To formally talk about a compression algorithm that stores a compressed version of the data with minimal error, we need to talk about what kind of errors are appropriate to discuss in the context of matrices. In the case of vectors, we can use the ℓ^2-norm, or more generally the ℓ^p norm, to define a distance function; then, the error would just be the distance between the true and the perturbed vectors. This motivates thinking about matrix norms, which allow us to quantify the distance between matrices, and thus create error functions.

There are two ways to think about a matrix. The first way is as a *block of numbers*. Similarly to how we thought of a vector as a block of numbers and found the norm based on this, we can think of the matrix as a big list of vectors and take the norm. This norm is called the *Frobenius norm*, and it corresponds to unrolling an $m \times n$ matrix into a length $m \cdot n$ vector and taking its ℓ^2-norm.

Definition 37 (Frobenius Norm)

For a matrix $A \in \mathbb{R}^{m \times n}$, its Frobenius norm is defined as

$$\|A\|_F \doteq \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}^2}. \tag{2.182}$$

The following property will help us when we study vector calculus.

Proposition 38

For a matrix $A \in \mathbb{R}^{m \times n}$, we have $\|A\|_F^2 = \operatorname{tr}\left(A^\top A\right)$.

The following pair of results will help us now.

Proposition 39

For a matrix $A \in \mathbb{R}^{m \times n}$ and orthonormal matrices $U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}$, we have

$$\|UAV\|_F = \|UA\|_F = \|AV\|_F = \|A\|_F . \tag{2.183}$$

We do not prove this property now, though it might be on homework.

Proposition 40

For a matrix $A \in \mathbb{R}^{m \times n}$ with rank r and singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$, we have

$$\|A\|_F^2 = \sum_{i=1}^{r} \sigma_i^2. \tag{2.184}$$

Proof. Let $A = U\Sigma V^\top$ be the SVD of A; then, we use the previous proposition to get

$$\begin{aligned} \|A\|_F^2 &= \left\|U\Sigma V^\top\right\|_F^2 && (2.185)\\ &= \|\Sigma\|_F^2 && (2.186)\\ &= \sum_{i=1}^{r} \sigma_i^2. && (2.187) \end{aligned}$$

□

Under this perspective, $\|A - B\|_F$ is small if each component of A and B is close; that is, they are very similar in terms of the block-of-numbers interpretation.

The second way to think about a matrix is as a *linear transformation*. In this case, the matrix is defined by how it acts on vectors via multiplication. A suitable notion of size in this case is the largest scaling factor of the matrix on any unit vector; this is called the *spectral norm* or the *matrix ℓ^2-norm*.

Definition 41 (Spectral Norm)

For a matrix $A \in \mathbb{R}^{m \times n}$, its spectral norm is defined by

$$\|A\|_2 \doteq \max_{\substack{\vec{x} \in \mathbb{R}^n \\ \|\vec{x}\|_2 = 1}} \|A\vec{x}\|_2 . \tag{2.188}$$

Fortunately, this optimization problem has a solution — what's more, we've actually seen this solution before.

Proposition 42

For a matrix $A \in \mathbb{R}^{m \times n}$ with rank r and singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$, we have

$$\|A\|_2 = \sigma_1. \tag{2.189}$$

Proof. We use the Rayleigh quotient to say that

$$\begin{aligned}
\|A\|_2 &\doteq \max_{\substack{\vec{x} \in \mathbb{R}^n \\ \|\vec{x}\|_2 = 1}} \|A\vec{x}\|_2 && (2.190)\\
&= \max_{\substack{\vec{x} \in \mathbb{R}^n \\ \|\vec{x}\|_2 = 1}} \sqrt{\|A\vec{x}\|_2^2} && (2.191)\\
&= \sqrt{\max_{\substack{\vec{x} \in \mathbb{R}^n \\ \|\vec{x}\|_2 = 1}} \|A\vec{x}\|_2^2} && (2.192)\\
&= \sqrt{\max_{\substack{\vec{x} \in \mathbb{R}^n \\ \|\vec{x}\|_2 = 1}} \vec{x}^\top A^\top A \vec{x}} && (2.193)\\
&= \sqrt{\lambda_{\max}\{A^\top A\}} && (2.194)\\
&= \sigma_1. && (2.195)
\end{aligned}$$

□

Connecting back to the ellipse transformations, the spectral norm captures how much the ellipse is stretched in its most stretched direction. Thus, the matrix is viewed as as a linear map.

To present our main theorems about how to approximate the matrix well under these norms, we need to define notation. Fix a matrix $A \in \mathbb{R}^{m \times n}$. For convenience, let $p \doteq \min\{m, n\}$. Suppose that A has rank $r \leq p$, and that A has SVD

$$A = \sum_{i=1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top \tag{2.196}$$

where we note that $\sigma_1 \geq \cdots \geq \sigma_r$ and define $\sigma_{r+1} = \sigma_{r+2} = \cdots = 0$. Then, for $k \leq p$, we can define

$$A_k \doteq \sum_{i=1}^{k} \sigma_i \vec{u}_i \vec{v}_i^\top . \tag{2.197}$$

Note that if $k \ll p$, then A_k can be stored much more efficiently than A. For instance, A_k needs to store k scalars (σs), k vectors in $\mathbb{R}^m$ ($\vec{u}$s), and k vectors in $\mathbb{R}^n$ ($\vec{v}$s), for a total storage of $k(m+n+1)$ floats. On the other hand, storing A requires mn floats naively, and even storing the (full) SVD of A is not much better. So the former is much more efficient to store. The only thing left to do is to show that it actually is a good approximation in A (otherwise what would be the point to storing it instead of A?)

It turns out that A_k indeed well-approximates A in the sense of the two norms — the Frobenius norm and the spectral norm — that we discussed previously. The two results are collectively known as the *Eckart-Young* (sometimes *Eckart-Young-Mirsky*) theorem(s). We state and prove these now.

We begin with the Eckart-Young theorem for the spectral norm, since it will help us prove the analogous result for Frobenius norms.

Theorem 43 (Eckart-Young Theorem for Spectral Norm)

We have

$$A_k \in \operatorname*{argmin}_{\substack{B \in \mathbb{R}^{m \times n} \\ \operatorname{rank}(B) \le k}} \|A - B\|_2 , \tag{2.198}$$

or, equivalently,

$$\|A - A_k\|_2 \le \|A - B\|_2 , \qquad \forall B \in \mathbb{R}^{m \times n} \colon \operatorname{rank}(B) \le k. \tag{2.199}$$

Proof of Theorem 43. The proof of Eckart-Young Theorem for Spectral Norm is partitioned into two parts which together jointly show the conclusion. First, we explicitly calculate $\|A - A_k\|_2$; then for an arbitrary B of rank $\le k$ we show that $\|A - B\|_2$ is lower-bounded by this quantity.

Part 1. First, we calculate $\|A - A_k\|_2$. Indeed, we have

$$A - A_k = \left(\sum_{i=1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top \right) - \left(\sum_{i=1}^{k} \sigma_i \vec{u}_i \vec{v}_i^\top \right) \tag{2.200}$$

$$= \sum_{i=k+1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top . \tag{2.201}$$

Since $\sigma_{k+1} \ge \sigma_{k+2} \ge \cdots$, the $\vec{u}_i$ are orthonormal, and the $\vec{v}_i$ are orthonormal, this is a valid singular value decomposition of the rank $r-k$ matrix $A - A_k$. Thus $A - A_k$ has largest singular value equal to σ_{k+1}, so

$$\|A - A_k\|_2 = \sigma_{k+1}. \tag{2.202}$$

Part 2. We now show that for any matrix $B \in \mathbb{R}^{m \times n}$ of rank $\le k$, we have $\|A - B\|_2 \ge \sigma_{k+1}$. Before we commence with the proof, let us first discuss the overall argument structure.

