Wasserstein GAN

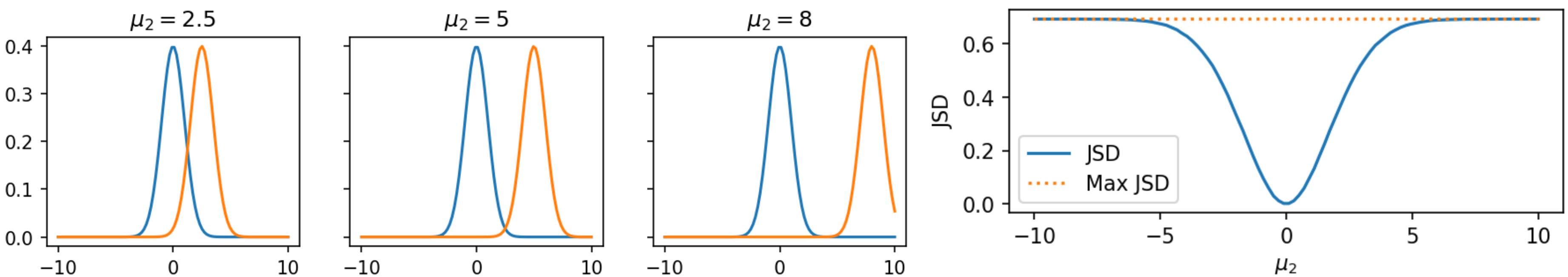
ECE57000: Artificial Intelligence

David I. Inouye

Motivation: **Vanishing gradients** for generator caused by a discriminator that is "too good"

From: https://developers.google.com/machine-learning/gan/problems

- Vanishing gradient means $\nabla_G V(D, G) \approx 0$.
  - Gradient updates do not improve G
- Theoretically, this is an issue of JSD



- Practically, careful balance during training required:
  - Optimizing D too much leads to vanishing gradient
  - **But** training too little means it is not close to JSD

Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

Wasserstein GAN: Better gradient values and better convergence (better stability)

- Better gradients even after significant training
- Convergent training even without batch normalization

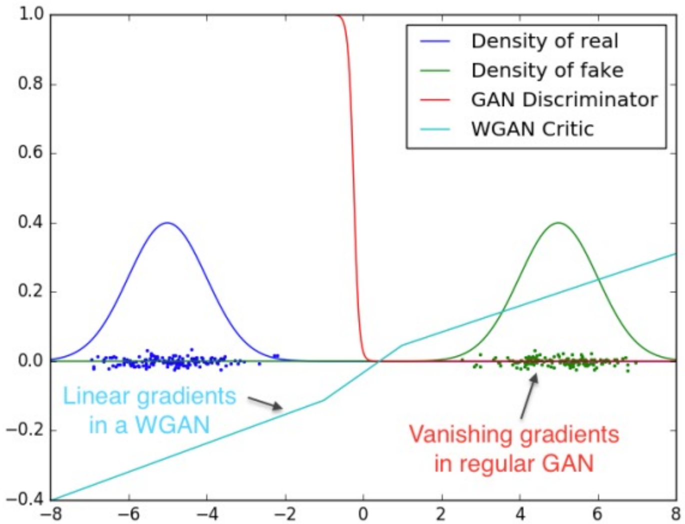


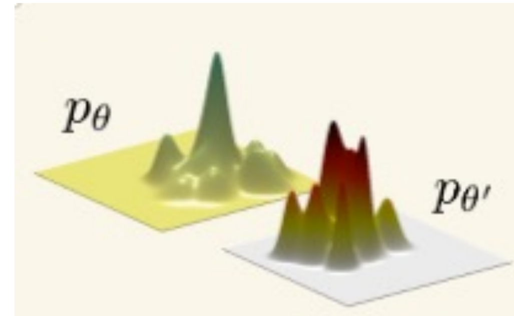
Figure 6: Algorithms trained with a generator without batch normalization and constant number of filters at every layer (as opposed to duplicating them every time as in [18]). Aside from taking out batch normalization, the number of parameters is therefore reduced by a bit more than an order of magnitude. Left: WGAN algorithm. Right: standard GAN formulation. As we can see the standard GAN failed to learn while the WGAN still was able to produce samples.

Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

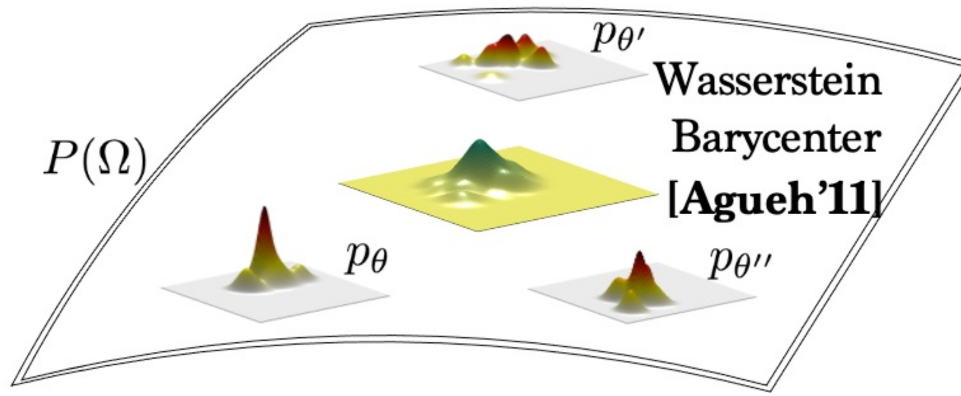
Outline of Wasserstein GANs

- Preliminaries
  - Optimal transport (OT) and Monge problem
  - Wasserstein distribution distance based on OT
  - Lipschitz continuous functions
- WGAN adversarial objective
  - Wasserstein distance as maximization problem
  - Comparison to standard GAN objective
- WGAN algorithms
  - Clipping algorithm (original WGAN)
  - Gradient penalty algorithm

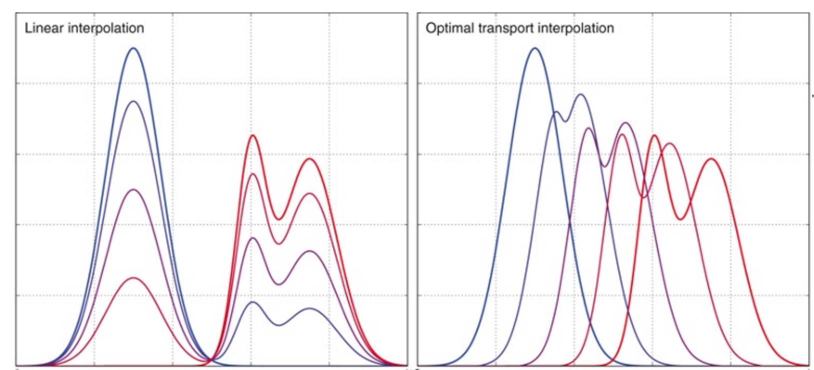
What is Optimal Transport?



- The natural geometry for **probability distributions**.



