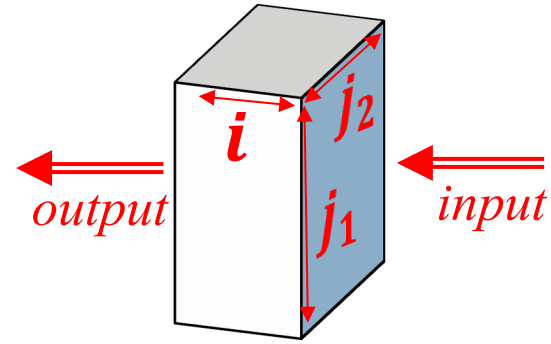


# Tensors

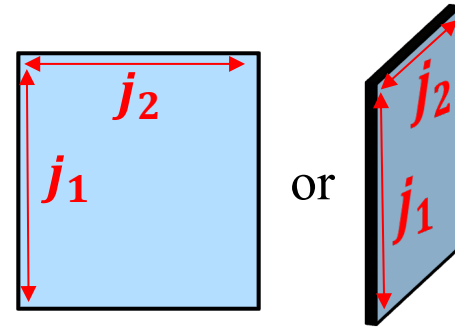
- Contravariant and covariant tensors
- Einstein notation
- Interpretation

# What is a Tensor?

- It is the generalization of a:
  - Matrix operator for  $>2$  dimensions



- Vector data to  $>1$  dimension



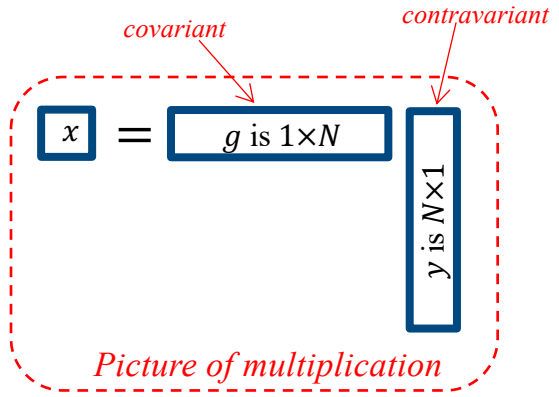
- Why do we need it?
  - Came out of differential geometry.
  - Was used by Albert Einstein to formulate the theory of General Relativity.
  - It's useful for many problems, including Deep NNs.

# Contravariant and Covariant Vectors

$$x = gy$$

- Contravariant vectors:
  - “Column Vectors” that represent data
  - Vectors that describe the position of something
  - $y^j$  for  $0 \leq j < N$  and  $x \in \mathfrak{R}$

- Covariant vector:
  - “Row vectors” that operate on data
  - Gradient vectors
  - $g_j$  for  $0 \leq j < N$



- Einstein notation

$$x = g_j y^j = \sum_{j=0}^{N-1} g_j y^j$$

- Leave out the sum to make notation cleaner
- Always sum over any two indices that appear twice

# Vector-Matrix Products as Tensors

$$x = Gy$$

- Rank 1 Contravariant vectors:

- $y^j$  for  $0 \leq j < N_y$
- $x^j$  for  $0 \leq j < N_x$

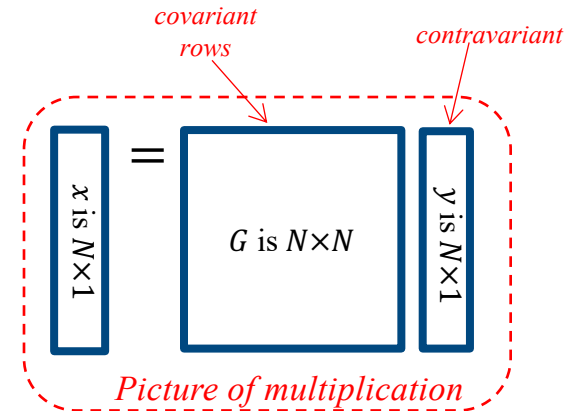
- Rank 2 tensor (i.e., matrix):

- $G^i_j$  for  $0 \leq i < N_x$  and  $0 \leq j < N_y$
- There is a gradient for each component  $x^i$

- Einstein notation

$$x^i = G^i_j y^j = \sum_{j=0}^{N_y-1} G^i_j y^j$$

- Leave out the sum to make notation cleaner
- Always sum over any two indices that appear twice



# Tensor Products

- Einstein notation

$$x^{i_1, i_2} = G^{i_1, i_2}_{j_1, j_2} y^{j_1, j_2}$$

*Sum over pairs  
of indices*

- 2-D Contravariant Tensors:

