} \sum_{i=1}^n x_i a_{i1} & \sum_{i=1}^n x_i a_{i2} & \cdots & \sum_{i=1}^n x_i a_{in} \end{bmatrix}.$$

Now, let us right-multiply by $\mathbf{x}$:

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} = \begin{bmatrix} \sum_{i=1}^n x_i a_{i1} & \sum_{i=1}^n x_i a_{i2} & \cdots & \sum_{i=1}^n x_i a_{in} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \sum_{j=1}^n \left(x_j \sum_{i=1}^n x_i a_{ij} \right).$$

Because the inner summation has no effect on x_j, we can treat it as a "constant" with respect to the inner summation and rewrite our expression as:

$$f(\mathbf{x}) = \sum_{j=1}^{n} \left(x_j \sum_{i=1}^{n} x_i a_{ij} \right) = \sum_{j=1}^{n} \sum_{i=1}^{n} (x_i x_j a_{ij}).$$

Now, abiding by our paradigm of proceeding element-wise and then generalizing, let us take the partial derivative of f with respect to an arbitrary x_k (we use k to avoid confusion with the indexing of our nested summations). One non-trivial challenge is to consider which terms in our nested summation are relevant to our partial derivative with respect to x_k – simply speaking, any terms in our nested summation that have an x_k are relevant terms. A particular term in our nested summation contains an x_k if $i = k$, $j = k$, or both $i, j = k$.

Naively, we could try to find the relevant terms by setting $j = k$ and $i = k$ in our nested summation separately to get:

$$\sum_{i=1}^{n} x_i x_k a_{ik} + \sum_{j=1}^{n} x_k x_j a_{kj},$$

as our relevant terms, but this is *not* correct!

The expression above is incorrect because we overcount the term $x_k x_k a_{kk}$, which appears in both (now-separated) summation expressions – once when $i = k$ in the first summation, and once when $j = k$ in the second summation. Thus, the true list of relevant terms is actually:

$$(\sum_{i=1}^{n} x_i x_k a_{ik}) + (\sum_{j=1}^{n} x_k x_j a_{kj}) - x_k x_k a_{kk}.$$

Having found the relevant terms (relevant, as in they do not disappear when taking the partial derivative with respect to x_k), we may be tempted to directly take the partial derivative with respect to x_k, but there is a dangerous nuance:

$$\frac{\partial (x_k x_k a_{kk})}{\partial x_k} = 2x_k a_{kk}, \text{ while } \frac{\partial (x_i x_k a_{ik})}{\partial x_k} = x_i a_{ik}, \text{ if } i \neq k.$$

To account for this nuance, let us rewrite the list of relevant terms. Let $f_{\text{relevant}}(\mathbf{x})$ represent the relevant terms to the partial derivative with respect to x_k:

$$f_{\text{relevant}}(\mathbf{x}) = (\sum_{i=1}^{n} x_i x_k a_{ik}) + (\sum_{j=1}^{n} x_k x_j a_{kj}) - (x_k x_k a_{kk}).$$

$$f_{\text{relevant}}(\mathbf{x}) = \left((\sum_{i=1, i\neq k}^{n} x_i x_k a_{ik}) + x_k x_k a_{kk} \right) + \left((\sum_{j=1, j\neq k}^{n} x_k x_j a_{kj}) + x_k x_k a_{kk} \right) - x_k x_k a_{kk}.$$

Now, we can finally take the partial derivative with respect to x_k:

$$\frac{\partial f}{\partial x_k} = \frac{\partial f_{\text{relevant}}}{\partial x_k} = \left((\sum_{i=1, i\neq k}^{n} x_i a_{ik}) + 2x_k a_{kk} \right) + \left((\sum_{j=1, j\neq k}^{n} x_j a_{kj}) + 2x_k a_{kk} \right) - 2x_k a_{kk}.$$

$$\frac{\partial f}{\partial x_k} = (\sum_{i=1, i\neq k}^{n} x_i a_{ik}) + (\sum_{j=1, j\neq k}^{n} x_j a_{kj}) + 2x_k a_{kk}.$$

$$\frac{\partial f}{\partial x_k} = (\sum_{i=1, i\neq k}^{n} x_i a_{ik}) + x_k a_{kk} + (\sum_{j=1, j\neq k}^{n} x_j a_{kj}) + x_k a_{kk}.$$

$$\frac{\partial f}{\partial x_k} = (\sum_{i=1}^{n} x_i a_{ik}) + (\sum_{j=1}^{n} x_j a_{kj}).$$

At this point, we proceed to generalize our expression for an arbitrary partial derivative to the larger matrix/vector derivative:

$$\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \begin{bmatrix} (\sum_{i=1}^{n} x_i a_{i1} + \sum_{j=1}^{n} x_j a_{1j}) & (\sum_{i=1}^{n} x_i a_{i2} + \sum_{j=1}^{n} x_j a_{2j}) & \dots & (\sum_{i=1}^{n} x_i a_{in} + \sum_{j=1}^{n} x_j a_{nj}) \end{bmatrix}.$$

