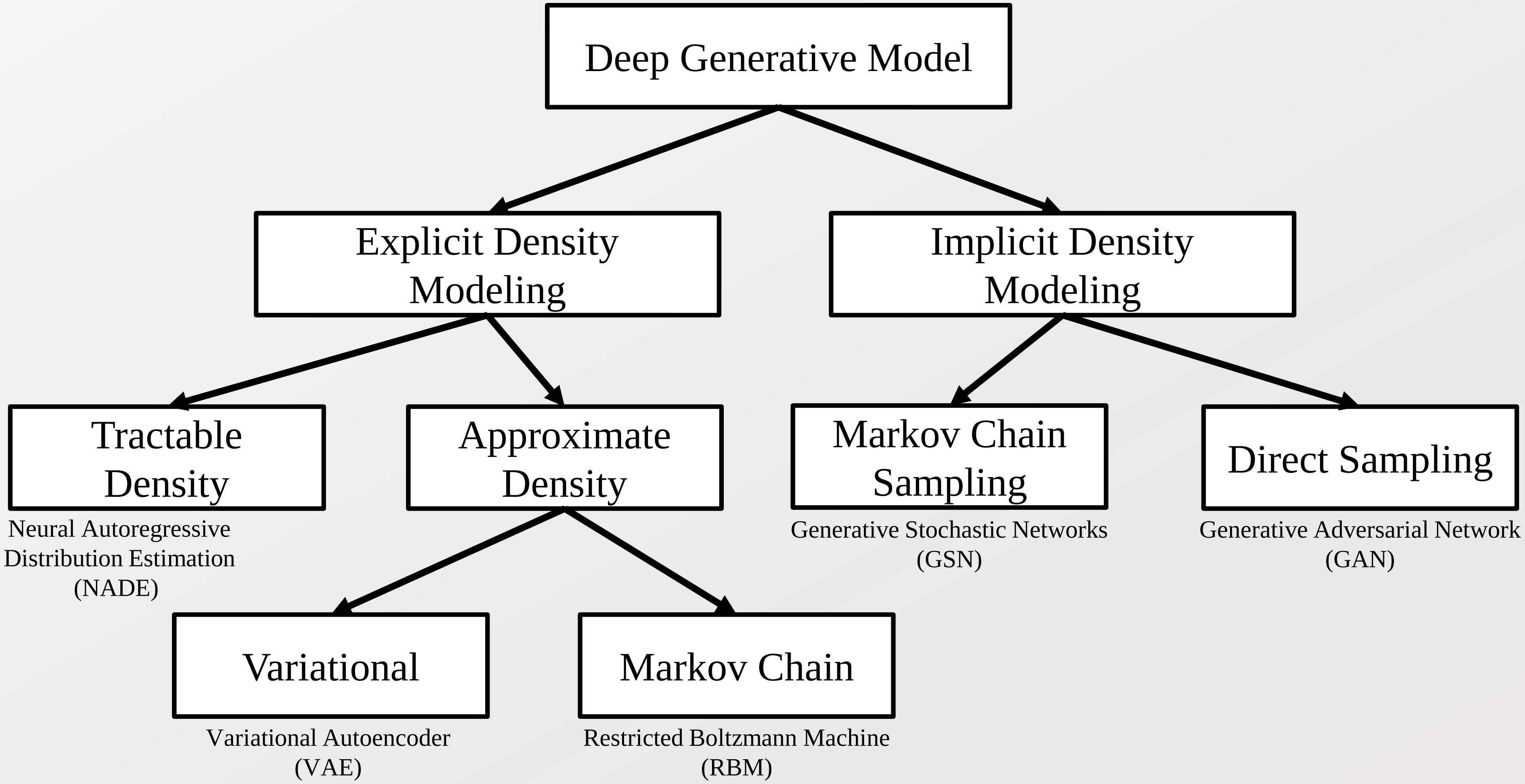
Implicit Deep Generative Model

Il-Chul Moon
Department of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Implicit Density Modeling

Taxonomy of Deep Generative Model

- **Deep Learning + Generative Modeling**
  - Why model a problem in a generative approach?
  - Good learning requires a generation of previous and new examples.



Detour: Variational Inference and Implicit Models

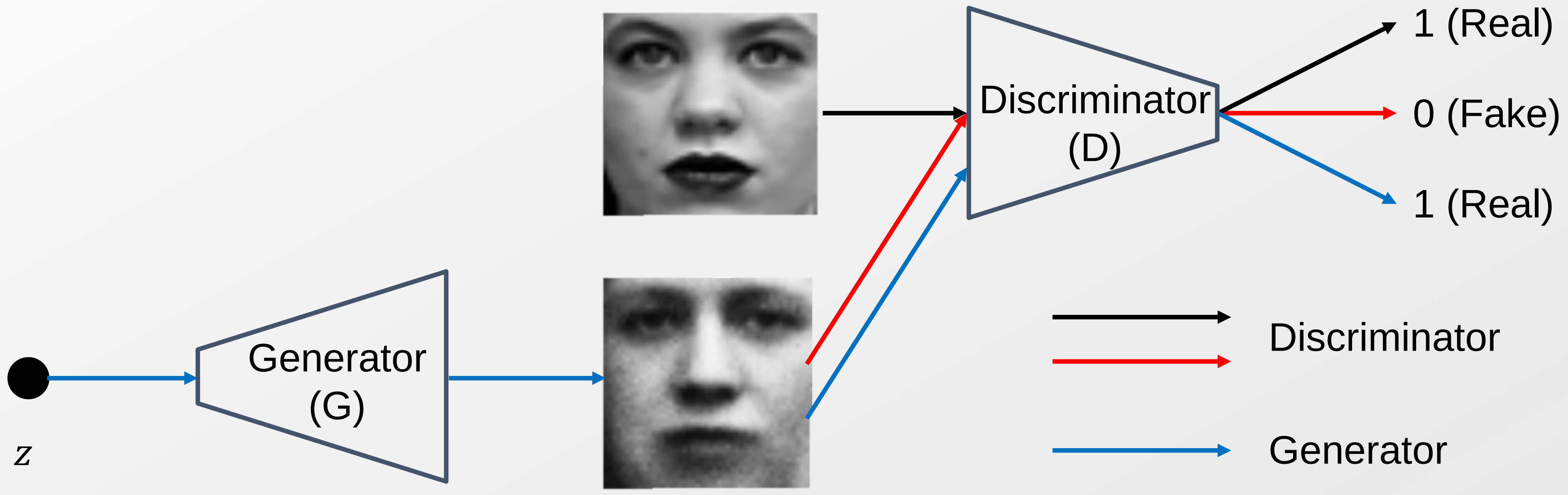
- Traditional VI requires conjugacy and tractable likelihood.
  - VAE resort the conjugacy issue by forming the inference networks for variational distribution.
    - VAE still requires an explicit likelihood function.

$$q^*(\mathbf{z}) = \underset{q \in Q}{\operatorname{argmin}}\, KL(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$$

$$q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{n=1}^{N} q(z_n; \lambda_n = \mathrm{NN}(x_n; \mathrm{NN}(x_n|\phi));\ \mathrm{q}(.) = \mathrm{Normal}$$

- What if we combine the methods of “learning in implicit models” with VI?
  - We can use an implicit form of q
    - More expressive than explicit forms
  - We also can use an implicit form of p
    - GAN, simulator...
  - Of course we can use both p and q in an implicit form.

