CS 182/282A Designing, Visualizing and Understanding Deep Neural Networks
Spring 2021 Sergey Levine

Matrix Calculus Review

This discussion deviates a bit from the week two discussion notes in order to provide you with a more comprehensive review of matrix calculus. We will first review some definitions that will be useful when differentiating vector functions, and then we will go through some examples. We will review the rules of differentiation, and extend them to vector functions. Finally, we will go through some easy, medium, and hard matrix calculus problems, with solutions to be released this weekend.

1 Vector and Matrix Calculus Review

In this section, we review vector and matrix calculus, and formalize the notation we will use. These notations will be required to understand backpropagation in the next lectures. Henceforth, we will denote scalars with lowercase letters (e.g., x), vectors with bolded lowercase letters (e.g., $\mathbf{x}$) and matrices with upper case letters (e.g., X). We will use similar conventions for functions depending on the shape of its output (e.g., $\mathbf{g}(\cdot)$ denotes a function with a vector valued output).

Gradients with respect to vectors We first define the gradient of a scalar function with respect to a vector input. Suppose we have a function $f : \mathbb{R}^d \to \mathbb{R}$, which maps a d-dimensional vector to a scalar. Then we define the gradient of f at a particular input x to be a column vector (the same shape as the input) consisting of partial derivatives at x:

$$\boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\mathbf{x}) \\ \vdots \\ \frac{\partial f}{\partial x_d}(\mathbf{x}) \end{bmatrix}.$$

We note that this choice of notation (laying out gradients to be the same shape as input) is not universally used; you will often find sources using the opposite convention (especially in mathematics) with gradients as row vectors (and Jacobians will be the transpose of what we describe next). However, we will use this convention for deep learning because it is intuitive (for example, in gradient descent, we often write $\theta \leftarrow \theta - \alpha \boldsymbol{\nabla} L(\theta)$, which only makes sense when θ and $\boldsymbol{\nabla} L(\theta)$ are the same shape) and because it is easily extended to gradients for matrices and higher dimensional arrays.

Example 1: Gradient of ℓ_1 norm

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$, and let $f(\mathbf{x}) = \|\mathbf{x}\|_1$. Compute the gradient $\boldsymbol{\nabla} f(\mathbf{x})$.

Example Solution 1: Gradient of ℓ_1 norm

1. The first step we can take is to explicitly write the function in terms of scalar components. This takes the form of the sum $\sum_{i=1}^{d} |x_i|$.
2. The second step is to take partial derivatives with respect to each of the scalar component of the input vector $\mathbf{x}$.

$$\frac{\partial f}{\partial x_j} = \frac{\partial}{\partial x_j} \sum_{i=1}^{d} |x_i| = \frac{\partial}{\partial x_j} |x_j|.$$

3. The third step is to evaluate the scalar derivative.

$$\frac{\partial f}{\partial x_j} = \begin{cases} 1 & x_j > 0 \\ -1 & x_j < 0 \end{cases}$$

4. The final step is to express the scalar components derivatives in vector form. In our notation, a scalar function applied to a vector applies that function elementwise.

$$\boldsymbol{\nabla}_{\mathbf{x}} f = \operatorname{sign}(\mathbf{x})$$

Guided Problem 2: Gradient of squared ℓ_2 norm

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$, and let $f(\mathbf{x}) = \|\mathbf{x}\|_2^2 = \mathbf{x}^\top \mathbf{x}$. Compute the gradient $\boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x})$.

Challenge Problem 3: Gradient of p norm

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$, and let $f(\mathbf{x}) = \|\mathbf{x}\|_p$. Compute the general form of the gradient $\boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x})$ as a function of p.

Jacobians We now consider the case where $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ has vector valued inputs and outputs. Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be the function that outputs the ith component of $\mathbf{f}$. Then, we can view our Jacobian (which we shall denote as $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$) as stacking together the gradients of f_i for $i \in \{1, \dots, m\}$. That is, the Jacobian $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ will be an $n \times m$ matrix with entries given by

$$\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)_{ij} = \frac{\partial f_j}{\partial x_i}.$$

Example Problem 5: Jacobian of a linear map

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d \times n}$. Let $\mathbf{f}(\mathbf{x}) = A^\top \mathbf{x} \in \mathbb{R}^n$. Compute the Jacobian of $\mathbf{f}$ with respect to $\mathbf{x}$.

