Generative Modeling

Wasserstein generative adversarial networks and friends

Denis Derkach, Artem Ryzhikov, Maxim Artemev

Laboratory for methods of big data analysis

LAMBDA • HSE

Spring 2022

In this Lecture

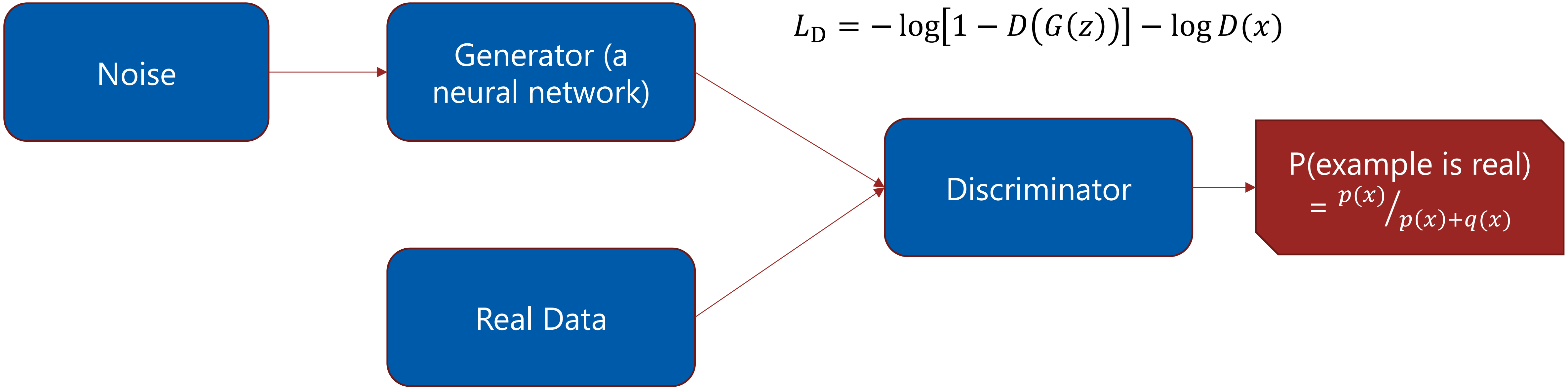
- f-GANs
- Wasserstein Distance
  - Kantorovich-Rubenstein Duality.
- WGAN
  - Algorithm.
  - Gradient Penalty.
  - Biased Gradients.

Reminder

JS GAN scheme

$$L_{\mathrm{G}} = -\log D(G(z))$$

$$L_{\mathrm{D}} = -\log\left[1 - D\left(G(z)\right)\right] - \log D(x)$$



JS GAN Summary

- Pros:
  - Can utilize power of back-prop.
  - No explicit intractable integral.
  - No MCMC needed.
- Cons:
  - Unclear stopping criteria
  - No explicit representation of PDF
  - Hard to train
  - No evaluation metric so hard to compare with other models
  - Easy to get trapped in local optima that memorize training data
  - Hard to invert generative model to get back latent z from generated x

f-GANs

Reminder: Variational Lower Bound

- For convex $f(.)$, P and Q some distributions, we define f-divergence:

$$\boxed{D_f(P||Q) = \int q(x) f\left(\frac{p(x)}{q(x)}\right) dx.}$$

- This is bounded:

$$D_f(P||Q) \geq \max_{T(x)} \mathbb{E}_{x\sim P} T(x) - \mathbb{E}_{x\sim Q} f^*\big(T(x)\big),$$

Name	$D_f(P\|Q)$	Generator $f(u)$	$T^*(x)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, dx$	$u \log u$	$1 + \log \frac{p(x)}{q(x)}$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, dx$	$-\log u$	$-\frac{q(x)}{p(x)}$
Pearson χ^2	$\int \frac{(q(x)-p(x))^2}{p(x)} \, dx$	$(u-1)^2$	$2(\frac{p(x)}{q(x)} - 1)$
Squared Hellinger	$\int \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 dx$	$(\sqrt{u} - 1)^2$	$(\sqrt{\frac{p(x)}{q(x)}} - 1) \cdot \sqrt{\frac{q(x)}{p(x)}}$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, dx$	$-(u+1) \log \frac{1+u}{2} + u \log u$	$\log \frac{2p(x)}{p(x)+q(x)}$

Variational Divergence Minimization

$$\boxed{D_f(P||Q) \geq \max_{T(x)} \mathbb{E}_{x\sim P} T(x) - \mathbb{E}_{x\sim Q} f^*\big(T(x)\big),}$$

- Work in GAN paradigm:
  - generator $x \sim Q: x = G_\theta(z)$;
  - test function $\mathrm{T}(x)$.

$$\min_\theta \max_\omega F(\omega, \theta) = \mathbb{E}_{x\sim P} T_\omega(x) - \mathbb{E}_{x\sim G_\theta(z)} f^*\big(T_\omega(x)\big)$$

- To have wider range of functions:

$$T_\omega(x) = g_f\big(V_\omega(x)\big),$$

here $g_f: \mathbb{R} \rightarrow dom_{f*}$ is output activation function for f-divergence used

S. Nowozin et al. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization

Output activation function

$$F(\omega, \theta) = \mathbb{E}_{x \sim P} g_f(V_\omega(x)) - \mathbb{E}_{x \sim G_\theta(z)} f^*\big(g_f(V_\omega(x))\big)$$

choice of output activation function is somewhat arbitrary.

Name	Output activation g_f
Kullback-Leibler (KL)	v
Reverse KL	$-\exp(-v)$
Pearson χ^2	v
Squared Hellinger	$1 - \exp(-v)$
Jensen-Shannon	$\log(2) - \log(1 + \exp(-v))$
GAN	$-\log(1 + \exp(-v))$

Example: GAN objective

$$F(\omega,\theta) = \mathbb{E}_{x\sim P} g_f(V_\omega(x)) - \mathbb{E}_{x\sim G_\theta(z)} f^*\big(g_f(V_\omega(x))\big)$$

$$g_{GAN} = -\log(1+\exp(-v))$$
$$f^*(t) = -\log(1 - \exp(t))$$

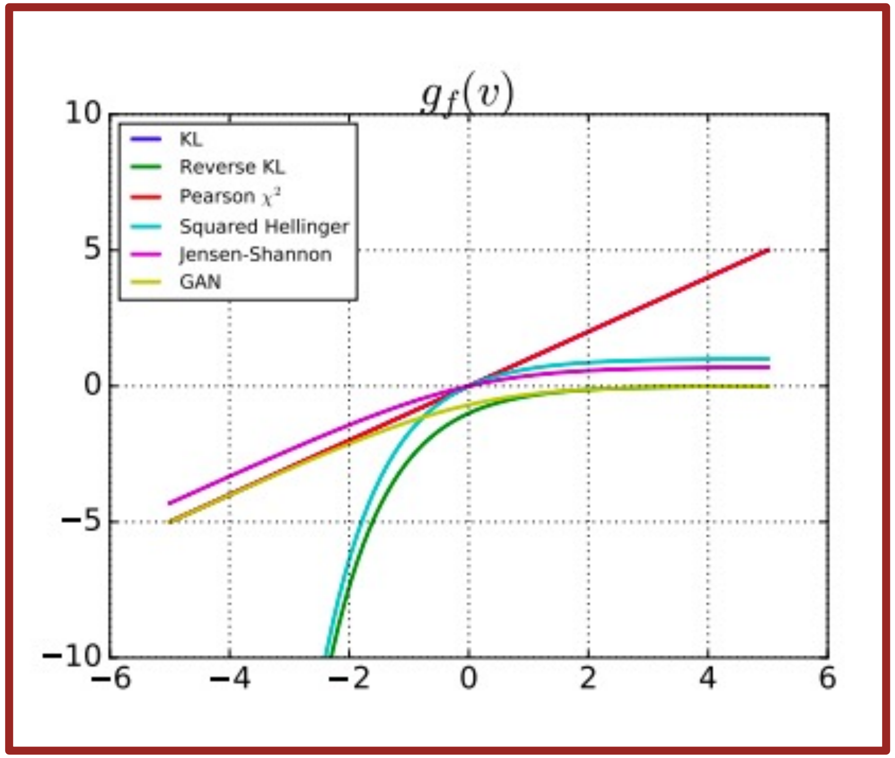
$$\boxed{F(\omega,\theta) = \mathbb{E}_{x\sim P} \log D_\omega(x) - \mathbb{E}_{x\sim G_\theta(z)} \log(1 - D_\omega(x)),}$$

