Backpropagation

TA: Yi Wen

April 17, 2020
CS231n Discussion Section

Slides credits: Barak Oshri, Vincent Chen, Nish Khandwala, Yi Wen

Agenda

- Motivation
- Backprop Tips & Tricks
- Matrix calculus primer

Agenda

- **Motivation**
- Backprop Tips & Tricks
- Matrix calculus primer

Motivation

Recall: Optimization objective is minimize loss

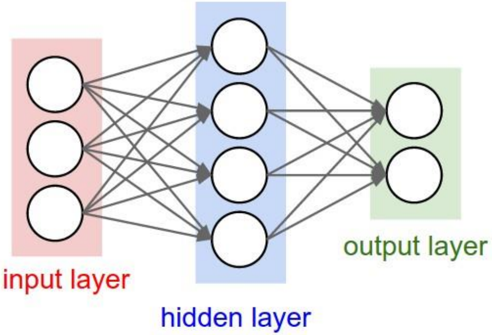
$$Loss = f(x, y; \theta)$$

Motivation

Recall: Optimization objective is minimize loss

$$Loss = f(x, y; \theta)$$

Goal: how should we tweak the parameters to decrease the loss?



Agenda

- Motivation
- **Backprop Tips & Tricks**
- Matrix calculus primer

A Simple Example

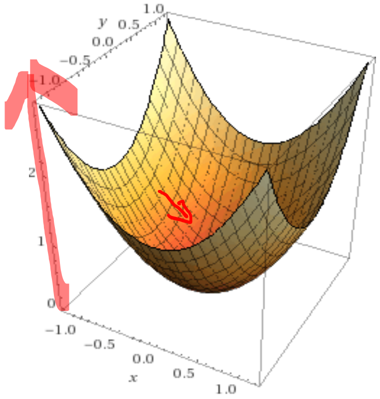
Loss $$Loss = f(\overbrace{x, y}^{\text{dataset}}; \theta)$$

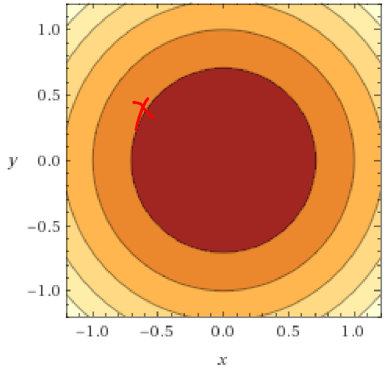
Goal: Tweak the parameters to minimize loss

=> **minimize a multivariable function** in parameter space

A Simple Example

=> minimize a multivariable function





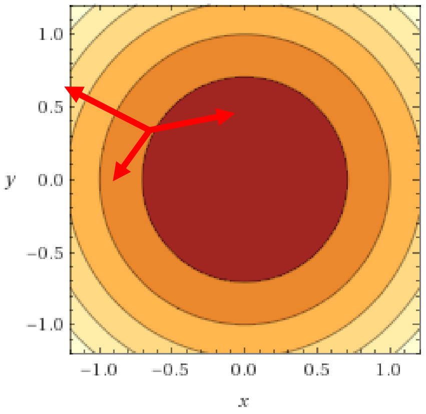
$$f(x, y) = x^2 + y^2$$

Plotted on WolframAlpha

Approach #1: Random Search

Intuition: the *step* we take in the domain of function

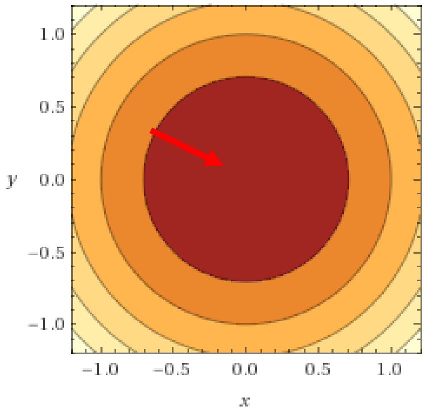
$$f(x, y) = x^2 + y^2$$



Approach #2: Numerical Gradient

Intuition: rate of change of a function with respect to a variable surrounding a small region