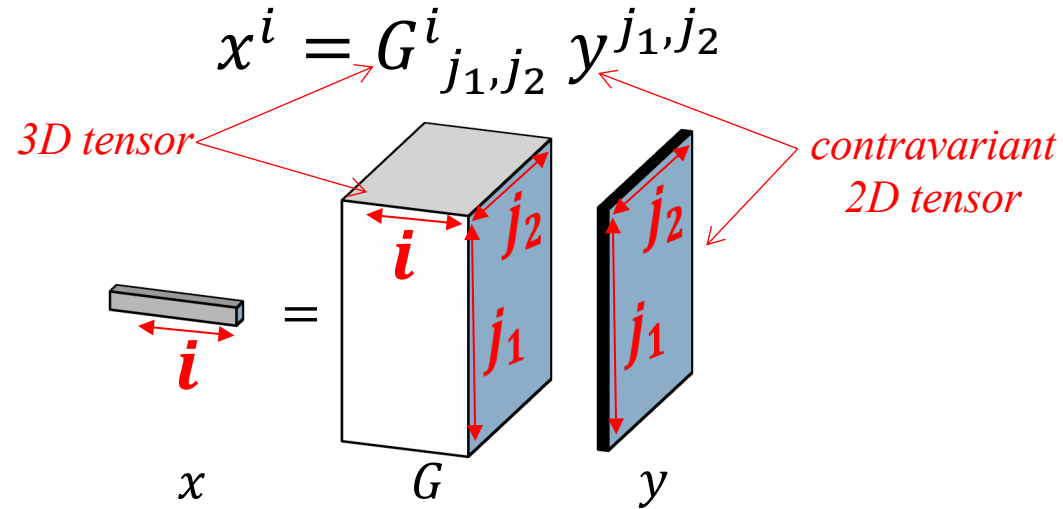
- $y^{j_1, j_2}$  for  $0 \leq j_1, j_2 < N_y$
- $x^{i_1, i_2}$  for  $0 \leq i_1, i_2 < N_x$

- 4-D Tensor:

- $G^{i_1, i_2}_{j_1, j_2}$  for  $0 \leq i_1, i_2 < N_x$  and  $0 \leq j_1, j_2 < N_y$
- $G$  is known as a tensor
  - 2D covariant input
  - 2D contravariant output

# Picture of a Tensor Product

- For example, if we have



- Tensor 3-D tensor  $G$  has  
Input: 2D image indexed by  $(j_1, j_2)$   
Output: 1D vector indexed by  $i$
- General idea:  $G \in \mathfrak{R}^{N_o \times N_i}$

# Some Useful Definitions

- Delta functions:

$$\delta^i_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

- So then we have that

$$G^k_j = G^k_i \delta^i_j$$

- Gradient w.r.t. vector

$$\nabla_y(Ay) = A$$

- In tensor notation, we have that

$$[\nabla_y(Ay)]^i_j = A^i_j$$

- Gradient w.r.t. matrix

$$\nabla_A(Ay) = ??$$

- First, it must have 1 output dimension and 2 input dimensions. So we have that

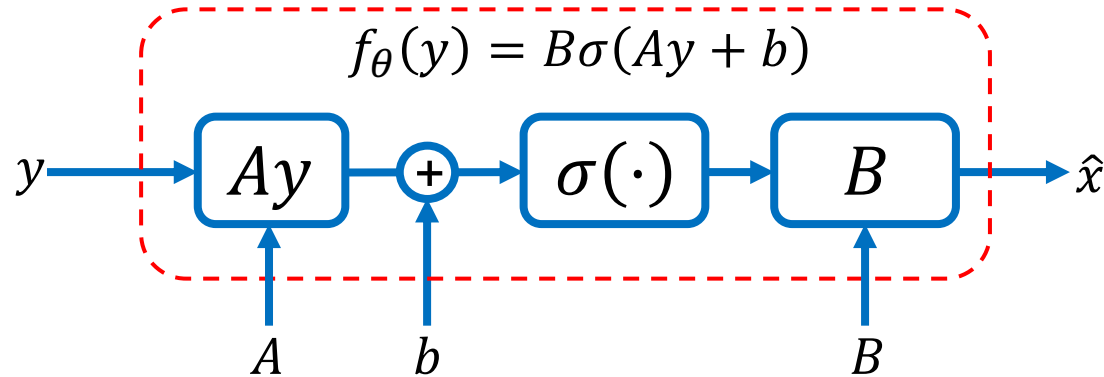
$$[\nabla_A(Ay)]^i_{j_1, j_2} = \delta^i_{j_1} y^{j_2}$$

# GD for Single Layer NN

- Structure of the Gradient
- Gradients for NN parameters
- Updates for NN parameters

# Gradient Direction for Single Layer NN

- Single layer NN:



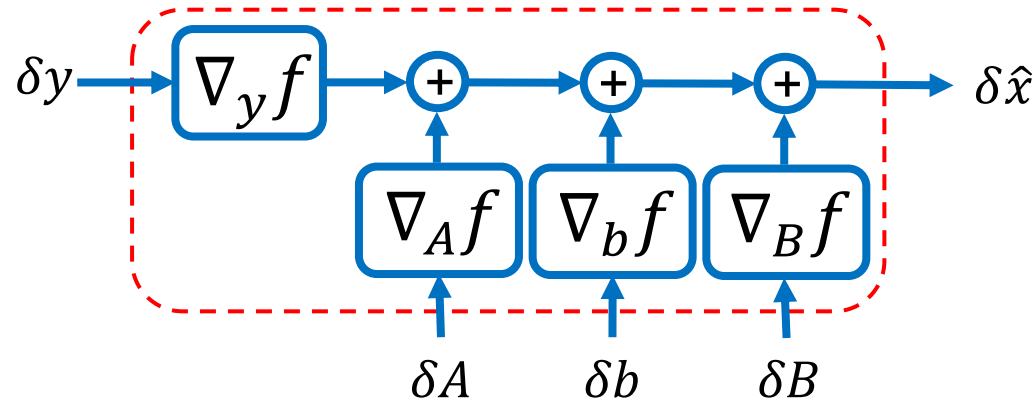
- We will need gradient w.r.t the parameters  $\theta = (A, b, B)$ :