To make things easier, let us rewrite the derivative as the sum of two vectors:

$$\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \begin{bmatrix} \sum_{i=1}^{n} x_i a_{i1} & \sum_{i=1}^{n} x_i a_{i2} & \dots & \sum_{i=1}^{n} x_i a_{in} \end{bmatrix} + \begin{bmatrix} \sum_{j=1}^{n} x_j a_{1j} & \sum_{j=1}^{n} x_j a_{2j} & \dots & \sum_{j=1}^{n} x_j a_{nj} \end{bmatrix}.$$

At this point we refer back to the visuals of $\mathbf{x}^T$ and $\mathbf{A}$:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \ \mathbf{x}^T = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}, \ \mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}.$$

And, we realize the following:

$$\mathbf{x}^T\mathbf{A} = \begin{bmatrix} \sum_{i=1}^{n} x_i a_{i1} & \sum_{i=1}^{n} x_i a_{i2} & \dots & \sum_{i=1}^{n} x_i a_{in} \end{bmatrix}.$$

We also realize that if

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \sum_{j=1}^{n} x_j a_{1j} \\ \sum_{j=1}^{n} x_j a_{2j} \\ \vdots \\ \sum_{j=1}^{n} x_j a_{nj} \end{bmatrix},$$

then

$$(\mathbf{A}\mathbf{x})^T = \mathbf{x}^T\mathbf{A}^T = \begin{bmatrix} \sum_{j=1}^{n} x_j a_{1j} & \sum_{j=1}^{n} x_j a_{2j} & \dots & \sum_{j=1}^{n} x_j a_{nj} \end{bmatrix}.$$

Lastly, substituting these compact matrix expressions back into our expression for the overall matrix derivative, we obtain the following final result:

$$\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^T\mathbf{A} + \mathbf{x}^T\mathbf{A}^T = \mathbf{x}^T(\mathbf{A} + \mathbf{A}^T).$$

One final observation about this result is that if $\mathbf{A}$ is symmetrical (which occurs quite often in the case of Kernel-Based Regression and other machine learning algorithms), we have the following neat form:

$$\frac{\partial \mathbf{x}^T \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^T (2\mathbf{A}).$$

$\square$

*note: some texts report the result as $(\mathbf{A} + \mathbf{A}^T)\mathbf{x}$, but once again, we are following the *numerator layout.*

3.2 Derivative of the Inverse of a Matrix

While we will present the majority of our derivations in this paper proceeding element-wise, we include this theorem to demonstrate the utility of the matrix product rule.

Theorem 4. Let $\mathbf{Y}$ be a $n \times n$ matrix with an inverse, $\mathbf{Y}^{-1}$, where all the elements of $\mathbf{Y}$ are functions of the scalar x. Then, the derivative of $\mathbf{Y}^{-1}$ is

$$\frac{\partial \mathbf{Y}^{-1}}{\partial x} = -\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x} \mathbf{Y}^{-1}.$$

Proof. By definition of the inverse of a matrix, we know

$$\mathbf{Y}\mathbf{Y}^{-1} = \mathbf{I}_n.$$

Differentiating, using the product rule for matrix differentiation, we get

$$\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x} + \frac{\partial \mathbf{Y}^{-1}}{\partial x} \mathbf{Y} = 0,$$

where the derivative of the identity matrix with respect to x equals zero since all elements of $\mathbf{I}_n$ are scalars of value 1. Now, we can rearrange the above equation, so we have

$$\frac{\partial \mathbf{Y}^{-1}}{\partial x} \mathbf{Y} = -\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x}.$$

Finally, we right-multiply both sides of the equation by $\mathbf{Y}^{-1}$ and applying the first property that $\mathbf{Y}\mathbf{Y}^{-1} = \mathbf{I}_n$ to get our final result [1, p. 364].

$$\frac{\partial \mathbf{Y}^{-1}}{\partial x} \mathbf{Y}\mathbf{Y}^{-1} = -\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x} \mathbf{Y}^{-1}. \tag{1}$$

$$\frac{\partial \mathbf{Y}^{-1}}{\partial x} \mathbf{I}_n = -\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x} \mathbf{Y}^{-1}. \tag{2}$$

$$\frac{\partial \mathbf{Y}^{-1}}{\partial x} = -\mathbf{Y}^{-1} \frac{\partial \mathbf{Y}}{\partial x} \mathbf{Y}^{-1}. \tag{3}$$

$\square$

3.3 Derivative of the Determinant of a Matrix with respect to Itself

The following theorem was selected because it is used in calculations involving the Multivariate Gaussian distribution, which is widely used in statistical and machine learning applications. Specifically, the probability density function of the Multivariate Gaussian distribution involves taking the determinant of the covariance matrix. We will omit further discussion of the Multivariate Gaussian distribution, and proceed to focus on the derivative.

Theorem 5. $\dfrac{\partial \det(\mathbf{A})}{\partial \mathbf{A}} = \mathbf{A}^{\#} = \det(\mathbf{A})\mathbf{A}^{-1}$, where $\mathbf{A}^{\#}$ is the *adjoint matrix* of $\mathbf{A}$.