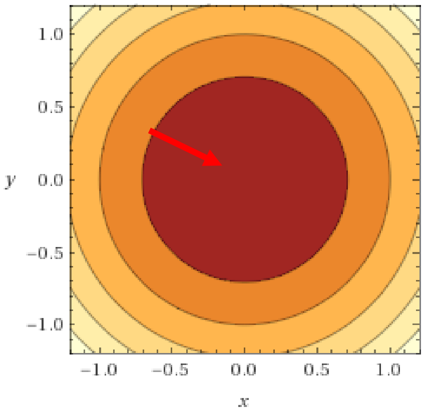
$$f(x, y) = x^2 + y^2$$



Approach #2: Numerical Gradient

Intuition: rate of change of a function with respect to a variable surrounding a small region

$$f(x, y) = x^2 + y^2$$



Finite Differences:

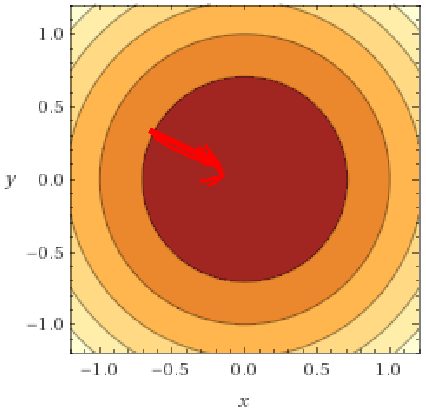
$$\frac{f(x+h, y) - f(x, y)}{h}$$

Approach #3: Analytical Gradient

Recall: partial derivative by limit definition

$$\frac{\partial f}{\partial x}(x, y) = \lim_{h \to 0} \frac{f(x + h, y) - f(x, y)}{h}$$

$f(x, y) = x^2 + y^2$



$\frac{\partial f}{\partial x} = 2x$

$\frac{\partial f}{\partial y} = 2y$

$-\begin{bmatrix} 2x \\ 2y \end{bmatrix}$

Approach #3: Analytical Gradient

Recall: chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}.$

Approach #3: Analytical Gradient

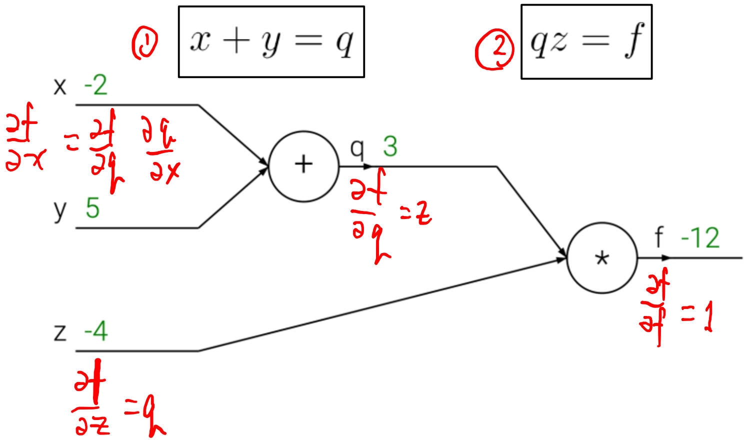
Recall: chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}.$