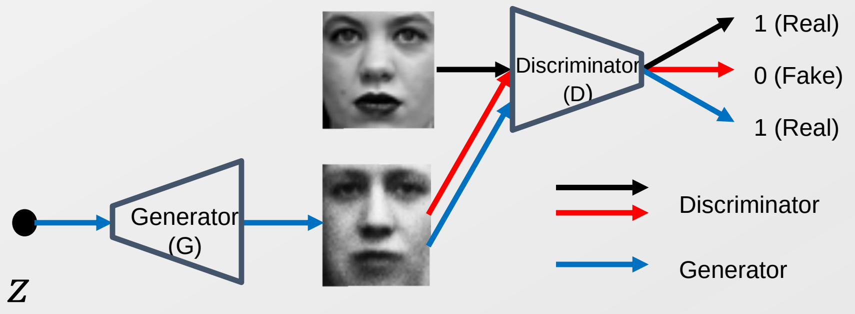
Generative Adversarial Network



- True image generation from a generator model
  - Generator is not able to distinguish the true image
  - Discriminator identifies the true or the generated images
    - Feedback enables the learning
- A discriminator model identifies the true or the fake image
  - True image is gathered from the dataset
  - Fake image is generated from the generator

Notations of GAN

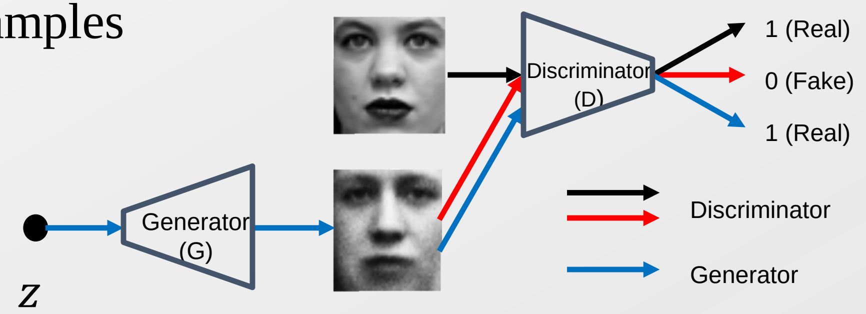
- $p_z(z)$: prior distribution on the input noise variables
  - $p_z(z) \sim N(0,1)$
- $p_{data}(x)$: data distribution over x
- $p_g(x)$: distribution of the sample $G(z)$ obtained when $z \sim p_z$



- $G(z;\theta_g)$: Generator
  - Differentiable function represented by a MLP with parameter θ_g
  - Mapping a input noise variables to the data space, **X**
- $D(x;\theta_d)$: Discriminator
  - Probability that x came from the data rather than p_g
    - $D(x;\theta_d) = 1 \Rightarrow$ the data x comes from real data
    - $D(x;\theta_d) = 0 \Rightarrow$ the data x comes from the generator
  - Outputs a single scalar

Formalization of GAN

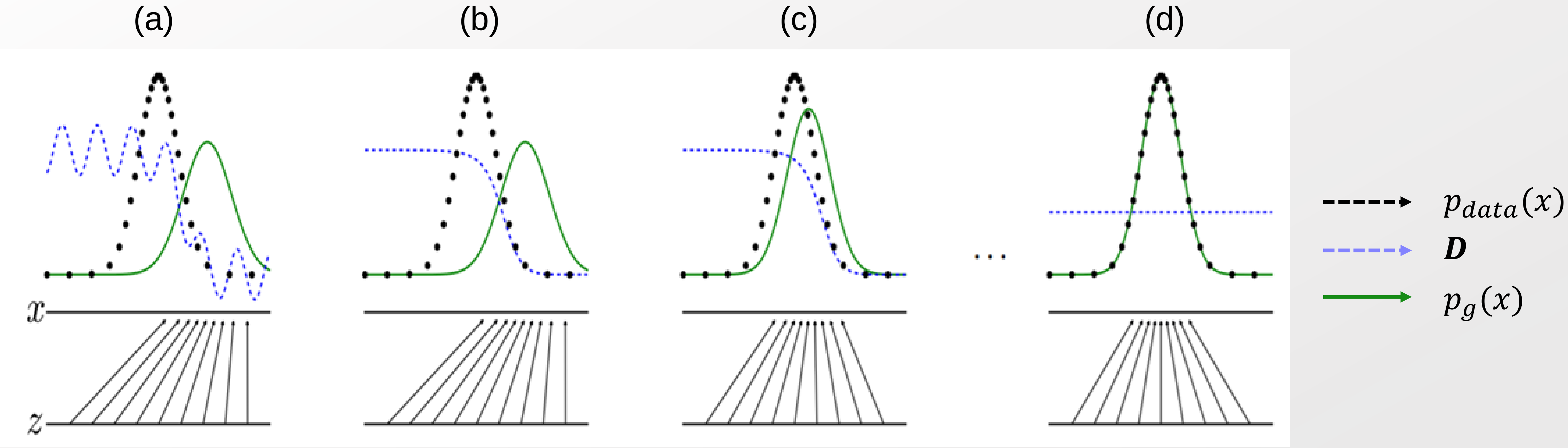
- For training Example x from the real world
  - Maximize the probability of assigning the correct label to training examples
  - For training examples, D should assign "True" label: $D(x) = 1$
  - Maximize $E_{x \sim p_{data}(x)}[logD(x)]$ w.r.t. D



- **Objective Function**
  - Binary case becomes the Bernoulli trial and the cross entropy
  - $V(D, G) = E_{x \sim p_{data(x)}}[logD(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z))]$
  - $\min_G \max_D V(D, G)$

- For input noise z
  - G maps z to data space, X, as close as possible
  - G should minimize $E_{z \sim p_z(z)}[\log(1 - D\big(G(x)\big))]$
    - Minimize $\log(1 - D\big(G(x)\big))$
    - Maximize $D(G(x)) \Rightarrow$ Ideal case : $D(G(x)) = 1$: Fool the Discriminator
  - D should maximize $E_{z \sim p_z(z)}[\log(1 - D\big(G(x)\big))]$
    - Minimize $D(G(x))$ $\Rightarrow$ Ideal case : $D(G(x)) = 0$

