Jacobians and Stuff

David S. Rosenberg

1 Derivative of a function $f : \mathbf{R} \to \mathbf{R}$

2 Directional derivative of a function $f : \mathbf{R}^n \to \mathbf{R}^m$

The **directional derivative** of $f : \mathbf{R}^n \to \mathbf{R}^m$ at c in the direction u is

$$f'(c; u) = \lim_{h \to 0} \frac{f(c + hu) - f(c)}{h},$$

whenever the limit on the right exists.

To gain some intuition, let's drop the limit and replace equality with approximate equality. Then we can rearrange the expression as

$$f(c + hu) - f(c) \approx hf'(c; u).$$

This has an easy interpretation: if we start at c and move to $c + hu$, then the value of f increases by approximately $hf'(c; u)$. This is called a **first order** approximation, because we used the first derivative information at x.

If $u = u_k$, the kth unit coordinate vector with a 1 in the kth position and 0's elsewhere, then $f'(c; u_k)$ is a special directional derivative called a **partial derivative** and is denoted by $D_k f(c)$. Note that $D_k f(c)$ is vector valued.

3 Total derivative of a function $f : \mathbf{R}^n \to \mathbf{R}^m$

The function $f : \mathbf{R}^n \to \mathbf{R}^m$ is said to be **differentiable** at c if there exists a *linear* function $T_c : \mathbf{R}^n \to \mathbf{R}^m$ such that

$$f(c + v) = f(c) + T_c(v) + \|v\| E_c(v),$$

where $E_c(v) \to 0$ as $v \to 0$. The linear function T_c is called the **total derivative** of f at c.

We can get all directional derivatives from a total derivative: If f is differentiable at $c \in \mathbf{R}^n$ with total derivative T_c, then the directional derivative $f'(c; v)$ exists for every v in $\mathbf{R}^n$ and we have

$$T_c(v) = f'(c; v).$$

We can also write the total derivative T_c as $f'(c)$. With this notation, we can write

$$f'(c)(v) = f'(c; v).$$

3.1 Total derivative in terms of partial derivatives (i.e. do we really understand linearity?)

Note that $f'(c)(v) \in \mathbf{R}^m$. In fact, it's a linear combination of the partial derivatives of f: If $v = (v_1, \ldots, v_n) \in \mathbf{R}^n$, then the directional derivative at c in the direction v is

$$f'(c)(v) = \sum_{k=1}^{n} v_k D_k f(c).$$

Let's ponder the significance of his for a moment.

If f is real-valued (i.e. $m = 1$) then

$$f'(c)(v) = \nabla f(c) \cdot v,$$

where $\nabla f(c) = (D_1 f(c), \ldots, D_n f(c))$ is called the **gradient vector** of f at c. It is defined at each point where the partials exist.

It is also convenient to write the first-order Taylor formula as

$$\begin{aligned} f(x) &\approx f(c) + f'(c)(x - c) \\ &= [f(c) - f'(c)(c)] + f'(c)(x) \end{aligned}$$

where in the first line we have simply taken $x = c + v$ in the definition of total derivative, and dropped the $E_c(v)$ term, which quickly converges to 0. In the second line, we used the linearity of $f'(c)$ to break up it's application to $x - c$ into two terms. If f is real-valued, this looks like

$$f(x) \approx [f(c) - \nabla f(c) \cdot c] + \nabla f(c) \cdot v.$$

[I think it will be easier to just start right in with the partial derivatives stuff, and then say that the larger groups follow trivially.]

We defined the total derivative of a function $f : \mathbf{R}^n \to \mathbf{R}^m$. Suppose an input $c \in \mathbf{R}^n$ naturally divides into two pieces, say $c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$ where $c_1 \in \mathbf{R}^{n_1}$ and $c_2 \in \mathbf{R}^{n_2}$, and $n = n_1 + n_2$.

Define $f_{c_2}(c_1) = f\begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$ as a function of c_1 for fixed c_2. And similarly define $f_{c_1}(c_2)$. And suppose we have $f'_{c_2}(c_1)$ and $f'_{c_1}(c_2)$. Then

Concept Check Question: Can we put these together to get the total derivative $f'(c)$?