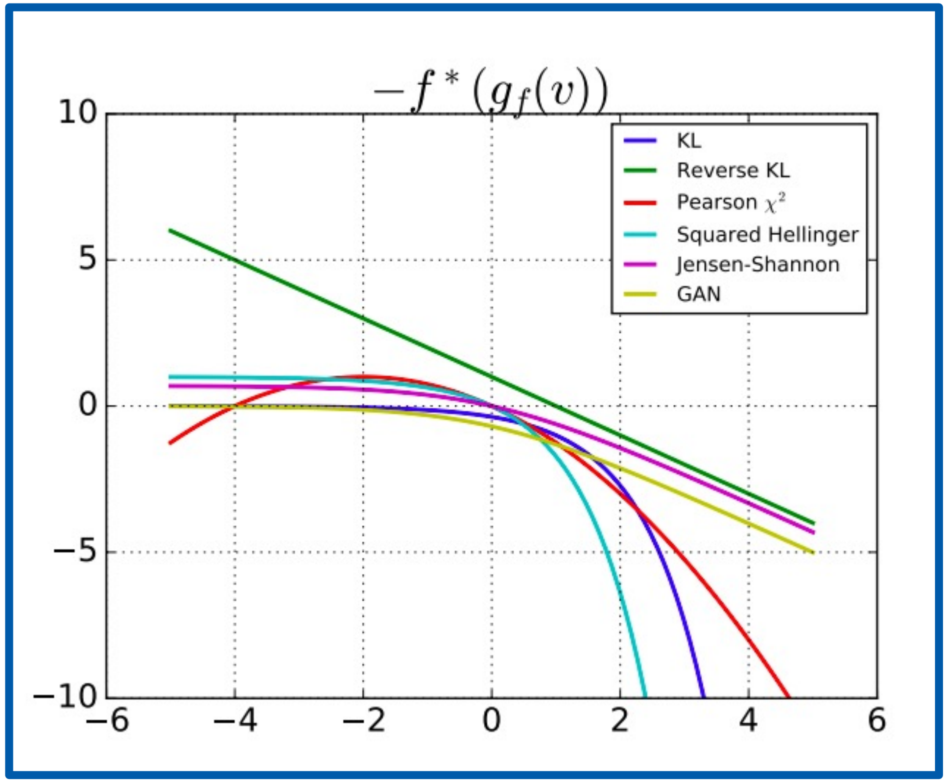
for the last nonlinearity in the discriminator taken as the sigmoid

$$D_\omega(x) = 1/(1+e^{-V_\omega(x)})$$

Variational Divergence Minimization

$$\min_{\theta} \max_{\omega} F(\omega, \theta) = \mathbb{E}_{x \sim P} g_f(V_\omega(x)) - \mathbb{E}_{x \sim G_\theta(z)} f^*\big(g_f(V_\omega(x))\big)$$





f-GAN results

Training divergence	KDE $\langle LL \rangle$ (nats)	$\pm$ SEM
Kullback-Leibler	416	5.62
Reverse Kullback-Leibler	319	8.36
Pearson χ^2	429	5.53
Neyman χ^2	300	8.33
Squared Hellinger	-708	18.1
Jeffrey	-2101	29.9
Jensen-Shannon	367	8.19
GAN	305	8.97
Variational Autoencoder [18]	445	5.36
KDE MNIST train (60k)	502	5.99

Table 4: Kernel Density Estimation evaluation on the MNIST test data set. Each KDE model is build from 16,384 samples from the learned generative model. We report the mean log-likelihood on the MNIST test set ($n = 10,000$) and the standard error of the mean. The KDE MNIST result is using 60,000 MNIST training images to fit a single KDE model.

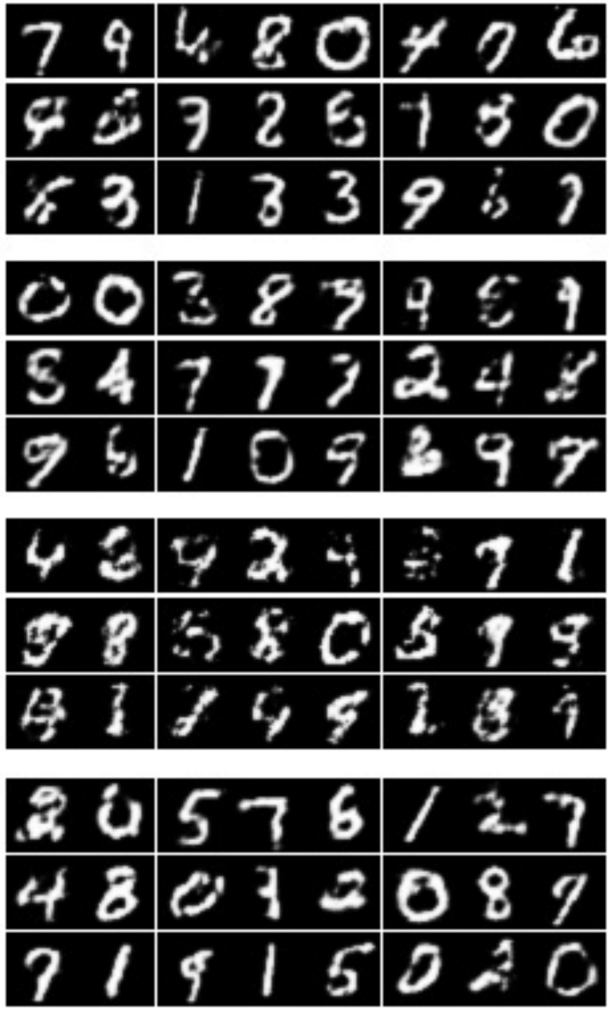


Figure 2: MNIST model samples trained using KL, reverse KL, Hellinger, Jensen from top to bottom.

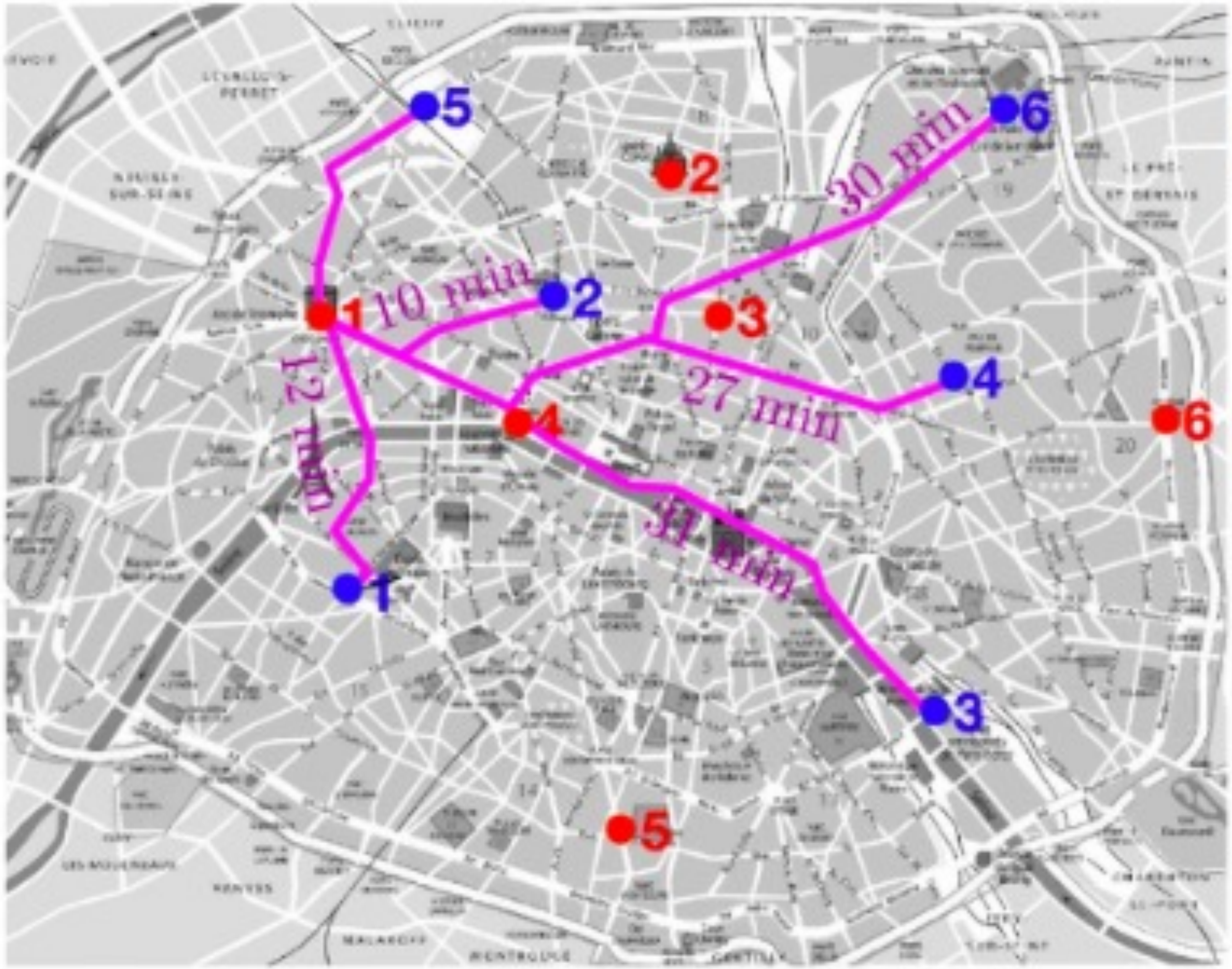
f-GAN Discussion

- Using f-GAN approach, one can estimate any f-divergence.
- Construction has some freedom in choice of function.
- Using different f-divergence leads to very different learning dynamics.
- Does not solve mode collapse problem.
- We need a better way to train GANs.