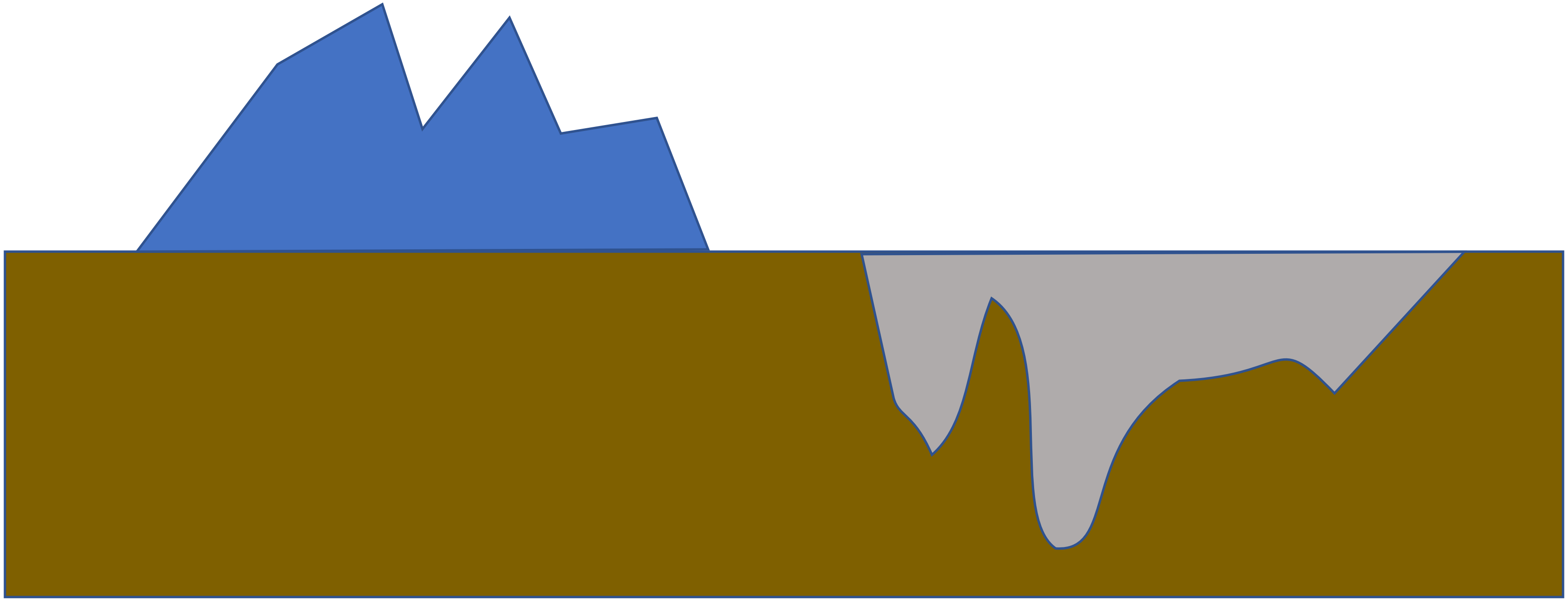
- How close are two distributions?
- Which distribution is between two distributions?
- What is the shortest path between two distributions?



Figures from Marco Cuturi & Justin M Solomon. A Primer on Optimal Transport, NeurIPS Tutorial, 2017.