Let's decompose $v = v_1^0 + v_2^0$. By the linearity of total derivatives, we have

$$\begin{aligned} f'(c)(v) &= f'(c)(v_1^0) + f'(c)(v_2^0) \\ &= f'_{c_2}(c_1)(v_1) + f'_{c_1}(c_2)(v_2) \end{aligned}$$

So there were two key steps. In the first step we used linearity to divide up the contribution The key step was th

The simplest idea would be that if you know how f changes in response to a change in c_1 (with all else fixed) and also how f changes in response to a change in c_2 (with all else fixed), then perhaps the change in f when we change both c_1 and c_2 is just the sum of the changes due to c_1 and the changes due to c_2. On the other hand, what if something non-obvious happens when we change c_1 and c_2 simultaneously. Could there be some interaction that leads to some different behavior when we change c_1 and c_2 simultaneously? The answer is that although there can certainly be interactions between the effects of c_1 and c_2, these will not show up in a first-order Taylor approximation. This idea is built right into the definition of the total derivative as a **linear** function, which we'll expand on now.

Let's make this concrete. Let $c_1^0 = \begin{pmatrix} c_1 \\ 0 \end{pmatrix} \in \mathbf{R}^n$ and $c_2^0 = \begin{pmatrix} 0 \\ c_2 \end{pmatrix} \in \mathbf{R}^n$. So $c = c_1^0 + c_2^0$. In the same way, let's decompose $v = v_1^0 + v_2^0$. By the linearity of total derivatives, we have

$$f'(c)(v) = f'(c)(v_1^0) + f'(c)(v_2^0)$$

anybody

Let's work with the directional derivative characterization $f'(c)(v) = f'(c; v)$, where $v = (v_1, v_2)$ with $v_1 \in \mathbf{R}^{n_1}$ and $v_2 \in \mathbf{R}^{n_2}$.

$$f(c_1 + v_1, c_2 + v_2)$$

$$f(c + v) = f(c) + T_c(v) + \|v\| E_c(v),$$

where $E_c(v) \to 0$ as $v \to 0$. The linear function T_c is called the **total derivative** of f at c.

Concept check:

$g_1 : \mathbf{R}^p \to \mathbf{R}^{n_1}$ is differentiable at a with total derivative $g_1'(a)$, and

- $g_2 : \mathbf{R}^p \to \mathbf{R}^{n_2}$ is differentiable at a with total derivative $g_2'(a)$, where

$n = n_1 + n_2$. And suppose we define the vector-valued function $g : \mathbf{R}^p \to \mathbf{R}^n$ by $g(a) = (g_1(a), g_2(a))$

Consider the function $f(x) = 1 - x + x^2$. So $D_1 f(x) = 2x - 1$. The total derivative of f at c is

$$f'(c)(h) = \frac{df(c)}{dx} v = (2c - 1)\, h.$$

for $v \in \mathbf{R}$. The interpretation is that if we start at c and move to $c + h$, f change by $(2c - 1)\, h$. The first-order Taylor approximation to f at c is then given by

$$\begin{aligned} f(x) &\approx f(c) + (2c - 1)(x - c) \\ &= \left[1 - c + c^2 - 2c^2 + c\right] + (2c - 1)\, x \\ &= \left[1 - c^2\right] + (2c - 1)\, x \end{aligned}$$

In the table below, $D_1 f(c)$ is the usual derivative of f at c, $f'(c)(h)$ is the total derivative of f at c

c	$f(c)$	$D_1 f(c)$	$f'(c)(h)$	1st-Order Taylor Approximation to f at c
0	1	-1	$f'(0)(h) = -h$	$f(x) \approx 1 + (-1)(x - 0) = 1 - x$
1	1	1	$f'(1)(h) = h$	$f(x) \approx 1 + (1)(x - 1) = x$
2	3	3	$f'(2)(h) = 3h$	$f(x) \approx 1 + 3(x - 2) = -5 + 3x$

4 The chain rule

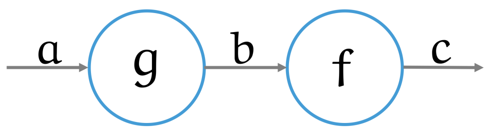
Suppose

- $g : \mathbf{R}^p \to \mathbf{R}^n$ is differentiable at a with total derivative $g'(a)$, and
- $f : \mathbf{R}^n \to \mathbf{R}^m$ is differentiable at $b = g(a)$ with total derivative $f'(b)$.

Then the composition $h = f \circ g$ is differentiable at a and has total derivative given by

$$h'(a) = f'(b) \circ g'(a).$$

The figure below depicts a computation graph corresponding to the this theorem:



We have $b = g(a)$ is the output of function g, and also the input of function f. Then $c = f(b) = f(g(a))$ is the output of the function f.