Parisian Bakeries

Problem Statement

Delivery Optimization

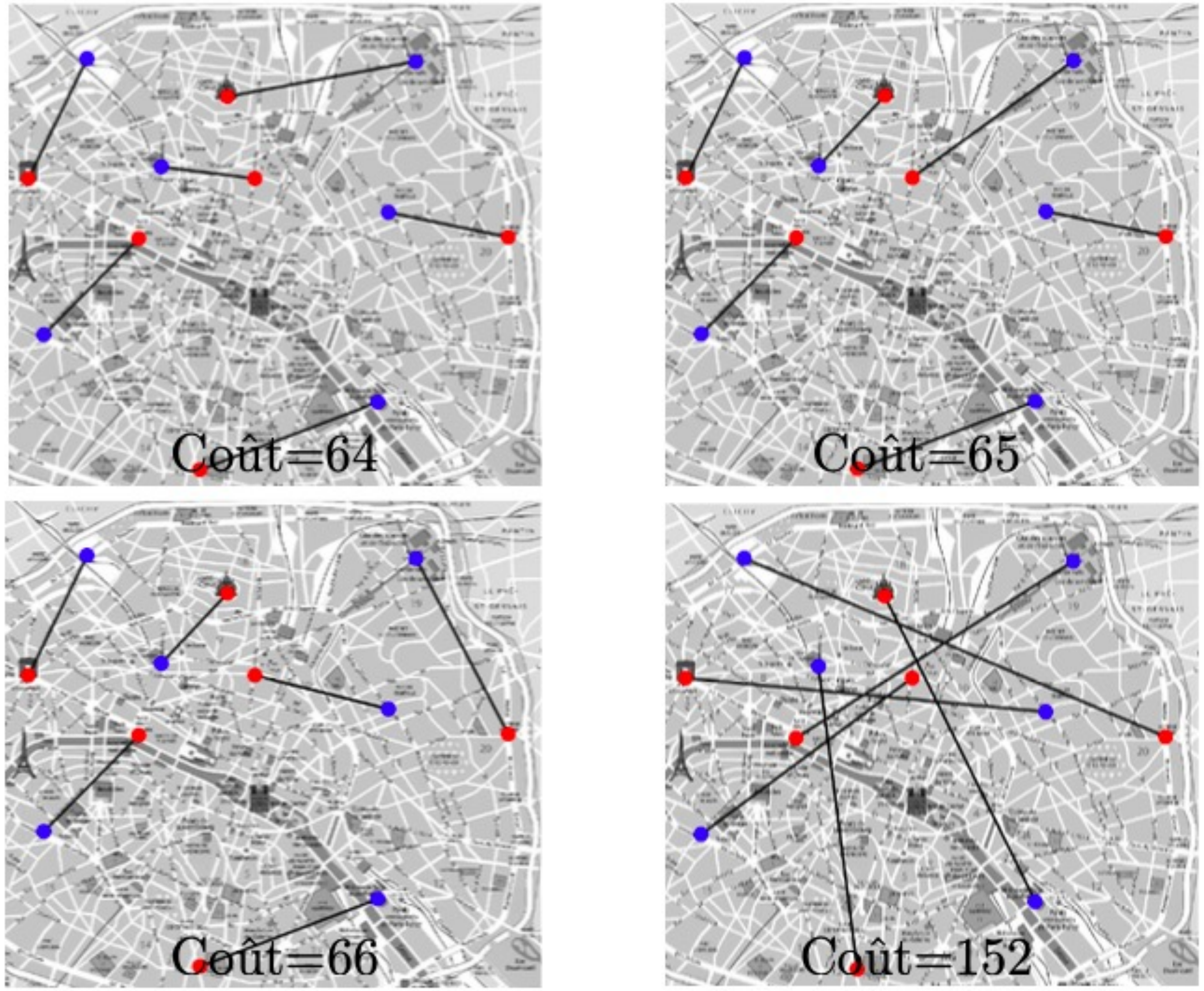


c_{ij}	y_1	y_2	y_3	y_4	y_5	y_6
x_1	12	10	31	27	10	30
x_2	22	7	25	15	11	14
x_3	19	7	19	10	15	15
x_4	10	6	21	19	14	24
x_5	15	23	14	24	31	34
x_6	35	26	16	9	34	15

given a set of N bakeries and M cafes, what is the optimal way to transport loaves of bread between them?

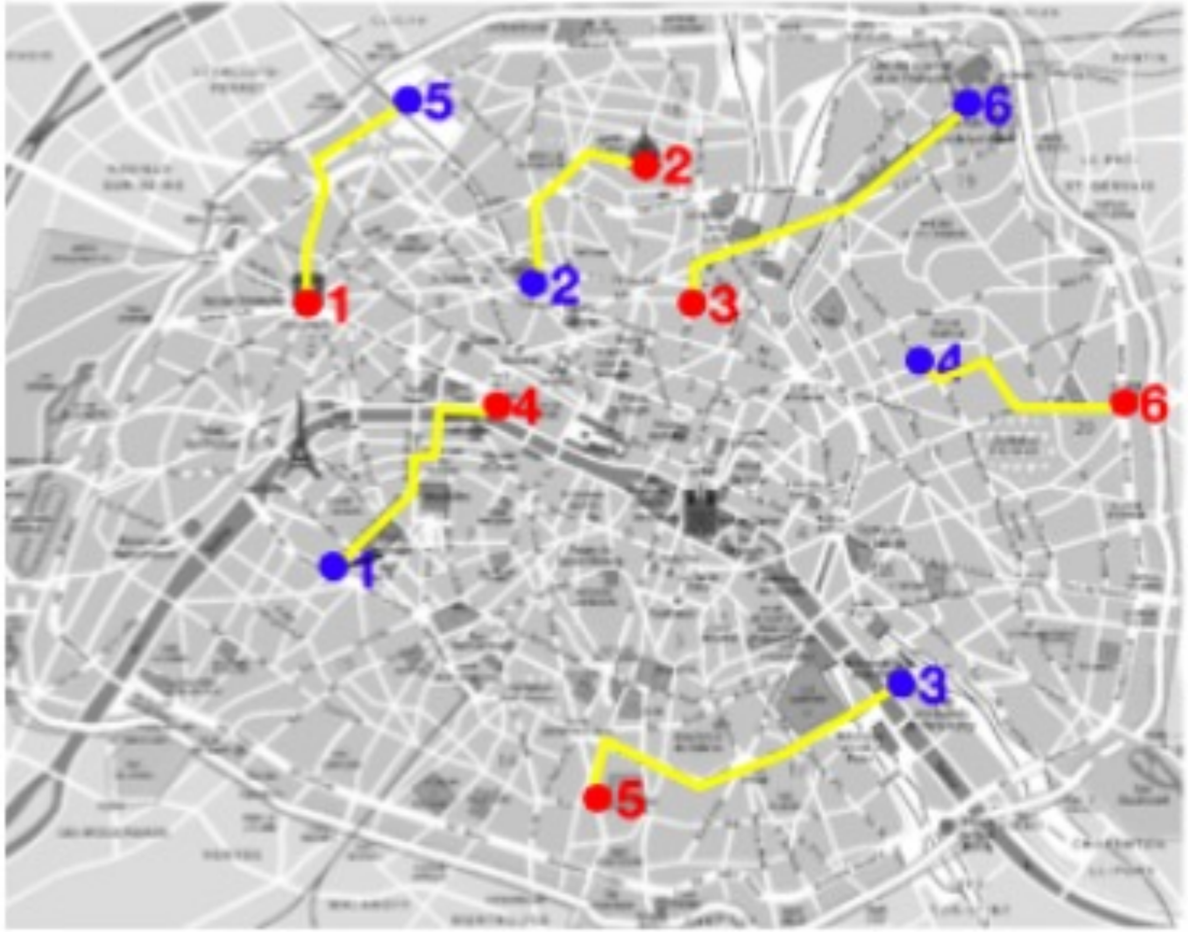
G. Peyre web-site

Delivery Solutions



We can estimate different possibilities, using the same matrix of costs.

Optimal Delivery



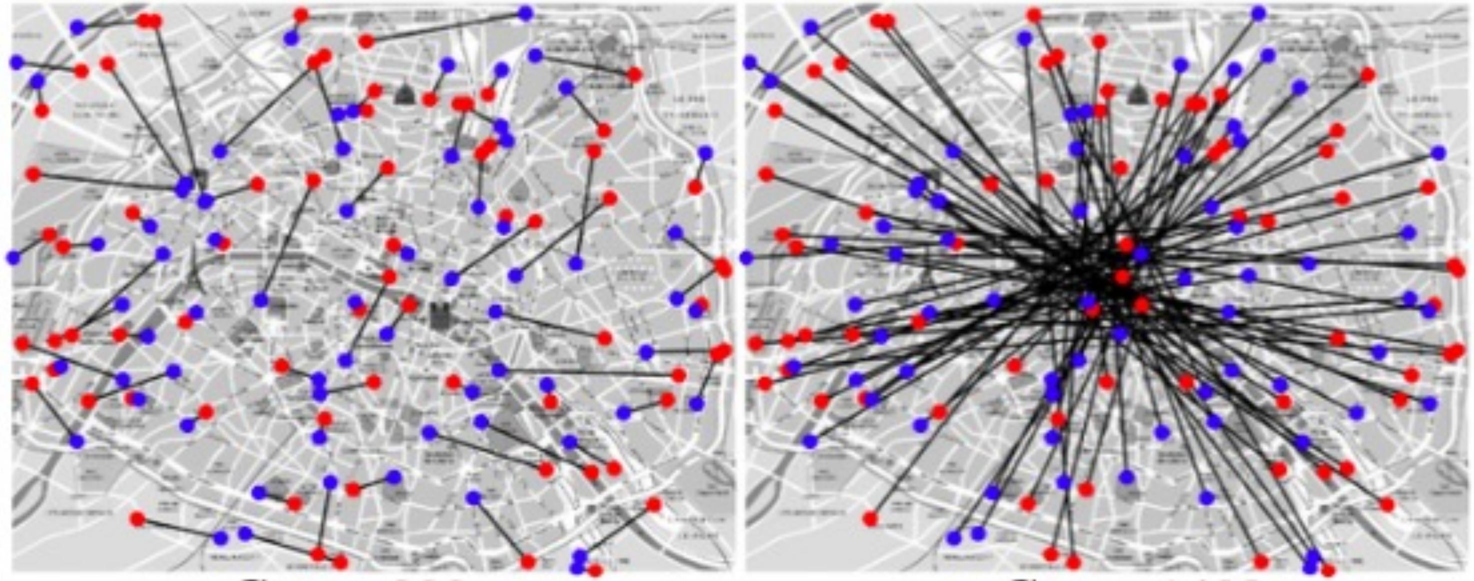
c_{ij}	y_1	y_2	y_3	y_4	y_5	y_6
x_1	12	10	31	27	10	30
x_2	22	7	25	15	11	14
x_3	19	7	19	10	15	15
x_4	10	6	21	19	14	24
x_5	15	23	14	24	31	34
x_6	35	26	16	9	34	15

Price = 10+7+15+10+14+9 = 65 min

Problem Statement: Monge

We thus need to solve a problem:

$$\min_{\sigma \in \mathrm{Perm}_n} \sum_{i=1}^{n} C_{i,\sigma(i)}$$



Cout=200 Cout=1493

The number of calculations rises as factorial.