Proof. Before we start the formal proof, we must introduce the *submatrix*, *cofactor matrix* and *adjoint matrix.*

Definition 3.1 (Submatrix)**.** If $\mathbf{A}$ is a matrix, then the *submatrix* $\mathbf{A_{ij}}$ (note the use of the uppercase $\mathbf{A_{ij}}$) of $\mathbf{A}$ is the matrix formed by deleting the i^{th} row and j^{th} column of $\mathbf{A}$ [8, p. A15].

Definition 3.2 (Cofactor Matrix)**.** For $n \times n$ square matrix $\mathbf{A}$, let a_{ij} be the entry in the i^{th} row and j^{th} column of $\mathbf{A}$. Let $\mathbf{A_{ij}}$ be the submatrix of $\mathbf{A}$ corresponding to a_{ij} (i.e. deleting row i and column j). The corresponding cofactor, C_{ij}, of a_{ij} is defined as $(-1)^{i+j}\det(\mathbf{A_{ij}})$ [9, p. 11].

Recall from Math 22a, that we can find the determinant of $\mathbf{A}$ through cofactor expansion down any row or column. For example, if we were to calculate the determinant via cofactor expansion of the first row of $\mathbf{A}$, then we have:

$$\det(\mathbf{A}) = \sum_{j=1}^{n} a_{1j}C_{1j}.$$

The *cofactor matrix*, $\mathbf{C}$ of $n \times n$ square matrix $\mathbf{A}$ is thus:

$$\mathbf{C} = \begin{bmatrix} C_{11} & \cdots & C_{1n} \\ C_{21} & \cdots & C_{2n} \\ \vdots & \ddots & \vdots \\ C_{n1} & \cdots & C_{nn} \end{bmatrix}.$$

Definition 3.3 (Adjoint Matrix)**.** For an $n \times n$ square matrix $\mathbf{A}$, if $\mathbf{C}$ is the cofactor matrix of $\mathbf{A}$, then the *adjoint matrix* of $\mathbf{A}$, denoted as $\mathbf{A}^{\#}$, is defined as $\mathbf{A}^{\#} = \mathbf{C}^T$ [9, p. 11].

Let us write out the terms of $\mathbf{A}^{\#}$ for intuition, as we will be using it quite frequently in our proof:

$$\mathbf{A}^{\#} = \begin{bmatrix} C_{11} & \cdots & C_{n1} \\ C_{12} & \cdots & C_{n2} \\ \vdots & \ddots & \vdots \\ C_{1n} & \cdots & C_{nn} \end{bmatrix}.$$

Lemma. If $\mathbf{A}$ is invertible, then $\mathbf{A}^{-1} = \dfrac{\mathbf{A}^{\#}}{\det(\mathbf{A})}$.

Proof. (of the lemma)

We can rewrite this statement as:

$$\det(\mathbf{A})\mathbf{I} = \mathbf{A}\mathbf{A}^{\#}, \text{ where } \mathbf{I} \text{ is the } n \times n \text{ identity matrix.}$$

Recall, by definition of cofactor expansion, that we can calculate the determinant of $n \times n$ square matrix $\mathbf{A}$ by expanding across any row k:

$$\det(\mathbf{A}) = \sum_{j=1}^{n} a_{kj}C_{kj}$$

Let $\mathbf{Z} = \mathbf{A}\mathbf{A}^{\#}$. If $i = j$, then by definition of matrix multiplication (taking row i from $\mathbf{A}$ and column j from $\mathbf{A}^{\#}$ as we wrote out above),

$$z_{ij} = z_{ii} = a_{i1}C_{i1} + a_{i2}C_{i2} + \cdots + a_{in}C_{in} = \det(\mathbf{A})$$

In other words, because what we proved above holds for all z_{ii}, we have shown that the diagonal entries of $\mathbf{Z}$ are all $\det(\mathbf{A})$. Now, take $i \neq j$. By definition of matrix multiplication, we have:

$$z_{ij} = a_{i1}C_{j1} + a_{i2}C_{j2} + \cdots + a_{in}C_{jn}$$

Consider the matrix $\mathbf{A}'$, which is obtained by replacing the j^{th} row of $\mathbf{A}$ with the i^{th} row, so that we have 2 rows with the same entries (inspired by [6]). By the Invertible Matrix Theorem, because the rows of $\mathbf{A}'$ are clearly not linearly independent, $\mathbf{A}'^T$ is not invertible and thus has a determinant of 0. Because the determinant of the transpose is equal to the determinant of the original matrix, it follows that $\det(\mathbf{A}') = 0$.