5 Chain rule with multiple inputs

Let's introduce the following two functions

- $g_1 : \mathbf{R}^p \to \mathbf{R}^{n_1}$ is differentiable at a with total derivative $g_1'(a)$, and
- $g_2 : \mathbf{R}^p \to \mathbf{R}^{n_2}$ is differentiable at a with total derivative $g_2'(a)$, where

$n = n_1 + n_2$. And suppose we define the vector-valued function $g : \mathbf{R}^p \to \mathbf{R}^n$ by $g(a) = (g_1(a), g_2(a))$. As before, let's take

- $f : \mathbf{R}^n \to \mathbf{R}^m$ is differentiable at $b = (g_1(a), g_2(a))$ with total derivative $f'(b)$.

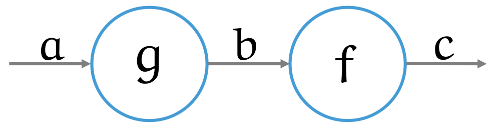
So

$$g'(a)(v_1, v_2) = g_1'(a)(v_1) + g_2'(a)(v_2).$$

Then the composition $h = f \circ g$ is differentiable at a and has total derivative given by

$$h'(a) = f'(b) \circ g'(a).$$

The figure below depicts a computation graph corresponding to the this theorem:



We have $b = g(a)$ is the output of function g, and also the input of function f. Then $c = f(b) = f(g(a))$ is the output of the function f.

6 affine transformation

Consider the function

$$f(x) = w^T x + b$$

where $w = (w_1, \ldots, w_d) \in \mathbf{R}^d$, $x = (x_1, \ldots, x_d) \in \mathbf{R}^d$, and $b \in \mathbf{R}$. It's useful to keep in mind that each entry w_i in the vector w has a very specific meaning in relation to the function f. In particular, it tells us the rate of change of f as we change x_i. To see that precisely, let $e_i = (0, 0, \ldots, 1, \ldots, 0)$ be the ith unit coordinate vector, which has a 1 in the ith position and 0's elsewhere. Then for any $h \in \mathbf{R}$, the change in the function value when we add h to the ith coordinate of x is

$$\begin{aligned} f(x + he_i) - f(x) &= w^T (x + he_i) + b - \left(w^T x + b\right) \\ &= hw^T e_i = hw_i. \end{aligned}$$

Now consider a more general linear mapping $f(x) = Wx$, where $W \in \mathbf{R}^{r \times d}$ and w_{ij} is the ij'th entry of W. Then if we again consider the change in the function evaluation when we increment x_i by $h \in \mathbf{R}$:

$$\begin{aligned} f(x + he_i) - f(x) &= W (x + he_i) - Wx = hWe_i \\ &= h \begin{pmatrix} w_{1i} \\ w_{2i} \\ \vdots \\ w_{ri} \end{pmatrix}. \end{aligned}$$

So when we change x_i by h, each of the the jth output changes by $w_{ji}h$. In words, the w in the ith row tells us how the ith output changes, and the jth column tells us the effect of changing the jth input.

why is this important?

7 Linear Mappings

Consider the simple linear neural network acting on a single input $x \in \mathbf{R}^d$.

$$h = W_1 x$$

and $W_1 \in \mathbf{R}^{r \times d}$ and

$$y = w^T h.$$

for $w \in \mathbf{R}^{r \times 1}$. Suppose we've already calculated the matrix $D_h y$. It's the row vector w^T. Then

$$\begin{aligned} D_{W_1} y &= (D_h y)(D_{W_1} h) \\ &= w^T \end{aligned}$$

$$\frac{\partial y}{\partial W_1} = \frac{\partial y}{\partial h}()$$

So we w

$$f(x; W_1, W_2) = W_2 W_1 x$$

8 Chain rule in two pieces

Consider a function $f : \mathbf{R}^n \to \mathbf{R}^m$, where the o.

9 Total derivative in two pieces

10 Affine Mapping

Consider the simple affine unit acting on a single input $x \in \mathbf{R}^d$ and producing b affine transformations of x:

$$g(x, W, b) = Wx + b$$

for $x \in \mathbf{R}^{d \times 1}$, $W \in \mathbf{R}^{r \times d}$ and $b \in \mathbf{R}^{r \times 1}$. And suppose the rest of the network is represented by a function $f : \mathbf{R}^r \to \mathbf{R}$. So the full network is represented by the function