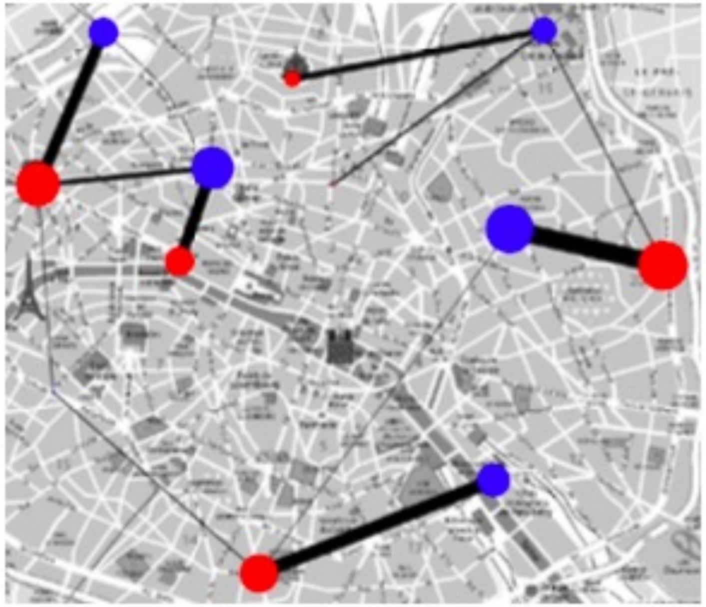
Kantorovich: Add Bread Masses

- $p_i, \ \ i \in \ 1 \dots N$ the mass of bread held by each bakery;
- $q_j, \ \ j \in \ 1 \dots M$ the mass of bread desired by each cafe;
- $\sum p_i = \sum q_j$, bread produced = bread needed
- x_i, y_j the positions of bakeries and cafes;
- $P_{i,j}$ transport of mass from i to j.
- $p_i = \sum_j P_{i,j}\,; q_j = \sum_i P_{i,j}$ limiting capacity exists.

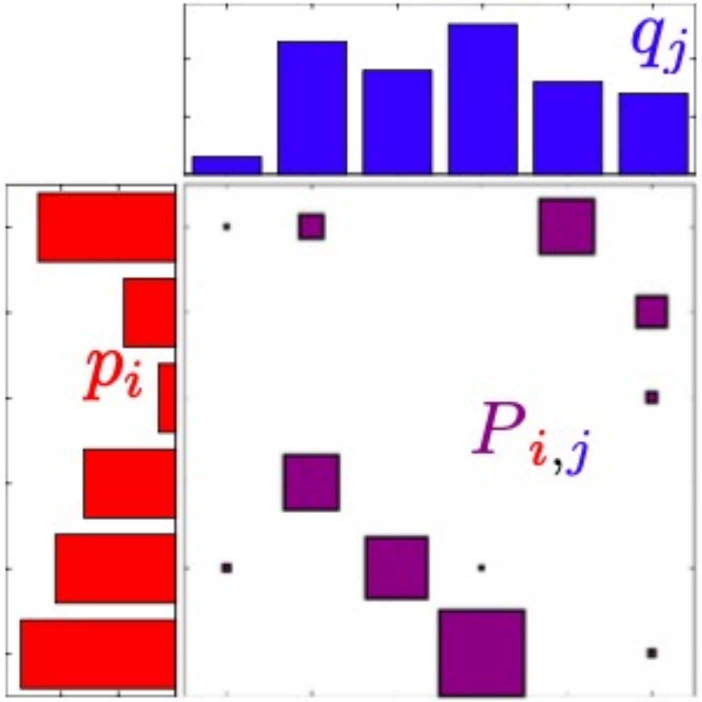
Paris deliveries

$\boxed{\mathcal{L}}\ \sum_j P_{i,j} = p_i$ $\quad$ $\boxed{\mathcal{C}}\ \sum_i P_{i,j} = q_j$

Deliver the bread to the café, which needs it



	3	23	18	26	16	14
24	1	7	0	0	16	0
9	0	0	0	0	0	9
3	0	0	0	0	0	3
16	0	16	0	0	0	0
21	2	0	18	1	0	0
27	0	0	0	25	0	2



Optimal plan for minimal efforts needed.

Kantorovich: Add Bread Masses

- $p_i, \;\; i \in \; 1 \dots N$ the mass of bread held by each bakery;
- $q_j, \;\; j \in \; 1 \dots M$ the mass of bread desired by each cafe;
- $\sum p_i = \sum q_j$, bread produced = bread needed
- x_i, y_j the positions of bakeries and cafes;
- $P_{i,j}$ transport of mass from i to j.
- $p_i = \sum_j P_{i,j}\,; q_j = \sum_i P_{i,j}$ limiting capacity exists.
- Need to solve a problem:

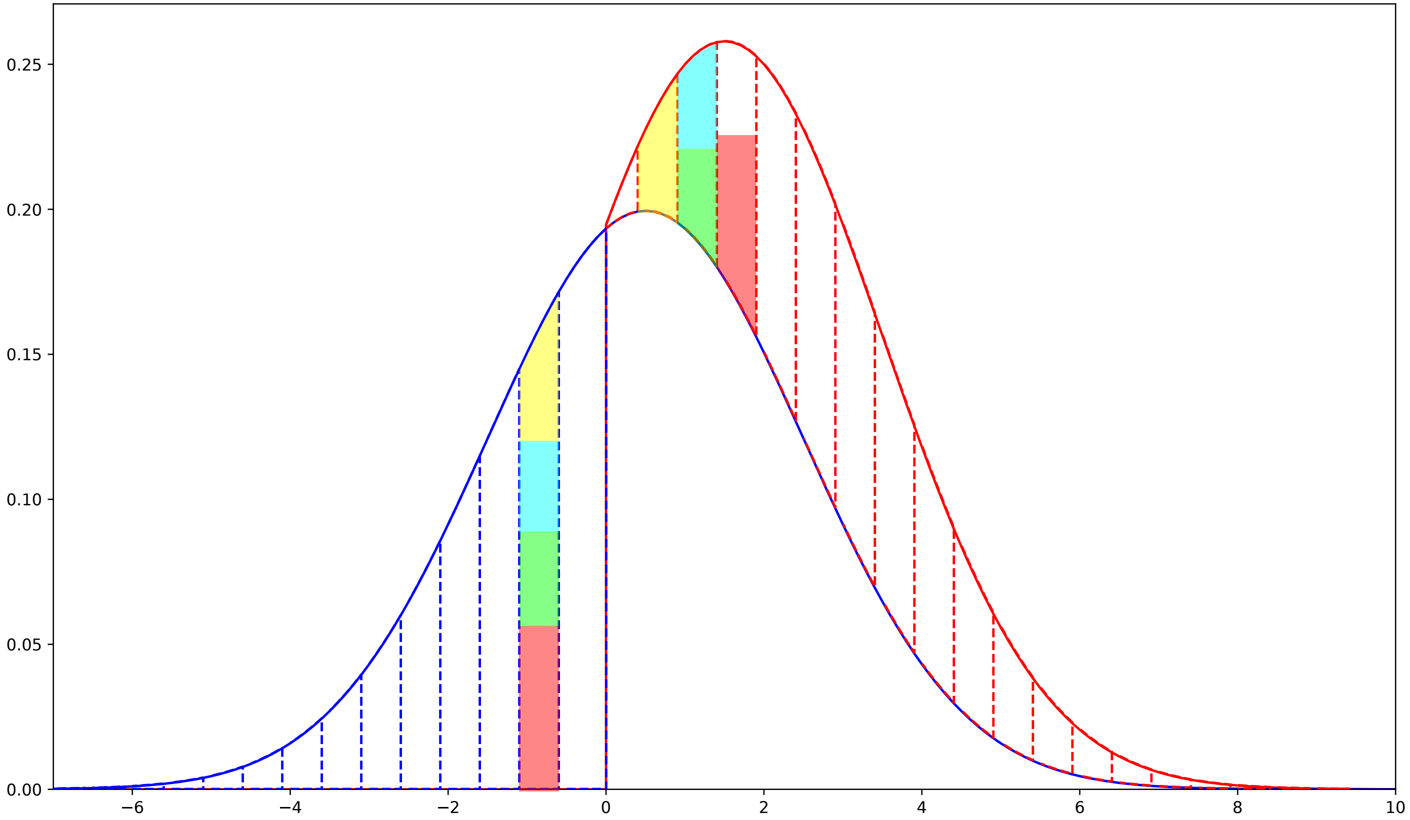
$$\min \sum P_{i,j} c_{i,j}$$

Wasserstein Distance

Wasserstein distance

Also called “Earth mover’s distance” (EMD)

- Distributions $P(x)$ and $Q(x)$ are viewed as describing the **amounts of “dirt” at point x**
- We want to convert one distribution into the other by **moving around** some amounts of dirt



- The cost of moving an amount $\boldsymbol{m}$ from $\boldsymbol{x_1}$ to $\boldsymbol{x_2}$ is $\boldsymbol{m}\times\|\boldsymbol{x_2}-\boldsymbol{x_1}\|$
- $\mathrm{EMD}(P,Q)$ = **minimum total cost** of converting P into Q

Idea of definition

- Say, we have a moving plan $\gamma(x_1, x_2) \geq 0$:

  $\gamma(x_1, x_2)dx_1 dx_2$ – how much dirt we're moving from $[x_1, x_1 + dx_1]$ to $[x_2, x_2 + dx_2]$