$$\nabla_{\theta} f_{\theta}(y) = [\nabla_A f_{(A,b,B)}(y), \nabla_b f_{(A,b,B)}(y), \nabla_B f_{(A,b,B)}(y)]$$

- Later, we will also need:

$$\nabla_y f_{\theta}(y)$$

# Gradient Structure for Single Layer NN



- Single layer NN:

- Parameters are  $\theta = (A, b, B)$

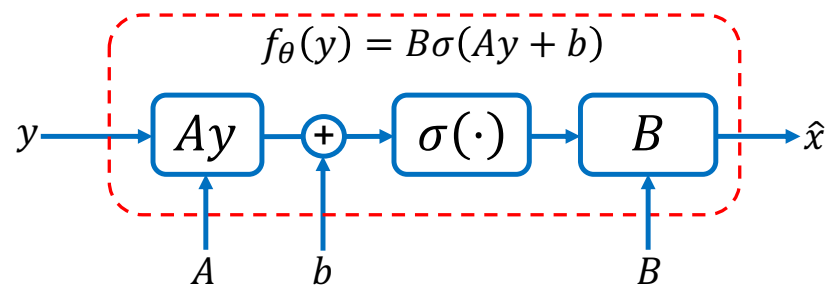
- We will need the parameter gradients:

$$\nabla_{\theta} f_{\theta}(y) = [\nabla_A f_{(A,b,B)}(y), \nabla_b f_{(A,b,B)}(y), \nabla_B f_{(A,b,B)}(y)]$$

- And the input gradient:

$$\nabla_y f_{\theta}(y)$$

# Gradient w.r.t. $y$



- For this case,

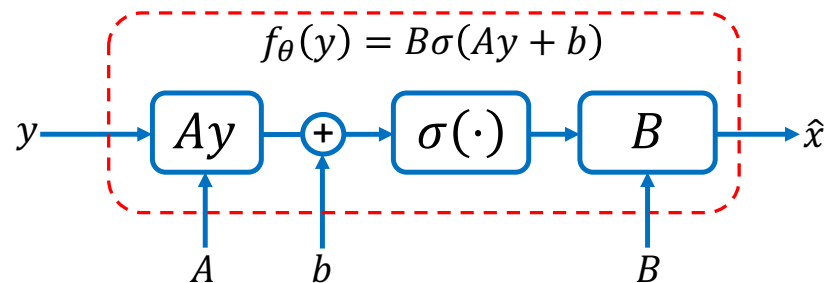
$$\begin{aligned}\nabla_y f &= \nabla_y f_{\theta}(y) = \nabla_y B[\sigma(Ay + b)] \\ &= B[\nabla\sigma(Ay + b)]A\end{aligned}$$

- Using Einstein notation

$$[\nabla_y f]^i_j = B^i_{i_1} [\nabla\sigma]^{i_1}_{i_2} A^{i_2}_j$$

*This is a matrix or  
2-D tensor*

# Gradient w.r.t. A



■ For this case,

$$\nabla_A f = B [\nabla \sigma(Ay + b)] \nabla_A(Ay)$$

– Using Einstein notation, since

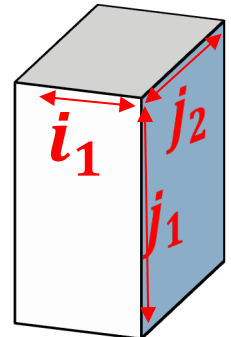
$$[\nabla_A(Ay)]^i_{j_1, j_2} = \delta^i_{j_1} y^{j_2}$$

*This is a tensor!*

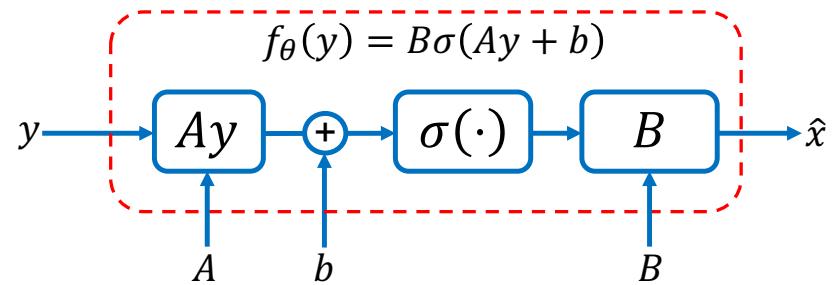
– Then

$$\begin{aligned} [\nabla_A f]^{i_1}_{j_1, j_2} &= B^{i_1}_{i_2} [\nabla \sigma]^{i_2}_{i_3} [\nabla_A(Ay)]^{i_3}_{j_1, j_2} \\ &= B^{i_1}_{i_2} [\nabla \sigma]^{i_2}_{i_3} \delta^{i_3}_{j_1} y^{j_2} \end{aligned}$$

$$[\nabla_A f]^{i_1}_{j_1, j_2} = B^{i_1}_{i_2} [\nabla \sigma]^{i_2}_{j_1} y^{j_2}$$



