TOTAL VARIATION
AND
CALCULUS

Lecture 3 - CMSC764
**IMAGE GRADIENT**

**Neumann**

\[
\begin{pmatrix}
  u_1 - u_0 \\
  u_2 - u_1 \\
  u_3 - u_2 \\
\end{pmatrix}
\]

**Circulant**

\[
\begin{pmatrix}
  u_1 - u_0 \\
  u_2 - u_1 \\
  u_3 - u_2 \\
  u_0 - u_3 \\
\end{pmatrix}
\]
TOTAL VARIATION

\[ TV(x) = |\nabla x| = \sum_{k} |x_{k+1} - x_{k}| \]

How to compute?
- Convolve with stencil \([0, 1, -1]\)
- Take absolute value
- Take sum
TV IN 2D

$$(\nabla x)_{ij} = (x_{i+1,j} - x_{i,j}, x_{i,j+1} - x_{i,j})$$

Anisotropic $\quad |(\nabla x)_{ij}| = |x_{i+1,j} - x_{i,j}| + |x_{i,j+1} - x_{i,j}|$

Isotropic $\quad \|(\nabla x)_{ij}\| = \sqrt{(x_{i+1,j} - x_{ij})^2 + (x_{i,j+1} - x_{ij})^2}$
COMPUTING TV ON IMAGES

• Two linear filters
  • $x$-stencil = $(-1 \ 1 \ 0)$
  • $y$-stencil = $(-1 \ 1 \ 0)'$

  ...or...

• Two linear convolutions
  • $x$-kernel = $(0 \ 1 \ -1)$
  • $y$-kernel = $(0 \ 1 \ -1)'$
COMPUTING TV ON IMAGES

- Two linear convolutions
- $x$-kernel = $(0 \ 1 \ -1)$
- $y$-kernel = $(0 \ 1 \ -1)'$

Fast Transforms: use FFT
FOURIER TRANSFORM

\[ \hat{x}_k = \sum_{n} x_n e^{-i2\pi kn/N} \]

\[ \hat{x}_k = \langle x, \mathcal{F}_k \rangle \]
CONVOLUTION

\[ f \ast g(x) = \sum_s f(x - s)g(s) = \sum_s f(x + s)g(-s) \]

**don’t forget**

WRONG!!
PROPERTIES OF FFT

• Classical FFT is **ORTHOGONAL**

\[ \mathcal{F}^H \mathcal{F} = I \]

• Computable in \( O(N \log N) \) time (Cooley-Tukey)

\[ \mathcal{F}x = \begin{pmatrix} I & B \\ I & -B \end{pmatrix} \begin{pmatrix} \mathcal{F}x_e \\ \mathcal{F}x_o \end{pmatrix} \]
The Convolution Theorem

- Convolution = multiplication in Fourier domain

\[ x \ast y = \mathcal{F}^T (\mathcal{F}x \cdot \mathcal{F}y) \]

What does this mean?
A BETTER WAY TO THINK ABOUT IT

\[ x \ast y = \mathcal{F}^T (\mathcal{F}x \cdot \mathcal{F}y) \]

- **FFT DIAGONALIZES** convolution matrices

\[ Kx = \mathcal{F}^T D \mathcal{F}x \]

**Trivia!**

- Is \[ K_1 K_2 = K_2 K_1 \]
- Does \[ \partial_x \partial_y u = \partial_y \partial_x u \]
HOW DO WE FIGURE OUT D?

- The diagonal matrix is just the FFT of backward filter stencil.

- Example: derivative
  - Define stencil: \([-1 \ 1 \ 0]\)
  - Convolve with: \([0 \ 1 \ -1]\)
  - Embed stencil in matrix:
    \[
    s = \begin{bmatrix}
      1 & -1 & 0 & 0 & 0 & 0 & 0 & 0
    \end{bmatrix}
    \]
  - Compute the diagonal:
    \[
    D = \mathcal{F} s
    \]
  - Perform convolution using EVD

\[
Kx = \mathcal{F}^T D \mathcal{F} x
\]
CONDITION NUMBER

\[ Kx = \mathcal{F}^T D \mathcal{F}x \]

\[ \text{diag}(D) = \mathcal{F}[1, -1, 0, 0, 0, 0] \]

How poorly conditioned is the FFT?

How poorly conditioned is a convolution?

Zero: why? What is the eigenvector
WHY IS THE CHIRP WELL-CONDITIONED?

Echolocation chirp: brown bat

Linear chirp
FFT IN THE WILD

Matlabs DFT is NOT ORTHOGONAL!

WHY?