Analogy of GAN



- Loop
  - (a) Sample z from uniform dist. and $G(z) = x$
  - (b) ***D*** is trained to discriminate samples from data
  - (c) ***G*** is updated to fool the ***D***
  - (d) ***D*** cannot discriminate at all ($D(x) = D\big(G(z)\big) = 0.5$)

Loss Function of GAN

- Objective Function of GAN
  - $V(D,G) = E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}[log(1-D(G(z)))]$

$$= \sum_x p_{data}(x)\ln\frac{P_{data}(x)}{P_g(x)+P_{data}(x)} + \sum_x p_g(x)\,ln\{1-\frac{P_{data}(x)}{P_g(x)+P_{data}(x)}\}$$

$$= \sum_x p_{data}(x)\,ln\frac{P_{data}(x)}{P_g(x)+P_{data}(x)} + \sum_x p_g(x)\,ln\{\frac{P_g(x)}{P_g(x)+P_{data}(x)}\}$$

$$= \sum_x p_{data}(x)\,ln\frac{P_{data}(x)}{2\times\frac{P_g(x)+P_{data}(x)}{2}} + \sum_x p_g(x)\,ln\frac{P_g(x)}{2\times\frac{P_g(x)+P_{data}(x)}{2}}$$

$$= \sum_x p_{data}(x)\,ln\frac{P_{data}(x)}{\frac{P_g(x)+P_{data}(x)}{2}} + \sum_x p_g(x)\,ln\frac{P_g(x)}{\frac{P_g(x)+P_{data}(x)}{2}} - ln\,2\sum_x p_{data}(x) - ln\,2\sum_x p_g(x)$$

$$= KL(P_{data}(x)|\left|\frac{P_g(x)+P_{data}(x)}{2}\right) + KL(P_g(x)|\left|\frac{P_g(x)+P_{data}(x)}{2}\right) - 2\ln 2$$

$$= 2JS(P_g||P_{data}) - \ln 4$$

$$KL(P||Q) = \sum_i P(i)\ln\left(\frac{P(i)}{Q(i)}\right)$$

$$z\sim p_z(z) \rightarrow x = G(z) \rightarrow x\sim P_g(x)$$

- Definition of **Jensen-Shannon Divergence**
  - $JS(P_g||P_{data}) = \frac{1}{2}KL(P_g(x)|\left|\frac{P_g(x)+P_{data}(x)}{2}\right) + \frac{1}{2}KL(P_{data}(x)|\left|\frac{P_g(x)+P_{data}(x)}{2}\right)$

Jensen Shannon Divergence

- $JS(P||Q) = \frac{1}{2}KL(P|| \frac{Q+P}{2}) + \frac{1}{2}KL(Q|| \frac{Q+P}{2})$
  - $0 \leq JS(P||Q) \leq \ln 2$
  - $JS(P||Q) = 0$, if and only if $P = Q$
  - $JS(P||Q) = JS(Q||P)$

- Close relation to the information theory
  - Assume X an abstract function on the events, or a mixture distribution, M, with a mode selection of Z; and with two mode components of P and Q
    - X samples from P distribution if Z=0
    - X samples from Q distribution if Z=1
    - The mode proportion between Z=0 and Z=1 is uniform
    - $X \sim M = \frac{P+Q}{2}$
  - $I(X;Z) = H(X) - H(X|Z) = -\sum MlogM + \frac{1}{2}[\sum PlogP + \sum QlogQ]$

$$= -\sum \frac{P+Q}{2} logM + \frac{1}{2}[\sum PlogP + \sum QlogQ]$$
$$= -\sum \frac{P}{2} logM - \sum \frac{Q}{2} logM + \frac{1}{2}[\sum PlogP + \sum QlogQ]$$
$$= \frac{1}{2}\sum P(logP - logM) + \frac{1}{2}\sum Q(logQ - logM)$$
$$= \frac{1}{2}\sum Plog\frac{P}{M} + \frac{1}{2}\sum Qlog\frac{Q}{M} = \frac{1}{2}KL(P||M) + \frac{1}{2}KL(Q||M) = JS(P||Q).$$

Training of GAN

- $min_{\theta_g} max_{\theta_d} V(D, G; \theta_g, \theta_d)$
  $= min_{\theta_g} max_{\theta_d} E_{x \sim p_{data(x)}}[logD(x; \theta_d)] + E_{z \sim p_{z(z)}}[log(1 - D(G(z; \theta_g); \theta_d)]$
  - Two sets of parameter θ_g and θ_d
  - Alternative gradient learning
    - Gradient ascent
      - $\theta_d^* = argmax_{\theta_d} E_{x \sim p_{data(x)}}[logD(x; \theta_d)] + E_{z \sim p_{z(z)}}[log(1 - D(G(z; \theta_g); \theta_d)]$
    - Gradient descent
      - $\theta_g^* = argmin_{\theta_g} E_{z \sim p_{z(z)}}[log(1 - D(G(z; \theta_g); \theta_d)]$

- Because the learning on θ_d requires the output from $G(z; \theta_g)$; and the learning on θ_g requires the feedback from $D(G(z; \theta_g); \theta_d)$
  - The simultaneous learning of θ_d and θ_g is infeasible
  - This suggest that the training is not a simultaneous game, rather a sequential game
    - Imagine a rock-paper-scissors game with a player who plays first
      - The other player who plays second has a deterministic result

Theoretical Results of GAN

- $V(D,G) = E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}[\log(1 - D(G(z))]$
  - $\min_G \max_D V(D,G)$
  - $\mathrm{C(G)} = \max_D V(D,G)$

- There exists the global minimum and its meaning
  - The global minimum of the virtual training criterion $\mathrm{C(G)}$ is achieved
    - if and only if $p_g = p_{data}$.
    - At that point, $\mathrm{C(G)}$ achieves the value –log 4.

  ⇒ For optimal D (fixed), global minimum is achieved iff $\boldsymbol{p_g = p_{data}}$

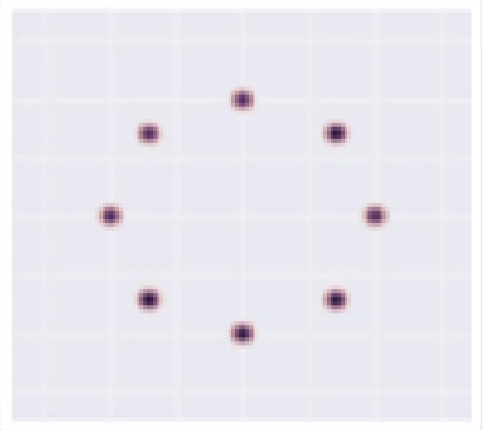
- There exists the convergence path to the global optimum
  - p_g converges to p_{data}
    - If G and D have enough capacity,
    - And at each step, the discriminator is allowed to reach its optimum given G
    - And, p_g is updated so as to improve the criterion $V(D^*, G)$,