# Gradient w.r.t. $b$



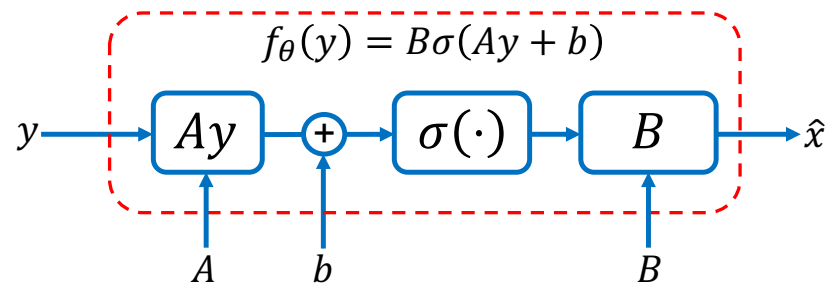
- For this case,

$$\begin{aligned}\nabla_b f &= B [\nabla\sigma(Ay + b)] \nabla_b(Ay + b) \\ &= B [\nabla\sigma]I = B \nabla\sigma\end{aligned}$$

- Using Einstein notation,

$$[\nabla_b f]^{i_1}_{j_1} = B^{i_1}_{i_2} [\nabla\sigma]^{i_2}_{j_1}$$

# Gradient w.r.t. $B$



- For this case,

$$\nabla_B f = \nabla_B (B\sigma)$$

*This is a tensor!*

- Using Einstein notation, since

$$[\nabla_B (B\sigma)]^{i_{j_1, j_2}} = \delta^{i_{j_1}} \sigma^{j_2}$$

- We have that

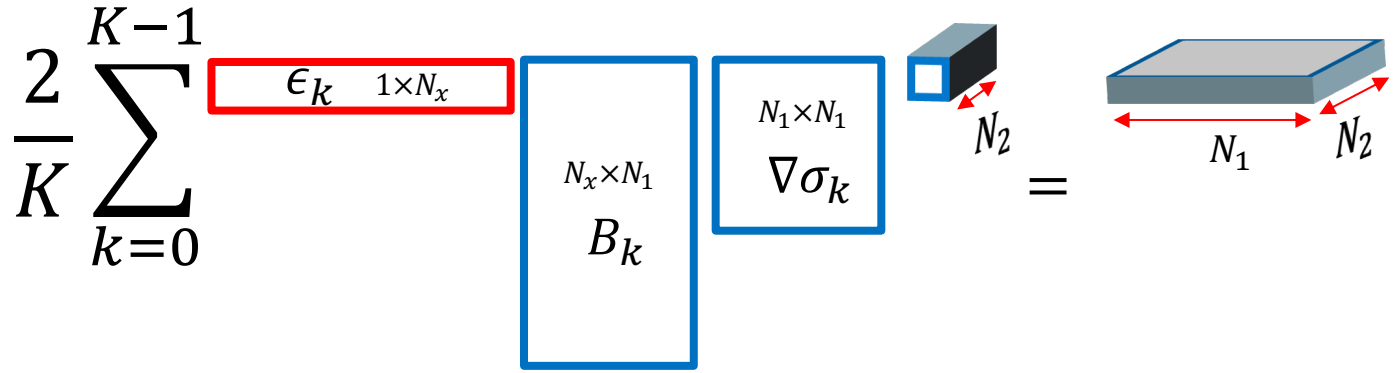
$$[\nabla_B f]^{i_1}_{j_1, j_2} = \delta^{i_1}_{j_1} \sigma^{j_2}$$

# Update Direction for A

Gradient step:  
 $A \leftarrow A + \alpha d^t$

where  $d$  is given by

$$d_{j_1, j_2} = -\nabla_A L(\theta) = \frac{2}{K} \sum_{k=0}^{K-1} \epsilon_{k, i_1} B^{i_1}_{i_2} [\nabla \sigma_k]^{i_2}_{j_1} y_k^{j_2}$$