Convolution with a delta function = identity

\[ F_\perp = \frac{1}{\sqrt{N}} F \quad F_{\perp}^{-1} = \sqrt{N} F^{-1} \]

FFT still diagonalizes convolutions!

\[ F^{-1} D F = \sqrt{N} F^{-1} D F^T \frac{1}{\sqrt{N}} = F_{\perp}^{-1} D F_{\perp} \]

Same D
CONTINUOUS GRADIENTS
WHY WE NEED GRADIENT

First-order conditions

\[ \nabla f(x^*) = 0 \]

First-order methods

Gradient descent
THE CALCULUS YO MOMMA TAUGHT YOU

\[ f(x, y, z) : \mathbb{R}^n \rightarrow \mathbb{R} \]

\[ \nabla f(x, y, z) = \partial_x f \mathbf{i} + \partial_y f \mathbf{j} + \partial_z f \mathbf{k} \]

What letter do we use for 4th variable?

What letter do we use for 1,000,000th variable?
DERIVATIVE

\[ f : \mathbb{R}^n \to \mathbb{R}^m \]

**Freshet derivative** is a linear operator \( D \) with

\[
\lim_{\|h\| \to 0} \frac{f(x + h) - f(x) - Dh}{\|h\|} = 0
\]

A linear operator that **locally** behaves like \( f \)

In optimization: variables are usually **column vectors**
DERIVATIVE

In optimization: variables are usually **column vectors**

\[ Df = \begin{pmatrix} D_{x1}f & D_{x2}f & D_{x3}f \end{pmatrix} \]

In this case the derivative is the **Jacobian**

\[
\begin{pmatrix}
D_{x1}f_1 & D_{x2}f_1 & D_{x3}f_1 \\
D_{x1}f_2 & D_{x2}f_2 & D_{x3}f_2 \\
D_{x1}f_3 & D_{x2}f_3 & D_{x3}f_3
\end{pmatrix}
\]
In optimization: variables are usually column vectors

**Definition:** Gradient is the adjoint of the derivative

For scalar-valued functions

\[
Df = \left( \frac{\partial x_1 f}{\partial x}, \frac{\partial x_2 f}{\partial x}, \frac{\partial x_3 f}{\partial x} \right) \quad \nabla f = \begin{pmatrix}
\frac{\partial x_1 f}{\partial x} \\
\frac{\partial x_2 f}{\partial x} \\
\frac{\partial x_3 f}{\partial x}
\end{pmatrix}
\]
GRADIENT

In optimization: variables are usually **column vectors**

**Better Definition:** The gradient of a scalar-valued function is a matrix of derivatives that is the **same shape** as the unknowns

example: matrix of unknowns

\[
x = \begin{pmatrix}
x_{11} & x_{12} & x_{13} \\
x_{21} & x_{22} & x_{23} \\
x_{31} & x_{32} & x_{33}
\end{pmatrix}
\]

\[
\nabla f = \begin{pmatrix}
\partial_{11} f & \partial_{12} f & \partial_{13} f \\
\partial_{21} f & \partial_{22} f & \partial_{23} f \\
\partial_{31} f & \partial_{32} f & \partial_{33} f
\end{pmatrix}
\]
WHAT’S THE GRADIENT?!

<table>
<thead>
<tr>
<th>Function</th>
<th>gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x) = Ax$</td>
<td>$\nabla f(x) = A^T$</td>
</tr>
<tr>
<td>$f(x) = |x|^2$</td>
<td>$\nabla f(x) = 2x$</td>
</tr>
<tr>
<td>$f(x) = \frac{1}{2} x^T Ax$</td>
<td>$\nabla f(x) = Ax$</td>
</tr>
</tbody>
</table>

symmetric $A$
CHAIN RULE

\[ h(x) = f \circ g(x) = f(g(x)) \]

**single-variable chain rule**

\[ \partial f(g(x)) = f'(g)g'(x) \]

**multi-variable chain rule**

\[ Df(g(x)) = Df \circ Dg(x) \]

*for gradients*

\[ \nabla f(g(x)) = \nabla g(x) \nabla f(g) \]

Swap
EXAMPLE: LEAST SQUARES

\[ f(g(x)) = \frac{1}{2} \|Ax - b\|^2 \]

\[ f = \frac{1}{2} \|x\|^2 \]

\[ Ax - b = g \]

\[ x \]

\[ \nabla \]

\[ A^T(Ax - b) \]

\[ \nabla g(x) \nabla f(g) \]

How would you find the gradient?
IMPORTANT RULES

If

\[ h(x) = f(Ax) \]

Then

\[ \nabla h(x) = A^T f'(Ax) \]

If

\[ h(x) = f(xA) \]

Then

\[ \nabla h(x) = f'(xA) A^T \]

Why?
EXAMPLE: RIDGE REGRESSION

\[ f(x) = \frac{\lambda}{2} \|x\|^2 + \frac{1}{2} \|Ax - b\|^2 \]