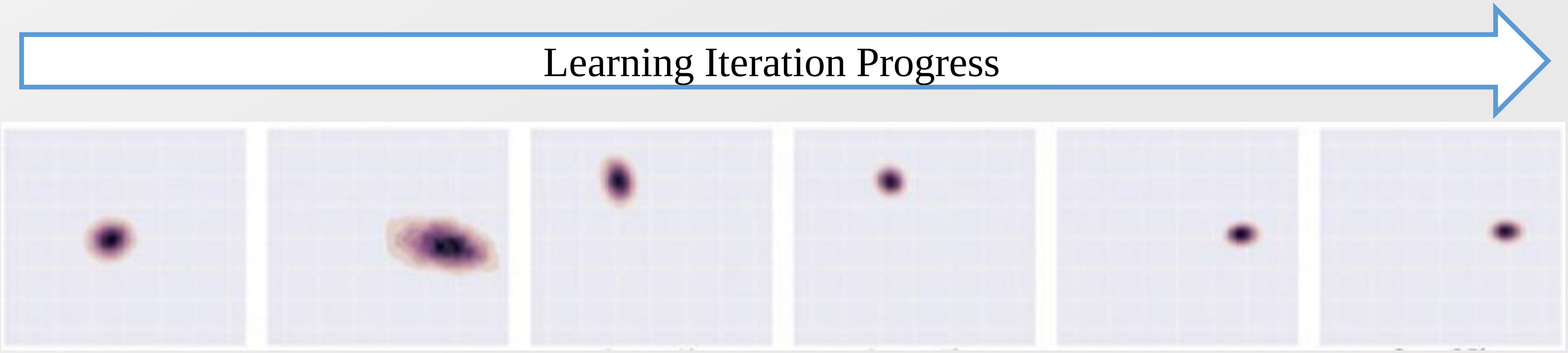
  ⇒ For optimal D (fixed), $V(D^*, G)$ converges to global minimum

  ⇒ For optimal D (fixed), p_g converges to p_{data}

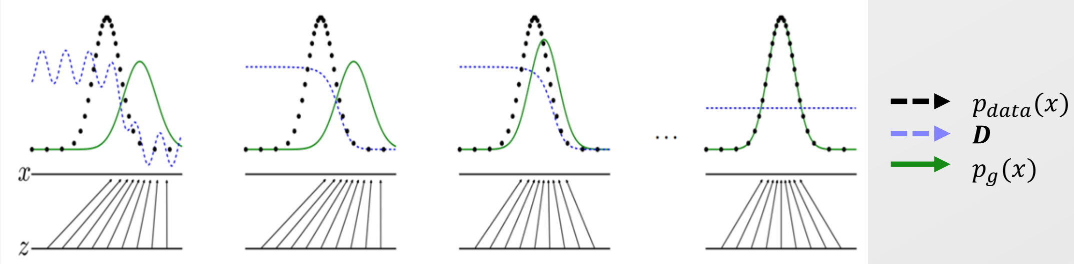
Mode Collapse

- The objective of $G(z)$ is $\min E_{z\sim p_{z(z)}}[\log(1-D(G(z))]$
  - $\min E_{z\sim p_{z(z)}}[\log(1-D(G(z))] \rightarrow \max E_{z\sim p_{z(z)}}[\log D(G(z))]$
  - In the generator perspective, the below is the potential possibility
    - $G(z) = x^*$ such that $x^* = argmax_x D(x)$
    - Here, x^* becomes a fixed output regardless of $z\sim p_{z(z)}$ sampling
      - Producing the most realistic image, yet a single fixed image regardless of the latent
      - Fixed discriminator → fixed optimal x^*





Unrolling Discriminator Learning

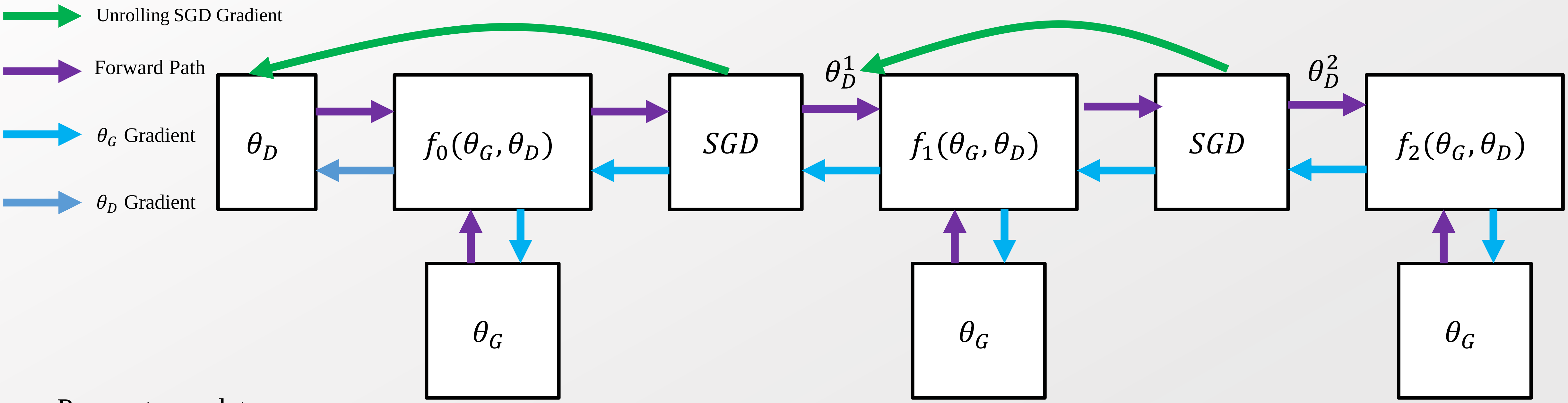


- Ordinary GAN
  - $\theta_G^* = argmin_{\theta_G} \max_{\theta_D} f(\theta_G, \theta_D) = argmin_{\theta_G} f(\theta_G, \theta_D^*(\theta_G))$
  - $\theta_D^*(\theta_G) = arg \max_{\theta_D} f(\theta_G, \theta_D)$
  - Optimal point : $\theta^* = \{\theta_G^*, \theta_D^*\}$
    - Multiple problem in reaching to the optimal points
      - $f(\theta_G, \theta_D)$ may not be a simple convex or concave function → Local optimum
      - Alteranating gradient approach → Depending on the gradient descent, the learning could be infeasible from the time perspective

- Considering the optimization of the generator, can we provide a good discriminator close to $\theta_D^*(\theta_G)$?
  - $\theta_G^* = argmin_{\theta_G} f(\theta_G, \theta_D^*(\theta_G))$
    - But, $\theta_D^*(\theta_G)$ is unreachable
    - Then, Let's approximate the learning
      - $\theta_D^0 = \theta_D$
      - $\theta_D^{k+1} = \theta_D^k + \eta^k \frac{df(\theta_G, \theta_D^k)}{d\theta_D^k}$
      - $\theta_D^*(\theta_G) = \lim_{k \to \infty} \theta_D^k$