$$(f \circ g)(x, W, b) = f(g(x, W, b)$$

Now let's fix some values of x, W, and b, say x_0, W_0, and b_0. Now let

$$\begin{aligned} h &= g(x_0, W_0, b_0) \\ y &= f(h) \end{aligned}$$

We want to find the gradient of y with respect to W and b and the points x_0, W_0, b_0. Equivalently, we want to know the total derivative of $(f \circ g)$ at (x_0, W_0, b_0). Suppose we've already calculated the total derivative of f at the point h, namely $f'(h)$. We can represent this by $\frac{\partial f}{\partial h}$ or $D_h f$. In either case, it's a $1 \times r$ row matrix. It's transpose would be the gradient $\nabla_h f$. So $f'(h)(v)$ is a first-order approximation to the difference $f(h+v) - f(h)$. Then by the chain rule, the total derivative of $(f \circ g)$ at (x_0, W_0, b_0) is

$$(f \circ g)'(x_0, W_0, b_0)(\Delta_x, \Delta_W, \Delta_b) = f'(h)(g'(x_0, W_0, b_0)(\Delta_x, \Delta_W, \Delta_b)).$$

Glad we simplified everything into such an easy notation.

As argued above, it's sufficient to compute the total derivative w.r.t. each input, and then just add them together.

$$\begin{aligned} g(x, W + \Delta_W, b) &= (W + \Delta_W)\, x + b \\ &= g(x, W, b) + \Delta_W x \end{aligned}$$

So

$$g'_W(x, W, b)(\Delta_W) = \Delta_W x$$

Now

$$\begin{aligned} (f \circ g_{x,b})'(W)(\Delta_W) &= f'(g_{x,b}(W)) \circ g'_{x,b}(W)(\Delta_W) \\ &= f'(g_{x,b}(W))(\Delta_W x) \end{aligned}$$

And for b we have

$$\begin{aligned} g(x, W, b + \Delta_b) - g(x, W, b) &= Wx + b + \Delta_b - (Wx + b) \\ &= \Delta_b \end{aligned}$$

So

$$g'_b(x, W, b)(\Delta_b) = \Delta_b$$

$$\frac{\partial g}{}$$

$$\begin{aligned} g(x+\Delta_x, W+\Delta_W, b+\Delta_b) &= (W+\Delta_W)(x+\Delta_x)+b+\Delta_b \\ &= g(x,W,b)+W\Delta_x+\Delta_W(x+\Delta_x)+\Delta_b \end{aligned}$$

$$\begin{aligned} (f\circ g)(x+\Delta_x, W+\Delta_W, b+\Delta_b) &= f() \\ (f\circ g)(x,W,b) \end{aligned}$$

(x,W,b)Suppose our network output is $y \in \mathbf{R}$, and our ultimate goal is to compute $D_W y$ and $D_b y$, for use in gradient descent. Suppose also that we've already calculated $D_h y$. $h \in \mathbf{R}^{r\times 1}$, so $D_h y \in \mathbf{R}^{1\times r}$. By the chain rule,

$$D_W y = D_h y D$$

$$y = w^T h.$$

for $w \in \mathbf{R}^{r\times 1}$. Suppose we've already calculated the matrix $D_h y$. It's the row vector w^T. Then

$$\begin{aligned} D_{W_1} y &= (D_h y)(D_{W_1} h) \\ &= w^T \end{aligned}$$

$$\frac{\partial y}{\partial W_1} = \frac{\partial y}{\partial h}()$$

So we w

$$f(x; W_1, W_2) = W_2 W_1 x$$

11 Linear mapping with a minibatch

Now supposed $X \in \mathbf{R}^{n\times d}$ and $W_1 \in \mathbf{R}^{r\times d}$ and

$$H = XW_1^T$$

where $H \in \mathbf{R}^{n\times r}$ and

$$y = \frac{1}{n}1^T Hw$$

for $w \in \mathbf{R}^{r\times 1}$. Now

$$\begin{aligned} \frac{1}{n}1^T (H+\Delta) w - \frac{1}{n}1^T Hw &= \frac{1}{n}1^T \Delta w \\ &= \mathrm{tr}\left(\frac{1}{n}w1^T\Delta\right) \end{aligned}$$

So

$$D_H y = \frac{1}{n}1w^T.$$

Now

$$D_{W_1}y = (D_H y)(D_{W_1}H)$$

and what's $D_{W_1}H$? This woudl be a 4-dimensional array. Let's avoid thinking about it explicitly. And go back to the transformation view. Suppose we change W_1 by Δ. Then H changes by

$$X(W_1+\Delta)^T - XW_1^T = X\Delta^T$$

and then from teh way linear maps work (and this should be clearly established above), we know that when we compose this with the next mapping the changes compose. So when W_1 increases by Δ, y increases by

$$\begin{aligned} \mathrm{tr}\left((D_H y)^T X\Delta^T\right) &= \mathrm{tr}\left(\Delta X^T (D_H y)\right) \\ &= \mathrm{tr}\left(X^T (D_H y)\Delta\right) \end{aligned}$$

So

$$D_{W_1}y = (D_H y)^T X$$

Does this mean $D_{W_1}H = X$? that doesn't make sense... does it? shouldn't it be a 4-dim array?