- Then, the cost of moving from $[x_1, x_1 + dx_1]$ to $[x_2, x_2 + dx_2]$ is:

$$\|x_2 - x_1\| \cdot \gamma(x_1, x_2)dx_1 dx_2$$

- and the total cost is:

$$C = \int_{x_1, x_2} \|x_2 - x_1\| \cdot \gamma(x_1, x_2)dx_1 dx_2 = \mathbb{E}_{x_1, x_2 \sim \gamma(x_1, x_2)} \|x_2 - x_1\|$$

- Since we want to convert P to Q, the plan has to satisfy:

$$\int_{x_1} \gamma(x_1, x_2)dx_1 = Q(x_2), \qquad \int_{x_2} \gamma(x_1, x_2)dx_2 = P(x_1)$$

Idea of Definition

- Let π be the set of all plans that convert P to Q, i.e.:

$$\pi = \left\{ \gamma: \quad \gamma \geq 0, \qquad \int_{x_1} \gamma(x_1, x_2) dx_1 = Q(x_2), \qquad \int_{x_2} \gamma(x_1, x_2) dx_2 = P(x_1) \right\}$$

- Then, the Wasserstein distance between P and Q is:

$$\mathrm{EMD}(P, Q) = \inf_{\gamma \in \pi} \mathbb{E}_{x_1, x_2 \sim \gamma} \|x_2 - x_1\|$$

Optimization over all transport plans – not too friendly

Wasserstein Distance

For continuous case, there are a set of p-Wasserstein distances, with $W_p(p_x, q_y)$ defined with $x \in M$, $y \in M$ and a distance D on x, y:

$$W_p(p_x, q_y) = \inf_{\gamma \in \Pi(x,y)} \int_{M \times M} D(x, y)^p d\gamma(x, y),$$

where $\Pi(x, y)$ is a set of all joint distributions having p_x, q_y as their marginals.

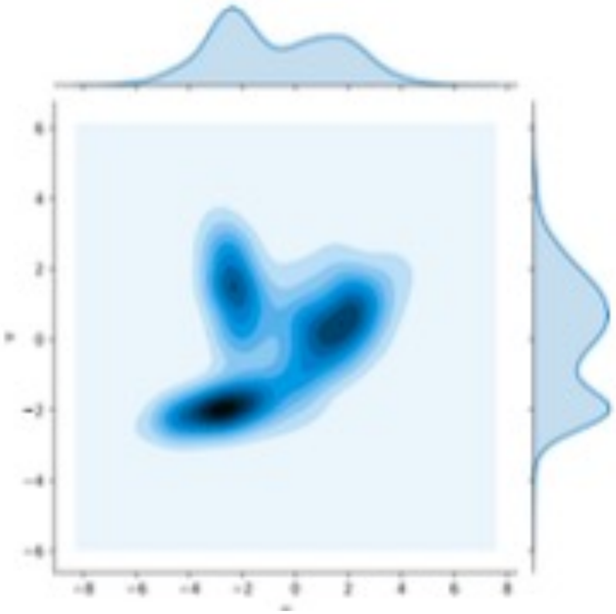
W_1 distance

In particular, W_1 distance with Euclidean norm is:

$$W(p_x, q_y) = \inf_{\gamma \in \Pi(x,y)} \int_{M\times M} D(x,y) d\gamma(x,y) = \inf_{\gamma \in \Pi(x,y)} \mathbb{E}(||x - y||)$$

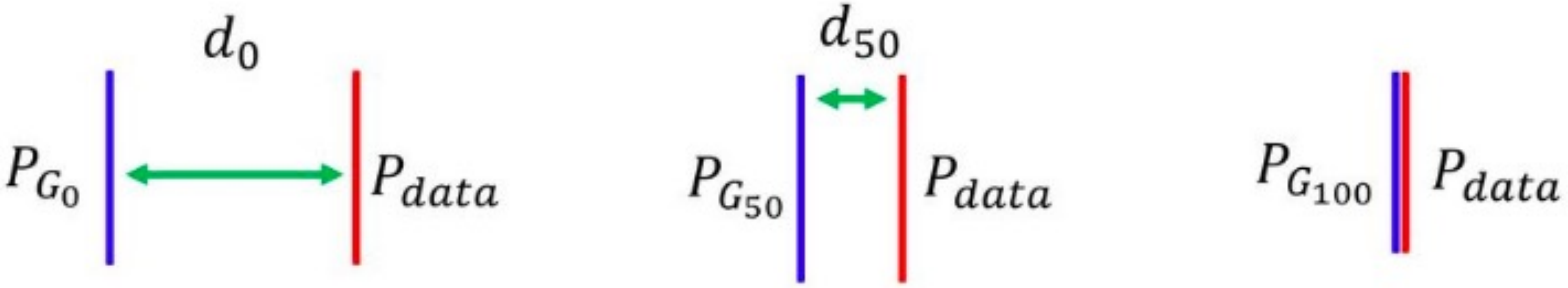
Which brings an evident connection to EMD.

Wikimedia



Two dimensional representation of the transport plan between horizontal (μ) and vertical ν pdfs. Note, that this is not unique plan. The inf must be taken over all possible plans.

Convergence Example

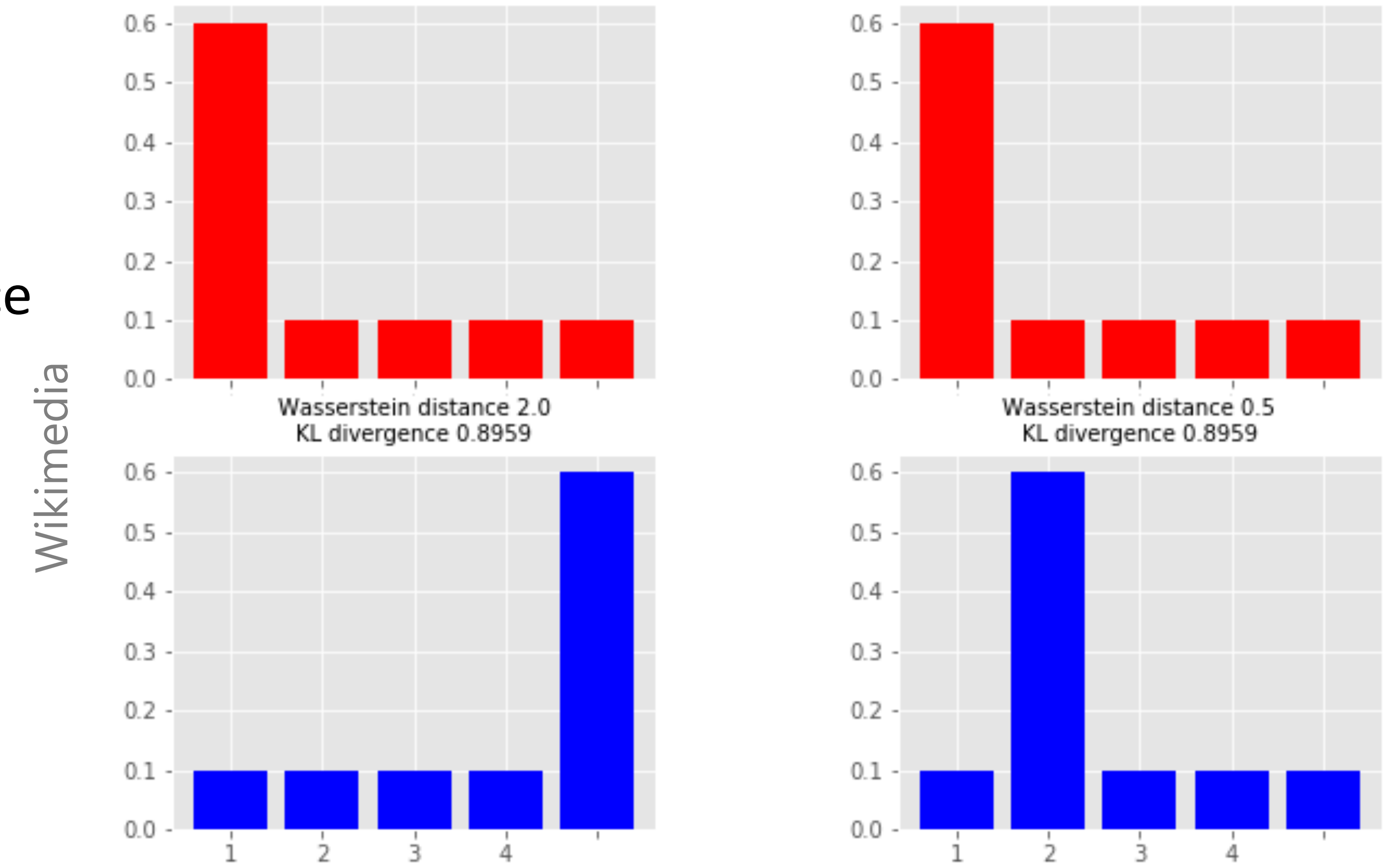


TV:	1	1	0
KL:	∞	∞	0
JS:	$\log 2$	$\log 2$	0
W:	d_0	d_{50}	0

Mass Attention

W takes into account the distance at which the differences in the distributions are located.

This is exactly what we need to take into account multiple solutions!



W properties hints

P – true PDF, Q – fitted PDF.

- For a sequence of distributions Q_n:

  $KL(P||Q_n) \to 0 \rightarrow JS(P;Q_n) \to 0 \rightarrow W(P;Q_n) \to 0,\ Q_n \xrightarrow{\mathrm{D}} P$

- For $Q_\theta \sim g_\theta(z)$, $g_\theta(z)$ continuos

  $W(Q_\theta;Q)$ is continuous and can be restricted to differentiable almost everywhere.

Should we use directly in GAN?