- Surrogate of $f(\theta_G, \theta_D^*(\theta_G))$: $f_K(\theta_G, \theta_D) = f(\theta_G, \theta_D^K(\theta_G, \theta_D))$
  - if $k = 0$, $f_K(\theta_G, \theta_D) = f(\theta_G, \theta_D)$
  - if $k \to \infty$, $f_K(\theta_G, \theta_D) = f(\theta_G, \theta_D^*(\theta_G))$
    - under the condition that $\theta_D^*(\theta_G)$ can be reached via the gradient method

Unrolled GAN



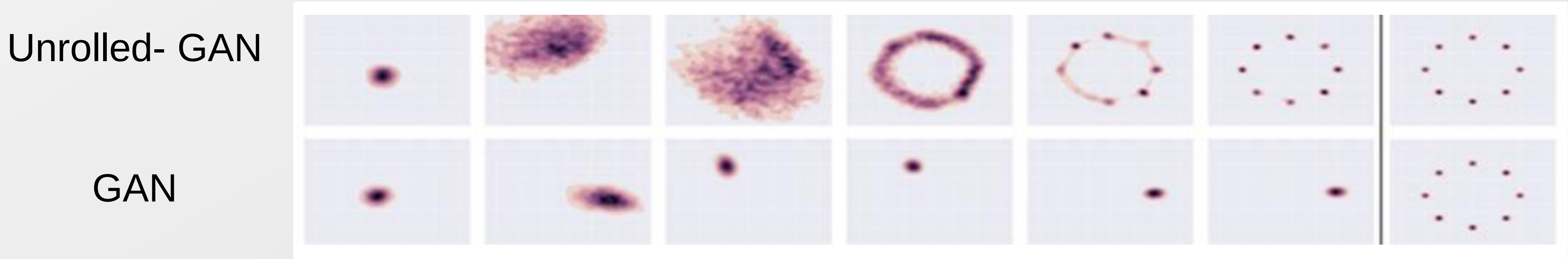
- Parameter update
  - $\theta_G \leftarrow \theta_G - \eta \frac{df_K(\theta_G,\theta_D)}{d\theta_G}$, $\theta_D \leftarrow \theta_D - \eta \frac{df(\theta_G,\theta_D)}{d\theta_D}$
- Effect on the generator learning
  - $\frac{df_K(\theta_G,\theta_D)}{d\theta_G} = \frac{\partial f(\theta_G,\theta_D^K(\theta_G,\theta_D))}{\partial \theta_G} + \frac{\partial f(\theta_G,\theta_D^K(\theta_G,\theta_D))}{\partial \theta_D^K(\theta_G,\theta_D)} \frac{d\theta_D^K(\theta_G,\theta_D)}{d\theta_G}$
  - as $k \rightarrow \infty$, $\frac{\partial f(\theta_G,\theta_D^K(\theta_G,\theta_D))}{\partial \theta_D^K(\theta_G,\theta_D)} \rightarrow 0$
    - This becomes the standard GAN with the optimal discriminator
  - as $k = 0$, this becomes the standard GAN with the iteratively optimized discriminator without optimality

Effects on Mode Collapsing by Unrolled GAN

- If G, D have enough capacity, and at each step,
  - Given G, the discriminator is allowed to reach its optimum
  - p_g is updated so as to improve the criterion
    - $E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}[log(1 - D(G(z))]$
    - then p_g converges to p_{data}

- **Surrogate Function**
  - $f_K(\theta_G, \theta_D) = f(\theta_G, \theta_D^K(\theta_G, \theta_D))$
  - Unrolling K-step
    - $\theta_G \leftarrow \theta_G - \eta \frac{f_K(\theta_G, \theta_D)}{d\theta_G}, \theta_D \leftarrow \theta_D + \eta \frac{f(\theta_G, \theta_D)}{d\theta_D}$
    - Anticipating the K-future discriminator parameter if we update the generator parameter

Learning Iteration Progress



Variants of Generative Adversarial Network

Conditional Generative Adversarial Network

- Role of original GAN from the generative modeling perspective
  - An implicit method of generating x by $G(z)$, such as $p(x|z)$
  - Original GAN has a pure noise modeling on z, i.e. $N(0,1)$
  - Therefore, modeling on a conditional probability is needed

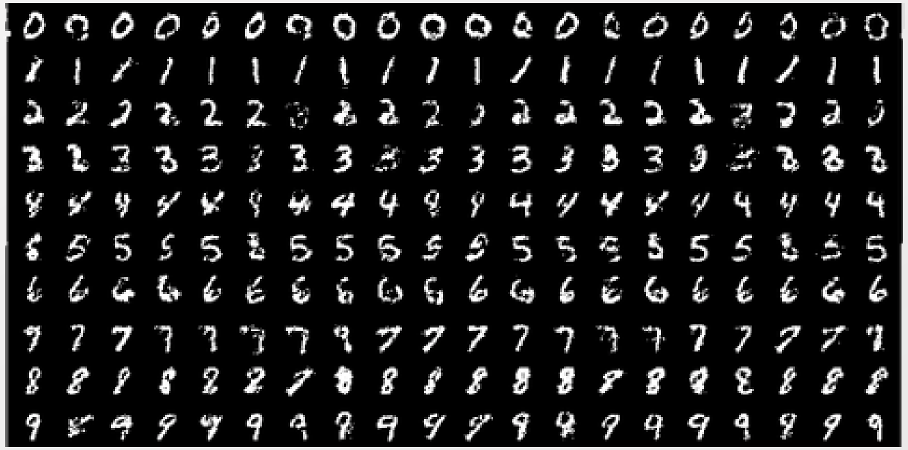
- **Original GAN**
  - $\min_G \max_D V(D,G)$
$$= \min_G \max_D E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z))]$$

- **Conditional GAN**
  - $\min_G \max_D V(D,G)$
$$= \min_G \max_D E_{x\sim p_{data(x)}}[logD(x|y)] + E_{z\sim p_z(z)}[\log(1 - D(G(z|y))]$$
  - $D(x) = p \rightarrow D(x|c) = p \rightarrow NN_D(x,c;w_D) = p$
    - The discriminator takes the condition as an additional input
  - $G(z) = x \rightarrow G(z|c) = x \rightarrow NN_G(z,c;w_G) = x$
    - The generator takes the condition as an additional input