E.g. $f(x, y, z) = (x + y)z$

$$\frac{\partial f}{\partial x} =$$

$$\frac{\partial f}{\partial y} =$$

$$\frac{\partial f}{\partial z} =$$

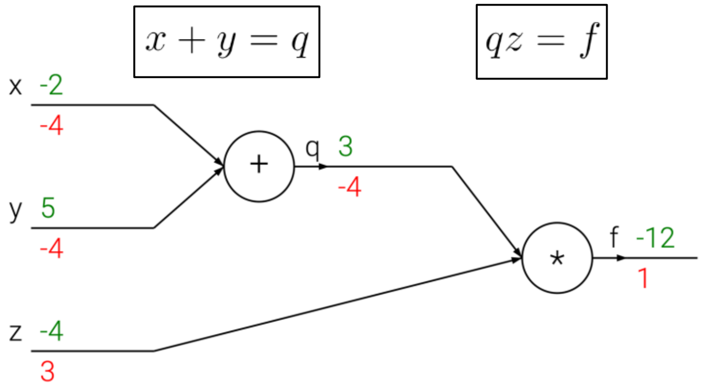


Approach #3: Analytical Gradient

Recall: chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$.

E.g. $f(x, y, z) = (x + y)z$

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x} = z \cdot 1 = z = -4$$
$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y} = z \cdot 1 = z = -4$$
$$\frac{\partial f}{\partial z} = q = x + y = -2 + 5 = 3$$



Approach #3: Analytical Gradient

Recall: chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}.$

Intuition: upstream gradient values propagate backwards -- we can reuse them!

Gradient

$$f : R^n \rightarrow R$$

$$\nabla f : R^n \rightarrow R^n$$

$$\mathbf{x} = [x_1, \cdots, x_n]^T \in R^n$$

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

"direction and rate of fastest increase"

Numerical Gradient vs Analytical Gradient

What about Autograd?

- Deep learning frameworks can automatically perform backprop!

- Problems might surface related to underlying gradients when debugging your models

"Yes You Should Understand Backprop"

https://medium.com/@karpathy/yes-you-should-understand-backprop-e2f06eab496b

Problem Statement: Backpropagation

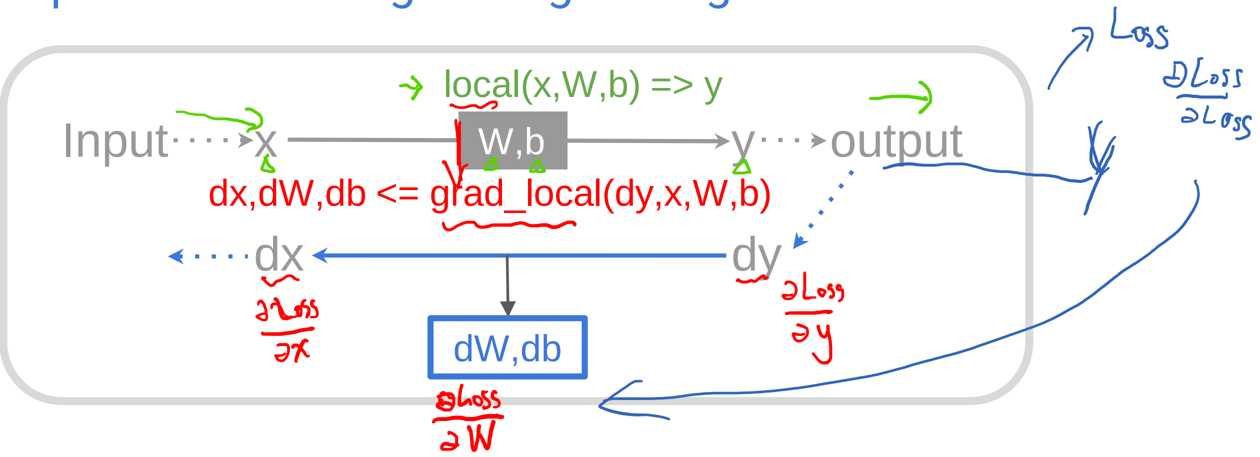
$$Loss = \frac{1}{N}\sum_i L_i(f(x_i,\ \theta\ ), y_i)$$

Given a function ***f*** with respect to inputs ***x***, labels ***y***, and parameters ***θ***
compute the gradient of ***Loss*** with respect to ***θ***