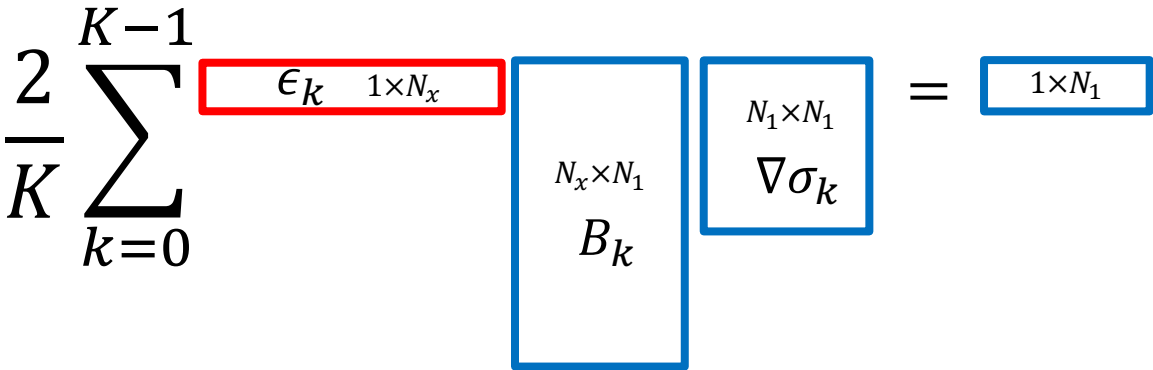
*For efficiency, computation goes this way!*

# Update Direction for $b$

Gradient step:  
 $b \leftarrow b + \alpha d^t$

where  $d$  is given by

$$d_{j_1} = -\nabla_b L(\theta) = \frac{2}{K} \sum_{k=0}^{K-1} \epsilon_{k,i_1} B^{i_1}_{i_2} [\nabla \sigma_k]^{i_2}_{j_1}$$



*For efficiency, computation goes this way!*

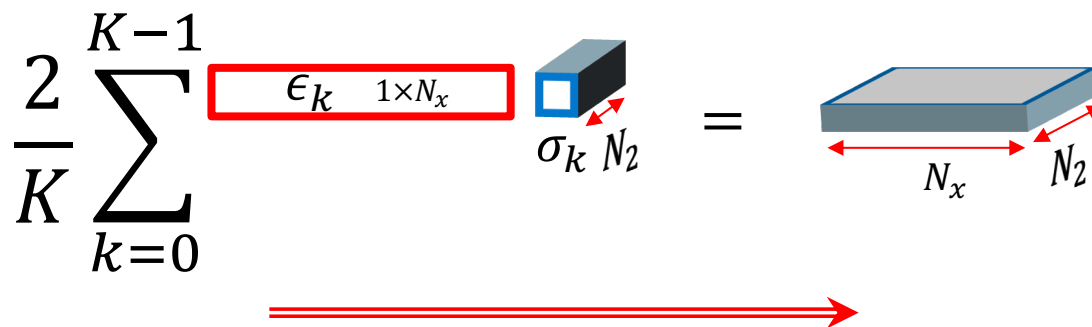
# Update Direction for $B$

Gradient step:

$$B \leftarrow B + \alpha d^t$$

where  $d$  is given by

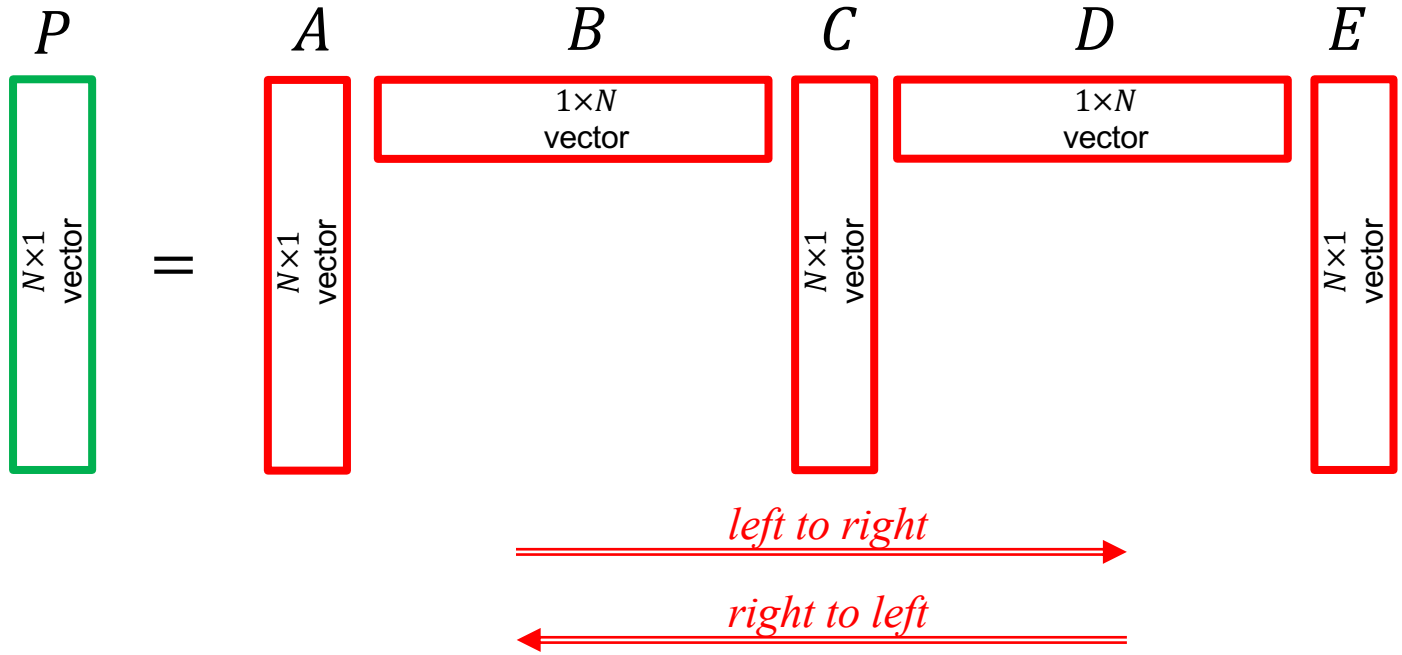
$$d_{j_1, j_2} = -\nabla_B L(\theta) = \frac{2}{K} \sum_{k=0}^{K-1} \epsilon_{k, j_1} \sigma_k^{j_2}$$