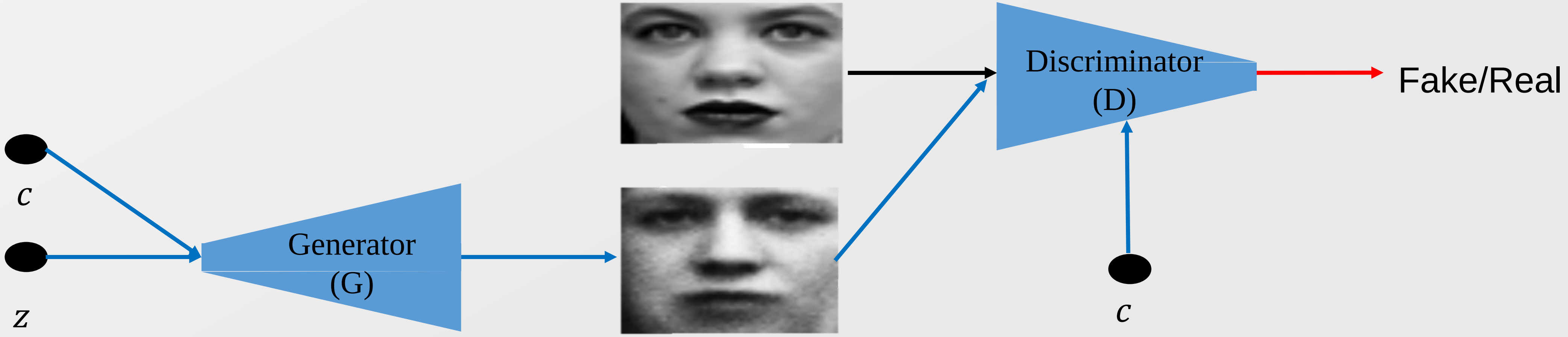
Structure of Conditional GAN

- Just a concatenated input of y
  - $NN_G(z, c; w_G) = x$
  - $NN_D(x, c; w_D) = p$

- Enables the conditioned sampling of x
  - Condition can be indicated as a vector value
  - i.e. a latent vector from autoencoder



MNIST Image generation through CGAN



Adding Latent Variable to GAN

- Original GAN objective
  - $\min_G \max_D V(D,G) = \min_G \max_D E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z))]$
- **Mutual information**
  - $I(X;Z) = D_{KL}(P_{X,Z}||P_X \otimes P_Z) = H(X) - H(X|Z) = H(Z) - H(Z|X)$
    - $I(X;Z) = \sum_{x\in X, z\in Z} P_{(X,Z)}(x,z) \log \frac{P_{(X,Z)}(x,z)}{P_X(x)P_Z(z)}$

$$= \sum_{x\in X, z\in Z} P_{(X,Z)}(x,z) \log \frac{P_{(X,Z)}(x,z)}{P_X(x)} - \sum_{x\in X, z\in Z} P_{(X,Z)}(x,z) \log P_Z(z) = \sum_{x\in X, z\in Z} P_X(x) P_{Z|X=x}(z) \log P_{Z|X=x}(z) - \sum_{x\in X, z\in Z} P_{(X,Z)}(x,z) \log P_Z(z)$$

$$= \sum_{x\in X} P_X(x) \left(\sum_{z\in Z} P_{Z|X=x}(z) \log P_{Z|X=x}(z) \right) - \sum_{z\in Z} \left(\sum_{x\in X} P_{(X,Z)}(x,z) \right) \log P_Z(z) = -\sum_{x\in X} P_X(x) H(Z|X=x) - \sum_{z\in Z} P_Z(z) \log P_Z(z)$$

$$= -H(Z|X) + H(Z)$$

- If we add a latent variable, c, to the generated data, p_g
  - The original GAN can be further extended
    - $\min_G \max_D V(D,G) - \lambda I(c; G(z,c))$
      - If c and $G(z,c)$ are independent, $I(c;G(z,c))$=0

InfoGAN

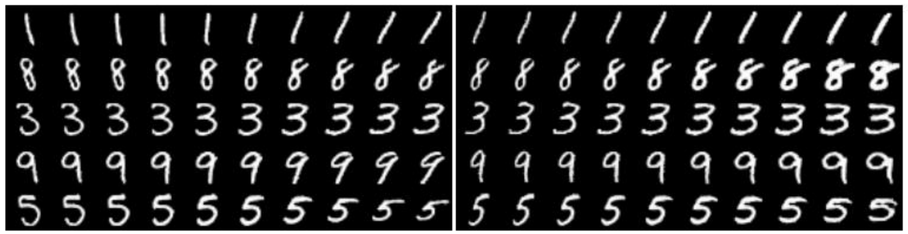
- $\min_G \max_D V(D,G) - \lambda I(c; G(z,c))$
  - Given the above objective function, $I(c; G(z,c))$ needs to be optimized
  - However, c is a latent variable that requires an approximation
  - **Variational mutual information maximization**
    - $I(c; G(z,c)) = H(c) - H(c|G(z,c))$
      $= H(c) + E_{x\sim G(z,c)}[\sum_{c'\sim P(c|x)} P(c'|x) \log P(c'|X)]$
      $= H(c) + E_{x\sim G(z,c)}[KL(P(c'|x)||Q(c'|x)) + E_{c'\sim P(c|x)}[\log Q(c'|X)]]$
      $\geq E_{x\sim G(z,c)}\left[E_{c'\sim P(c|x)}[\log Q(c'|X)]\right] + H(c)$
    - Introduced the implicit variational distribution $Q(c'|x)$
  - $L_I(G,Q) = E_{x\sim G(z,c)}\left[E_{c'\sim P(c|x)}[\log Q(c'|X)]\right] + H(c) \leq I(c; G(z,c))$

- $\min_G \max_D V(D,G) - \lambda I(c; G(z,c)) \leq \min_{G,Q} \max_D V(D,G) - \lambda L_I(G,Q)$
  - From the perspective of Q,
    - $V(D,G) - \lambda L_I(G,Q)$ should be minimized
      - Discriminator should not distinguish the generation from the prior noise and the latent variable

Implementation of InfoGAN

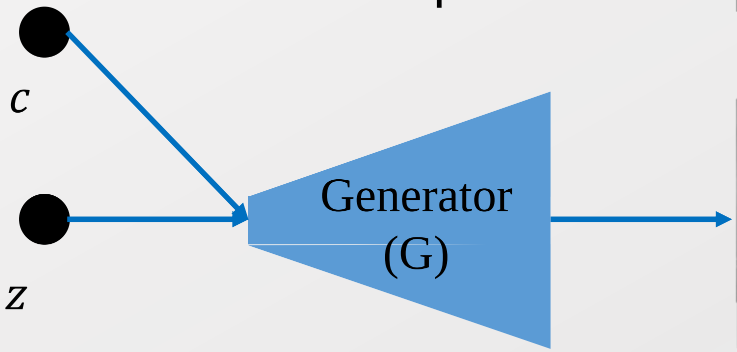
- $\min_{G,Q} \max_{D} V(D,G) - \lambda L_I(G,Q)$
  - $L_I(G,Q) = E_{x\sim G(z,c)}\left[E_{c'\sim P(c|x)}[\log Q(c'|X)]\right] + H(c)$
  - $Q(c'|X)$ becomes a distribution that produces the estimation of c given x