We have two characterizations of the spectral norm: the maximum singular value, and the maximal Rayleigh coefficient. Up until now, we don't have any advanced machinery such as singular value inequalities to directly show that the maximum singular value of $A-B$ is lower-bounded by some constant. However, we do understand matrix-vector products, and how to characterize their optima, pretty well, so this motivates the use of the maximal Rayleigh coefficient. In particular, suppose $f(\vec{x}) = \|(A-B)\vec{x}\|_2 / \|\vec{x}\|_2$. Then we know that for any specific value of $\vec{x}_0$, we have $\|A - B\|_2 = \max_{\vec{x}} f(\vec{x}) \ge f(\vec{x}_0)$. Thus, one way to get a lower bound on $\|A - B\|_2$ is to plug in any $\vec{x}_0$ into f. The trick is, given B, to find the right value of $\vec{x}_0$ such that $f(\vec{x}_0) = \sigma_{k+1}$. The first part of the argument will be finding an appropriate $\vec{x}_0$; the next will show that it works.

Okay, let's start with the proof.

Step 1. Here we show the existence of an appropriate choice for $\vec{x}_0$.

Because B has rank $\leq k$, the rank-nullity theorem gives

$$\dim(\mathcal{N}(B)) = n - \operatorname{rank}(B) \geq n - k > 0. \tag{2.203}$$

Now, define $V_{k+1} \doteq \begin{bmatrix}\vec{v}_1, \ldots, \vec{v}_{k+1}\end{bmatrix}$. We know by the SVD that the $\vec{v}_i$ are orthonormal, and so are linearly independent. Thus

$$\operatorname{rank}(V_{k+1}) = \dim(\mathcal{R}(V_{k+1})) = k + 1. \tag{2.204}$$

Thus

$$\dim(\mathcal{N}(B)) + \dim(\mathcal{R}(V_{k+1})) \geq (n - k) + (k + 1) = n + 1 > n. \tag{2.205}$$

Claim. There exists a unit vector in $\mathcal{N}(B) \cap \mathcal{R}(V_{k+1})$.

Proof. First, we note that since $\mathcal{N}(B)$ and $\mathcal{R}(V_{k+1})$ are subspaces of $\mathbb{R}^n$, their intersection is a subspace of $\mathbb{R}^n$. Thus, it is closed under scalar multiplication. Thus the existence of a unit vector is equivalent to the existence of a nonzero vector, since one can scale this nonzero vector by its (inverse) norm to get the unit vector.

Suppose for the sake of contradiction that $\mathcal{N}(B)$ and $\mathcal{R}(V_{k+1})$ have no nonzero vectors in common. Fix a basis S_1 for $\mathcal{N}(B)$ and a basis S_2 for $\mathcal{R}(V_{k+1})$. By assumption, we have that $S_1 \cap S_2 = \emptyset$, and $S_1 \cup S_2$ is a linearly independent set. Here, there are two cases.

A. $S_1 \cup S_2$ contains $< n + 1$ vectors. This means that S_1 and S_2 have a vector in common, so $S_1 \cap S_2 \neq \emptyset$ and we have reached a contradiction.

B. $S_1 \cup S_2$ has exactly $n + 1$ vectors. As a set of $n + 1$ vectors in $\mathbb{R}^n$, $S_1 \cup S_2$ is linearly dependent and we have reached a contradiction.

In both cases we have reached a contradiction, so there is a nonzero vector in $\mathcal{N}(B) \cap \mathcal{R}(V_{k+1})$. We may scale this vector by its inverse norm to get a unit vector in $\mathcal{N}(B) \cap \mathcal{R}(V_{k+1})$. □

Let $\vec{x}_0 \in \mathcal{N}(B) \cap \mathcal{R}(V_{k+1})$ be any such unit vector. We use this vector for the Rayleigh coefficient.

Step 2. Now we show that our choice of $\vec{x}_0$ plugged into the Rayleigh coefficient gives σ_{k+1} as a lower bound.

Indeed, we have

$$\|A - B\|_2 = \max_{\vec{x} \neq \vec{0}} \frac{\|(A - B)\vec{x}\|_2}{\|\vec{x}\|_2} \tag{2.206}$$

$$\geq \frac{\|(A - B)\vec{x}_0\|_2}{\|\vec{x}_0\|_2} \tag{2.207}$$

$$= \|(A - B)\vec{x}_0\|_2 \tag{2.208}$$

$$= \|A\vec{x}_0 - B\vec{x}_0\|_2 . \tag{2.209}$$

Since $\vec{x}_0 \in \mathcal{N}(B)$, we have $B\vec{x}_0 = \vec{0}$, so

$$\|A - B\|_2 \geq \|A\vec{x}_0 - B\vec{x}_0\|_2 = \|A\vec{x}_0\|_2 . \tag{2.210}$$

Since $\vec{x}_0 \in \mathcal{R}(V_{k+1})$, there are constants $\alpha_1, \ldots, \alpha_{k+1}$, not all zero, such that $\vec{x}_0 = \sum_{i=1}^{k+1} \alpha_i \vec{v}_i$. Thus

$$\|A - B\|_2 \geq \|A\vec{x}_0\|_2 \tag{2.211}$$

$$= \left\| A \sum_{i=1}^{k+1} \alpha_i \vec{v}_i \right\|_2 \tag{2.212}$$

$$= \left\| \left(\sum_{i=1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top \right) \left(\sum_{i=1}^{k+1} \alpha_i \vec{v}_i \right) \right\|_2 \tag{2.213}$$

$$= \left\| \sum_{i=1}^{p} \sum_{j=1}^{k+1} \sigma_i \alpha_j \vec{u}_i \vec{v}_i^\top \vec{v}_j \right\|_2 . \tag{2.214}$$

By vector algebra, the fact that the $\vec{u}_i$ are orthonormal, and the fact that the $\vec{v}_i$ are orthonormal, one can mechanically show that

$$\|A - B\|_2 \geq \left\| \sum_{i=1}^{p} \sum_{j=1}^{k+1} \sigma_i \alpha_j \vec{u}_i \vec{v}_i^\top \vec{v}_j \right\|_2 \tag{2.215}$$

$$= \left\| \sum_{i=1}^{k+1} \alpha_i \sigma_i \vec{u}_i \right\|_2 \tag{2.216}$$

$$= \sqrt{\left\| \sum_{i=1}^{k+1} \alpha_i \sigma_i \vec{u}_i \right\|_2^2} \tag{2.217}$$

$$= \sqrt{\sum_{i=1}^{k+1} \alpha_i^2 \sigma_i^2} \tag{2.218}$$

$$\geq \sqrt{\sigma_{k+1}^2 \sum_{i=1}^{k+1} \alpha_i^2} \tag{2.219}$$

$$= \sigma_{k+1} \sqrt{\sum_{i=1}^{k+1} \alpha_i^2} \tag{2.220}$$

$$= \sigma_{k+1} \|\vec{x}_0\|_2 \tag{2.221}$$

$$= \sigma_{k+1}. \tag{2.222}$$

This proves the claim.

□

We now state and prove the other result, which is the Eckart-Young theorem for the Frobenius norm.

Theorem 44 (Eckart-Young Theorem for Frobenius Norm)

We have

$$A_k \in \underset{\substack{B \in \mathbb{R}^{m \times n} \\ \operatorname{rank}(B) \leq k}}{\operatorname{argmin}} \|A - B\|_F , \tag{2.223}$$

or, equivalently,

$$\|A - A_k\|_F^2 \leq \|A - B\|_F^2 , \qquad \forall B \in \mathbb{R}^{m \times n} \colon \operatorname{rank}(B) \leq k. \tag{2.224}$$

Proof of Theorem 44. Like the Frobenius norm, the proof of Eckart-Young Theorem for Frobenius Norm is partitioned into two parts which together jointly show the conclusion. First, we explicitly calculate $\|A - A_k\|_F^2$; then for an arbitrary B of rank $\leq k$ we show that $\|A - B\|_F^2$ is lower-bounded by this quantity.