Example Solution 4: Jacobian of a linear map

1. The first step we can take is to explicitly write the jth component of the vector function in terms of scalar components. This takes the form of the sum below.

$$f_j = \sum_{i=1}^{d} A_{ij} x_i$$

2. The second step is to take partial derivatives of the jth entry of $f(\mathbf{x})$ with respect to the ith entry of the input x_j.

$$\frac{\partial f_j}{\partial x_i}(\mathbf{x}) = \frac{\partial}{\partial x_i} \sum_{k=1}^{d} A_{kj} x_k$$

3. The third step is to evaluate the scalar derivative.

$$\frac{\partial f_j}{\partial x_i}(\mathbf{x}) = \frac{\partial}{\partial x_i} A_{ij} x_i = A_{ij}$$

4. The final step is to express the scalar component derivatives in matrix notation. For this problem, the solution has the following matrix form.

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}) = A$$

Multivariate Chain Rule We first recall the basic chain rule when everything is scalar valued. Suppose we have an input x, compute $y = g(x)$, then compute $z = f(y)$. Then the chain rule says

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x}$$

Now let's consider the case where $\mathbf{y}$ is vector valued in $\mathbb{R}^n$ (x and z remain scalars). Summing over the contributions of each entry of $\mathbf{y}$, we see $\frac{\partial z}{\partial x}$ is now a scalar given by

$$\sum_{i=1}^{n} \frac{\partial y_i}{\partial x} \frac{\partial z}{\partial y_i}.$$

Finally, let's consider the case when $\mathbf{x}$ is also a vector in $\mathbb{R}^m$. From our calculation with scalar x and vector $\mathbf{y}$, we know the jth entry of $\frac{\partial z}{\partial \mathbf{x}}$ is given by the partial derivative

$$\frac{\partial z}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial x_j}.$$

Stacking together the entries $\frac{\partial z}{\partial x_j}$ into a vector, we see that the gradient of the output z with respect to x is given by the product of the Jacobian matrix of $\mathbf{y}$ with respect to $\mathbf{x}$ and the gradient of z with respect to $\mathbf{y}$:

$$\overbrace{\frac{\partial z}{\partial \mathbf{x}}}^{\mathbb{R}^m} = \overbrace{\frac{\partial \mathbf{y}}{\partial \mathbf{x}}}^{\mathbb{R}^{m \times n}} \overbrace{\frac{\partial z}{\partial \mathbf{y}}}^{\mathbb{R}^n}.$$

Guided Problem 5: Combining the two previous calculations with the chain rule

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d \times n}$. Let $\mathbf{g}(\mathbf{x}) = A^\top \mathbf{x} \in \mathbb{R}^n$, and let $f(\mathbf{y}) = \|\mathbf{y}\|_2^2$. Compute the gradient of $f(\mathbf{g}(\mathbf{x}))$ with respect to $\mathbf{x}$.

Challenge Problem 6: Gradient of a single neural network layer

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$, a matrix $A \in \mathbb{R}^{d \times n}$, and a matrix $B \in \mathbb{R}^{n \times m}$. Let $\mathbf{f}(\mathbf{x}) = B^\top \sigma(A^\top \mathbf{x})$, where $\sigma(\cdot)$ is the sigmoid function. Calculate the gradient $\boldsymbol{\nabla}_\mathbf{x} \mathbf{f}(\mathbf{x})$.

Gradients with respect to matrices and higher dimensional arrays Now suppose we have a function $f : \mathbb{R}^{d_1 \times d_2} \to \mathbb{R}$, which maps a d_1 by d_2 matrix to a scalar. We will again define the gradient at a particular input matrix X to be a matrix of the same shape as X, consisting of the partial derivatives with respect to each entry of the matrix.

$$\boldsymbol{\nabla}_X f(X) = \begin{bmatrix} \frac{\partial f}{\partial X_{11}} & \cdots & \frac{\partial f}{\partial X_{1,d_2}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial X_{d_1,1}} & \cdots & \frac{\partial f}{\partial X_{d_1,d_2}} \end{bmatrix}.$$

Similarly, we can generalize this convention of having the gradient match the shape of the input when our input were higher dimensional arrays (e.g. in the weights of a convolutional layer).