Anyway, doesn't seem we're actually using this form: $D_{W_1}y = (D_H y)(D_{W_1}H)$ of the chain rule, but rather the more basic composition of total derivatives form.

12 Affine mapping with a minibatch

Now supposed $X \in \mathbf{R}^{n\times d}$ and $W \in \mathbf{R}^{r\times d}$ and $b \in \mathbf{R}^{r\times 1}$. We need to add b^T to every row of H. For math purposes,

$$H = XW_1^T$$

where $H \in \mathbf{R}^{n\times r}$ and

$$y = \frac{1}{n}1^T Hw$$

for $w \in \mathbf{R}^{r\times 1}$. Now

$$\begin{aligned} \frac{1}{n}1^T\left(H+\Delta\right)w - \frac{1}{n}1^T Hw &= \frac{1}{n}1^T\Delta w \\ &= \operatorname{tr}\left(\frac{1}{n}w1^T\Delta\right) \end{aligned}$$

So

$$D_H y = \frac{1}{n}1w^T.$$

Now

$$D_{W_1}y = \left(D_H y\right)\left(D_{W_1}H\right)$$

and what's $D_{W_1}H$? This woudl be a 4-dimensional array. Let's avoid thinking about it explicitly. And go back to the transformation view. Suppose we change W_1 by Δ. Then H changes by

$$X\left(W_1+\Delta\right)^T - XW_1^T = X\Delta^T$$

and then from teh way linear maps work (and this should be clearly established above), we know that when we compose this with the next mapping the changes compose. So when W_1 increases by Δ, y increases by

$$\begin{aligned} \operatorname{tr}\left(\left(D_H y\right)^T X\Delta^T\right) &= \operatorname{tr}\left(\Delta X^T\left(D_H y\right)\right) \\ &= \operatorname{tr}\left(X^T\left(D_H y\right)\Delta\right) \end{aligned}$$

So

$$D_{W_1}y = \left(D_H y\right)^T X$$

Does this mean $D_{W_1}H = X$? that doesn't make sense... does it? shouldn't it be a 4-dim array?

Anyway, doesn't seem we're actually using this form: $D_{W_1}y = (D_H y)(D_{W_1}H)$ of the chain rule, but rather the more basic composition of total derivatives form.

13 asdf

Consider the linear function $f(x;w) = w^T x$, for $w, x \in \mathbf{R}^{d\times 1}$. So the total derivative w.r.t. w is a linear mapping from $\mathbf{R}^d$ to $\mathbf{R}$, which is represented by the gradient $\nabla_w f = x$.

Now suppose we want to compute this function for a batch of x's, which we stack into a design matrix $X \in \mathbf{R}^{n\times d}$ in the usual way. Then we have the linear mapping $f(X;w) = Xw$ for $w \in \mathbf{R}^d$ and $X \in \mathbf{R}^{n\times d}$. As a function of w, this is a mapping from $\mathbf{R}^d \to \mathbf{R}^n$, and thus the total derivative w.r.t. w is a linear mapping from $\mathbf{R}^d$ to $\mathbf{R}^n$, which we can represent naturally by the $n \times d$ Jacobian matrix, which is actually just X.

Now suppose we have a vector-valued function $f(x;W) = Wx$, where $W \in \mathbf{R}^{r\times d}$ and $x \in \mathbf{R}^{d\times 1}$. As a function of W, this is a mapping from $\mathbf{R}^{r\times d}$ to $\mathbf{R}^r$we have Then the total derivative w.r.t. W is a linear mapping from $\mathbf{R}^{r\times d}$ to $\mathbf{R}^r$. So representing this linear mapping requires a tensor $\mathbf{R}^{r\times d\times r}$. This is a little confusing, so let's introduce some notation. Let $y = Wx$, where $y = (y_1, \ldots, y_r) \in \mathbf{R}^r$. Then $\nabla_W y_i$ is an $\mathbf{R}^{r\times d}$ matrix. And we'll get one of those for each y_i. And then we'd stack them in an array of shape $r \times d \times r$. So $y_i = (Wx)_i = \sum_{j=1}^d W_{ij}x_j$. So

$$\frac{\partial y_i}{\partial W_{ab}} = \begin{cases} x_b & \text{if } a = i \\ 0 & \text{otherwise.} \end{cases}.$$

$$\nabla_W y_i = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ x_1 & x_2 & \cdots & & x_d \\ & 0 & 0 & & \\ & & & & \end{pmatrix}$$

In other words, all rows are 0 except the ith row, since only the ith row contributes to the ith output y_i. And the ith row is just going to be x.