Part 1. First, we calculate $\|A - A_k\|_F^2$. Indeed, we have

$$A - A_k = \left(\sum_{i=1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top\right) - \left(\sum_{i=1}^{k} \sigma_i \vec{u}_i \vec{v}_i^\top\right) \tag{2.225}$$

$$= \sum_{i=k+1}^{p} \sigma_i \vec{u}_i \vec{v}_i^\top. \tag{2.226}$$

Since $\sigma_{k+1} \geq \sigma_{k+2} \geq \cdots$, the $\vec{u}_i$ are orthonormal, and the $\vec{v}_i$ are orthonormal, this is a valid singular value decomposition of the rank $r - k$ matrix $A - A_k$. Thus $A - A_k$ has largest singular value equal to σ_{k+1}, so

$$\|A - A_k\|_F^2 = \sum_{i=k+1}^{p} \sigma_i^2. \tag{2.227}$$

Part 2. We now show that for any matrix $B \in \mathbb{R}^{m \times n}$ of rank $\leq k$, we have $\|A - B\|_F^2 \geq \sum_{i=k+1}^{p} \sigma_i^2$. Again, let us begin by discussing the overall argument structure.

Our goal is to show an inequality between two sums of squares of singular values. To do this, it is simplest to match up terms in the two sums and show the inequality between each pair of terms. Then summing over all terms preserves the inequality.

More concretely, define $C \doteq A - B$. Suppose C has SVD $C = \sum_{i=1}^{p} \gamma_i \vec{y}_i \vec{z}_i^\top$. We want to show

$$\|A - B\|_F^2 = \sum_{i=1}^{p} \gamma_i^2 \underbrace{\geq}_{\text{still need to show}} \sum_{i=k+1}^{p} \sigma_i^2. \tag{2.228}$$

Thus, we claim that $\gamma_i \geq \sigma_{i+k}$ for every $i \in \{1, \ldots, p\}$, with the understanding that $\sigma_{p+1} = \sigma_{p+2} = \cdots = 0$.

To use our earlier knowledge about the spectral norm to our advantage, we write the singular value γ_i as the spectral norm of the approximation error matrix:

$$\gamma_i = \|C - C_{i-1}\|_2 . \tag{2.229}$$

We can expand to get

$$\gamma_i = \|C - C_{i-1}\|_2 = \|(A - B) - C_{i-1}\|_2 = \|A - (B + C_{i-1})\|_2 . \tag{2.230}$$

Now this looks somewhat like the low-rank approximation setup, because we are computing the spectral norm of the difference between A and some matrix. But what is the rank of this matrix? We have

$$\operatorname{rank}(B + C_{i-1}) \leq \operatorname{rank}(B) + \operatorname{rank}(C_{i-1}) \leq k + i - 1. \tag{2.231}$$

Thus, by applying Eckart-Young Theorem for Spectral Norm to the rank $i + k - 1$ approximation of A, that

$$\gamma_i = \|A - (B + C_{i-1})\|_2 \geq \|A - A_{i+k-1}\|_2 = \sigma_{i+k} \tag{2.232}$$

which is what we wanted to show.

To finish the proof, we use the following inequalities:

$$\gamma_i^2 \geq \sigma_{i+k}^2, \qquad \forall i \in \{1, \ldots, p\} \tag{2.233}$$

$$\sum_{i=1}^{p} \gamma_i^2 \geq \sum_{i=1}^{p} \sigma_{i+k}^2 \qquad \text{summing over all } i, \tag{2.234}$$

$$\|A - B\|_F^2 \geq \sum_{i=k+1}^{p} \sigma_i^2 \tag{2.235}$$

as desired.

□

2.8 (OPTIONAL) Block Matrix Identities

In this section, we list many ways to manipulate block matrices. Since each fact in here is something you can derive yourself using definitions (e.g. of matrix multiplication), you may use any of them without proof.

2.8.1 Transposes of Block Matrices

$$\begin{bmatrix} \vec{x}_1 & \cdots & \vec{x}_n \end{bmatrix}^\top = \begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} \tag{2.236}$$

$$\begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} = \begin{bmatrix} \vec{x}_1 & \cdots & \vec{x}_n \end{bmatrix}^\top \tag{2.237}$$

$$\begin{bmatrix} A & B \end{bmatrix}^\top = \begin{bmatrix} A^\top \\ B^\top \end{bmatrix} \tag{2.238}$$

$$\begin{bmatrix} A \\ B \end{bmatrix}^\top = \begin{bmatrix} A^\top & B^\top \end{bmatrix} \tag{2.239}$$

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^\top = \begin{bmatrix} A^\top & C^\top \\ B^\top & D^\top \end{bmatrix}. \tag{2.240}$$

2.8.2 Block Matrix Products

In the following, $\vec{e}_i$ is the i^{th} standard basis vector – it has a 1 in the i^{th} coordinate and 0 in all other coordinates.

$$\begin{bmatrix} \vec{x}_1 & \cdots & \vec{x}_n \end{bmatrix} \begin{bmatrix} \vec{y}_1^\top \\ \vdots \\ \vec{y}_n^\top \end{bmatrix} = \sum_{i=1}^{n} \vec{x}_i \vec{y}_i^\top \tag{2.241}$$

$$\begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} \begin{bmatrix} \vec{y}_1 & \cdots & \vec{y}_n \end{bmatrix} = \begin{bmatrix} \vec{x}_1^\top \vec{y}_1 & \cdots & \vec{x}_1^\top \vec{y}_n \\ \vdots & \ddots & \vdots \\ \vec{x}_n^\top \vec{y}_1 & \cdots & \vec{x}_n^\top \vec{y}_n \end{bmatrix} \tag{2.242}$$

$$\vec{e}_i^\top \begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} = \vec{x}_i^\top \tag{2.243}$$

$$\begin{bmatrix} \vec{x}_1 & \cdots & \vec{x}_n \end{bmatrix} \vec{e}_i = \vec{x}_i \tag{2.244}$$

$$A \begin{bmatrix} \vec{x}_1 & \cdots & \vec{x}_n \end{bmatrix} = \begin{bmatrix} A\vec{x}_1 & \cdots & A\vec{x}_n \end{bmatrix} \tag{2.245}$$

$$\begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} A = \begin{bmatrix} \vec{x}_1^\top A \\ \vdots \\ \vec{x}_n^\top A \end{bmatrix} \tag{2.246}$$

$$A \begin{bmatrix} B & C \end{bmatrix} = \begin{bmatrix} AB & AC \end{bmatrix} \tag{2.247}$$

$$\begin{bmatrix} A \\ B \end{bmatrix} C = \begin{bmatrix} AC \\ BC \end{bmatrix}. \tag{2.248}$$

2.8.3 Block Diagonal Matrices

$$\begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{bmatrix} \begin{bmatrix} \vec{x}_1^\top \\ \vec{x}_2^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix} = \begin{bmatrix} d_1\vec{x}_1^\top \\ \vdots \\ d_n\vec{x}_n^\top \end{bmatrix} \tag{2.249}$$

$$\begin{bmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_n \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} A_1B_1 \\ A_2B_2 \\ \vdots \\ A_nB_n \end{bmatrix} \tag{2.250}$$

2.8.4 Quadratic Forms

$$\vec{x}^\top A\vec{y} = \sum_i \sum_j A_{ij}x_iy_j \tag{2.251}$$

$$\begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix}^\top \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix} = \vec{x}^\top A\vec{x} + \vec{x}^\top B\vec{y} + \vec{y}^\top C\vec{x} + \vec{y}^\top D\vec{y}. \tag{2.252}$$

Chapter 3

Vector Calculus

Relevant sections of the textbooks:

- [1] Appendix A.4.

3.1 Gradient, Jacobian, and Hessian

To motivative this section, we start with a familiar concept which should have been covered in the prerequisites: the derivatives of a *scalar* function $f\colon \mathbb{R} \to \mathbb{R}$ which takes in scalar input and produces a scalar output. The derivative quantifies the (instantaneous) rate of change of the function due to the change of its input. We recall the limit definition of the derivative.

Definition 45 (Derivative for Scalar Functions)
Let $f\colon \mathbb{R} \to \mathbb{R}$ be differentiable. The *derivative* of f with respect to x is the function $\frac{\mathrm{d}f}{\mathrm{d}x}\colon \mathbb{R} \to \mathbb{R}$ defined by

$$\frac{\mathrm{d}f}{\mathrm{d}x}(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}. \tag{3.1}$$

The derivative of f has several other notations, including f' and $\dot{f}$.