Reminder: Noisy Supports

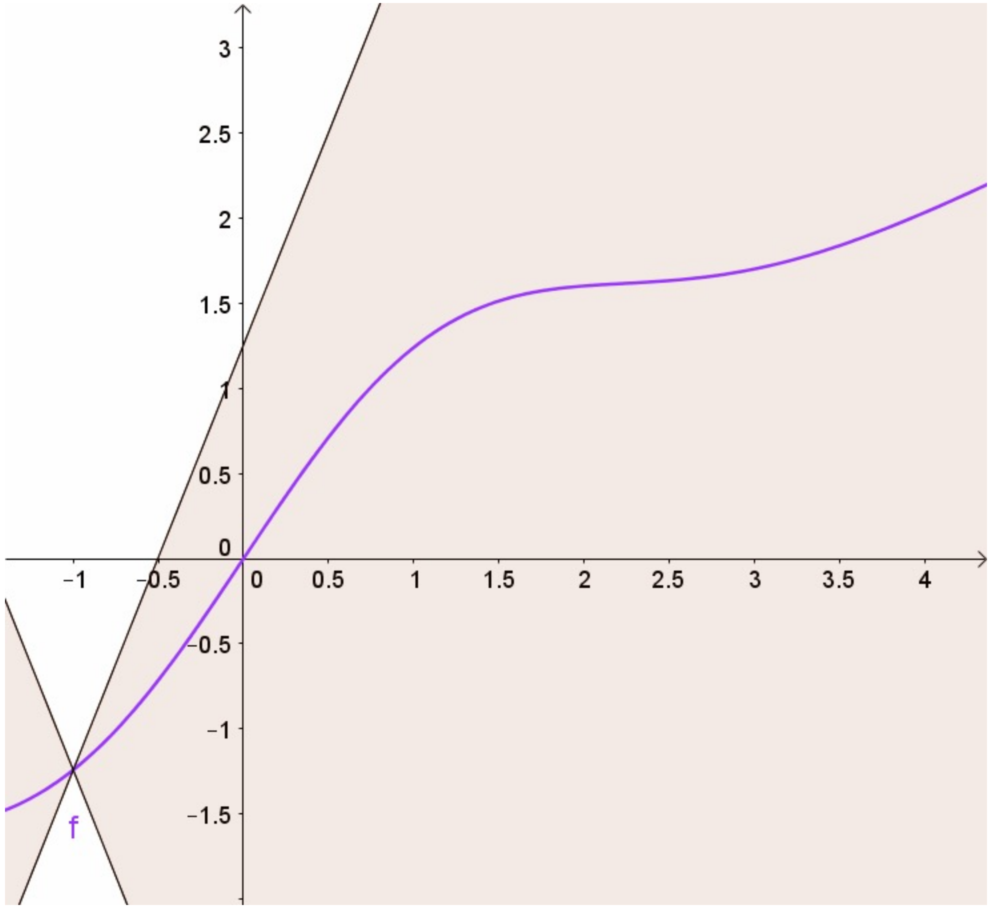
- Let's make the problem harder: introduce random noise $\varepsilon \sim N(0; \sigma^2 I)$:
$$\mathbb{P}_{x+\varepsilon(x)} = \mathbb{E}_{y\sim P(x)}\mathbb{P}_\varepsilon(x - y).$$
- This will make noisy supports, that makes it difficult for discriminator.

- For $V = \mathbb{E}||\epsilon||^2$
$$W(P,Q) \leq 2\sqrt{V} + \sqrt{JS(P,Q)}.$$
- We thus approximated W with JS

Lipschitz continuity

- f is Lipschitz-k continuous if
- there exists a constant $k \geq 0$, such that for all x_1 and x_2:

$$|f(x_1) - f(x_2)| \leq \mathrm{k} \cdot \|x_1 - x_2\|$$



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

Kantorovich-Rubinstein Duality

P – true PDF, Q – fitted PDF.

$$W(P;Q) = \sup_{f \in Lip_1} \left(\mathbb{E}_{x\sim P} f(x) - \mathbb{E}_{x\sim Q} f(x) \right),$$

where Lip_1 is 1-Lipshitz condition.

Back to Bakeries

In fact, the duality works also for EMD, we can say:

$$EMD = \sup_{||f||_L \leq 1} \sum_i f_i q_i - \sum_j f_j p_j$$

For example of bakeries, f can be interpreted as a price of buying or selling at points x_i and y_j.

Integral Probability Metrics

$p(x), q(x)$ – PDF.

$$\gamma_{\mathcal{F}}(P, Q) = \sup\left\{\left|\int f\, dp(x) - \int f\, dq(x)\right| : f \in \mathcal{F}\right\}$$

$\mathcal{F}$ is a class of real-valued bounded measurable functions on S.

For $\mathcal{F} = \{f : ||f||_L \leq 1\}$, with 1-Lipschitz condition:

W_1 is **IPM** but **not f-divergence**

B. Sriperumbudur et al. On the empirical estimation of integral probability metrics
S. Nowozin, NIPS2016 workshop talk

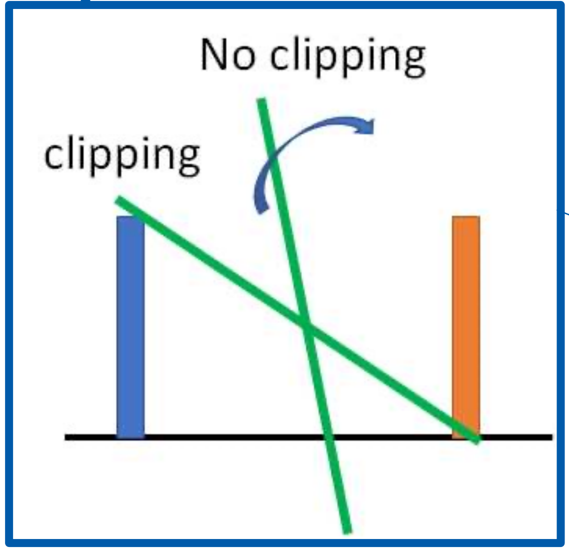
Conclusions

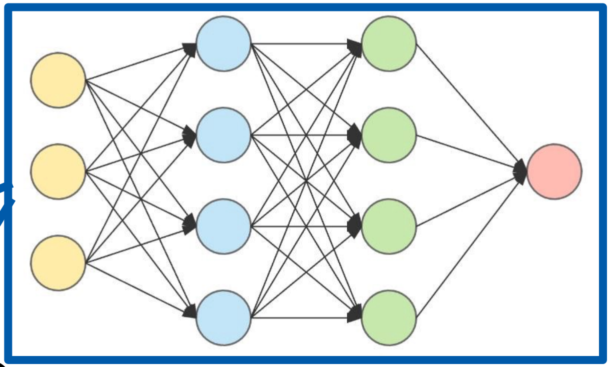
- Wasserstein-1 is a distance with desired properties.
- Kantorovich-Rubinstein duality connects Wasserstein-1 distance to IPM.
- Lipschitzeness is needed for above to work.
- Wasserstein-1 distance cannot directly be inserted into f-GAN style*.

*J. Song et al., Bridging the Gap Between f-GANs and Wasserstein GANs, ICML 2020

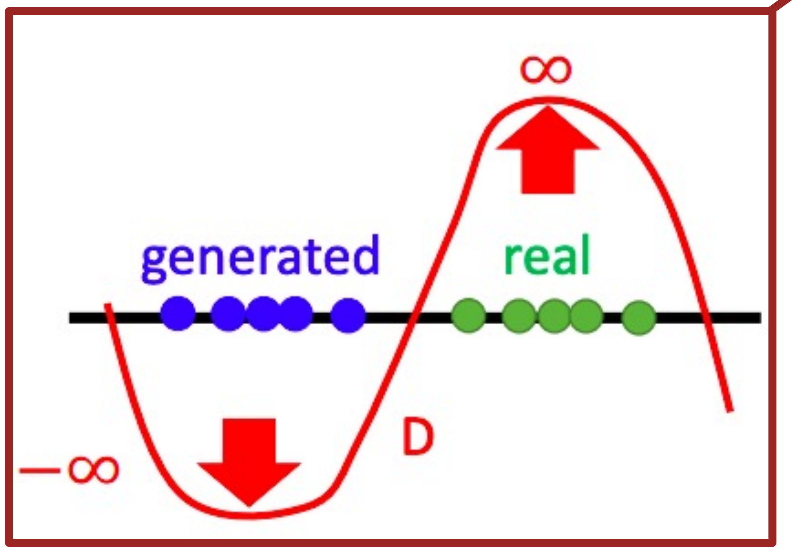
Wasserstein GAN

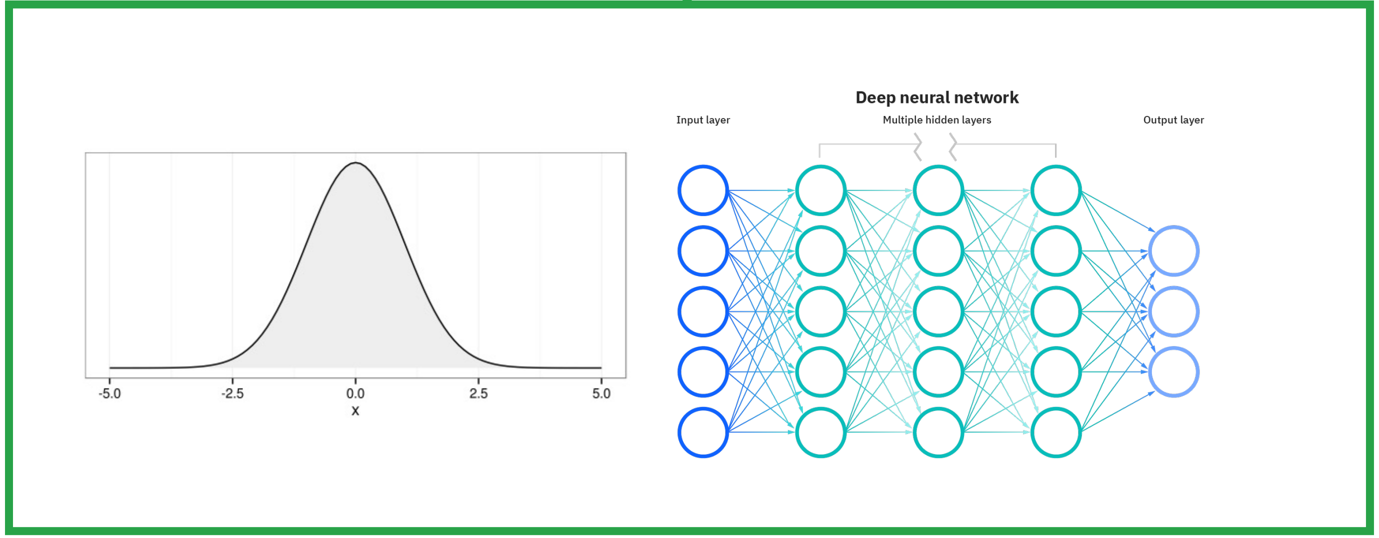
Lipschitz-1 Condition and Neural Networks





$$W(P;Q) = \sup_{f \in Lip_1} \left(\mathbb{E}_{x\sim P} f(x) - \mathbb{E}_{x\sim Q} f(x) \right),$$





