# Agenda

- Motivation
- Backprop Tips & Tricks
- Matrix calculus primer

# Agenda

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

# Motivation

Recall: Optimization objective is to minimize loss

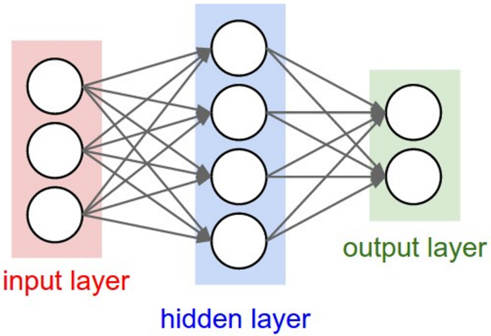
$$Loss = f(x, y; \theta)$$

# Motivation

Recall: Optimization objective is to 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

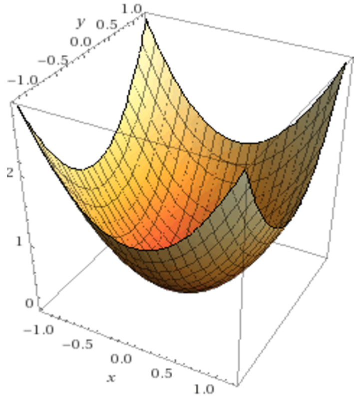
Loss $$Loss = f(x, y; \theta)$$

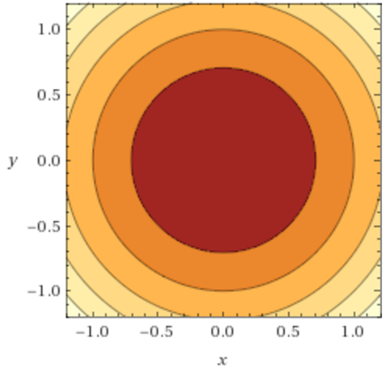
Goal: Tweak the parameters to minimize loss