In this section, we aim to generalize the concept of derivatives beyond scalar functions. We will focus on two types of functions:

1. *Multivariate functions* $f\colon \mathbb{R}^n \to \mathbb{R}$ which take a vector $\vec{x} \in \mathbb{R}^n$ as input and produce a scalar $f(\vec{x}) \in \mathbb{R}$ as output. Familiar examples of such functions include $f(\vec{x}) = \|\vec{x}\|_p$ and $f(\vec{x}) = \vec{a}^\top \vec{x}$.
2. *Vector-valued functions* $\vec{f}\colon \mathbb{R}^n \to \mathbb{R}^m$ which take a vector $\vec{x} \in \mathbb{R}^n$ as input and produce another vector $\vec{f}(\vec{x}) \in \mathbb{R}^m$ as output. A familiar example of such functions is $f(\vec{x}) = A\vec{x}$.

One tool that allows us to compute derivatives of scalar functions is the chain rule, which describes the derivative of the composition of two functions.

Theorem 46 (Chain Rule for Scalar Functions)
Let $f\colon \mathbb{R} \to \mathbb{R}$ and $g\colon \mathbb{R} \to \mathbb{R}$ be two differentiable scalar functions, and let $h\colon \mathbb{R} \to \mathbb{R}$ be defined as $h(x) =$

$f(g(x))$ for all $x \in \mathbb{R}$. Then h is differentiable, and

$$\frac{\mathrm{d}h}{\mathrm{d}x}(x) = \frac{\mathrm{d}f}{\mathrm{d}g}(g(x)) \cdot \frac{\mathrm{d}g}{\mathrm{d}x}(x). \tag{3.2}$$

Here, we use some — perhaps unfamiliar — notation not previously introduced. In particular, the derivative $\frac{\mathrm{d}f}{\mathrm{d}g}(g(x))$ is a little strange. When we write $\frac{\mathrm{d}f}{\mathrm{d}g}$ we take the derivative of f with respect to the output of g. In this case, we know that the input of f is exactly the output of g, so really $\frac{\mathrm{d}f}{\mathrm{d}g}$ takes the derivative of f with respect to its input. Thus we can more compactly write the chain rule using the f' notation as

$$h'(x) = f'(g(x)) \cdot g'(x). \tag{3.3}$$

We will see in the next section that the chain rule can be generalized to settings of multivariate functions. To aid our study of such generalizations, we will introduce here a *computational graph* perspective of the chain rule. At the moment this graphical perspective looks trivial, but it will help us understand more complicated cases.

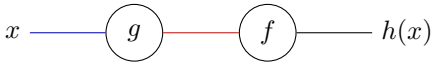


Figure 3.1: Graphical depiction of the single-variable chain rule corresponding to $h = f \circ g$.

In this computational graph, one computes the derivative by summing along all paths from x to $h(x)$. There is only one path, so we get

$$\frac{\mathrm{d}h}{\mathrm{d}x}(x) = \frac{\mathrm{d}f}{\mathrm{d}g}(g(x)) \cdot \frac{\mathrm{d}g}{\mathrm{d}x}(x). \tag{3.4}$$

3.1.1 Partial Derivatives

For multivariate functions $f\colon \mathbb{R}^n \to \mathbb{R}$, when we talk about the rate of change of the function with respect to its input, we need to specify which input we are talking about. Partial derivatives quantify this and give us the rate of change of the function due to the change of one of its inputs, say x_i, while keeping all other inputs fixed. Keeping all but one input fixed renders a scalar function, for which we know how to compute the derivative. To formalize this we introduce the limit definition of the partial derivative.

Definition 47 (Partial Derivative)

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be differentiable. The *partial derivative* of f with respect to x_i is the function $\frac{\partial f}{\partial x_i}\colon \mathbb{R}^n \to \mathbb{R}$ defined by

$$\frac{\partial f}{\partial x_i}(\vec{x}) = \lim_{h \to 0} \frac{f(x_1, \ldots, x_i + h, \ldots, x_n) - f(\vec{x})}{h}, \tag{3.5}$$

or equivalently,

$$\frac{\partial f}{\partial x_i}(\vec{x}) = \lim_{h \to 0} \frac{f(\vec{x} + h \cdot \vec{e}_i) - f(\vec{x})}{h} \tag{3.6}$$

where $\vec{e}_i$ is the i^{th} standard basis vector.

This limit definition gives an alternative way of interpreting partial derivatives: $\frac{\partial f}{\partial x_i}$ is the rate of change of the function along the direction of the standard basis vector $\vec{e}_i$.

In practice, we do not use the limit definition to compute regular derivatives. Similarly, we do not use the limit definition to compute partial derivatives. The main way to compute partial derivatives uses the following tip.

Problem Solving Strategy 48. *To compute the partial derivative $\frac{\partial f}{\partial x_i}$, pretend that all x_j for $j \neq i$ are constants, then take the ordinary derivative in x_i.*

Example 49. Consider the function $f(\vec{x}) = \vec{a}^\top \vec{x}$. Then

$$\frac{\partial f}{\partial x_i}(\vec{x}) = \frac{\partial}{\partial x_i} \vec{a}^\top \vec{x} \tag{3.7}$$

$$= \frac{\partial}{\partial x_i} \sum_{j=1}^{n} a_j x_j \tag{3.8}$$

$$= \frac{\partial}{\partial x_i} (a_1 x_1 + \cdots + a_n x_n) \tag{3.9}$$

$$= a_i. \tag{3.10}$$

Let us consider the case where the input $\vec{x}$ is not an independent vector but rather depends on another variable $t \in \mathbb{R}$, i.e., $\vec{x} \colon \mathbb{R} \to \mathbb{R}^n$ is a function. In such case the function $f(\vec{x}(t))$ has *one independent* input, which is t. If we are interested in finding the derivative of f with respect to t, we can utilize a chain rule to do so.

Theorem 50 (Chain Rule For Multivariate Functions)
Let $f \colon \mathbb{R}^n \to \mathbb{R}$ and $\vec{g} \colon \mathbb{R} \to \mathbb{R}^n$ be differentiable functions. Define the function $h \colon \mathbb{R} \to \mathbb{R}$ by $h(x) = f(\vec{g}(x))$ for all $x \in \mathbb{R}$. Then h is differentiable and has derivative

$$\frac{\mathrm{d}h}{\mathrm{d}x}(x) = \sum_{i=1}^{n} \frac{\partial f}{\partial g_i}(\vec{g}(x)) \cdot \frac{\mathrm{d}g_i}{\mathrm{d}x}(x). \tag{3.11}$$

Here we again review this computational graph perspective. There are now n separate paths from x to h, like so:

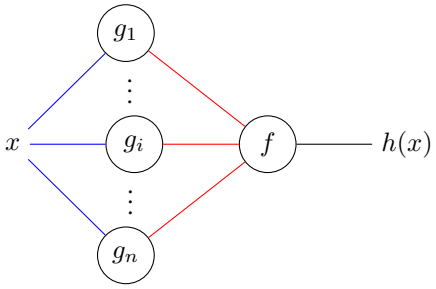


Figure 3.2: Graphical depiction of the multivariate chain rule corresponding to $h = f \circ \vec{g}$.

In this computational graph, one computes the derivative by summing across the n paths from x to $h(x)$, obtaining

$$\frac{\mathrm{d}h}{\mathrm{d}x}(x) = \sum_{i=1}^{n} \frac{\partial f}{\partial g_i}(\vec{g}(x)) \cdot \frac{\mathrm{d}g_i}{\mathrm{d}x}(x). \tag{3.12}$$

Here again we use the notation $\frac{\partial f}{\partial g_i}$ to denote the derivative of f with respect to the output of g_i. Since the output of g_i is really the i^{th} input of f, this derivative is just the derivative of f with respect to its i^{th} input.

3.1.2 Gradient

We will now use the definition of partial derivatives to introduce the gradient of multivariate functions.