With this in mind, let us write out the expression for $\det(\mathbf{A}')$ via cofactor expansion across its j^{th} row, noting that the j^{th} row of $\mathbf{A}'$ is, by our intention, the i^{th} row of $\mathbf{A}$. Also, note that because we are doing cofactor expansion across the j^{th} row of $\mathbf{A}'$, this means that the cofactors we are concerned with are calculated from submatrices that all do not include any elements of the j^{th} row. Thus, the cofactors we are working with can be directly copied from the cofactors corresponding to the elements in the j^{th} row of the original matrix $\mathbf{A}$. Thus, we have:

$$\det(\mathbf{A}') = 0 = a_{i1}C_{j1} + a_{i2}C_{j2} + \cdots + a_{in}C_{jn} = z_{ij},$$
$$\text{from above.}$$

The key result here is that if $i \neq j$, then $z_{ij} = 0$, for $\mathbf{Z} = \mathbf{A}\mathbf{A}^{\#}$. Thus, we conclude that $\mathbf{A}\mathbf{A}^{\#}$ is a matrix with its diagonal entries equal to $\det(\mathbf{A})$, and all other entries equal to 0.

In other words, we have proven the following:

$$\det(\mathbf{A})\mathbf{I} = \mathbf{A}\mathbf{A}^{\#}.$$

□

Now, we can proceed uninterrupted with our proof of the derivative of the determinant with respect to the matrix itself. Note that the determinant of a $n \times n$ square matrix $\mathbf{A}$ is always a scalar. By our understanding of the derivative of a scalar with respect to a matrix, we know that the resultant derivative should have the same dimensions as the *transpose* of the original matrix, to stay consistent with notation.

Following our paradigm of proceeding element-wise, we have the following:

$$\frac{\partial \det(\mathbf{A})}{\partial \mathbf{A}} = \begin{bmatrix} \frac{\partial \det(\mathbf{A})}{\partial a_{11}} & \cdots & \frac{\partial \det(\mathbf{A})}{\partial a_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \det(\mathbf{A})}{\partial a_{n1}} & \cdots & \frac{\partial \det(\mathbf{A})}{\partial a_{nn}} \end{bmatrix}^T.$$

*note: the presence of the transpose is to stay consistent with our *numerator layout* notational norms. We work with the transpose of the derivative matrix (as opposed to just relabeling our indices) because we want our indices to be consistent across $\mathbf{A}$, $\mathbf{C}$, and $\mathbf{A}^{\#}$ as much as possible.

The crucial intuition here is that we can find the determinant of a square matrix via cofactor expansion along *any* of its rows. Thus, to calculate each of the n^2 entries of our derivative matrix (or its transpose, in our case), we can be selective in how we express $\det(\mathbf{A})$ [7, p. 10]. Namely, when trying to find any entry in the i^{th} row of (the transpose) of the derivative matrix, we will express the determinant as the cofactor expansion down the i^{th} row of $\mathbf{A}$.

The entry in the i^{th} row and j^{th} column of (the transpose) of this derivative matrix, for all valid indices i, j, can be written as below:

$$\frac{\partial \det(\mathbf{A})}{\partial a_{ij}} = \frac{\partial(\sum_{k=1}^{n} a_{ik} C_{ik})}{\partial a_{ij}}.$$

Note that a_{ij} itself appears in one term, and one term only, in this particular summation expression for the determinant. Also, note that a_{ij} has no influence on C_{ij}, as we explicitly cross out the i^{th} row and j^{th} column when calculating C_{ij}. Thus, we have our result:

$$\frac{\partial \det(\mathbf{A})}{\partial a_{ij}} = \frac{\partial(\sum_{k=1}^{n} a_{ik} C_{ik})}{\partial a_{ij}} = C_{ij}.$$

Having found the partial derivative (i.e. one entry) with respect to an arbitrary a_{ij}, we conclude the following:

$$\frac{\partial \det(\mathbf{A})}{\partial \mathbf{A}} = \begin{bmatrix} \frac{\partial \det(\mathbf{A})}{\partial a_{11}} & \cdots & \frac{\partial \det(\mathbf{A})}{\partial a_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \det(\mathbf{A})}{\partial a_{n1}} & \cdots & \frac{\partial \det(\mathbf{A})}{\partial a_{nn}} \end{bmatrix}^T = \begin{bmatrix} C_{11} & \cdots & C_{1n} \\ \vdots & \ddots & \vdots \\ C_{n1} & \cdots & C_{nn} \end{bmatrix}^T = \mathbf{C}^T.$$

At this point, we know that $\mathbf{C}^T = \mathbf{A}^{\#}$. From our lemma, we also know that $\mathbf{A}^{-1} = \dfrac{\mathbf{A}^{\#}}{\det(\mathbf{A})}$. Rewriting this expression to isolate the adjoint matrix, we have (presuming $\mathbf{A}$ is invertible):

$$\det(\mathbf{A})\mathbf{A}^{-1} = \mathbf{A}^{\#}.$$

Next, to bring in $\mathbf{C}^T$, we have:

$$\mathbf{C}^T = \mathbf{A}^{\#} = \det(\mathbf{A})\mathbf{A}^{-1}.$$

Finally, we have proven our theorem:

$$\frac{\partial \det(\mathbf{A})}{\partial \mathbf{A}} = \mathbf{A}^{\#} = \det(\mathbf{A})\mathbf{A}^{-1}.$$

□

4 Application to Statistics and Machine Learning

Finally, we will show how matrix calculus is integral to statistics and machine learning by demonstrating its applications to linear regression.