But what we really want to compute is

$$\nabla_w J = (\nabla_y J) \nabla_w y$$

$$J'$$

So we have $W \mapsto y \mapsto J$. Then

$$\begin{aligned} T_W J(\Delta) &= T_y J(T_W y(\Delta)) \\ &= T_y J(\Delta x) \\ &= (\nabla_y J)^T \Delta x \\ &= \operatorname{trace}\left(x (\nabla_y J)^T \Delta\right). \end{aligned}$$

Suppose Δ is 1 in the ijth entry, and 0 elsewhere. Then we have the partial derivative in the direction Δ_{ij}. Which is $\left[(\nabla_y J) x^T\right]_{ij}$. And so the gradient can be represented as the matrix $(\nabla_y J) x^T$.

A Linear mappings between $f : \mathbf{R}^{a\times b} \to \mathbf{R}^{m\times n}$.

A.1 Is there even anything new going on here?

On the one hand, there really is nothing new going on with $f : \mathbf{R}^{a\times b} \to \mathbf{R}^{m\times n}$ compared to $f : \mathbf{R}^n \to \mathbf{R}^m$. A matrix is nothing but a particular way of arranging as presenting a vector of numbers. Given a matrix, one can convert it to a vector just by stacking the columns into one giant vector. In fact, there is a standard notation for that operation. Formally, if A is an $m \times n$ matrix, and we write a_i for the ith column of A, then we define the vec of A as the column $mn \times 1$ column vector

$$\operatorname{vec}(A) = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}.$$

So we can always think of a function $f : \mathbf{R}^{a\times b} \to \mathbf{R}^{m\times n}$ mapping matrices to matrices as a function $f : \mathbf{R}^{ab} \to \mathbf{R}^{mn}$ mapping vectors to vectors.

However, this vectorized version of matrix functions is not always convenient. Many matrix functions are much easier to work with and express in matrix form. For example, a function $f : X \mapsto X^{-1}$ defined on the space of $n \times n$ invertible matrices. Expressing this function explicity in terms of entries of X will be very difficult except for very small matrices.

A.2 Ok, seems worthwhile. Let's Investigate.

How can we represent a general linear function from $a \times b$ matrices to $m \times n$ matrices? Let's go through some false starts. First, it definitely can't be of the form $f(X) = MX$, for some matrix M. Let $M \in \mathbf{R}^{m\times a}$ so that at least MX is defined. But the product matrix will be $m \times b$, rather tham $m \times n$.

To end up with an $m \times n$ matrix, the simplest thing we can do is $f(X) = MXN^T$, for $M \in \mathbf{R}^{m\times a}$ and $N \in \mathbf{R}^{n\times b}$. Now every entry of $f(X)$ is given by

$$[f(X)]_{ij} = m_i X n_j^T,$$

where m_i is the ith row of M and n_j is the jth row of N. It's the right shape, but clearly not the most general function we could have. The most general form we could have is

$$[f(X)]_{ij} = \operatorname{tr}\left(A_{ij}^T X\right),$$

where A_{ij} is an $a \times b$ matrix, and we have a different one for every output coordinate ij.

So to specify such a linear mapping, we'd need a whole matrix of matrices.

A Derivative of a function $f : \mathbf{R}^{a\times b} \to \mathbf{R}^{m\times n}$.

B Affine transformation

$$y = Wx + b$$

where y and b are $m \times 1$, x is $d \times 1$, and W is $m \times d$.