Definition 51 (Gradient)

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. The *gradient* of f is the function $\nabla f\colon \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{x}) \end{bmatrix}. \tag{3.13}$$

Note that the gradient is a column vector. The transpose of the gradient is (confusingly!) referred to as the *derivative* of the function. We will now list two important geometric properties of the gradient. The first can be stated straight away:

Proposition 52

Let $\vec{x} \in \mathbb{R}^n$. The gradient $\nabla f(\vec{x})$ points in the direction of *steepest ascent* at $\vec{x}$, i.e., the direction around $\vec{x}$ in which f has the maximum rate of change. Furthermore, this rate of change is quantified by the norm $\|\nabla f(\vec{x})\|_2$.

Proof. Let $\vec{u} \in \mathbb{R}^n$ be a unit vector (i.e., representing an arbitrary direction in $\mathbb{R}^n$). Using the Cauchy-Schwarz inequality we can write:

$$\vec{u}^\top [\nabla f(\vec{x})] \leq \|\vec{u}\|_2 \|\nabla f(\vec{x})\|_2 \tag{3.14}$$

$$= \|\nabla f(\vec{x})\|_2, \tag{3.15}$$

so the maximum value that the expression can take is $\|\nabla f(\vec{x})\|_2$. Now it remains to show that this value is attained for the choice $\vec{u} = \frac{\nabla f(\vec{x})}{\|\nabla f(\vec{x})\|_2}$.

$$\left[\frac{\nabla f(\vec{x})}{\|\nabla f(\vec{x})\|_2}\right]^\top [\nabla f(\vec{x})] = \frac{\|\nabla f(\vec{x})\|_2^2}{\|\nabla f(\vec{x})\|_2} \tag{3.16}$$

$$= \|\nabla f(\vec{x})\|_2. \tag{3.17}$$

□

To list the second property, first we need a quick definition.

Definition 53 (Level Set)

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function, and $\alpha \in \mathbb{R}$ be a scalar.

- The α-*level set* of f is the set of points $\vec{x}$ such that $f(\vec{x}) = \alpha$:

$$L_\alpha(f) = \{\vec{x} \in \mathbb{R}^n \mid f(\vec{x}) = \alpha\}. \tag{3.18}$$

- The α-*sublevel set* of f is the set of points $\vec{x}$ such that $f(\vec{x}) \leq \alpha$:

$$L_{\leq\alpha}(f) = \{\vec{x} \in \mathbb{R}^n \mid f(\vec{x}) \leq \alpha\}. \tag{3.19}$$

- The α-*superlevel set* of f is the set of points $\vec{x}$ such that $f(\vec{x}) \geq \alpha$:

$$L_{\geq\alpha}(f) = \{\vec{x} \in \mathbb{R}^n \mid f(\vec{x}) \geq \alpha\}. \tag{3.20}$$

Proposition 54

Let $\vec{x} \in \mathbb{R}^n$ and suppose $f(\vec{x}) = \alpha$. Then $\nabla f(\vec{x})$ is orthogonal to the hyperplane which is tangent at $\vec{x}$ to the α-level set of f.

We illustrate the two properties through examples.

Example 55 (Gradient of the Squared ℓ^2 Norm)**.** In this example we will compute and visualize the gradient of the function $f(\vec{x}) = \|\vec{x}\|_2^2$ where $\vec{x} \in \mathbb{R}^2$.

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}) \\ \frac{\partial f}{\partial x_2}(\vec{x}) \end{bmatrix} \tag{3.21}$$

$$= \begin{bmatrix} \frac{\partial}{\partial x_1}(x_1^2 + x_2^2) \\ \frac{\partial}{\partial x_2}(x_1^2 + x_2^2) \end{bmatrix} \tag{3.22}$$

$$= \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix} \tag{3.23}$$

$$= 2\vec{x}. \tag{3.24}$$

This function has a paraboloid-shaped graph, as shown in Figure 3.3a. Let us now find the α-level set of the function f for some constant α.

$$\alpha = f(\vec{x}) \tag{3.25}$$

$$= x_1^2 + x_2^2 \tag{3.26}$$

For a given $\alpha \geq 0$, the α-level set is a circle centered at the origin which has radius $\sqrt{\alpha}$. Now we evaluate the gradient at a few points on these level sets:

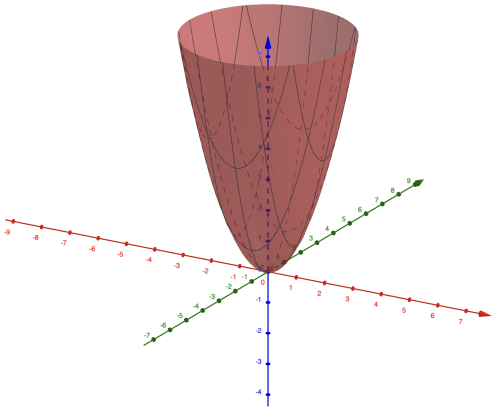
$$\nabla f(-1, 0) = (-2, 0) \tag{3.27}$$

$$\nabla f(1/\sqrt{2}, 1/\sqrt{2}) = (\sqrt{2}, \sqrt{2}) \tag{3.28}$$

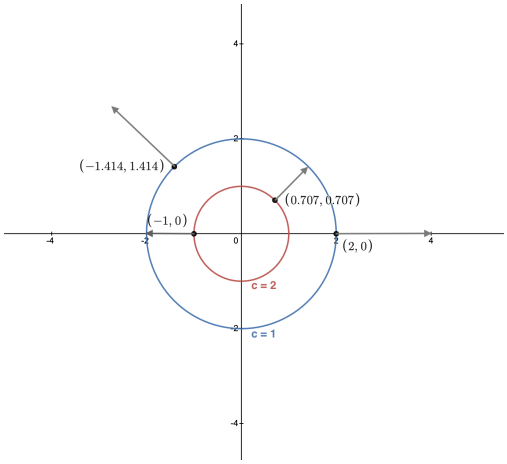
$$\nabla f(2, 0) = (4, 0) \tag{3.29}$$

$$\nabla f(-\sqrt{2}, \sqrt{2}) = (-2\sqrt{2}, 2\sqrt{2}) \tag{3.30}$$

We plot the level sets for $\alpha = 1$ and $\alpha = 4$ and visualize the gradient directions in Figure 3.3b.



(a) Plot of the function $f(\vec{x}) = \|\vec{x}\|_2^2$



(b) The level sets of the function $f(\vec{x}) = \|\vec{x}\|_2^2 = \alpha$ for $\alpha = 1$ and $\alpha = 4$, and the gradients at some points along the level sets.

Figure 3.3: Example 55

We make the following observations:

1. At each point on a given level set, the gradient vector at that point is orthogonal to the line tangent to the level set at that point.
2. The length of the gradient vector increases as we move away from the origin. This means that the function gets steeper in that direction (i.e., it changes more rapidly).

Example 56 (Gradient of Linear Function)**.** In this example we will compute and comment on the gradient of the linear function $f\colon \mathbb{R}^n \to \mathbb{R}$ is defined by $f(\vec{x}) = \vec{a}^\top \vec{x}$ where $\vec{a} \in \mathbb{R}^n$ is fixed. We have

$$f(\vec{x}) = \vec{a}^\top \vec{x}, \tag{3.31}$$

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{x}) \end{bmatrix} \tag{3.32}$$

$$= \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \tag{3.33}$$

$$= \vec{a}. \tag{3.34}$$

We make the following observations:

1. The α-level sets of this function are the hyperplanes given by all $\vec{x}$ such that $\vec{a}^\top \vec{x} = \alpha$. These hyperplanes have normal vector $\vec{a}$. Notice that the normal vector (which is orthogonal to all vectors in the hyperplane) is exactly the gradient vector.
2. The gradient is constant, meaning that the function has a constant rate of change everywhere.