4.1 Matrix Calculus Derivation of Optimal Least-Squares Linear Regression

4.1.1 Introducing the Problem

Given a set of n datapoints in the form $(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots (\mathbf{x_n}, y_n)$, where $\mathbf{x_i} \in \mathbb{R}^D$ and $y_i \in \mathbb{R}^1$ for $i = 1, 2, \ldots n$, we want to find the line-of-best-fit (or plane, or hyperplane, depending on the dimensions of our data) that best captures the relationship between our input variables $\mathbf{x_1}, \ldots \mathbf{x_n}$ and our output variables $y_1, \ldots y_n$ [5, p. 6]. For sake of notation, we will treat each individual $\mathbf{x_i}$ as a *column* vector.

Remark: In this section, we will generalize the word "line" to mean not just a literal line in 2D, but also its analogs in higher dimensions (i.e., plane, hyperplane, etc.).

Given that our input variables are all in $\mathbb{R}^D$ (D as in "dimension"), our line-of-best-fit can be written in the following form, where $\hat{y_i}$ represents our predicted value (i.e. best guess) of y_i for a given input $\mathbf{x_i}$. Let $\mathbf{x_i}$ be expressed as $(x_{i1}, \dots x_{iD})$.

$$\hat{y_i} = w_0 + w_1 x_{i1} + \cdots + w_D x_{iD}.$$

Note the inclusion of the constant term w_0 in this expression. Let the *column* vector $\mathbf{w} = (w_0, w_1, \dots w_D)$ be defined as the weights of our line-of-best-fit. Observe that $\mathbf{w} \in \mathbb{R}^{D+1}$, as opposed to $\mathbb{R}^D$.

To account for the constant coefficient term, w_0, and to make calculations cleaner, from this point onwards, we will redefine each $\mathbf{x_i}$ by appending a 1 to the beginning of each of our input datapoints: now, $\mathbf{x_i} = (1, x_{i1}, \dots x_{iD})$ for $i = 1, 2, \dots n$. Now, note that each $\mathbf{x_i} \in \mathbb{R}^{D+1}$. This trick is called the the *bias trick* [5, p. 7]. This is useful because we can now write each of our predicted values $\hat{y_i}$ as $\hat{y_i} = \mathbf{w}^T \mathbf{x_i}$.

Our goal now is to find the best set of weights, $\mathbf{w}$ for our dataset. Observe that $y_i - \hat{y_i}$ is the error/difference between the actual y-value and our prediction. We define the *best* set of weights, $\mathbf{w}^*$, as the set of weights that minimizes the sum of the squared errors across our data:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \hat{y_i})^2.$$

We include the squared error for the following intuitive reason: If our set of weights overpredicts y_j by 1 and underpredicts y_i by 1, the sum of the errors of these two datapoints would be 0, which is clearly not very reflective of the goodness of our model. The squared error helps us minimize the *magnitude* of the errors. We call this expression for the error the *least-squares error.* Now, we can proceed to set up our optimization.

4.1.2 Setting Up the Optimization

Recall that because of the bias trick, we can write each of our predicted outputs as $\hat{y_i} = \mathbf{w}^T \mathbf{x_i}$ (note that the *same* set of weights $\mathbf{w}$ are used across all datapoints!). Thus, we can rewrite the equation we want to minimize as:

$$f(\mathbf{w}) = \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x_i})^2.$$

4.1.3 Taking the Matrix Derivative [5, p. 11]

To find the minima, we will proceed to take the derivative/gradient of f with respect to the entire vector $\mathbf{w}$ and set it equal to 0. By the chain rule and the sum rule, we have:

$$\frac{\partial f}{\partial \mathbf{w}} = \sum_{i=1}^{n} 2(y_i - \mathbf{w}^T \mathbf{x_i}) \frac{\partial(-\mathbf{w}^T \mathbf{x_i})}{\partial \mathbf{w}} = 0.$$

We are confident that taking the gradient will help us find the true optimal weights that minimize our least-squares error because f is a convex function. For sake of time, we will omit the proof of why f is a convex function. But, the implication of f being a convex function, for our purposes, is that any local minimum of f is guaranteed to be a global minimum of f [3, p. 2]. Here, we encounter a matrix derivative that we had proven earlier in this paper:

$$\frac{\partial(-\mathbf{w}^T\mathbf{x_i})}{\partial \mathbf{w}} = -\frac{\partial(\mathbf{w}^T\mathbf{x_i})}{\partial \mathbf{w}} = -\mathbf{x_i}^T.$$

Substituting, we have:

$$\frac{\partial f}{\partial \mathbf{w}} = \sum_{i=1}^{n} 2(y_i - \mathbf{w}^T\mathbf{x_i})(-\mathbf{x_i}^T) = 0.$$