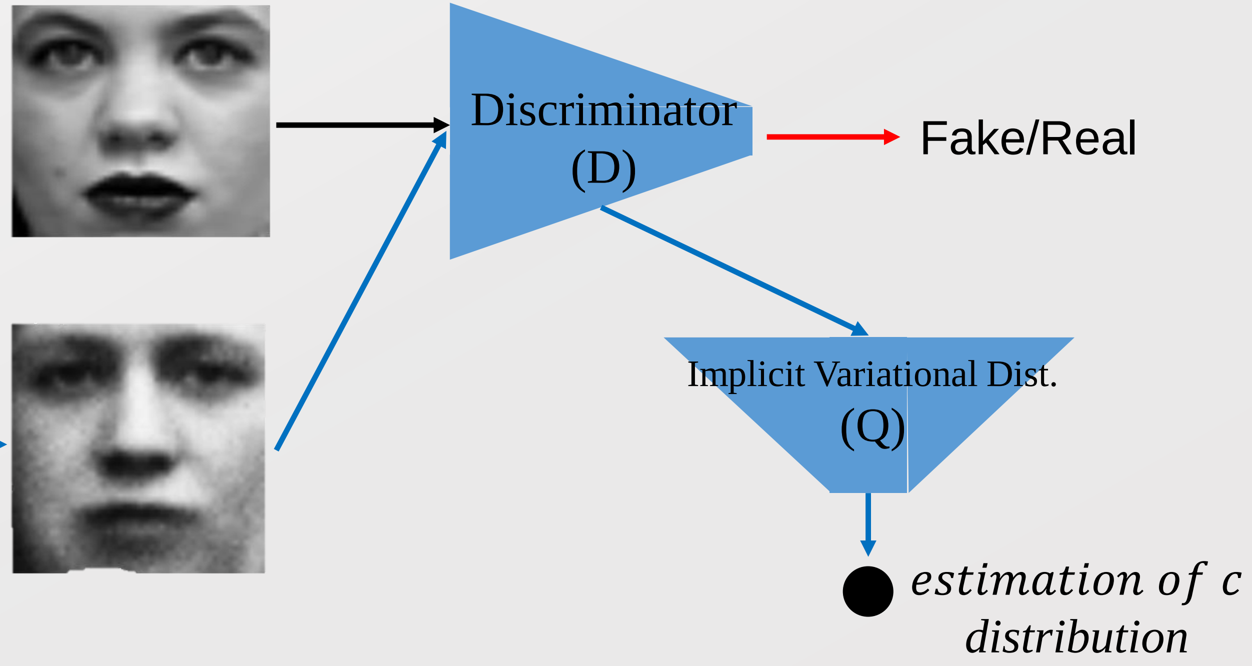
Chen, Xi, et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets." *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016.



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation) (d) Varying c_3 from -2 to 2 on InfoGAN (Width)

- Virtual example is generated from the designed c and the noise z
  - Sample c from a selected prior distribution
- Real example is evaluated to produce c

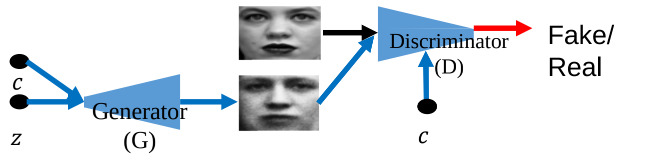




Comparison between Conditional GAN and InfoGAN

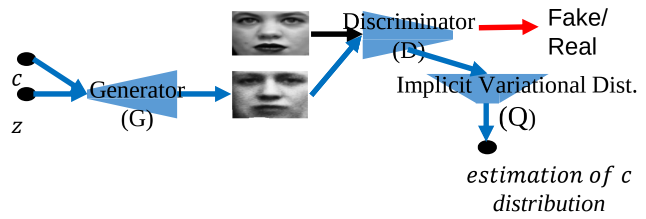
- Both Conditional GAN and InfoGAN uses the code as an input to the generator.

- Conditional GAN
  - $\min_G \max_D E_{x\sim p_{data(x)}}[logD(x|y)] + E_{z\sim p_z(z)}[\log(1 - D(G(z|y))]$
  - Generator : y=[0,0,1,0,0,0,0,0,0,0], Z → G(z,y) = x = image of ‘2’
  - Discriminator : y=[0,0,1,0,0,0,0,0,0,0], x=image of ‘2’ → D(y,x) = p in [0,1]



Conditional GAN Structure

- InfoGAN
  - $\min_{G,Q} \max_D V(D,G) - \lambda L_I(G,Q)$
  - $\min_{G,Q} \max_D E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z,c))] - \lambda\{E_{x\sim G(z,c)}\left[E_{c'\sim P(c|x)}[\log Q(c'|x)]\right] + H(c)\}$
  - Generator same as Conditional GAN
    - y=[0,0,1,0,0,0,0,0,0,0], Z → G(z,y) = x = image of ‘2’
    - However, here, y is sampled, not a supervised output
  - Discriminator
    - x=image of ‘2’ → D(x) = p in [0,1]
  - Auxiliary structure
    - x=image of ‘2’ → Q(c|x)
    - Q(c|x) : the probability distribution of c given x
      - c=[0.05, 0.1, 0.7,….] : c is estimated, not embedded as the label in the dataset
      - If Q(c|x) follows the multinomial distribution, it could have a final softmax layer
      - If Q(c|x) follows the Gaussian distribution, it could have a layer to produce the mean and the variance



InfoGAN Structure

Modifying the Loss Characteristics

Generalizing Loss of Divergence

- GAN minimizes the loss of the Jensen-Shannon divergence
  - $V(D,G) = E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}[log(1-D(G(z)))]$
    $= 2JS(P_g||P_{data}) - \ln 4$
  - Divergence
    - The difference between two probability distributions
      - Jensen-Shannon divergence
      - Kullback-Liebler divergence
    - This is not the distance measure. Distance is defined to be a function, d
      - $d(x,y) \geq 0$ and $d(x,y) = 0 \Leftrightarrow x = y$
      - $d(x,y) = d(y,x)$: Not satisfied by divergence
      - $d(x,y) + d(y,z) \geq d(x,z)$: Not satisfied by divergence
    - Usually, x and y are assumed to be a vector, but a function can be a vector, as well
      - Abstract vector space
      - Natually, it is feasible to define a distance between two vectors representing functions
        - Probability density function is a function
- Then, our question becomes how to generalize the function
  - In terms of the divergence and the distance