Example 57 (Gradient of the Quadratic Form)**.** Let $A \in \mathbb{R}^{n\times n}$. In this example we will compute the gradient of the quadratic function $f\colon \mathbb{R}^n \to \mathbb{R}$ defined by $f(\vec{x}) = \vec{x}^\top A\vec{x}$. Indeed, we have

$$f(\vec{x}) = \vec{x}^\top A\vec{x} \tag{3.35}$$

$$= \sum_{i=1}^n \sum_{j=1}^n A_{ij}x_i x_j. \tag{3.36}$$

Now fix $k \in \{1,\dots,n\}$, and we find the partial derivative with respect to x_k. We have

$$\begin{aligned}
f(\vec{x}) &= \sum_{i=1}^n \sum_{j=1}^n A_{ij}x_i x_j \\
&= \sum_{j=1}^n A_{kj}x_k x_j + \sum_{\substack{i=1\\ i\neq k}}^n \sum_{j=1}^n A_{ij}x_i x_j \\
&= \sum_{j=1}^n A_{kj}x_k x_j + \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i x_k + \sum_{\substack{i=1\\ i\neq k}}^n \sum_{\substack{j=1\\ j\neq k}}^n A_{ij}x_i x_j \\
&= A_{kk}x_k^2 + \sum_{\substack{j=1\\ j\neq k}}^n A_{kj}x_k x_j + \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i x_k + \sum_{\substack{i=1\\ i\neq k}}^n \sum_{\substack{j=1\\ j\neq k}}^n A_{ij}x_i x_j \\
&= A_{kk}x_k^2 + x_k \sum_{\substack{j=1\\ j\neq k}}^n A_{kj}x_j + x_k \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i + \sum_{\substack{i=1\\ i\neq k}}^n \sum_{\substack{j=1\\ j\neq k}}^n A_{ij}x_i x_j.
\end{aligned}$$

Then, taking the derivatives, we have

$$\frac{\partial f}{\partial x_k}(\vec{x}) = \frac{\partial}{\partial x_k}\left(A_{kk}x_k^2 + x_k \sum_{\substack{j=1\\ j\neq k}}^n A_{kj}x_j + x_k \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i + \sum_{\substack{i=1\\ i\neq k}}^n \sum_{\substack{j=1\\ j\neq k}}^n A_{ij}x_i x_j \right) \tag{3.37}$$

$$= 2A_{kk}x_k + \sum_{\substack{j=1\\ j\neq k}}^n A_{kj}x_j + \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i \tag{3.38}$$

$$= \left(A_{kk}x_k + \sum_{\substack{j=1\\ j\neq k}}^n A_{kj}x_j \right) + \left(A_{kk}x_k + \sum_{\substack{i=1\\ i\neq k}}^n A_{ik}x_i \right) \tag{3.39}$$

$$= \sum_{j=1}^n A_{kj}x_j + \sum_{i=1}^n A_{ik}x_i \tag{3.40}$$

$$= (A\vec{x})_k + (A^\top \vec{x})_k \tag{3.41}$$

$$= ((A + A^\top)\vec{x})_k. \tag{3.42}$$

Thus computing the gradient via stacking the partial derivatives gets

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{x}) \end{bmatrix} \tag{3.43}$$

$$= \begin{bmatrix} ((A + A^\top)\vec{x})_1 \\ \vdots \\ ((A + A^\top)\vec{x})_n \end{bmatrix} \tag{3.44}$$

$$= (A + A^\top)\vec{x}. \tag{3.45}$$

That was a very involved computation! Luckily, in the near future, we will see ways to simplify the process of computing gradients.

3.1.3 Jacobian

We now have the tools to generalize the notion of derivatives to vector-valued functions $\vec{f}\colon \mathbb{R}^n \to \mathbb{R}^m$.

Definition 58 (Jacobian)
Let $\vec{f}\colon \mathbb{R}^n \to \mathbb{R}^m$ be a differentiable function. The *Jacobian* of $\vec{f}$ is the function $D\vec{f}\colon \mathbb{R}^n \to \mathbb{R}^{m\times n}$ defined as

$$D\vec{f}(\vec{x}) = \begin{bmatrix} \nabla f_1(\vec{x})^\top \\ \vdots \\ \nabla f_m(\vec{x})^\top \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\vec{x}) & \cdots & \frac{\partial f_1}{\partial x_n}(\vec{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(\vec{x}) & \cdots & \frac{\partial f_m}{\partial x_n}(\vec{x}) \end{bmatrix}. \tag{3.46}$$

One big thing to note is that *the Jacobian is different from the gradient!* If $f\colon \mathbb{R}^n \to \mathbb{R}^1 = \mathbb{R}$, then its Jacobian $Df\colon \mathbb{R}^n \to \mathbb{R}^{1\times n}$ is a function which outputs a *row vector*. This row vector is the *transpose* of the gradient.

We can develop a nice and general chain rule with the Jacobian.

Theorem 59 (Chain Rule for Vector-Valued Functions)
Let $\vec{f}\colon \mathbb{R}^p \to \mathbb{R}^m$ and $\vec{g}\colon \mathbb{R}^n \to \mathbb{R}^p$ be differentiable functions. Let $\vec{h}\colon \mathbb{R}^n \to \mathbb{R}^m$ be defined as $\vec{h}(\vec{x}) = \vec{f}(\vec{g}(\vec{x}))$ for all $\vec{x} \in \mathbb{R}^n$. Then $\vec{h}$ is differentiable, and

$$D\vec{h}(\vec{x}) = [D\vec{f}(\vec{g}(\vec{x}))][D\vec{g}(\vec{x})]. \tag{3.47}$$

Here, as before, the notation $D\vec{f}(\vec{g}(\vec{x}))$ means that we compute $D\vec{f}$ and then evaluate it on the point $\vec{g}(\vec{x})$.

This is a broad chain rule which we can apply to many problems, but we must always remember that *for a function $f\colon \mathbb{R}^n \to \mathbb{R}$, its Jacobian is the transpose of its gradient*. One typical chain rule we can derive from the general one follows below.

Corollary 60. *Let $f\colon \mathbb{R}^p \to \mathbb{R}$ and $\vec{g}\colon \mathbb{R}^n \to \mathbb{R}^p$ be differentiable functions. Let $h\colon \mathbb{R}^n \to \mathbb{R}$ be defined as $h(\vec{x}) = f(\vec{g}(\vec{x}))$ for all $\vec{x} \in \mathbb{R}^n$. Then h is differentiable, and*

$$\nabla h(\vec{x}) = [D\vec{g}(\vec{x})]^\top \nabla f(\vec{g}(\vec{x})). \tag{3.48}$$

Example 61. In this example we will use the chain rule to compute the gradient of the function $h(\vec{x}) = \|A\vec{x} - \vec{y}\|_2^2$. It can be written as $h(\vec{x}) = f(\vec{g}(\vec{x}))$ where $f(\vec{x}) = \|\vec{x}\|_2^2$ and $\vec{g}(\vec{x}) = A\vec{x} - \vec{y}$. We have that

$$D\vec{g}(\vec{x}) = A \tag{3.49}$$

(the proof is in discussion or homework), and also from earlier

$$\nabla f(\vec{x}) = 2\vec{x}. \tag{3.50}$$

Thus applying the chain rule obtains

$$\nabla h(\vec{x}) = [D\vec{g}(\vec{x})]^\top \nabla f(\vec{g}(\vec{x})) \tag{3.51}$$

$$= 2A^\top(A\vec{x} - \vec{y}) \tag{3.52}$$

as desired. This gradient matches what would be computed if we had used the componentwise partial derivatives.

Finally, we again revisit the computational graph perspective. Here, the graph looks like:

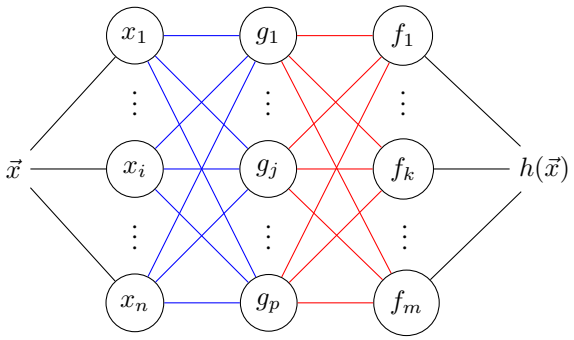


Figure 3.4: Graphical depiction of the multivariate chain rule corresponding to $\vec{h} = \vec{f} \circ \vec{g}$.