Optimality Condition

\[ \nabla f(x) = \lambda x + A^T (Ax - b) = 0 \]

\[ (A^T A + \lambda I)x = A^T b \]

\[ x^* = (A^T A + \lambda I)^{-1} A^T b \]
REGULARIZED GRADIENTS

Huber regularization

\[ |x| \approx h(x) = \begin{cases} 
\frac{1}{2} x^2, & \text{for } |x| \leq \delta \\
\delta(|a| - \frac{1}{2}\delta), & \text{otherwise} 
\end{cases} \]

Hyperbolic regularization

\[ |x| \approx \sqrt{x^2 + \epsilon^2} \]
GRADIENT OF TV

\[ |\nabla x| + \frac{\mu}{2} \|x - f\|^2 \]

Regularize L1

\[ h(\nabla x) + \frac{\mu}{2} \|x - f\|^2 \]

Differentiate

\[ \nabla^T h'(\nabla x) + \mu(x - f) \]

\[ h'(x) = \frac{x}{\sqrt{x^2 + \epsilon^2}} \]

divide component-wise
LOGISTIC REGRESSION

Data vectors \( \{x_i\} \)

Labels \( \{y_i\} \)

Model

\[
p(y_i = 1|x_i) = \sigma(x_i w) = \frac{e^{x_i w}}{1 + e^{x_i w}}
\]

Maximum Likelihood (MAP) Estimator

\[
\text{maximize} \quad p(w|X, Y)
\]

\[
\text{minimize} \quad \sum_{k=1}^{m} \log(1 + \exp(-y_k \cdot x_k w))) = f_{lr}(Y X w)
\]
LOGISTIC REGRESSION

\[
\text{minimize}_{w} \sum_{k=1}^{m} \log(1 + \exp(-y_k \cdot x_k w))) = f_{lr}(YXw)
\]

Break the problem down

\[
f_{lr}(z) = \sum_{k=1}^{m} \log(1 + \exp(-z_k)) \quad \hat{X} = YX
\]

Coordinate separable

Linear operator

\[
\text{minimize}_{x} f_{lr}(\hat{X}w)
\]

\[
\nabla\{f_{lr}(\hat{X}w)\} = \hat{X}^T \nabla f_{lr}(\hat{X}w)
\]

\[
\nabla f_{lr}(z) = \frac{-\exp(-z)}{1 + \exp(-z)}
\]
NON-NEGATIVE MATRIX FACTORIZATION

\[
\begin{align*}
\text{minimize} & \quad E(X, Y) = \frac{1}{2} \| XY - S \|^2 \\
\text{subject to} & \quad X \geq 0, Y \geq 0
\end{align*}
\]

\[
\begin{align*}
\partial_Y E &= X^T (XY - S) \\
\partial_X E &= (XY - S) Y^T
\end{align*}
\]
NEURAL NETS: LOGISTIC REGRESSION ON STEROIDS

\[ \sigma(\sigma(\sigma(x_i W_1)W_2)W_3) = y_3 \]

MLP = “Multilayer Perception”
GRADIENT OF NET

\[ \sigma(\sigma(\sigma(x_i W_1) W_2) W_3) = y_3 \]
Break network apart into simple operations

Training data: row vector

"Pre-activations"
FORWARD PASS

Compute the intermediate values

\[ x_i \xrightarrow{W_1} z_1 \xrightarrow{\sigma_1} y_1 \xrightarrow{W_2} z_2 \xrightarrow{\sigma_2} y_2 \xrightarrow{W_3} z_3 \xrightarrow{\text{loss}} \ell \]

Store the \( y \)'s for later
BACKWARD PASS

Let's find derivative of loss with respect to $x$

\[
\frac{\partial \ell}{\partial z_3}
\]
BACKWARD PASS

Let's find derivative of loss with respect to $x$

$$x_i \xrightarrow{W_1} z_1 \xrightarrow{\sigma_1} y_1 \xrightarrow{W_2} z_2 \xrightarrow{\sigma_2} y_2 \xrightarrow{W_3} z_3 \xrightarrow{\text{loss}} l$$

The derivative with respect to this is:

$$\frac{\partial l}{\partial z_3}$$
BACKWARD PASS

Let's find derivative of loss with respect to $x$

$$x_i \rightarrow z_1 \rightarrow y_1 \rightarrow z_2 \rightarrow y_2 \rightarrow z_3 \rightarrow \ell$$

$\sigma'_2 W_3 \frac{\partial \ell}{\partial z_3}$

derivative with respect to this
Let's find derivative of loss with respect to $x$.