Problem Statement: Backpropagation

An algorithm for computing the gradient of a **compound** function as a series of **local, intermediate gradients**:

1. Identify intermediate functions (forward prop)
2. Compute local gradients (chain rule)
3. Combine with upstream error signal to get full gradient



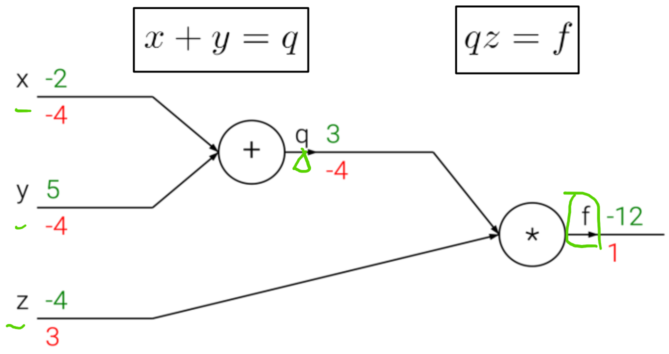
Modularity: Previous Example

Compound function $$f(x, y, z) = (x + y)z$$

Intermediate Variables
(forward propagation)

$$q = x + y$$

$$f = qz$$



Modularity: 2-Layer Neural Network

Compound function

$$Loss = L\Big(\sigma(xW_1 + b_1)W_2 + b_2, y\Big)$$

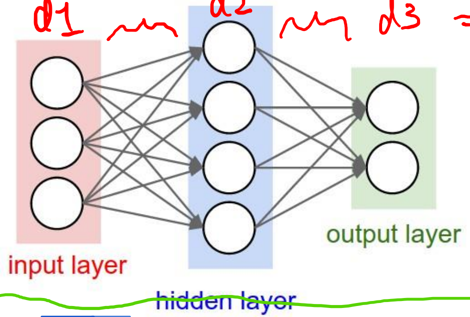
Intermediate Variables

(forward propagation)

classifier

d1 d2 d3 => # classes

x



$$\begin{tikzcd}
& \boxed{W_1 \mid b_1} \arrow[d] & & \boxed{W_2 \mid b_2} \arrow[d] & & \\
x \arrow[r] & {} \arrow[r] & h_1 \arrow[r] & z_1 \arrow[r] & z_2 \arrow[r] & Loss \\
y \arrow[rrrrru] & & & & &
\end{tikzcd}$$

$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1W_2 + b_2$$

$Loss$ => Squared Euclidean Distance between z_2 and y

Intermediate **Variables**

(forward propagation)

coding , Batch size = N

$$h_1 = \underbrace{x}_{N\times d1}\underbrace{W_1}_{\boxed{d1\times d2}} + b_1$$

N×d2

$$z_1 = \sigma(h_1)$$

$$z_2 = \underbrace{z_1}_{N\times d2}\underbrace{W_2}_{d2\times d3} + b_2$$

N×d3

only one image

$\rightarrow$? f(x;W,b) = $\underbrace{W}_{\boxed{d2\times d1}}\underbrace{x}_{[\,]_{d_1\times 1}}$ + b ?

$\Rightarrow [\,]_{d_2\times 1}$

(↑lecture note) Input one feature **vector**

(←here) Input a batch of data (**matrix**)

Intermediate **Variables**

(forward propagation)

1. intermediate functions
2. local gradients
3. full gradients

Intermediate **Gradients**

(backward propagation)