To make things cleaner, we can multiply everything by $-\frac{1}{2}$ and then split apart the summation expression:

$$\sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x_i})(\mathbf{x_i}^T) = \sum_{i=1}^{n}\left(y_i\mathbf{x_i}^T - (\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}^T\right) = \sum_{i=1}^{n} y_i\mathbf{x_i}^T - \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}^T = 0.$$

Discarding the intermediate expressions, we have:

$$\sum_{i=1}^{n} y_i\mathbf{x_i}^T - \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}^T = 0.$$

It follows that

$$\sum_{i=1}^{n} y_i\mathbf{x_i}^T = \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}^T.$$

Because $\mathbf{w}^T\mathbf{x}$ and y_i are scalars, we can take the transpose of both sides of our equation to get rid of the transpose signs and clean things up:

$$(\sum_{i=1}^{n} y_i\mathbf{x_i}^T)^T = \left(\sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}^T\right)^T.$$

$$\sum_{i=1}^{n} y_i\mathbf{x_i} = \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}.$$

Recall from property of dot products, that the output of a dot product between two vectors is always scalar, and recall from general vector properties, that for scalar c and vector $\mathbf{x}$, $c\mathbf{x} = \mathbf{x}c$. Also, recall that dot products are commutative: $\mathbf{w}^T\mathbf{x} = \mathbf{x}^T\mathbf{w}$. Thus, because we are trying to solve for the optimal $\mathbf{w}$, we have:

$$\sum_{i=1}^{n} y_i\mathbf{x_i} = \sum_{i=1}^{n}(\mathbf{w}^T\mathbf{x_i})\mathbf{x_i} = \sum_{i=1}^{n}\mathbf{x_i}(\mathbf{w}^T\mathbf{x_i}) = \sum_{i=1}^{n}\mathbf{x_i}(\mathbf{x_i}^T\mathbf{w}).$$

Once again, discarding the intermediate expressions, we have:

$$\sum_{i=1}^{n} y_i\mathbf{x_i} = \sum_{i=1}^{n}\mathbf{x_i}(\mathbf{x_i}^T\mathbf{w}).$$

4.1.4 Converting to Matrix Form

Our end goal is to find and isolate the optimal $\mathbf{w}$. To get rid of the summation signs, we can rewrite the sums as the products of matrices. At this point, let us define $\mathbf{X}$ to be an n (no. of datapoints) $\times$ $(D+1)$ (dimensions of each $\mathbf{x_i}$, with the appended 1) matrix:

$$\mathbf{X} = \begin{bmatrix} \leftarrow \mathbf{x_1}^T \rightarrow \\ \leftarrow \mathbf{x_2}^T \rightarrow \\ \vdots \\ \leftarrow \mathbf{x_n}^T \rightarrow \end{bmatrix} = \begin{bmatrix} x_{11} & \cdots & x_{1(D+1)} \\ x_{21} & \cdots & x_{2(D+1)} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{n(D+1)} \end{bmatrix}.$$

*note: we add in the transpose symbol because, as defined above, $\mathbf{x_i}$ are column vectors.
*note: because of the bias trick, $x_{i1} = 1$, for $i = 1, 2, \ldots, n$.

For clarity, let us write out $\mathbf{X}^T \in \mathbb{R}^{(D+1)\times n}$ as well: it will become useful for intuition very soon.

$$\mathbf{X}^T = \begin{bmatrix} x_{11} & \cdots & x_{n1} \\ x_{12} & \cdots & x_{n2} \\ \vdots & \ddots & \vdots \\ x_{1(D+1)} & \cdots & x_{n(D+1)} \end{bmatrix}.$$

Let us define $\mathbf{y}$ to be an $n \times 1$ *column* vector with entries $y_1, \ldots y_n$:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

The key observation here is that, by definition of matrix multiplication, $\mathbf{X}^T\mathbf{y} = \sum_{i=1}^{n} y_i \mathbf{x_i}$.

Next, note that because $\mathbf{w}$ can be treated as a constant and is unchanged by the indices of the summation, as opposed to $\mathbf{x_i}$, which clearly changes with i, we can pull the $\mathbf{w}$ out of the summation:

$$\sum_{i=1}^{n} \mathbf{x_i}(\mathbf{x_i}^T\mathbf{w}) = (\sum_{i=1}^{n} \mathbf{x_i}\mathbf{x_i}^T)\mathbf{w}.$$

Observe that for all $\mathbf{x_i}$, $\mathbf{x_i}\mathbf{x_i}^T$ returns an $\mathbb{R}^{(D+1)\times(D+1)}$ matrix (as $\mathbf{x_i}$ is a column vector and $\mathbf{x_i}^T$ is a row vector). Naturally, the summation of all the $\mathbf{x_i}\mathbf{x_i}^T$ should also be $\mathbb{R}^{(D+1)\times(D+1)}$.

To find an elegant, compact matrix expression for this not-so-friendly-looking summation, we proceed by the element-wise paradigm. Note that if $\mathbf{Q} = \mathbf{x_k}\mathbf{x_k}^T$ (using k to avoid confusion), then $q_{ij} = x_{ki}x_{kj}$.