Lipschitz-1 Condition and Neural Networks

$$W(P;Q) = \sup_{f \in Lip_1} \left(\mathbb{E}_{x\sim P} f(x) - \mathbb{E}_{x\sim Q} f(x) \right),$$

- f is a neural net – **discriminator** (‘**critic**’ in the original paper).
- The expectations is estimated from samples.
- Lipschitz-1 continuity can be replaced with Lipschitz-k continuity
  - estimate $k \times \mathrm{W}(P,Q)$
  - achieved **by clipping the weights** of the critic: $w \to \mathrm{clip}(w, -c, c)$ with some constant c.

M. Arjovsky et al. **Wasserstein GAN**

WGAN

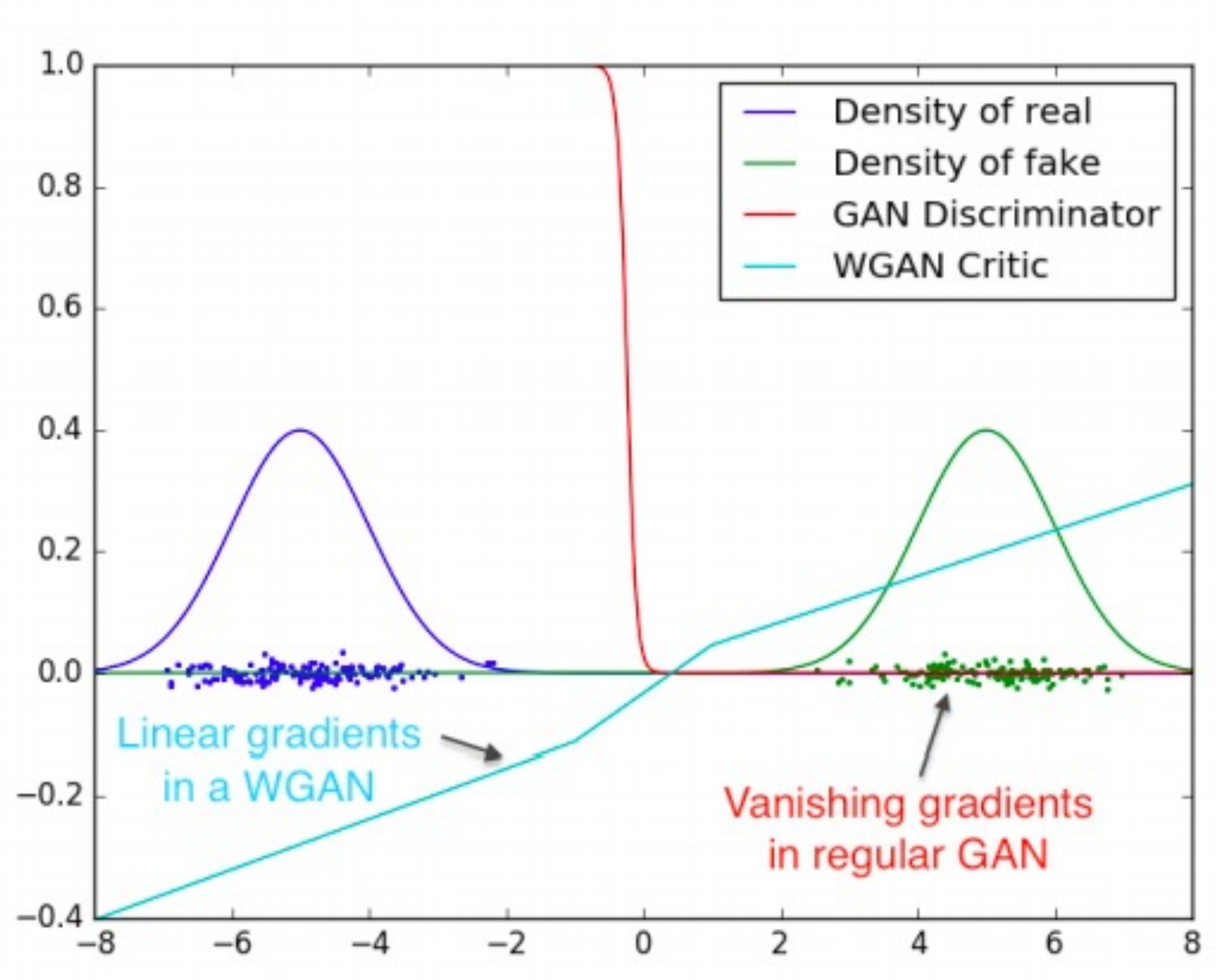
Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

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

WGAN: problems solved

- the vanishing gradient problem is **solved**;
- mode collapse problem is **addressed;**
- rom authors: Weight clipping is a clearly terrible way to enforce a Lipschitz constraint. :



https://arxiv.org/abs/1701.07875

WGAN: results

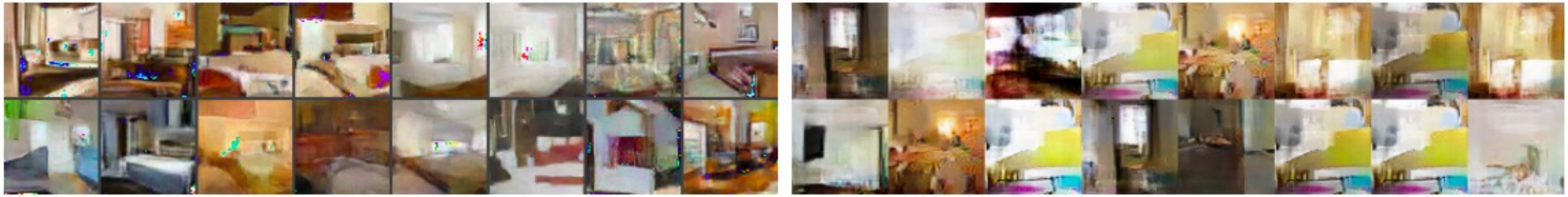


Figure 7: Algorithms trained with an MLP generator with 4 layers and 512 units with ReLU nonlinearities. The number of parameters is similar to that of a DCGAN, but it lacks a strong inductive bias for image generation. Left: WGAN algorithm. Right: standard GAN formulation. The WGAN method still was able to produce samples, lower quality than the DCGAN, and of higher quality than the MLP of the standard GAN. Note the significant degree of mode collapse in the GAN MLP.

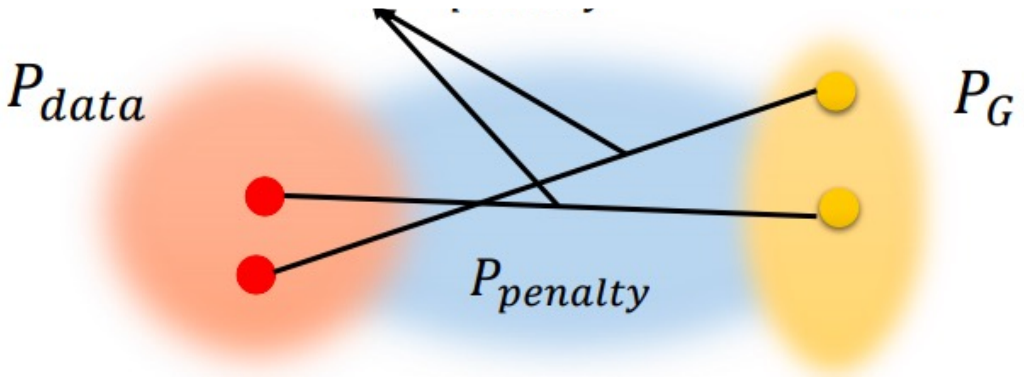
WGAN-GP

- Weight clipping makes the critic less expressive and the training harder to converge
- Optimal f should satisfy $\|\nabla f\| = 1$ almost everywhere under P and Q
- Also: $\|f\|_L \leq 1 \iff \|\nabla f\| \leq 1$
- Can replace weight clipping with a gradient penalty term:

$$\text{GP} = \lambda \int \max[(\|\nabla_{\tilde{x}} f(\tilde{x})\| - 1)^2]\, dx$$

$$\text{GP} = \lambda \mathbb{E}_{\tilde{x} \sim \mathbb{P}_{\tilde{x}}}[(\|\nabla_{\tilde{x}} f(\tilde{x})\| - 1)^2]$$

Largest gradient constrained here



$$\mathbb{P}_{\tilde{x}} : \begin{bmatrix} \tilde{x} = \alpha x_1 + (1-\alpha) x_2 \\ \alpha \sim \text{Uniform}(0,1) \\ x_1 \sim P \\ x_2 \sim Q \end{bmatrix}$$