Now there is also some function $f : \mathbf{R}^m \to \mathbf{R}$, and let's write $J = f(Wx + b)$. Our goal is to find the partial derivative of J with respect to each element of W, namely $\partial J/\partial W_{ij}$. Suppose we have already computed the of all partial derivatives of J with respect to the intermediate variable y, namely $\frac{\partial J}{\partial y_i}$ for $i = 1, \ldots, m$. Then by the chain rule, we have

$$\frac{\partial J}{\partial W_{ij}} = \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} \frac{\partial y_r}{\partial W_{ij}}.$$

Now $y_r = W_{r\cdot}x + b_r = b_r + \sum_{k=1}^{d} W_{rk}x_k$. So

$$\frac{\partial y_r}{\partial W_{ij}} = x_k \delta_{ir}\delta_{jk} = x_j \delta_{ir}$$

Putting it together we get

$$\begin{aligned}\frac{\partial J}{\partial W_{ij}} &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} x_j \delta_{ir} \\ &= \frac{\partial J}{\partial y_i} x_j\end{aligned}$$

We can represent these partial derivatives as a matrix and compute it where the ij'th entry of $\frac{\partial J}{\partial W}$ is $\frac{\partial J}{\partial W_{ij}}$, i.e. the partial derivative of J w.r.t. the parameter W_{ij}. It's gonna be

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial y} x^T,$$

where $\frac{\partial J}{\partial y}$ is $m \times 1$ and x is $d \times 1$. So this is an outer product of two vectors, yielding an $m \times d$ matrix.

We'll also need the derivative w.r.t x – if it's actually data, we don't need the derivative w.r.t. x, but when we chain things together, x will be the output of another unit:

$$\frac{\partial y_r}{x_i} = W_{ri}$$

$$\begin{aligned}\frac{\partial J}{\partial x_i} &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} \frac{\partial y_r}{\partial x_i} \\ &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} W_{ri} \\ &= \left(\frac{\partial J}{\partial y}\right)^T W_{\cdot i}\end{aligned}$$

and

$$\frac{\partial J}{\partial x} = W^T \left(\frac{\partial J}{\partial y}\right)$$

will give us a column vector.

Similarly,

$$\begin{aligned}\frac{\partial J}{\partial b_i} &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r}\frac{\partial y_r}{\partial b_i} \\ &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r}\delta_{ir} \\ &= \frac{\partial J}{\partial y_i}\end{aligned}$$

Let's repeat the same calculations for a minibatch. Let's suppose we have n inputs $x_1, \ldots, x_n \in \mathbf{R}^d$, and we stack them in the usual way as rows in a $n \times d$ design matrix X. For each x_i there's an intermediate output $y_i = Wx_i + b$. Let's consider stacking these as rows as well, so each row is $y_i^T = x_i^T W^T + b^T$. Let's write Y for the $n \times m$ matrix, which stacks the n row vectors y_i^T on top of each other. Then we have

$$Y = XW^T + b^T,$$

and the rs'th entry is given by

$$\begin{aligned}Y_{rs} &= X_{r\cdot}\left(W^T\right)_{\cdot s} + 1b^T, \\ &= \sum_{k=1}^{d} X_{rk}\left(W^T\right)_{ks} + b_s \\ &= \sum_{k=1}^{d} X_{rk}W_{sk} + b_s\end{aligned}$$

whee 1 is an $n \times 1$ column vector. where the notation $X_{r\cdot}$ refers the the rth row of X, as a row matrix, and similarly $X_{\cdot s}$ refers to the sth column of X, as a column matrix. Now

$$\begin{aligned}\frac{\partial Y_{rs}}{\partial W_{ij}} &= X_{rk}\delta_{is}\delta_{jk} = X_{rj}\delta_{is} \\ \frac{\partial Y_{rs}}{\partial b_i} &= \delta_{is} \\ \frac{\partial Y_{rs}}{\partial X_{ij}} &= \sum_{k=1}^{d} W_{sk}\delta_{ir}\delta_{jk} = W_{sj}\delta_{ir}\end{aligned}$$

(Note – the necessity for the δ_{ir} should be obvious if we understand what rows of Y and X are.)

Now we have a function $f : \mathbf{R}^{n\times m} \to \mathbf{R}$ that operates on a full minibatch and produces a single scalar. This would typically be the average of the $f(Wx_i + b)$ over $i = 1, \ldots, n$. So

$$
\begin{aligned}
\frac{\partial J}{\partial W_{ij}} &= \sum_{r=1}^{n}\sum_{s=1}^{m} \frac{\partial J}{\partial Y_{rs}} \frac{\partial Y_{rs}}{\partial W_{ij}} \\
&= \sum_{r=1}^{n}\sum_{s=1}^{m} \frac{\partial J}{\partial Y_{rs}} X_{rj}\delta_{is} \\
&= \sum_{r=1}^{n} \frac{\partial J}{\partial Y_{ri}} X_{rj} \\
&= \left[\left(\frac{\partial J}{\partial Y}\right)_{\cdot i}\right]^T X_{\cdot j}
\end{aligned}
$$