Now, consider $\mathbf{Z} = \sum_{k=1}^{n} \mathbf{x_k}\mathbf{x_k}^T$: it follows that $z_{ij} = \sum_{k=1}^{n} x_{ki}x_{kj}$.
Let $\mathbf{S} = \mathbf{X}^T\mathbf{X}$, with $\mathbf{X}$ as defined earlier. Recall that $\mathbf{X}$ is constructed by vertically stacking $\mathbf{x_i}^T$'s.

With this definition of $\mathbf{S}$, and our intuition about matrix multiplication, we conclude that $s_{ij} = \sum_{k=1}^{n} x_{ki}x_{kj}$.

Because $\mathbf{S} \in \mathbb{R}^{(D+1)\times(D+1)}$, $\mathbf{Z} \in \mathbb{R}^{(D+1)\times(D+1)}$, and because $s_{ij} = z_{ij} = \sum_{k=1}^{n} x_{ki}x_{kj}$, it follows that $\mathbf{X}^T\mathbf{X} = \mathbf{S} = \mathbf{Z} = \sum_{k=1}^{n} \mathbf{x_k}\mathbf{x_k}^T$.

4.1.5 Moore-Penrose Pseudoinverse Result

Rewriting our optimization results from **4.1.3**, we have:

$$\sum_{i=1}^{n} y_i \mathbf{x_i} = \sum_{i=1}^{n} (\mathbf{w}^T\mathbf{x_i})\mathbf{x_i}.$$

$$\sum_{i=1}^{n} y_i \mathbf{x_i} = (\sum_{i=1}^{n} \mathbf{x_i}\mathbf{x_i}^T)\mathbf{w}.$$

Substituting our matrix expressions for these summations, we have:

$$\mathbf{X}^T\mathbf{y} = (\mathbf{X}^T\mathbf{X})\mathbf{w}.$$

Assuming that $\mathbf{X}^T\mathbf{X}$ is invertible (we omit the proof for now), we can solve for $\mathbf{w}^*$ [5, p. 11]:

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

*note: the $\mathbf{w}^*$ is used to indicate that this is our optimal set of weights.
*note: [9, p. 287-288] arrives at the same result using a different proof.

The expression $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ is known as the *Moore-Penrose Pseudoinverse*, which itself has many unique properties that are outside the scope of this paper. Intuitively, we can think of it as the equivalent of the "inverse" of a non-square matrix [5, p. 11].

4.2 Toy Example

To provide a more numerical, illustrative example of the Optimal Least-Squares problem, we created a toy data set and attempted to find the line (or in this case, plane, because each $\mathbf{x_i} \in \mathbb{R}^2$) of best fit.

The Python code implementation can be found here.

For this toy example, we will be working with data of the form $(\mathbf{x_i} = (x_{i1}, x_{i2}), y_i)$. In other words, for each data point, our $\mathbf{x_i} \in \mathbb{R}^2$, and our $y_i \in \mathbb{R}^1$.

To generate our toy data set, we randomly generated 20 datapoints, with each x_{i1} and x_{i2} generated from a random Uniform distribution between 0 and 20. We generated each y_i through the following simple formula, where $\epsilon \sim N(0, 0.1^2)$ (i.e., mean 0 and standard deviation 0.1) is included to generate some random noise:

$$y_i = 2x_{i1} + 3x_{i2} + \epsilon.$$

Thus, we would expect the calculated vector of optimal weights $\mathbf{w}^*$ to be somewhere around $w_0^* = 0$ (because the mean of the noise term is 0), $w_1^* = 2$, $w_2^* = 3$, for our plane of best fit:

$$\hat{y_i} = w_0^* + w_1^* x_{i1} + w_2^* x_{i2}.$$

To form our data matrix, $\mathbf{X}$, we append a column of 1s as discussed earlier in the bias trick to get the $\mathbf{X}$ matrix shown below (only showing 9 datapoints for formatting), where the second column is all of our x_1's and the third column is all of our x_2's. Note that in this example, $\mathbf{X} \in \mathbb{R}^{20\times 3}$. Our $\mathbf{y}$ vector of true targets is just a column vector in $\mathbb{R}^{20}$.

$$\mathbf{X} = \begin{bmatrix} 1.00 & 4.17 & 15.36 \\ 1.00 & 9.63 & 13.76 \\ 1.00 & 8.41 & 7.74 \\ 1.00 & 17.18 & 12.30 \\ 1.00 & 3.42 & 8.55 \\ \vdots & \vdots & \vdots \\ 1.00 & 5.41 & 14.05 \\ 1.00 & 19.16 & 2.57 \\ 1.00 & 14.04 & 9.69 \\ 1.00 & 5.95 & 10.32 \end{bmatrix}, \ \mathbf{y} = \begin{bmatrix} 54.39 \\ 60.55 \\ 39.96 \\ 71.27 \\ 32.51 \\ \vdots \\ 44.98 \\ 45.94 \\ 57.21 \\ 42.90 \end{bmatrix}.$$