We can define a version of a Jacobian for vector-valued functions with matrix inputs that preserves the matrix dimensions (similarly for higher dimensional arrays as well). Suppose $\mathbf{f} : \mathbb{R}^{d_1, d_2} \to \mathbb{R}^n$, then we can define the Jacobian to be a rank-3 tensor (an array with 3 indices) in $\mathbb{R}^{d_1 \times d_2 \times n}$ with each entry given by

$$\left(\frac{\partial \mathbf{f}}{\partial X}\right)_{ijk} = \frac{\partial f_k}{\partial X_{ij}}.$$

We will now go through the chain rule calculation again, this time with a matrix input. Suppose $X \in \mathbb{R}^{d_1, d_2}$, $\mathbf{y} = \mathbf{g}(X) \in \mathbb{R}^n$ and $z = f(\mathbf{y}) \in \mathbb{R}$. Again, we have that the partial derivative with respect to each entry of the matrix X_{ij} is given by

$$\frac{\partial z}{\partial X_{ij}} = \sum_{k=1}^{n} \frac{\partial z}{\partial y_i} \frac{\partial y_i}{\partial X_{ij}}.$$

Similarly to the vector input case, we can again succinctly write out the full gradient with respect to the matrix X as

$$\overbrace{\frac{\partial z}{\partial X}}^{\mathbb{R}^{d_1 \times d_2}} = \overbrace{\frac{\partial y}{\partial X}}^{\mathbb{R}^{d_1 \times d_2 \times n}} \overbrace{\frac{\partial z}{\partial y}}^{\mathbb{R}^n}.$$

Note that the product of the rank-3 tensor (or 3-dimensional array) and vector can be a seen as a generalization of a matrix vector multiplication. Multiplying a matrix $X \in \mathbb{R}^{m \times n}$ by a vector $y \in \mathbb{R}^n$ results in a vector in $\mathbb{R}^m$ where each entry is the inner product of a row of X with y. The product of a rank-3 tensor $A \in \mathbb{R}^{d_1 \times d_2 \times n}$ with a vector $\mathbf{b} \in \mathbb{R}^n$ then forms a $d_1 \times d_2$ matrix, where each entry is the inner product of a "row" of A and $\mathbf{b}$.

We also note that this calculation of the gradient with respect to a matrix X is equivalent to first flattening X to a vector, computing the gradient with respect to the flattened X using the previous multivariate chain rule for vectors, and then reshaping the gradients back to match the original matrix shape of X.

Guided Problem 7: Revisiting with a matrix derivative instead

In problem 5, we computed the gradient of $z = \left\|A^\top \mathbf{x}\right\|_2^2$ with respect to $\mathbf{x}$. We will now repeat this exercise, but instead compute the gradient with respect to A.
Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d\times n}$. Let $\mathbf{g}(A) = A^\top \mathbf{x} \in \mathbb{R}^n$, and let $f(\mathbf{y}) = \|\mathbf{y}\|_2^2$. Compute the gradient of $f(\mathbf{g}(A))$ with respect to A.

Finally, as a matter of notation, note that the way we order derivatives in our chain rule (with the final output on the rightmost side) is again a result of our chosen convention for gradients and Jacobians. You may notice in other texts that the chain rule is written in the reversed order using a different convention for Jacobians.

Challenge Problem 8: Gradient with respect to the first layer weights

Suppose we have a vector $\mathbf{x} \in \mathbb{R}^d$, a vector $\mathbf{c} \in \mathbb{R}^m$, a matrix $A \in \mathbb{R}^{d\times n}$, and a matrix $B \in \mathbb{R}^{n\times m}$. Define a loss function $\mathcal{L}(A) = \|B^\top \sigma(A^\top \mathbf{x}) - \mathbf{c}\|_2^2$. Calculate the gradient $\boldsymbol{\nabla}_A \mathcal{L}(A)$.