To find the partial derivative $\frac{\partial f_k}{\partial x_i}(\vec{x})$, one sums across all paths in the graph from x_i to f_k, i.e.,

$$[D\vec{h}(\vec{x})]_{k,i} = \frac{\partial f_k}{\partial x_i}(\vec{x}) \tag{3.53}$$

$$= \sum_{j=1}^{p} \frac{\partial f_k}{\partial g_j}(\vec{g}(\vec{x})) \cdot \frac{\partial g_j}{\partial x_i}(\vec{x}) \tag{3.54}$$

$$= \sum_{j=1}^{p} [D\vec{f}(\vec{g}(\vec{x}))]_{k,j} \cdot [D\vec{g}(\vec{x})]_{j,i} \tag{3.55}$$

$$= \{[D\vec{f}(\vec{g}(\vec{x}))][D\vec{g}(\vec{x})]\}_{k,i}. \tag{3.56}$$

Here again we use the notation $\frac{\partial f_k}{\partial g_j}$ to denote the derivative of f_k with respect to the output of g_j. Since the output of g_j is really the j^{th} input of f_k, this derivative is just the derivative of f_k with respect to its j^{th} input.

Example 62 (Neural Networks and Backpropagation)**.** In this example, we will develop a basic understanding of what deep neural networks are and how to train them via gradient-based optimization methods, such as those we will study in this class.

The most basic class of neural networks are "multi-layer perceptrons," or functions of the form $\vec{f}\colon \mathbb{R}^n \times \Theta \to \mathbb{R}^p$, where Θ is the so-called "parameter space", and have the form

$$\vec{f}(\vec{x}, \boldsymbol{\theta}) = W^{(m)}\vec{\sigma}^{(m)}(W^{(m-1)}(\cdots \vec{\sigma}^{(1)}(W^{(0)}\vec{x} + \vec{b}^{(0)})\cdots) + \vec{b}^{(m-1)}) + \vec{b}^{(m)} \tag{3.57}$$

$$\text{where} \quad \boldsymbol{\theta} = (W^{(0)}, \vec{b}^{(0)}, W^{(1)}, \vec{b}^{(1)}, W^{(2)}, \vec{b}^{(2)}, \ldots, W^{(m)}, \vec{b}^{(m)}) \in \Theta, \tag{3.58}$$

and $\vec{\sigma}^{(1)}, \ldots, \vec{\sigma}^{(m)}$ are "activation functions," which we treat as generic differentiable nonlinear functions. Here $\boldsymbol{\theta}$ is bolded because it should not be thought of as a scalar, vector, or matrix, but rather as a collection of matrices and vectors, an object we haven't discussed so far in this class.

More precisely, if we define the functions $(\vec{z}^{(i)})_{i=0}^m$ and $(\vec{f}^{(i)})_{i=0}^m$ as

$$\vec{z}^{(0)}(\vec{x}, \boldsymbol{\theta}) = \vec{f}^{(0)}(\vec{x}, \boldsymbol{\theta}) = W^{(0)}\vec{x} + \vec{b}^{(0)} \tag{3.59}$$

$$\vec{z}^{(i)}(\vec{x}, \boldsymbol{\theta}) = \vec{f}^{(i)}(\vec{z}^{(i-1)}(\vec{x}, \boldsymbol{\theta}), \boldsymbol{\theta}) = W^{(i)}\vec{\sigma}^{(i)}(\vec{z}^{(i-1)}) + \vec{b}^{(i)}, \qquad \forall\, i \in \{1, \ldots, m\}, \tag{3.60}$$

then this formulation gives

$$\vec{f}(\vec{x}, \boldsymbol{\theta}) = \vec{z}^{(m)}(\vec{x}, \boldsymbol{\theta}) = \vec{f}^{(m)} \circ \cdots \circ \vec{f}^{(0)}(\vec{x}, \boldsymbol{\theta}). \tag{3.61}$$

When we say that we "train" this neural network, we mean just that we optimize $\boldsymbol{\theta}$ to minimize some loss function evaluated on the data. Given a (differentiable) loss function $\ell\colon \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ and a data point $(\vec{x}, \vec{y}) \in \mathbb{R}^n \times \mathbb{R}^p$, we evaluate the loss on this data point as $\ell(\vec{f}(\vec{x}, \boldsymbol{\theta}), \vec{y})$. The true "empirical loss" across a batch of q data points $(\vec{x}_i, \vec{y}_i)$ is

$$L(\boldsymbol{\theta}) \doteq \sum_{i=1}^{q} \ell(\vec{f}(\vec{x}_i, \boldsymbol{\theta}), \vec{y}_i), \tag{3.62}$$

and a "training algorithm" attempts to optimize $\boldsymbol{\theta}$ to minimize L.

For now, we concentrate on the case where we have $q = 1$, i.e., we have a single data point $(\vec{x}, \vec{y})$. Here we have

$$L(\boldsymbol{\theta}) = \ell(\vec{f}(\vec{x}, \boldsymbol{\theta}), \vec{y}). \tag{3.63}$$

We now explore how to compute $\nabla L(\boldsymbol{\theta})$. In this situation, we are only differentiating with respect to $\boldsymbol{\theta}$, so we fix $\vec{x}$ and $\vec{y}$, which gives the following computational graph:[1]

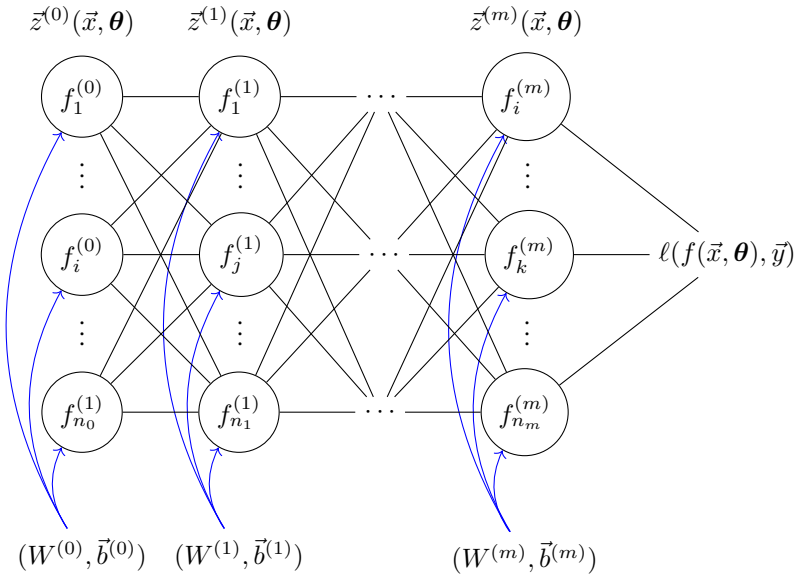


Figure 3.5: Computational graph corresponding to the multi-layer perceptron.

Since we have not defined gradients with respect to matrices (yet), we will focus on computing $\nabla_{\vec{b}^{(i)}} L(\boldsymbol{\theta})$, i.e., the gradient of the loss function L with respect to $\vec{b}^{(i)}$, as well as $\nabla_{\vec{z}^{(i)}} L(\boldsymbol{\theta})$, i.e., the gradient of the loss function L with respect to $\vec{z}^{(i)}(\vec{x}, \boldsymbol{\theta})$. They may be interpreted as sub-vectors (i.e., literally subsets of the entries) of the full gradient $\nabla L(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$. This subscript notation is common in more involved problems which demand derivatives with respect to multiple vector-valued variables.

We first compute $\nabla_{\vec{z}^{(m)}} L(\boldsymbol{\theta})$. We have

$$\nabla_{\vec{z}^{(m)}} L(\boldsymbol{\theta}) = \nabla_{\vec{z}^{(m)}} \ell(\vec{z}^{(m)}, \vec{y}). \tag{3.64}$$

[1] Note that we did not break the graph down to matrix multiplications, vector additions, and applications of $\vec{\sigma}^{(i)}$; we do this for clarity, because the full graph would be rather messy.