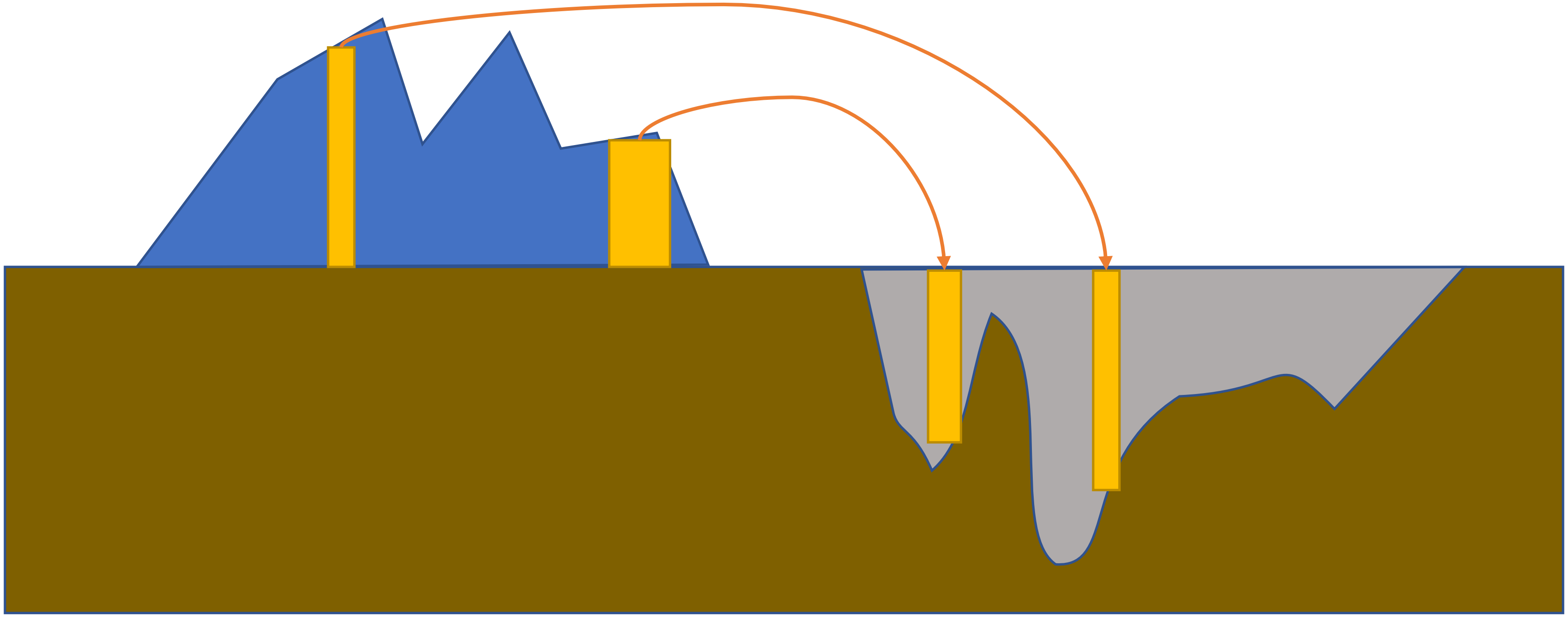
<u>OT Monge map problem</u>: Find the **minimum (optimal)** transportation plan between **two distributions**

- We want a map (i.e., a function) that moves the mass from the mountain to fill the hole (exactly)



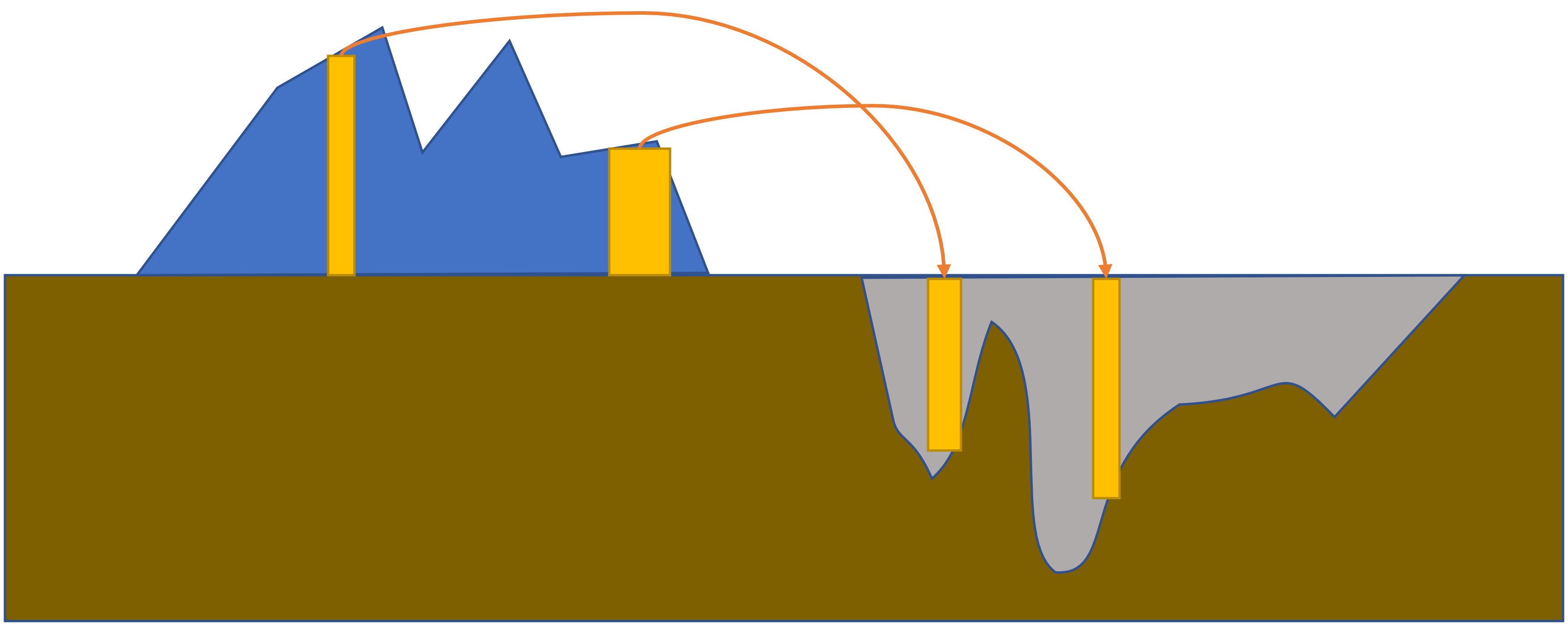
<u>OT Monge map problem</u>: Find the **minimum (optimal)** transportation plan between **two distributions**