$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1 W_2 + b_2$$

$$Loss = \frac{1}{N}||z_2 - y||_F^2$$

$$||A||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

$$\frac{\partial h_1}{\partial W_1}, \frac{\partial h_1}{\partial b_1} \qquad \frac{\partial h_1}{\partial x} \; ? \; ? \; ? \quad W_1$$

$$\frac{\partial z_1}{\partial h_1} \; ? \; ? \quad z_1 ? (1 - z_1)$$

not scalar

$$\boxed{\frac{\partial z_2}{\partial W_2}, \frac{\partial z_2}{\partial b_2} \qquad \frac{\partial z_2}{\partial z_1} \; ? \; ?} \; ? \quad W_2$$

$$\frac{\partial Loss}{\partial z_2} = \frac{2}{N}(z_2 - y)$$

Agenda

- Motivation
- Backprop Tips & Tricks
- **Matrix calculus primer**

Previous : $\frac{\partial \text{ Scalar1}}{\partial \text{ Scalar2}}$

Derivative w.r.t. Vector

Scalar-by-Vector

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

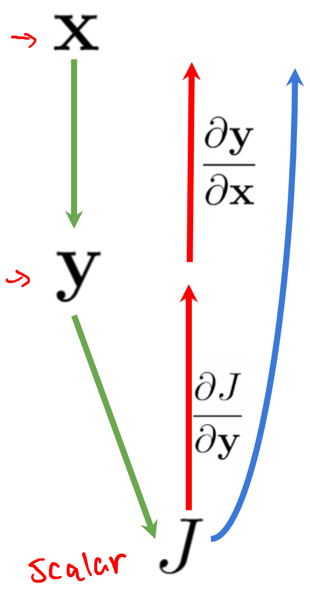
shape change   col $[\;]_{n\times 1}$ $[\;\;\text{row}\;\;]_{1\times n}$

Vector-by-Vector

$$\frac{\partial \mathbf{y}_m}{\partial \mathbf{x}_n} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}_{m\times n}$$

Derivative w.r.t. Vector: Chain Rule

1. intermediate functions
2. local gradients
3. full gradients



$$\frac{\partial J}{\partial \mathbf{x}} = ?$$

$$\frac{\partial J}{\partial x_j} = \sum_i \frac{\partial J}{\partial y_i} \frac{\partial y_i}{\partial x_j}$$

$$= \sum_i \left(\frac{\partial J}{\partial \mathbf{y}}\right)_{1i} \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)_{ij}$$

$$\boxed{\underbrace{\frac{\partial J}{\partial \mathbf{x}}}_{1\times n} = \underbrace{\frac{\partial J}{\partial \mathbf{y}}}_{1\times m} \underbrace{\frac{\partial \mathbf{y}}{\partial \mathbf{x}}}_{m\times n}}$$

Derivative w.r.t. Vector: Takeaway

$$\mathbf{y} = A_{m\times n}\mathbf{x} \qquad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = A$$

$$\frac{\partial J}{\partial \mathbf{x}} = \frac{\partial J}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \frac{\partial J}{\partial \mathbf{y}} A$$

---

elementwise function

$$\mathbf{y} = \omega(\mathbf{x}) = \begin{bmatrix} \omega(x_1) \\ \vdots \\ \omega(x_n) \end{bmatrix}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \omega'(x_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \omega'(x_n) \end{bmatrix}$$

$$\frac{\partial J}{\partial \mathbf{x}} = \frac{\partial J}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \left[\frac{\partial J}{\partial y_1}\omega'(x_1), \cdots, \frac{\partial J}{\partial y_n}\omega'(x_n)\right] = \frac{\partial J}{\partial \mathbf{y}} \circ \omega'(\mathbf{x})^T$$

Derivative w.r.t. Matrix

Loss input/param

Scalar-by-Matrix

$$\underset{\text{matrix}}{\overset{\text{scalar}}{\frac{\partial y}{\partial A_{m\times n}}}} = \begin{bmatrix} \frac{\partial y}{\partial A_{11}} & \frac{\partial y}{\partial A_{12}} & \cdots & \frac{\partial y}{\partial A_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial A_{m1}} & \frac{\partial y}{\partial A_{m2}} & \cdots & \frac{\partial y}{\partial A_{mn}} \end{bmatrix}_{m\times n}$$

e.g. $y = \sum_{i,j} A_{ij} = \boxed{A_{11}} + A_{12} + \cdots + A_{1n}$

$+ A_{21} + \cdots$

$+ \cdots$

$+ A_{m1} + \cdots + A_{mn}$

?