Plugging in these values for $\mathbf{X}$ and $\mathbf{y}$ into our derived formula for the Optimal Least Squares Linear Regression weights, $\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$, we calculate our optimal $\mathbf{w}^*$ to be:

$$\mathbf{w}^* = \begin{bmatrix} w_0^* = -0.0763 \\ w_1^* = 2.0008 \\ w_2^* = 3.0040 \end{bmatrix}.$$

It follows that our plane of best fit equation is:

$$\hat{y_i} = -0.0763 + 2.0008x_{i1} + 3.0040x_{i2}.$$

This is very close to the true data-generating formula, noting that the mean of ϵ is 0:

$$y_i = \epsilon + 2x_{i1} + 3x_{i2}.$$

We include the following plot to show how our calculated optimal weights $\mathbf{w}^*$ really do capture the linear regression relationship of our data. The plane-of-best-fit does fit all of our data points very well:

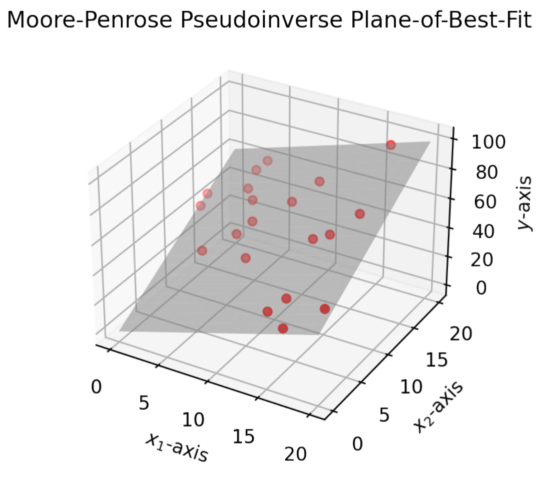

Figure 1: Visualization of our Optimal Weights' Performance

4.3 Remarks

Linear regression is a fundamental, yet very powerful technique in applied machine learning and statistics. The results we derived above tell us that for any data set with scalar y_i's, we can easily find the line-of-best-fit (or plane, or hyperplane, depending on dimensions) through our points (as long as each data point is of the same dimensions, of course). With our $\mathbf{w}^*$ and the resultant line-of-best-fit, we can effectively predict the output variable of a new datapoint, $\hat{y_{new}}$, given a new input variable $\mathbf{x_{new}}$. For example, we could use linear regression to predict children's heights as a function of their parents' heights and other environmental factors. We could also use linear regression to predict a company's next-quarter sales based on certain market variables and consumer data. The uses are endless.

5 Conclusion

From this brief overview of matrix calculus, it is evident that simplifying calculations involving complicated derivatives by condensing these derivatives into elegant matrices with flexible properties allows us to derive and perform a wide array of computations and proofs, a principle that we have already seen frequently in 22b with vector calculus. We also saw that these matrix derivatives have extremely useful applicability to fields such as statistics and machine learning, as detailed in this paper with the example of least squares regression. This being said, there are countless other identities involving matrix derivatives and other applications in the above fields that delve further into the relevancy of matrix calculus in these interdisciplinary applications, such as the multivariate distributions in statistics, namely the multivariate Gaussian distribution and the derivation of its maximum likelihood estimator (MLE), principal components in psychometrics [9, p. 273-448], and many more.

References

[1] Abadir, Karim M., and Jan R. Magnus. *Matrix Algebra. Vol. 1.* Cambridge University Press, 2005.

[2] Adams, Ryan. “COS 302 Precept 6.” COS 302 / SML 305: Mathematics for Numerical Computing and Machine Learning, Princeton University, 2020, www.cs.princeton.edu/courses/archive/spring20/cos302/files/COS_302_Precept_6.pdf.

[3] Ahmadi, Amir Ali. “Lecture 4.” ORF523: Convex and Conic Optimization. Princeton University, February 16, 2016. https://www.princeton.edu/ aaa/Public/Teaching/ORF523/S16/ORF523_S16_Lec4_gh.pdf.

[4] Barnes, Randal J. “Matrix differentiation.” Springs Journal (2006): 1-9.

[5] Deuschle, William J. 2019. *Undergraduate Fundamentals of Machine Learning.* Bachelor’s thesis, Harvard College (CS181 course textbook).

[6] Green, Larry. “Cofactors.” Matrices and Applications, Lake Tahoe Community College, ltcconline.net/greenl/courses/203/MatricesApps/cofactors.htm.

[7] Hu, Pili. *Matrix Calculus: Derivation and Simple Application.* Technical report, City University of Hong Kong, 2012.

[8] Lay, David C., Steven R. Lay, and Judi J. McDonald. *Linear Algebra and its Applications.* (2016).

[9] Magnus, Jan R., and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics.* John Wiley & Sons, 2019.

[10] Minka, Thomas P. Old and New Matrix Algebra Useful for Statistics. https://tminka.github.io/papers/matrix/minka-matrix.pdf.

[11] Petersen, Kaare Brandt, and Michael Syskind Pedersen. *The Matrix Cookbook.* Technical University of Denmark, 2012.