- One plan for showing movement of parts of the mass



__OT Monge map problem__: Find the **minimum (optimal)** transportation plan between **two distributions**

- Better transport plan if using squared Euclidean cost



<u>OT Monge map problem</u>: Find the optimal transportation plan between **two distributions**

- We denote the source distribution by $X \sim p_X$ and the target distribution by $Y \sim p_Y$.
  - p_X is like the mound of dirt
  - p_Y is like the hole in the ground
- The OT Monge problem can be formulated as the optimization problem

$$\min_T \mathbb{E}_{p_X}\left[c\left(x, T(x)\right)\right]$$
$$\text{s.t.} \quad p_{T(X)} = p_Y$$

  - Where the constraint makes sure that the transformed source distribution is aligned with target distribution (i.e., the moved dirt fills the hole exactly).

The Wasserstein-1 distribution distance can be derived from the solution to this OT problem

- The Wasserstein-1 distance is defined as:

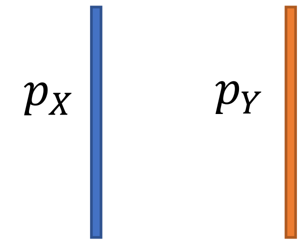
$$W_1(p_X, p_Y) = \begin{pmatrix} \min_T \mathbb{E}_{p_X}[\|x - T(x)\|_2] \\ \text{s.t.} \quad p_{T(X)} = p_Y \end{pmatrix}$$

- This is merely the optimal value of the Monge problem with $c\big(x, T(x)\big) = \|x - T(x)\|_2$

- Other similar Wasserstein distances can be defined such as Wasserstein-2

Comparison of Wasserstein distance to JSD for disjoint uniform distributions in 1D

- Suppose both distributions are on a line segment in 2D
  - JSD gives no information
  - Wasserstein (also known as Earth Mover distance) gives nice information
- Wasserstein distance gives how far you need to move the line