where $\frac{\partial J}{\partial Y}$ is the $n \times m$ matrix with $\frac{\partial J}{\partial Y_{ij}}$ in the ij'th entry. So

$$
\frac{\partial J}{\partial W} = \left(\frac{\partial J}{\partial Y}\right)^T X
$$

and

$$
\begin{aligned}
\frac{\partial J}{\partial b_i} &= \sum_{r=1}^{n}\sum_{s=1}^{m} \frac{\partial J}{\partial Y_{rs}} \frac{\partial Y_{rs}}{\partial b_i} \\
&= \sum_{r=1}^{n}\sum_{s=1}^{m} \frac{\partial J}{\partial Y_{rs}} \delta_{is} \\
&= \sum_{r=1}^{n} \frac{\partial J}{\partial Y_{ri}} \\
&= 1^T \left(\frac{\partial J}{\partial Y}\right)_{\cdot i}
\end{aligned}
$$

and if we let $\frac{\partial J}{\partial b}$ be the $b \times 1$ vector of derivatives $\frac{\partial J}{\partial b_i}$, then we can write

$$
\frac{\partial J}{\partial b} = \left(\frac{\partial J}{\partial Y}\right)^T 1.
$$

Finally,

$$\begin{aligned}
\frac{\partial J}{\partial X_{ij}} &= \sum_{r=1}^{n}\sum_{s=1}^{m}\frac{\partial J}{\partial Y_{rs}}\frac{\partial Y_{rs}}{\partial X_{ij}} \\
&= \sum_{r=1}^{n}\sum_{s=1}^{m}\frac{\partial J}{\partial Y_{rs}}W_{sj}\delta_{ir} \\
&= \sum_{s=1}^{m}\frac{\partial J}{\partial Y_{is}}W_{sj}
\end{aligned}$$

So

$$\frac{\partial J}{\partial X} = \frac{\partial J}{\partial Y}W$$

C Softmax

Consider an input vector of scores s is $d \times 1$ and output vector y also $d \times 1$, where y encodes a probability distribution over d classes. Then the ith entry of the output is given by

$$\begin{aligned}
d(ab^{-1}) &= (da)\,b^{-1} + ad\left(b^{-1}\right) = (da)\,b^{-1} - ab^{-2}d(b) \\
&= \frac{bda - adb}{b^2}
\end{aligned}$$

$$y_i = \frac{\exp(s_i)}{\sum_{c=1}^{k}\exp(s_c)}.$$

Then

$$\begin{aligned}
\frac{\partial y_i}{\partial s_j} &= \frac{\frac{\partial}{\partial s_j}\left(\exp(s_i)\right)}{\sum_{c=1}^{k}\exp(s_c)} - \frac{\exp(s_i)\,\frac{\partial}{\partial s_j}\left(\sum_{c=1}^{k}\exp(s_c)\right)}{\left[\sum_{c=1}^{k}\exp(s_c)\right]^2} \\
&= \frac{\exp(s_i)\,\delta_{ij}}{\sum_{c=1}^{k}\exp(s_c)} - \frac{\exp(s_i)\exp(s_j)}{\left[\sum_{c=1}^{k}\exp(s_c)\right]^2} \\
&= \sigma(s_i)\delta_{ij} - \sigma(s_i)\sigma(s_j) \\
&= \sigma(s_i)\left(\delta_{ij} - \sigma(s_j)\right)
\end{aligned}$$

Now there is also some function $f : \mathbf{R}^d \to \mathbf{R}$, and let's write $J = f(\sigma(s))$. Our goal is to find the partial derivative of J with respect to each element of s, namely $\partial J/\partial s_j$. Suppose we have already computed all partial derivatives of J with respect to the intermediate vector $y = \sigma(s)$, namely $\frac{\partial J}{\partial y_i}$ for $i = 1, \ldots, d$. Then by the chain rule, we have

$$\begin{aligned}
\frac{\partial J}{\partial s_j} &= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} \frac{\partial y_r}{\partial s_j} \\
&= \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} \sigma(s_r) \left(\delta_{rj} - \sigma(s_j)\right) \\
&= \frac{\partial J}{\partial y_j} \sigma(s_j) - \sum_{r=1}^{m} \frac{\partial J}{\partial y_r} \sigma(s_r)\sigma(s_j)
\end{aligned}$$

so

$$\frac{\partial J}{\partial s} = \left(\frac{\partial J}{\partial y} - \left[\left(\frac{\partial J}{\partial y} \right)^T \sigma(s) \right] 1 \right) * \sigma(s)$$

Now suppose we are using a minibatch, in which case we have