$\frac{\partial y}{\partial A_{ij}} = 1$

$$\frac{\partial y}{\partial A_{11}} = \frac{\partial (A_{11} + \cdots)}{\partial A_{11}} = 1$$

1D 2D

Vertor-by-Matrix $\rightarrow$ 3D

Derivative w.r.t. Matrix: Dimension Balancing

When you take **scalar-by-matrix** gradients

↓

The gradient has **shape of denominator**

- Dimension balancing is the “cheap” but **efficient** approach to gradient calculations in most practical settings

Derivative w.r.t. Matrix: Takeaway

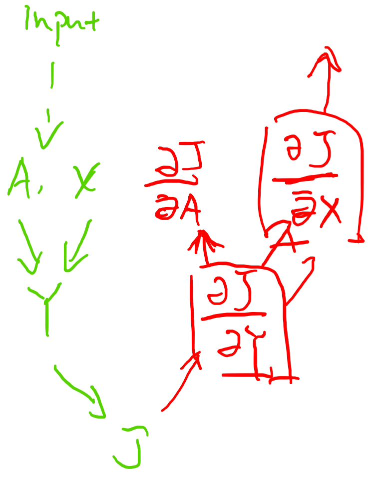
$$Y_{m\times l} = A_{m\times n} X_{n\times l}$$

scalar

known $\frac{\partial J}{\partial Y}$

$$\underbrace{\frac{\partial J}{\partial X}}_{n\times l} = \underbrace{A^T}_{n\times m} \underbrace{\frac{\partial J}{\partial Y}}_{m\times l}$$

$$\underbrace{\frac{\partial J}{\partial A}}_{m\times n} = \underbrace{\frac{\partial J}{\partial Y}}_{m\times l} \underbrace{X^T}_{l\times n}$$



---

elementwise function : activation function

$$Y_{m\times n} = \omega(X_{m\times n})$$

elementwise mult

$$\underbrace{\frac{\partial J}{\partial X}}_{m\times n} = \underbrace{\frac{\partial J}{\partial Y}}_{m\times n} \odot \underbrace{\omega'(X)}_{m\times n}$$

1. intermediate functions
2. local gradients
3. full gradients

Intermediate **Variables**

(forward propagation)

Batch size : N

$N \times d_1$

$$h_1 = xW_1 + b_1$$

$N \times d_2$

$$z_1 = \sigma(h_1)$$

$z_2 = z_1 W_2$, $\frac{\partial Loss}{\partial W_2} = z_1^T \frac{\partial Loss}{\partial z_2}$

$N \times d_3$

$$z_2 = z_1 W_2 + b_2$$

$$Loss = \frac{1}{N} \|z_2 - y\|_F^2$$

Intermediate **Gradients**

(backward propagation)

$$\boxed{\frac{\partial Loss}{\partial W_1}, \frac{\partial Loss}{\partial b_1}} \quad \frac{\partial Loss}{\partial x}$$

$$\frac{\partial Loss}{\partial h_1}$$

$$\boxed{\frac{\partial Loss}{\partial W_2}, \frac{\partial Loss}{\partial b_2}} \quad \frac{\partial Loss}{\partial z_1} = \frac{\partial Loss}{\partial z_2} \frac{\partial z_2}{\partial z_1}$$

$$\frac{\partial Loss}{\partial z_2}$$

Backprop Menu for Success

$$\begin{tikzcd}[row sep=small] & & & u \arrow[dr] & & \\ \text{input} \cdots \arrow[r] & x \arrow[urr] \arrow[drr] & & & y \arrow[r] & {} \\ & & & v \arrow[ur] & & \end{tikzcd}$$