$\theta =$ distance between line distributions

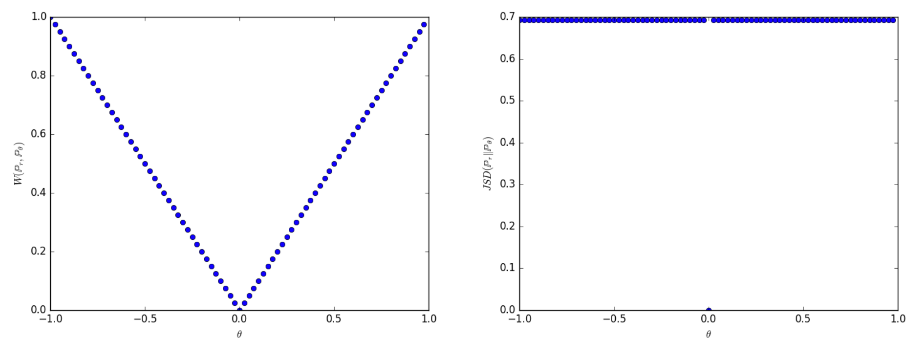
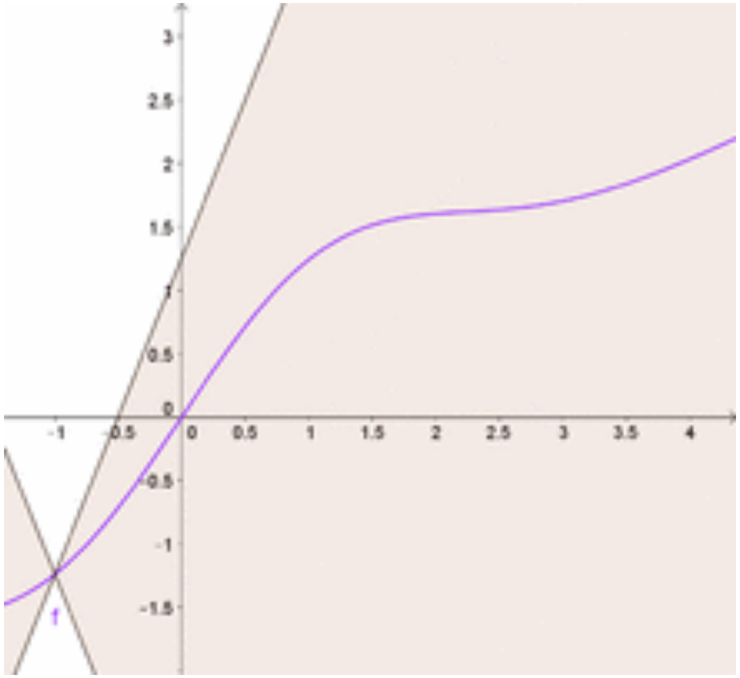


Figure 1: These plots show $\rho(\mathbb{P}_\theta, \mathbb{P}_0)$ as a function of θ when ρ is the EM distance (left plot) or the JS divergence (right plot). The EM plot is continuous and provides a usable gradient everywhere. The JS plot is not continuous and does not provide a usable gradient.

Figure from Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

Preliminaries for WGAN: What is a Lipschitz smooth function?

- Informally, a Lipschitz continuous function means that the function does not change too quickly
- Intuitively, a double cone whose origin can be moved along the function so that the whole function always stays outside the double cone



https://en.wikipedia.org/wiki/Lipschitz_continuity

Preliminaries for WGAN: What is a Lipschitz smooth function?

- Formally, the slope of the line connecting any two points on the function is bounded

$$\frac{\|f(x_2) - f(x_1)\|_2}{\|x_2 - x_1\|_2} \le K$$

- If the function is continuously differentiable, then

$$\|\nabla_x f(x)\|_2 \le K, \qquad \forall x$$

- Examples
  - $f(x) = ax$, with $K = a$
  - $f(x) = |x|$, with $K = 1$
  - $f(x) = \sin(x)$, with $K = 1$
- Counterexamples
  - $f(x) = x^2$
  - $f(x) = \exp(x)$
  - $f(x) = \begin{cases} 1, & \text{if } x \ge 0 \\ 0, & \text{otherwise} \end{cases}$

Wasserstein GAN first reformulates the minimization over T to maximization over f

- The original Wasserstein problem:

$$W_1(p_X, p_Y) = \begin{pmatrix} \min_T \mathbb{E}_{p_X}[\|x - T(x)\|_1] \\ \text{s.t.} \quad p_{T(X)} = p_Y \end{pmatrix}$$