Ishaan Gulrajani **Improved Training of Wasserstein GANs**

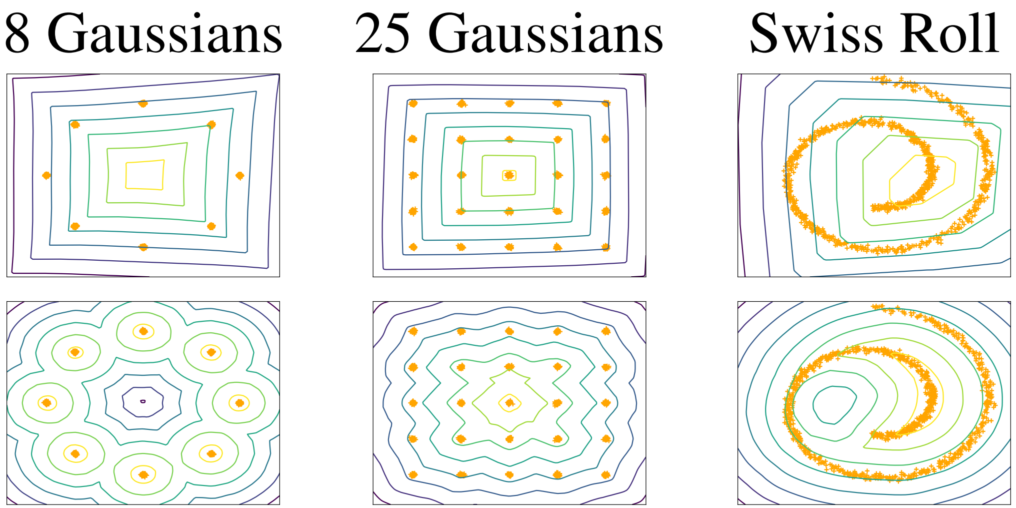
WGAN-GP

- Weight clipping makes the critic less expressive and the training harder to converge
- Optimal f should satisfy $\|\nabla f\| = 1$ almost everywhere under P and Q
- Also: $\|f\|_L \leq 1 \iff \|\nabla f\| \leq 1$
- Can replace weight clipping with a gradient penalty term:

$$\text{GP} = \lambda \mathbb{E}_{\tilde{x}\sim\mathbb{P}_{\tilde{x}}}[(\|\nabla_{\tilde{x}} f(\tilde{x})\| - 1)^2]$$

or alternatively ('one-sided' penalty):

$$\text{GP} = \lambda \mathbb{E}_{\tilde{x}\sim\mathbb{P}_{\tilde{x}}}[\max(0, \|\nabla_{\tilde{x}} f(\tilde{x})\| - 1)^2]$$



$$\mathbb{P}_{\tilde{x}} : \begin{bmatrix} \tilde{x} = \alpha x_1 + (1-\alpha) x_2 \\ \alpha \sim \text{Uniform}(0,1) \\ x_1 \sim P \\ x_2 \sim Q \end{bmatrix}$$

Ishaan Gulrajani **Improved Training of Wasserstein GANs**

WGAN: spectral normalization

- Spectral normalisation proposes to use normalised weights:

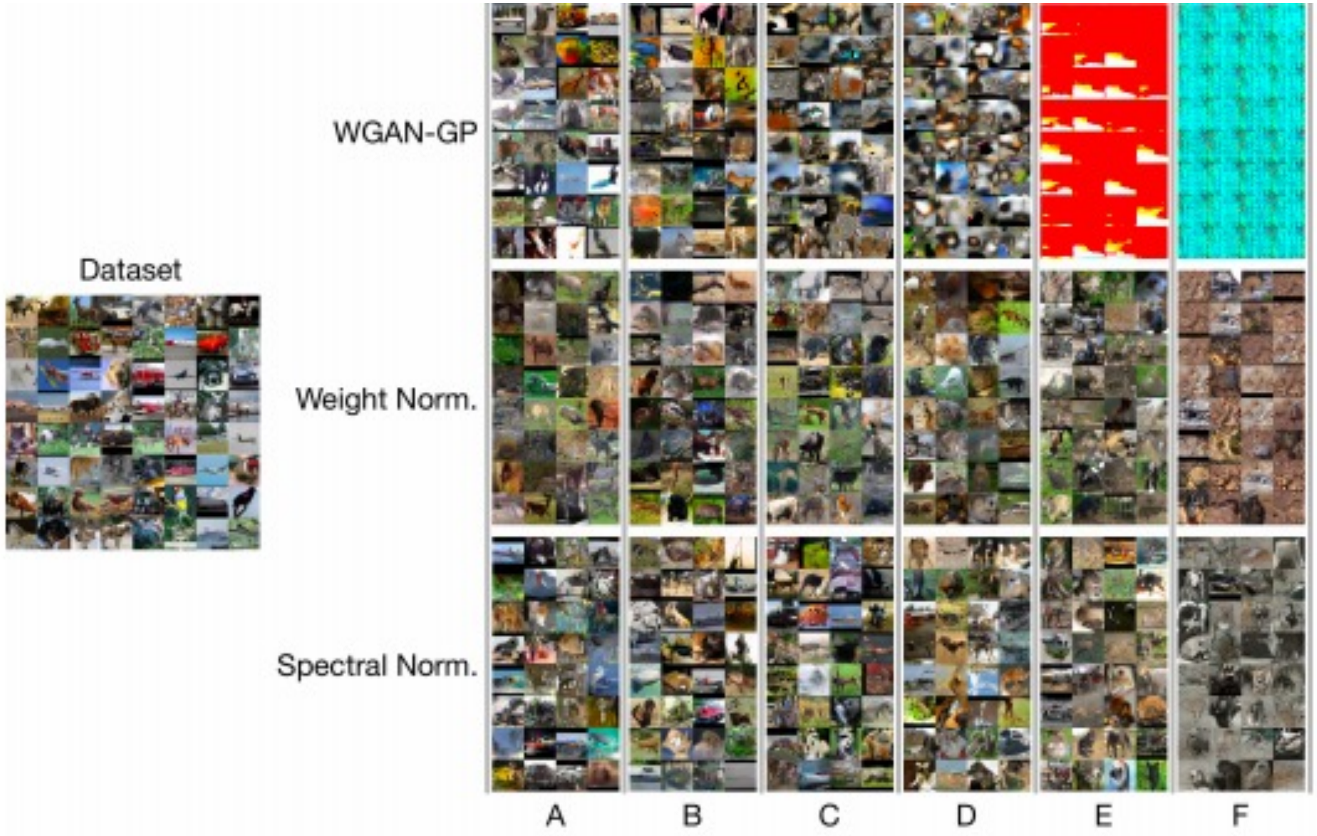
$$W_{SN} = \frac{W}{\sigma(W)}$$

where:

$$\sigma(W) = \max_{h:h\neq 0} \frac{||Wh||_2}{||h||_2}$$

- this gives constraints on gradient:

$$||f||_{Lip} \leq \prod_{i=1}^{l} \sigma(W_l).$$



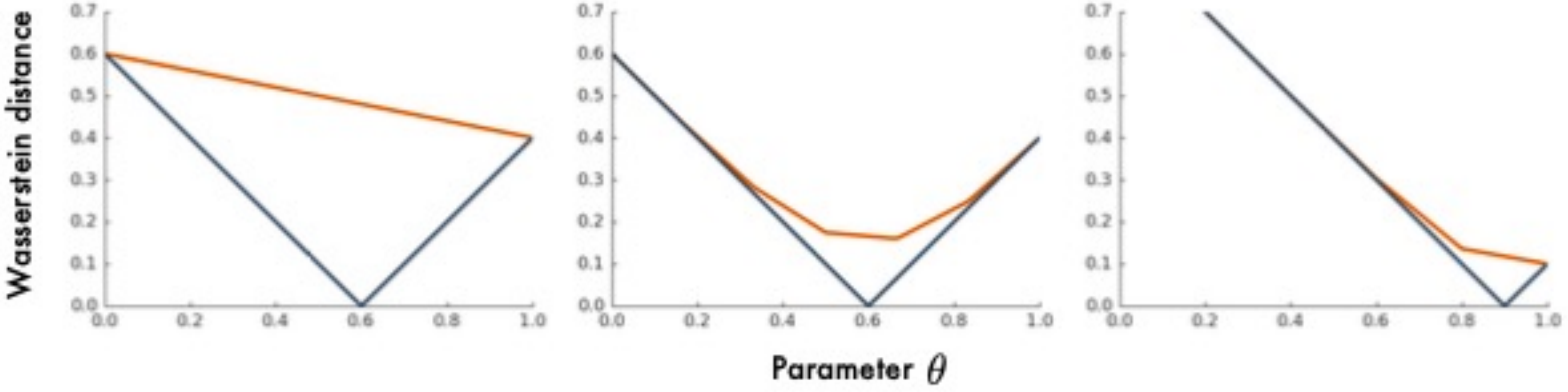
(b) STL-10

Figure 6: Generated images on different methods: WGAN-GP, weight normalization, and spectral normalization on CIFAR-10 and STL-10.

Miyato et al. Spectral Normalization for Generative Adversarial Networks, ICLR 2018

WGAN: problems

- The expected EMD gradients can differ from the true gradients.
- This leads to problems even for Bernoulli distribution.



Red for sample gradient expectation, blue is for real gradients solution. Left to right $\theta^* = 0.6; 0.6; 0.9.$

M. Bellemare et al. The Cramer Distance as a Solution to Biased Wasserstein Gradients

Conclusions

- WGAN is a power generative model.
- Simpler training procedure but need to control Lipschitz continuity
- Several ideas how do this.
- Still problems:
  - Kantorovich-Rubinstein duality only mimicked;
  - gradient is stuck near solutions.