Indeed we cannot simplify this further, it being just the gradient of ℓ in its first argument.

Now we handle general $i \in \{1, \dots, m\}$. By chain rule we have the recursion:

$$\nabla_{\vec{z}^{(i-1)}} L(\boldsymbol{\theta}) = [D_{\vec{z}^{(i-1)}} L(\boldsymbol{\theta})]^\top \tag{3.65}$$
$$= [\{D_{\vec{z}^{(i)}} L(\boldsymbol{\theta})\}\{D_{\vec{z}^{(i-1)}} \vec{z}^{(i)}(\vec{x}, \boldsymbol{\theta})\}]^\top \tag{3.66}$$
$$= [D_{\vec{z}^{(i-1)}} \vec{z}^{(i)}(\vec{x}, \boldsymbol{\theta})]^\top [\nabla_{\vec{z}^{(i)}} L(\boldsymbol{\theta})]. \tag{3.67}$$

Now we want to compute $D_{\vec{z}^{(i-1)}} \vec{z}^{(i)}$. Recall that

$$\vec{z}^{(i)} = W^{(i)} \vec{\sigma}^{(i)}(\vec{z}^{(i-1)}) + \vec{b}^{(i)}. \tag{3.68}$$

For convenience, write

$$\vec{u}^{(i)} \doteq \vec{\sigma}^{(i)}(\vec{z}^{(i-1)}), \qquad \text{so that} \quad \vec{z}^{(i)} = W^{(i)} \vec{u}^{(i)} + \vec{b}^{(i)}. \tag{3.69}$$

Now we compute derivatives, obtaining by the chain rule:

$$D_{\vec{z}^{(i-1)}} \vec{z}^{(i)} = [D_{\vec{u}^{(i)}} \vec{z}^{(i)}][D_{\vec{z}^{(i-1)}} \vec{u}^{(i)}] \tag{3.70}$$
$$= W^{(i)} \cdot D\vec{\sigma}^{(i)}(\vec{z}^{(i-1)}). \tag{3.71}$$

This gives us

$$D_{\vec{z}^{(i-1)}} \vec{z}^{(i)} = W^{(i)} \cdot D\vec{\sigma}^{(i)}(\vec{z}^{(i-1)}), \tag{3.72}$$

and so

$$\nabla_{\vec{z}^{(i-1)}} L(\boldsymbol{\theta}) = [W^{(i)} \cdot D\vec{\sigma}^{(i)}(\vec{z}^{(i-1)})]^\top [\nabla_{\vec{z}^{(i)}} L(\boldsymbol{\theta})]. \tag{3.73}$$

To see the true value of this computation, we now compute $\nabla_{\vec{b}^{(i)}} L(\boldsymbol{\theta})$, for each $i \in \{0, \dots, m\}$. By another application of chain rule, we obtain

$$\nabla_{\vec{b}^{(i)}} L(\boldsymbol{\theta}) = [D_{\vec{b}^{(i)}} L(\boldsymbol{\theta})]^\top \tag{3.74}$$
$$= [\{D_{\vec{z}^{(i)}} L(\boldsymbol{\theta})\}\{D_{\vec{b}^{(i)}} \vec{z}^{(i)}\}]^\top \tag{3.75}$$
$$= [\{D_{\vec{z}^{(i)}} L(\boldsymbol{\theta})\} \cdot I]^\top \tag{3.76}$$
$$= [D_{\vec{z}^{(i)}} L(\boldsymbol{\theta})]^\top \tag{3.77}$$
$$= \nabla_{\vec{z}^{(i)}} L(\boldsymbol{\theta}). \tag{3.78}$$

Now to do automatic differentiation by backpropagation, your favorite machine learning framework (such as PyTorch) computes all the gradients $\nabla_{\vec{z}^{(i)}} L(\boldsymbol{\theta})$, starting at $i = m$ and recursing until $i = 0$. *Computing the gradients in this order saves a lot of work, since each derivative is only computed once — this is the main idea of backpropagation*. Once these derivatives are computed, one can use them to compute $\nabla_{\vec{b}^{(i)}} L(\boldsymbol{\theta})$. If we can also compute $\nabla_{W^{(i)}} L(\boldsymbol{\theta})$, then we can compute $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, and thus will be able to optimize L via gradient-based optimization algorithms such as gradient descent, as discussed later in the class. However, this computation will have to wait until slightly later when we cover matrix calculus.

3.1.4 Hessian

So far, we have appropriately generalized the notion of first derivative to vector-valued functions, i.e., functions $\vec{f}\colon \mathbb{R}^n \to \mathbb{R}^m$. We now turn to doing the same with second derivatives.

It turns out that for general vector-valued functions $\vec{f}$, defining a second derivative is possible, but such an object will live in $\mathbb{R}^{m \times n \times n}$ and thus not be hard to work with using the linear algebra we have discussed in this class. However,

for multivariate functions $f\colon \mathbb{R}^n \to \mathbb{R}$, defining this second derivative as a particular matrix becomes possible; this matrix, called the *Hessian*, has great conceptual and computational importance.

Recall that to find the second derivative of a scalar-valued function, we merely take the derivative of the derivative. Our notion of the gradient suffices as a first derivative; to take the derivative of this, we need to use the Jacobian. Indeed, the Hessian is exactly the Jacobian of the gradient, and defined precisely below.

Definition 63 (Hessian)

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice differentiable. The *Hessian* of f is the function $\nabla^2 f\colon \mathbb{R}^n \to \mathbb{R}^{n\times n}$ defined by

$$\nabla^2 f(\vec{x}) = D(\nabla f)(\vec{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(\vec{x}) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}(\vec{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(\vec{x}) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(\vec{x}) \end{bmatrix}. \tag{3.79}$$

It turns out that under some mild conditions, the Hessian is symmetric; this is called Clairaut's theorem.

Theorem 64 (Clairaut's Theorem)

Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, and fix $\vec{x} \in \mathbb{R}^n$. Then $\nabla^2 f(\vec{x})$ is a symmetric matrix, i.e., for every $1 \le i, j \le n$ we have

$$\frac{\partial^2 f}{\partial x_i \, \partial x_j}(\vec{x}) = \frac{\partial^2 f}{\partial x_j \, \partial x_i}(\vec{x}). \tag{3.80}$$

The vast majority of functions we work with in this course are twice continuously differentiable, with some notable exceptions (i.e., the ℓ^1 norm is not even once-differentiable). Thus, in most cases, the Hessian is symmetric.

Example 65 (Hessian of Squared ℓ^2 Norm)**.** In this example we will compute the Hessian of the function $f(\vec{x}) = \|\vec{x}\|_2^2$ where $\vec{x} \in \mathbb{R}^2$. Recall the gradient of this function computed in Example 55 as

$$\nabla f(\vec{x}) = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix}. \tag{3.81}$$

The Hessian can then be computed as

$$\nabla^2 f(\vec{x}) = D(\nabla f)(\vec{x}) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}. \tag{3.82}$$

Example 66. In this example we will compute the gradient and Hessian of the function $h(\vec{x}) = \log(1 + \|\vec{x}\|_2^2)$. Let $f\colon \mathbb{R} \to \mathbb{R}$ be defined by $f(x) = \log(1+x)$, and $g\colon \mathbb{R}^n \to \mathbb{R}$ be defined as $g(\vec{x}) = \|\vec{x}\|_2^2$. Then we have $Df = f'$ is the derivative of f, and $\nabla g(\vec{x}) = 2\vec{x}$ by previous examples. By the Jacobian chain rule, we have

$$\begin{aligned} \nabla h(\vec{x}) &= (Dh(\vec{x}))^\top && (3.83)\\ &= (D(f \circ g)(\vec{x}))^\top && (3.84)\\ &= ([Df(g(\vec{x}))][Dg(\vec{x})])^\top && (3.85)\\ &= ([Dg(\vec{x})]^\top [Df(g(\vec{x}))]^\top) && (3.86)\\ &= [\nabla g(\vec{x})][Df(g(\vec{x}))]^\top && (3.87)\\ &= [\nabla g(\vec{x})][f'(g(\vec{x}))] && (3.88) \end{aligned}$$