- The equivalent dual problem (Kantorovich-Rubinstein duality):

$$W_1(p_X, p_Y) = \begin{pmatrix} \max_f \mathbb{E}_{p_X}[f(x)] - \mathbb{E}_{p_Y}[f(y)] \\ \text{s.t.} \ \|f\|_L \le 1 \end{pmatrix}$$

  - Where $\|f\|_L \le 1$ means the Lipschitz constant of f is less than 1

- Very informally, this switches the objective with the constraints and vice versa

Villani, C. (2009). *Optimal transport: old and new* (Vol. 338, p. 23). Berlin: Springer.
Marco Cuturi. (2019). A Primer on Optimal Transport Part 2. Accessed on 11/4/2021. https://www.youtube.com/watch?v=R49Xb9eAUBA

Wasserstein GAN first reformulates the minimization over T to maximization over f

- Wasserstein-1 dual problem:

$$W_1(p_X, p_Y) = \begin{pmatrix} \max_f \mathbb{E}_{p_X}[f(x)] - \mathbb{E}_{p_Y}[f(y)] \\ \mathrm{s.t.}\ \|f\|_L \leq 1 \end{pmatrix}$$

- Compare with JSD maximization problem:

$$JSD(p_X, p_Y) = \begin{pmatrix} \max_D \mathbb{E}_{p_X}[\log D(x)] + \mathbb{E}_{p_Y}\big[\log\big(1 - D(y)\big)\big] \\ \mathrm{s.t.}\ D: \mathbb{R}^d \to [0,1] \end{pmatrix}$$

Putting it all together for the final adversarial min-max problem of Wasserstein GAN

- Wasserstein GAN objective

$$\min_G \max_f \mathbb{E}_{p_{data}}[f(x)] - \mathbb{E}_{p_z}\left[f\big(G(z)\big)\right]$$
$$\text{s.t. } \|f\|_L \leq 1$$

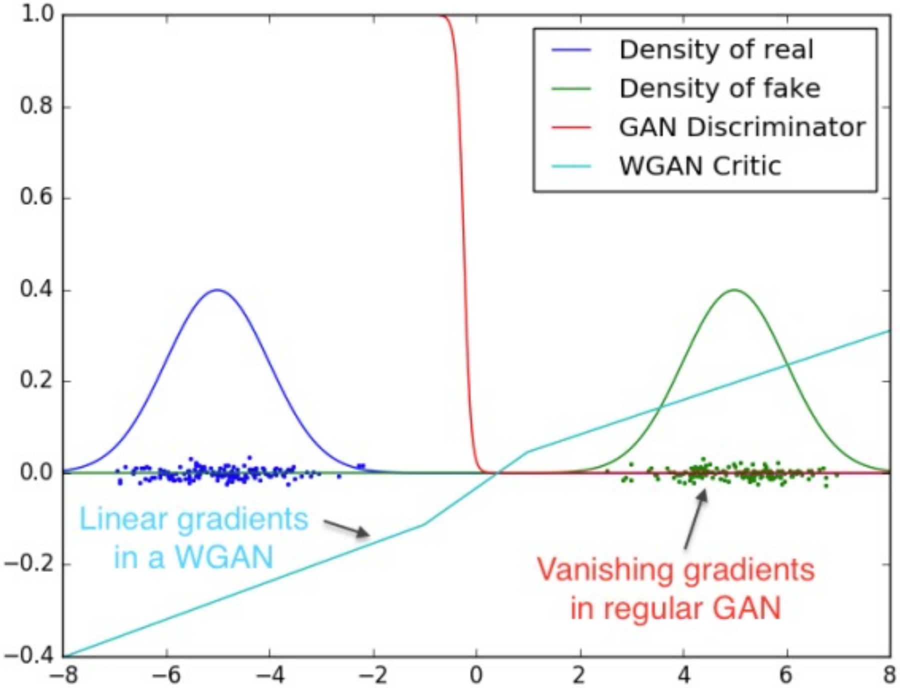
- Original (JSD) GAN objective

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}\left[\log\Big(1 - D\big(G(z)\big)\Big)\right]$$
$$\text{s.t. } D: \mathbb{R}^d \to [0,1]$$

Comparison of Wasserstein distance to JSD in 1D

- This Lipschitz constraint rather than the classifier constraint produces better gradients

- No balancing of training objective required (in theory)



Key question:
How do we enforce the Lipschitz constraint?