$$\frac{2}{K} \sum_{k=0}^{K-1} \left[ \epsilon_k \quad 1 \times N_x \right] \begin{matrix} \text{blue box} \\ \sigma_k \quad N_2 \end{matrix} = \begin{matrix} \text{gray box} \\ N_x \quad N_2 \end{matrix}$$


*For efficiency, computation goes this way!*

# Order of Operation Matters!

- Multiplication is associative  $\Rightarrow$  order doesn't matter
  - $P = (((A * B) * C) * D) * E = A * (B * (C * (D * E)))$
- But for computation, order matters!!
  - Left-to-Right: Dense matrix-vector products  $\Rightarrow 4N^2$  multiplies
  - Right-to-Left: Vector inner products  $\Rightarrow 4N$  multiplies



# Local and Global Minima

- Open and Closed Sets
- Convex Sets and Functions
- Properties of Convex Functions
- Local Minimum, Saddle Points, and Global Minima
- Optimization Theorems

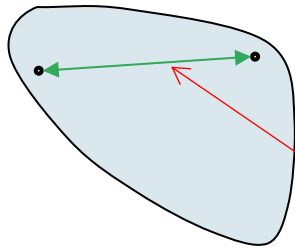
# Open and Closed Sets

- Define:
  - $\mathcal{A} \subset \mathfrak{R}^N$
  - Open ball of radius  $\epsilon$  is  $B(r, \epsilon) = \{r \in \mathfrak{R}^N : \|r - r_o\| < \epsilon\}$ .
- A set  $\mathcal{A}$  is open if
  - At every point, there is an open ball contained in  $\mathcal{A}$ .
  - $\forall r \in \mathcal{A}, \exists \epsilon > 0$  s.t.  $B(r, \epsilon) \subset \mathcal{A}$ .
- A set  $\mathcal{A}$  is closed if  $\mathcal{A}^c = \mathfrak{R}^N - \mathcal{A}$  is open.
- A set  $\mathcal{A}$  is compact if it is closed and bounded.
- Facts:
  - $\mathfrak{R}^N$  is both open and closed, but it is not compact.
  - If  $\mathcal{A}$  is compact, then every sequence in  $\mathcal{A}$  has a limit point in  $\mathcal{A}$ .

# Convexity Sets

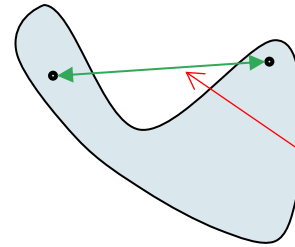
- A set  $\mathcal{A}$  is convex if

$$\forall \lambda \in (0,1), \forall s, r \in \mathcal{A}, \text{ then } \lambda r + (1 - \lambda)s \in \mathcal{A}$$



Line connecting points is always in set

Convex Set



Line connecting points is sometimes outside set

Nonconvex Set

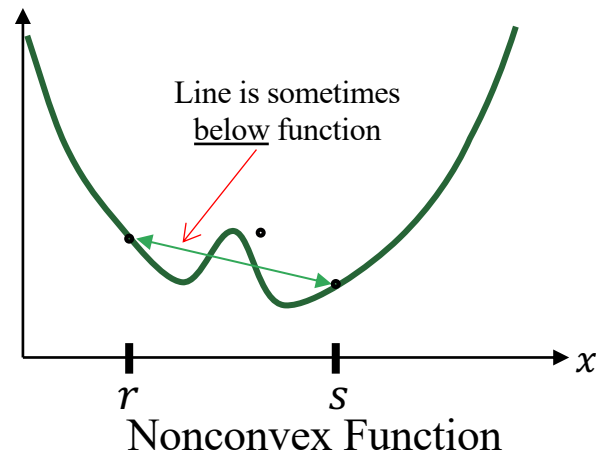
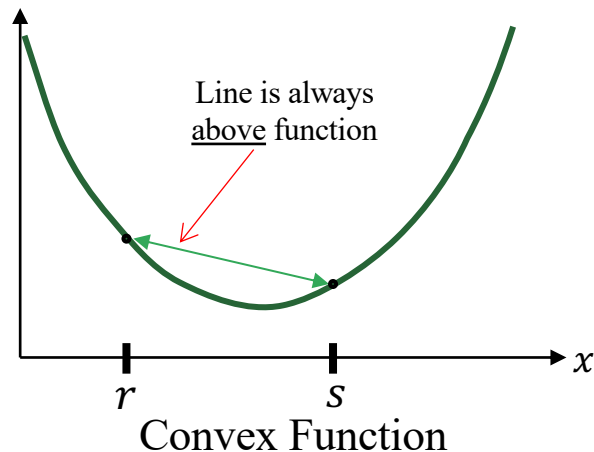
- Properties:

- The intersection of convex sets is convex

# Convexity Functions

- Let  $f: \mathcal{A} \rightarrow \mathfrak{R}$  where  $\mathcal{A}$  is a convex set. Then we say that  $f$  is a convex function if

$$\forall \lambda \in (0,1), \forall s, r \in \mathcal{A}, \quad \text{then } f(\lambda r + (1 - \lambda)s) \leq \lambda f(r) + (1 - \lambda)f(s)$$



- Properties:

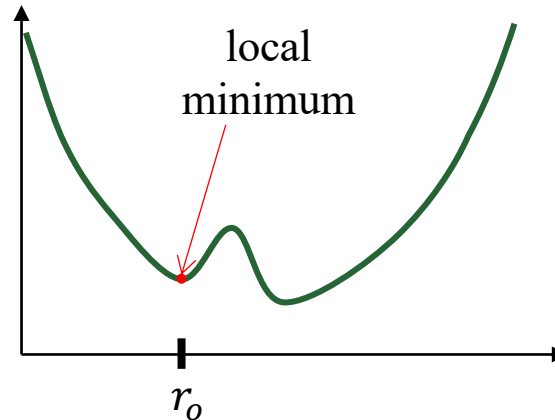
- The sums of convex functions is convex
- The maximum of a set of convex functions is convex
- If  $f(y)$  is convex, then  $f(Ay)$  is also convex.
- $f(y)$  is concave if  $-f(y)$  is convex.
- $f(y) = Ay + b$  is both convex and concave

# Properties of Convex Functions

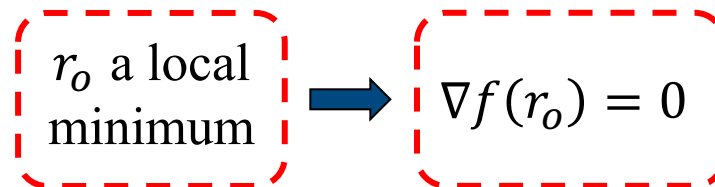
- Valuable properties of convex functions
  - The sums of convex functions is convex.
    - Let  $f(x) = \sum_n f_n(x)$ . If  $f_n(x)$  are convex, then  $f(x)$  is convex.
  - The maximum of a set of convex functions is convex.
    - Let  $f(x) = \max_n f_n(x)$ . If  $f_n(x)$  are convex, then  $f(x)$  is convex.
  - The second derivative of a convex function is positive.
    - Let  $f(y)$  have two continuous derivatives, and let  $H_{i,j}(y) = \frac{\partial^2 f}{\partial y_i \partial y_j}$  be the Hessian of  $f$  at  $y$ . Then  $f(y)$  is convex if and only if  $H(y)$  is non-negative definite for all  $y$ .
  - A convex function of a linear transform is convex.
    - If  $f(y)$  is convex, then  $f(Ay)$  is also convex.
  - An affine transform is both convex and concave.
    - $f(y) = Ay + b$  is both convex and concave.
  - A function is concave if its negative is convex.
    - $f(y)$  is concave if  $-f(y)$  is convex.

# Local Minimum

- Let  $f: \mathcal{A} \rightarrow \mathfrak{R}$  where  $\mathcal{A} \subset \mathfrak{R}^N$ .
  - We say that  $r_o \in \mathcal{A}$  is a local minimum of  $f$  if there  $\exists \epsilon > 0$  s.t.  $\forall r \in \mathcal{A} \cap B(r, \epsilon), f(r) \geq f(r_o)$ .

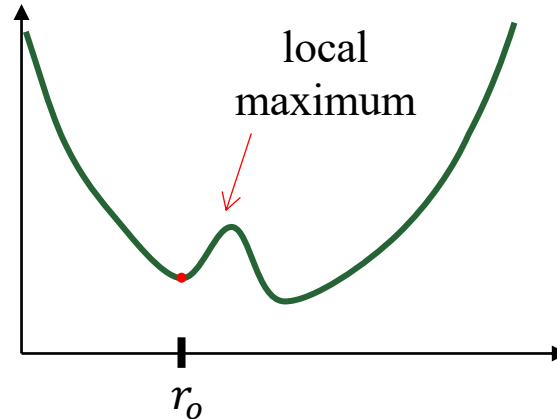


- Necessary condition for local minima:
  - Let  $f$  be continuously differentiable and let  $r_o$  be a local minimum, then  $\nabla f(r_o) = 0$ .