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

# A Simple Example

**=> minimize a multivariable function**



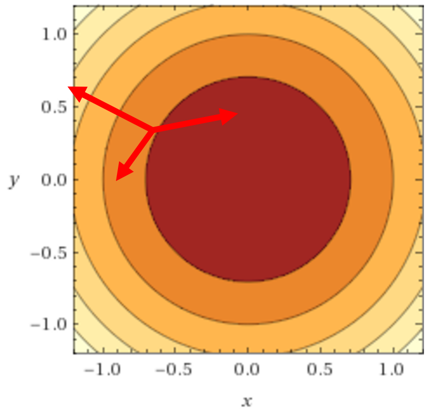

$$f(x, y) = x^2 + y^2$$

Plotted on WolframAlpha

# Approach #1: Random Search

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

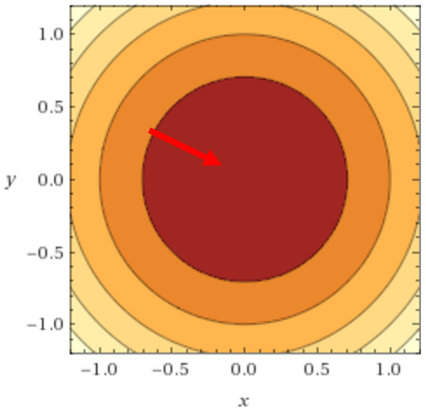
$$f(x, y) = x^2 + y^2$$



# Approach #2: Numerical Gradient

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

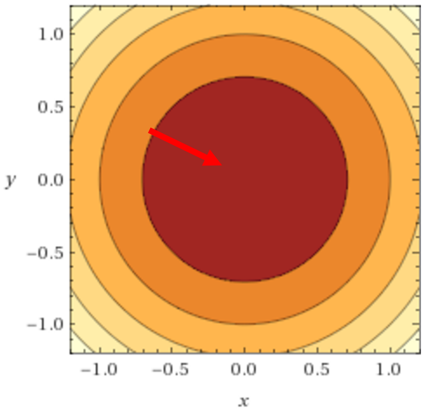
$$f(x, y) = x^2 + y^2$$



# 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:**

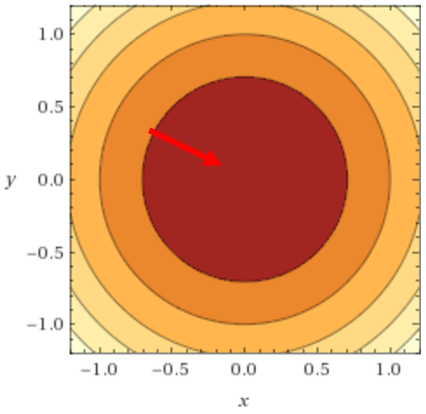
$$\frac{f(x+h, y) - f(x, y)}{h}$$

# Approach #3: Analytical Gradient

**Recall:** **partial derivative by limit definition**

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

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



# Approach #3: Analytical Gradient

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

# Approach #3: Analytical Gradient

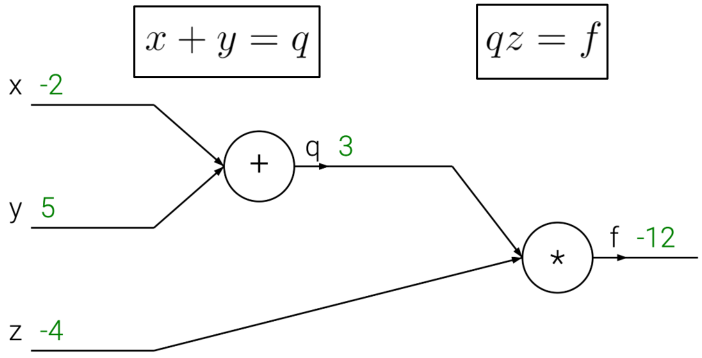
**Recall:** chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}.$

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

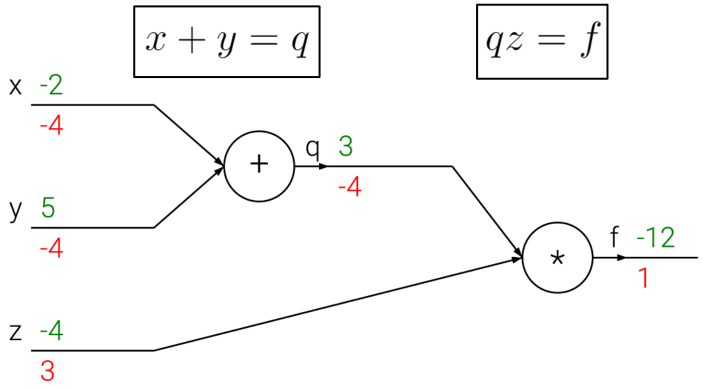
**Recall:** chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}.$

**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

$$f : R^n \to R$$

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

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

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

**"direction and rate of fastest increase"**

**Numerical Gradient vs Analytical Gradient**

# What about Autograd?

Q: “Why do we have to write the backward pass when frameworks in the real world, such as TensorFlow, compute them for you automatically?”

A: Problems might surface related to underlying gradients when debugging your model (e.g. vanishing or exploding gradients)

“Yes You Should Understand Backprop”

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

# Problem Statement: Backpropagation

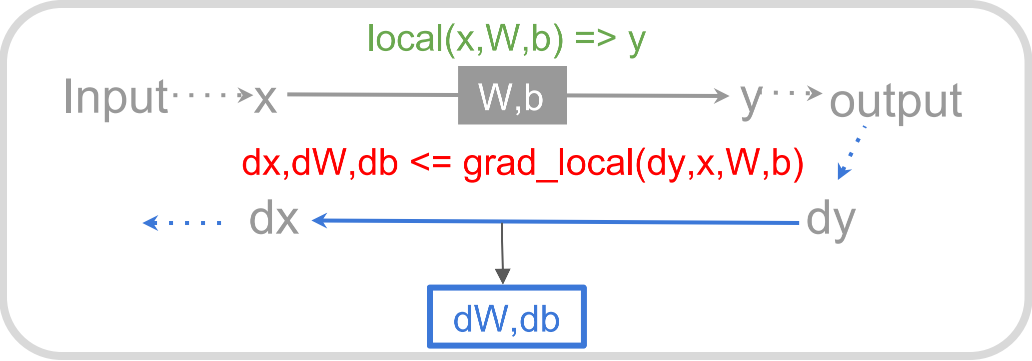
$$Loss = \frac{1}{N} \sum_i L_i(f(x_i,\ \theta\ ), y_i)$$

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



# Modularity: Previous Example

**Compound function**

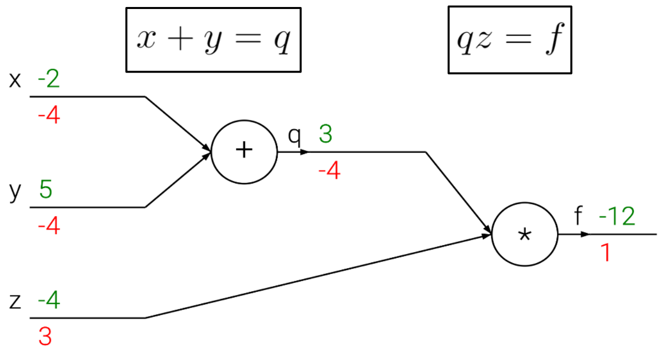
$$f(x, y, z) = (x + y)z$$

**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**

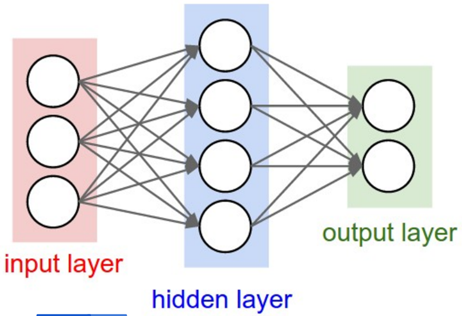
**(forward propagation)**

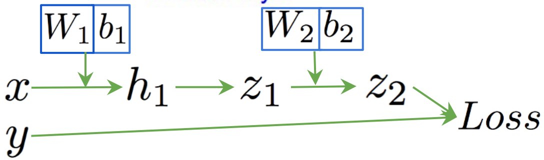
$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1 W_2 + b_2$$

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



## Intermediate Variables

(forward propagation)

$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1W_2 + b_2$$

? f(x;W,b) = Wx + b ?