- Clip the parameter weights
- Why would this partially work?
  - If all weights are bounded, then the Lipschitz constant is bounded.
  - If the Lipschitz constant is bounded, it is equivalent to scaled Lipschitz constraint

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

Algorithm from Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.

Better idea: Empirically encourage this constraint by adding a penalty term

- Original problem:

$$\min_G \max_{f:\|f\|_L\leq 1} \mathbb{E}_{p_{data}}[f(x)] - \mathbb{E}_{p_z}\left[f\big(G(z)\big)\right]$$

- Relaxed Lipschitz constraint via gradient penalty

$$\min_G \max_f \mathbb{E}_{p_{data}}[f(x)] - \mathbb{E}_{p_z}\left[f\big(G(z)\big)\right]$$
$$-\lambda \mathbb{E}_{p_{\tilde{x}}}[(\|\nabla_x f(\tilde{x})\|_2 - 1)^2]$$

- For $p_{\tilde{x}}$, they use interpolated samples between real and fake samples:

$$\tilde{x} = \epsilon x + (1-\epsilon)G(z)$$

  - Where $x \sim p_X, z \sim p_z, \epsilon \sim \mathrm{Uniform}([0,1])$

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems* (pp. 5767-5777).

How do we implement the gradient penalty?

- Key problem: We don’t know gradients in closed-form so how do we compute the objective?
- First note that backprop itself is a computation!
- Solution: Use autograd to compute gradient and then backprop through that (gradient of gradient)

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.

```
while θ has not converged do
    for t = 1, ..., n_critic do
        for i = 1, ..., m do
            Sample real data x ~ P_r, latent variable z ~ p(z), a random number ε ~ U[0, 1].
            x̃ ← G_θ(z)
            x̂ ← εx + (1 − ε)x̃
            L^(i) ← D_w(x̃) − D_w(x) + λ(‖∇_x̂ D_w(x̂)‖_2 − 1)^2
        end for
        w ← Adam(∇_w (1/m) Σ_{i=1}^m L^(i), w, α, β1, β2)
    end for
    Sample a batch of latent variables {z^(i)}_{i=1}^m ~ p(z).
    θ ← Adam(∇_θ (1/m) Σ_{i=1}^m −D_w(G_θ(z)), θ, α, β1, β2)
end while
```

Algorithm from Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems* (pp. 5767-5777).

Outline of Wasserstein GANs

- Preliminaries
  - Optimal transport (OT) and Monge problem
  - Wasserstein distribution distance based on OT
  - Lipschitz continuous functions
- WGAN adversarial objective
  - Wasserstein distance as maximization problem
  - Comparison to standard GAN objective
- WGAN algorithms
  - Clipping algorithm (original WGAN)
  - Gradient penalty algorithm

Additional resources for optimal transport and Wasserstein distance

- A primer on Optimal Transport slides:
  - https://nips.cc/Conferences/2017/Schedule?showEvent=8736
  - Alternative link: https://www.dropbox.com/s/55tb2cf3zipl6xu/aprimeronOT.pdf?dl=0
- Optimal transport tutorial videos
  - Video Part 1 - https://www.youtube.com/watch?v=6iR1E6t1MMQ
  - Video Part 2 - https://www.youtube.com/watch?v=R49Xb9eAUBA
  - Video Part 3 - https://www.youtube.com/watch?v=SZHumKEhgtA
- Additional resources
  - https://optimaltransport.github.io/