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u}\frac{\partial u}{\partial x} \boxed{+} \frac{\partial y}{\partial v}\frac{\partial v}{\partial x}$$

1. Write down variable graph
2. Keep track of error signals
3. Compute derivative of loss function
4. Enforce shape rule on error signals, especially when deriving over a linear transformation

Vector-by-vector

$$\boxed{\mathbf{y} = A_{m \times n}\mathbf{x}}$$

$$y_i = \sum_{j=1}^{n} A_{ij} x_j$$

$$\frac{\partial y_i}{\partial x_j} = A_{ij}$$

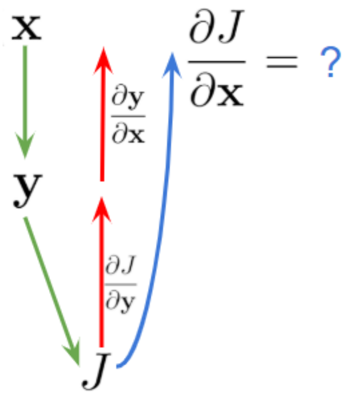
$$\boxed{\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = A}$$

Vector-by-vector

$$\boxed{\begin{aligned}\mathbf{y} &= A_{m\times n}\mathbf{x}\\ \lambda^T &= \frac{\partial J}{\partial \mathbf{y}}\end{aligned}}$$

$$y_i = \sum_{j=1}^{n} A_{ij}x_j \qquad \frac{\partial y_i}{\partial x_j} = A_{ij}$$

$$\lambda_i = \frac{\partial J}{\partial y_i}$$



$$\frac{\partial J}{\partial x_j} = \sum_i \frac{\partial J}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \sum_{i=1}^{m} \lambda_i A_{ij}$$

$$\boxed{\frac{\partial J}{\partial \mathbf{x}} = \lambda^T A = \frac{\partial J}{\partial \mathbf{y}} A}$$

Vector-by-vector

$$\boxed{\mathbf{y} = \omega(\mathbf{x})}$$

$$y_i = \omega(x_i)$$

$$\frac{\partial y_i}{\partial x_i} = \omega'(x_i)$$

$$\frac{\partial y_i}{\partial x_j} = 0 \qquad (i \neq j)$$

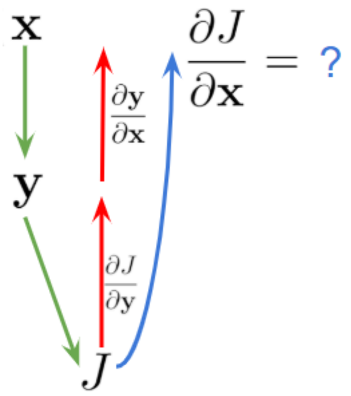
$$\boxed{\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \omega'(x_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \omega'(x_n) \end{bmatrix}}$$

Vector-by-vector

$$\boxed{\begin{aligned}\mathbf{y} &= \omega(\mathbf{x})\\ \lambda^T &= \frac{\partial J}{\partial \mathbf{y}}\end{aligned}}$$

$$y_i = \omega(x_i) \qquad \frac{\partial y_i}{\partial x_i} = \omega'(x_i)$$

$$\lambda_i = \frac{\partial J}{\partial y_i} \qquad \frac{\partial y_i}{\partial x_j} = 0 \qquad (i \neq j)$$



$$\frac{\partial J}{\partial x_j} = \sum_i \frac{\partial J}{\partial y_i}\frac{\partial y_i}{\partial x_j} = \frac{\partial J}{\partial y_j}\frac{\partial y_j}{\partial x_j} = \lambda_j \omega'(x_j)$$

$$\boxed{\frac{\partial J}{\partial \mathbf{x}} = \lambda^T \circ \omega'(\mathbf{x})^T = \frac{\partial J}{\partial \mathbf{y}} \circ \omega'(\mathbf{x})^T}$$

$$\mathbf{y} = A_{m\times n}\mathbf{x} \qquad \frac{\partial J}{\partial \mathbf{x}} = \frac{\partial J}{\partial \mathbf{y}} A$$

$$\frac{\partial J}{\partial \mathbf{x}}^T = A^T \frac{\partial J}{\partial \mathbf{y}}^T$$