(↑ 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)

$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1 W_2 + b_2$$

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

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

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

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

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

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

# Agenda

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

# Derivative w.r.t. Vector

**Scalar-by-Vector**

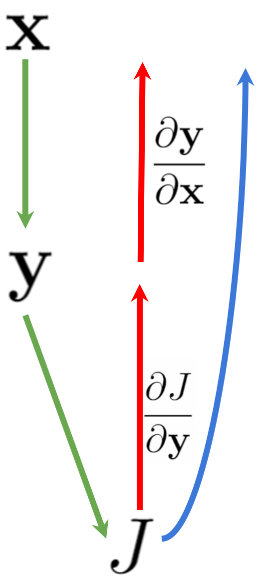
$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} \cdots \frac{\partial y}{\partial x_n} \end{bmatrix}$$

**Vector-by-Vector**

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

# Derivative w.r.t. Vector: Chain Rule

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



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

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

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

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

# Derivative w.r.t. Vector: Takeaway

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

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

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

---

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

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

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

# Derivative w.r.t. Matrix

**Scalar-by-Matrix**

$$\frac{\partial y}{\partial A} = \begin{bmatrix} \frac{\partial y}{\partial A_{11}} & \frac{\partial y}{\partial A_{12}} & \cdots & \frac{\partial y}{\partial A_{1n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial A_{m1}} & \frac{\partial y}{\partial A_{m2}} & \cdots & \frac{\partial y}{\partial A_{mn}} \end{bmatrix}$$

**Vector-by-Matrix**

?

# 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

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

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

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

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

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

**Intermediate Variables**

(forward propagation)

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

**Intermediate Gradients**

(backward propagation)

$$h_1 = xW_1 + b_1$$

$$z_1 = \sigma(h_1)$$

$$z_2 = z_1 W_2 + b_2$$

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

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

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

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

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

# Backprop Menu for Success

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

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

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

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

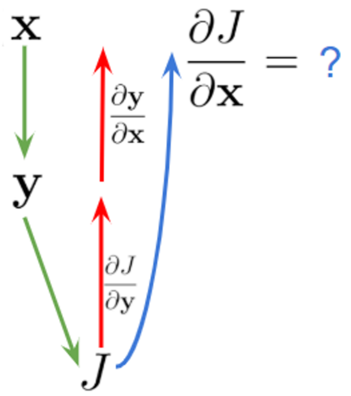
$$\boxed{\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = A}$$

# 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

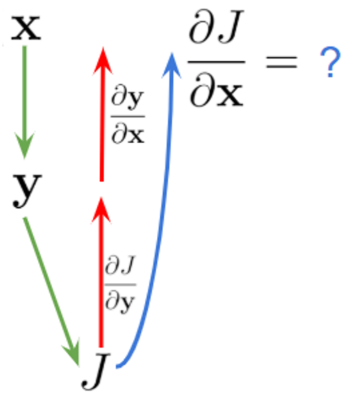
$$\boxed{\begin{aligned} \mathbf{y} &= \omega(\mathbf{x}) \\ \lambda^T &= \frac{\partial J}{\partial \mathbf{y}} \end{aligned}}$$

$$y_i = \omega(x_i)$$

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

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

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




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

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

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

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

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

# Matrix multiplication [Backprop]

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

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

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

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

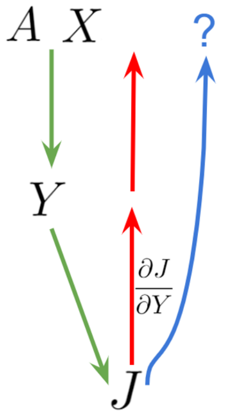
$$= \sum_{i'=1}^{m} A_{i'i} \Lambda_{i'j} = \sum_{k=1}^{m} A_{ki} \Lambda_{kj}$$

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

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

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

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



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

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

# Elementwise function [Backprop]

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

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

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

$$\begin{aligned} \frac{\partial J}{\partial X_{ij}} &= \sum_{i',j'} \frac{\partial J}{\partial Y_{i'j'}} \frac{\partial Y_{i'j'}}{\partial X_{ij}} \\ &= \frac{\partial J}{\partial Y_{ij}} \frac{\partial Y_{ij}}{\partial X_{ij}} = \Lambda_{ij}\omega'(X_{ij}) \end{aligned}$$

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