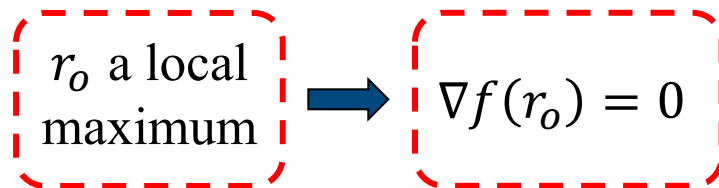


# Local Maximum

- Let  $f: \mathcal{A} \rightarrow \mathbb{R}$  where  $\mathcal{A} \subset \mathbb{R}^N$ .
  - We say that  $r_o \in \mathcal{A}$  is a local minimum of  $f$  if there  $\exists \epsilon > 0$  s.t.  $\forall r \in \mathcal{A} \cap B(r, \epsilon), f(r) \leq f(r_o)$ .



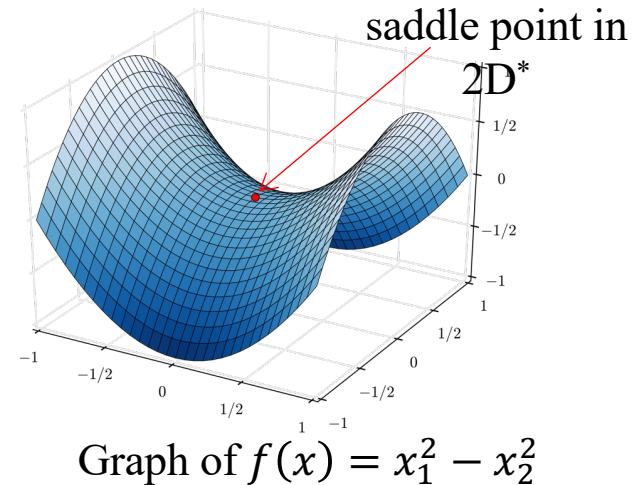
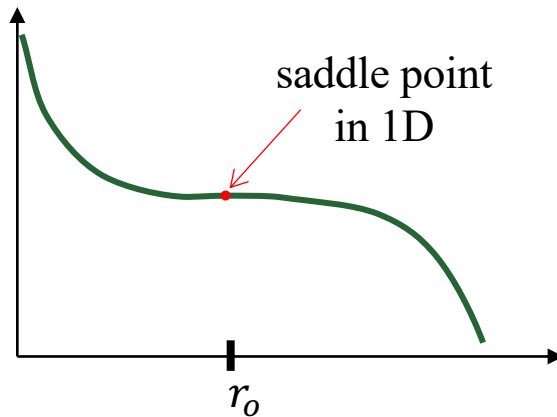
- Necessary condition for local maximum:
  - Let  $f$  be continuously differentiable and let  $r_o$  be a local minimum, then  $\nabla f(r_o) = 0$ .



# Saddle Point

■ Let  $f: \mathcal{A} \rightarrow \mathfrak{R}$  where  $\mathcal{A} \subset \mathfrak{R}^N$  be a continuously differentiable function.

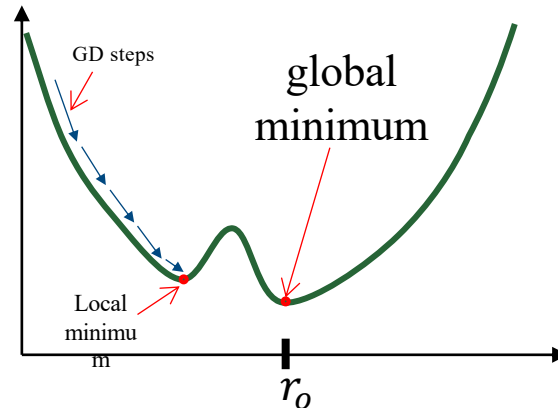
- We say that  $r_o \in \mathcal{A}$  is a saddle point of  $f$  if  $\nabla f(r_o) = 0$  and  $r_o$  is not a local minimum or maximum.



- Saddle points can cause more problems than you might think.

# Global Minimum

- Let  $f: \mathcal{A} \rightarrow \mathbb{R}$  where  $\mathcal{A} \subset \mathbb{R}^N$ .
  - We say that  $r_0 \in \mathcal{A}$  is a global minimum of  $f$  if  $\forall r \in \mathcal{A}, f(r) \geq f(r_0)$ .



- Comments:
  - In general, finding global minimum is difficult.
  - Gradient descent optimization typically becomes trapped in local minima.

# Optimization Theorems

- Let  $f: \mathcal{A} \rightarrow \mathfrak{R}$  where  $\mathcal{A} \subset \mathfrak{R}^N$ .
  - If  $f$  is continuous and  $\mathcal{A}$  is compact, then  $f$  takes on a global minimum in  $\mathcal{A}$ .
  - If  $f$  is convex on  $\mathcal{A}$ , then any local minimum is a global minimum.
  - If  $f$  is continuously differentiable and convex on  $\mathcal{A}$ , then  $\nabla f(r_o) = 0$  implies that  $r_o \in \mathcal{A}$  is a global minimum of  $f$ .
- Important facts:
  - Global minimum may not be unique.
  - If  $\mathcal{A}$  is closed but not bounded, then it may not take on a global minimum.
  - Generally speaking, gradient descent algorithms converge to the global minimum of continuously differentiable convex functions.
  - Most interesting functions in ML are not convex!