Matrix multiplication [Backprop]

$$\boxed{\begin{aligned} Y_{m\times l} &= A_{m\times n} X_{n\times l} \\ \frac{\partial J}{\partial Y} &= \Lambda_{m\times l} \end{aligned}}$$

$$Y_{ij} = \sum_{k=1}^{n} A_{ik} X_{kj}$$

$$\Lambda_{ij} = \frac{\partial J}{\partial Y_{ij}}$$

$$\frac{\partial J}{\partial X_{ij}} = \sum_{i',j'} \frac{\partial J}{\partial Y_{i'j'}} \frac{\partial Y_{i'j'}}{\partial X_{ij}} = \sum_{i'=1}^{m} \frac{\partial J}{\partial Y_{i'j}} \frac{\partial Y_{i'j}}{\partial X_{ij}}$$

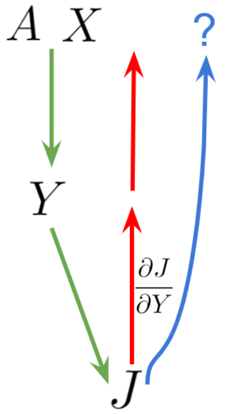
$$= \sum_{i'=1}^{m} A_{i'i} \Lambda_{i'j} = \sum_{k=1}^{m} A_{ki} \Lambda_{kj}$$

$$\boxed{\frac{\partial J}{\partial X} = A^T \Lambda = A^T \frac{\partial J}{\partial Y}}$$

$$\frac{\partial J}{\partial A_{ij}} = \sum_{i',j'} \frac{\partial J}{\partial Y_{i'j'}} \frac{\partial Y_{i'j'}}{\partial A_{ij}} = \sum_{j'=1}^{l} \frac{\partial J}{\partial Y_{ij'}} \frac{\partial Y_{ij'}}{\partial A_{ij}}$$

$$= \sum_{j'=1}^{l} \Lambda_{ij'} X_{jj'} = \sum_{k=1}^{l} \Lambda_{ik} X_{jk}$$

$$\boxed{\frac{\partial J}{\partial A} = \Lambda X^T = \frac{\partial J}{\partial Y} X^T}$$



$$\mathbf{y} = \omega(\mathbf{x}) \qquad \frac{\partial J}{\partial \mathbf{x}} = \frac{\partial J}{\partial \mathbf{y}} \circ \omega'(\mathbf{x})^T$$

$$\frac{\partial J}{\partial \mathbf{x}}^T = \frac{\partial J}{\partial \mathbf{y}}^T \circ \omega'(\mathbf{x})$$

Elementwise function [Backprop]

$$\boxed{\begin{aligned} Y_{m\times n} &= \omega(X_{m\times n}) \\ \frac{\partial J}{\partial Y} &= \Lambda_{m\times n} \end{aligned}}$$

$$Y_{ij} = \omega(X_{ij})$$

$$\Lambda_{ij} = \frac{\partial J}{\partial Y_{ij}}$$

$$\frac{\partial J}{\partial X_{ij}} = \sum_{i',j'} \frac{\partial J}{\partial Y_{i'j'}} \frac{\partial Y_{i'j'}}{\partial X_{ij}}$$

$$= \frac{\partial J}{\partial Y_{ij}} \frac{\partial Y_{ij}}{\partial X_{ij}} = \Lambda_{ij}\omega'(X_{ij})$$

$$\boxed{\frac{\partial J}{\partial X} = \Lambda \circ \omega'(X) = \frac{\partial J}{\partial Y} \circ \omega'(X)}$$