$W_2 \sigma'_2 W_3 \frac{\partial \ell}{\partial z_3}$
BACKWARD PASS

Let's find derivative of loss with respect to $x$

$$x_i \xrightarrow{W_1} z_1 \xrightarrow{\sigma_1} y_1 \xrightarrow{W_2} z_2 \xrightarrow{\sigma_2} y_2 \xrightarrow{W_3} z_3 \xrightarrow{\text{loss}} \ell$$

derivative
with respect to this

$$\sigma'_1 W_2 \sigma'_2 W_3 \frac{\partial \ell}{\partial z_3}$$
Let's find derivative of loss with respect to $x$
GRADIENT VS DERIVATIVE

derivative

\[ \frac{\partial l}{\partial x_1} = W_1 \sigma'_1 W_2 \sigma'_2 W_3 \frac{\partial l}{\partial z_3} \]

gradient

\[ \nabla l_{x_1} = \nabla_{z_3} l \ W_3^\top \sigma'_2 \ W_2^\top \sigma'_1 \ W_1^\top \]
SANITY CHECK

\[ \nabla \ell_{x_1} = \nabla_{z_3} \ell \ W_T^3 \sigma_2' \ W_T^2 \sigma_1' \ W_T^1 \]

\[ \ell' = \nabla \ell \]

What shape should the output be?
INSANITY CHECK:
WHAT ABOUT WEIGHTS?

Let’s find derivative of loss with respect to $W_2$

Use chain rule

$$\frac{\partial l}{\partial W_2} = \frac{\partial l}{\partial z_2} \frac{\partial z_2}{\partial W_2}$$

$$\nabla_{W_2} l = \nabla_{W_2} z_2 \nabla_{z_2} l$$
INSANITY CHECK: WHAT ABOUT WEIGHTS?

Let's find derivative of loss with respect to $W_2$

$$
\sigma_2' \ W_3 \ \frac{\partial \ell}{\partial z_3}
$$
Let's find derivative of loss with respect to $W_2$
INSANITY CHECK: WHAT ABOUT WEIGHTS?

Let’s find gradient of loss with respect to $W_2$

$$x_i \xrightarrow{W_1} z_1 \xrightarrow{\sigma_1} y_1 \xrightarrow{W_2} z_2 \xrightarrow{\sigma_2} y_2 \xrightarrow{W_3} z_3 \xrightarrow{\text{loss}} l$$

$$y_1^\top \nabla_{z_3} l \quad W_3^\top \sigma'_2$$
COMPLETE BACKPROP

\[ x_i \rightarrow z_1 \rightarrow y_1 \rightarrow z_2 \rightarrow y_2 \rightarrow z_3 \rightarrow \ell \]

\[ \nabla_{z_3} \ell \]

\[ \nabla_{z_2} \ell = \nabla_{z_3} \ell W_3^\top \sigma'_2 \]

\[ \nabla_{z_1} \ell = \nabla_{z_3} \ell W_3^\top \sigma'_2 W_2^\top \sigma'_1 \]

\[ \nabla W_3 \ell = y_2^\top \nabla_{z_3} \ell \]

\[ \nabla W_2 \ell = y_1^\top \nabla_{z_2} \ell \]

\[ \nabla W_1 \ell = x_i^\top \nabla_{z_1} \ell \]
CONVOLUTIONAL NETS

Convolution-based template matching
CONVOLUTIONAL NETS

Regular-old MLP:

Fancy-schmance convnet:

\[ y_1 = \sigma(x_i W_1) \]

\[ y_1 = \sigma(x_i K_1) = \sigma(x_i * k_1) \]
CONVOLUTIONAL NETS

\[ y = \sigma(\sigma(x_i K_1) K_2) K_3 \]

\[ \nabla z_1 \ell = \nabla z_3 \ell K_3^\top \sigma_2' K_2^\top \sigma_1' \]

Adjoint of convolution
CONVOLUTIONAL NETS

\[ \nabla_x \sigma_i (xK_i) = \text{diag}(\sigma_i') K_i^T \]

Stencil

\[ B \rightarrow \mathcal{F}_k \rightarrow \text{Diagonal} \rightarrow \text{Adjoint of convolution} \]

\[ K = \mathcal{F}^H D \mathcal{F} \rightarrow \text{Adjoint} \rightarrow ??? \]
\[ \nabla_x \sigma_i(xK_i) = \text{diag}(\sigma'_i)K_i^T \]

\[
K = \mathcal{F}^H D \mathcal{F}
\]

Adjoint

\[
K = \mathcal{F}^H \bar{D} \mathcal{F}
\]

???

Stencil

Diagonal

Adjoint of convolution
CONVOLUTIONAL NETS

\[ \nabla_x \sigma_i(xK_i) = \text{diag}(\sigma'_i)K^T_i \]

Stencil \hspace{1cm} F_k \hspace{1cm} Diagonal

K = F^HDF \hspace{1cm} Adjoint

K = F^H \bar{D}F

F \text{flip}(k) \hspace{1cm} Stencil