Detour: Convex Duality

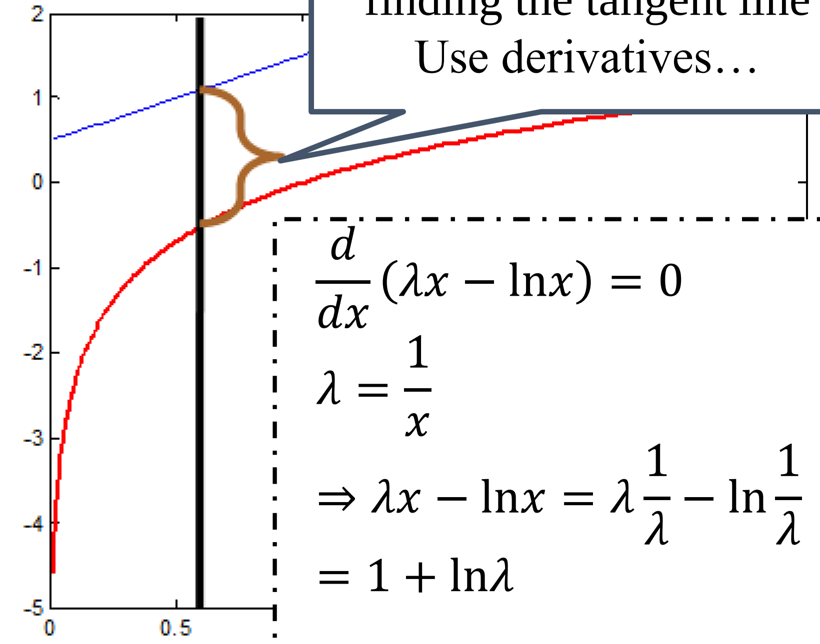
- Systematic variational transform?
  - Utilize the convex duality

- Concave function f(x), such as log function
  - Can be represented via a conjugate or dual function as follows
  - Remember that if f(x) is not a concave function
    - You can always use the log-concave function
      - Transform using the log function
      - Re-transform using the exp function

- $f(x) = min_{\lambda}\{\lambda^T x - f^*(\lambda)\}$

$$\Leftrightarrow f^*(\lambda) = min_x\{\lambda^T x - f(x)\}$$

Dual function or Conjugate function



Minimization, or optimization, through finding the tangent line
Use derivatives…

$$\frac{d}{dx}(\lambda x - \ln x) = 0$$

$$\lambda = \frac{1}{x}$$

$$\Rightarrow \lambda x - \ln x = \lambda\frac{1}{\lambda} - \ln\frac{1}{\lambda}$$

$$= 1 + \ln\lambda$$

Convex Conjugate Function

- For a function $f: X \longrightarrow R$, the convex conjugate function $f^*: X \longrightarrow R$ is defined by
  - $f^*(a) \coloneqq \sup\{< a, x > - f(x)\}$
    - $f^*(a) \geq [< a + x > -f(x)]$
  - Also, known as **Fenchel conjugate**, which is convex regardless of the convexity of f

- Property of conjugate functions
  - **Fenchel's inequality**: for any function f and its convex conjugate f^*
    - for all $a, x \in X$, $f^*(a) + f(x) \geq < a, x >$
  - Order reversing: if $f(x) \leq g(x)$ for all $x \in X \Longrightarrow g^*(a) \leq f^*(a)$ for all $a \in X$
  - Convex conjugate function f^* is always convex and lower semi-continuous
    - But, not necessarily proper
  - $a = f'(x) \Longrightarrow \forall y \in X, f(y) \geq f(x) + < a, y - x >$

$$\Leftrightarrow < a, y > -f(y) \leq < a, x > -f(x)$$
$$\Leftrightarrow sup_{y \in X}\{< a, y > -f(y)\} = f^*(a) \leq < a, x > -f(x)$$
$$\Leftrightarrow f^*(a) + f(x) \leq < a, x > \Leftrightarrow f^*(a) + f(x) = < a, x >$$

    - By the convexity of f at the first step
    - By the Fenchel's inequality at the last step

f - divergence

- Let's generalize the divergence as the below
  - $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$
    - f : generator function, convex, $f(1) = 0$
    - We can define the Fenchel conjugate of the generator function, $f^*(t), t \in T$
      - $f(u) = sup_{t \in T}\{tu - f^*(t)\}$

$$KL(P||Q) = \sum_i P(i) \ln\left(\frac{P(i)}{Q(i)}\right)$$

- $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx = \int_x q(x) sup_{t \in T}\left\{t\frac{p(x)}{q(x)} - f^*(t)\right\} dx$

$$\geq \sup_{\tau \in \mathrm{T}}\left\{\int_x p(x)\tau(x)dx - \int_x q(x) f^*\big(\tau(x)\big)dx\right\}$$

$$= \sup_{\tau \in \mathrm{T}}\{E_{x\sim p(x)}[\tau(x)] - E_{x\sim q(x)}[f^*(\tau(x))]\}$$

  - The domain of f is $\frac{p(x)}{q(x)}$
  - t becomes the function by varying x: $t \ at \ x \rightarrow \tau(x)$
    - Optimal setting of $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right)$
      - By following the property of Fenchel conjugate: $a = f'(x) \Leftrightarrow f^*(a) + f(x) = < a, x >$
    - $\tau \in \mathrm{T}$: T is an arbitrary class of functions $\tau: X \longrightarrow R$
  - Samples from $p(x)$ are real images
  - Samples from q(x) are generated images
  - Two steps of the lower bound
    - Jensen's inequality : swapping the supremum and the integration
    - The limited search space of T

Derivations of Optimal τ of Fenchel Conjugate

- Let's calculate some examples of Fenchel conjugate of some divergences
  - Only applicable to the family of f-divergence: $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$
  - KL divergence
    - $D_f(P||Q) = \int p(x) \log\frac{p(x)}{q(x)} dx$
    - $f(u) = u \log u,\ f'(u) = \log u + u\frac{1}{u} = 1 + \log u$
    - Optimal setting of $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right) = 1 + \log\frac{p(x)}{q(x)}$
  - GAN divergence
    - $V(D,G) = E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}[log(1 - D(G(z))]$

$$= \int p(x)\log\frac{p(x)}{p(x)+q(x)} + q(x)\log\frac{q(x)}{p(x)+q(x)} dx$$

$$= \int p(x)\log\frac{2p(x)}{p(x)+q(x)} + q(x)\log\frac{2q(x)}{p(x)+q(x)} - p(x)\log 2 - q(x)\log 2\, dx$$

$$= \int p(x)\log\frac{2\frac{p(x)}{q(x)}}{\frac{p(x)}{q(x)}+1} + q(x)\log\frac{2}{\frac{p(x)}{q(x)}+1} dx - \log 4$$

$$= \int p(x)\log\frac{p(x)}{q(x)} - (p(x)+q(x))\log\left(\frac{p(x)}{q(x)}+1\right) + (p(x)+q(x))\log 2\, dx - \log 4$$

$$= \int q(x)\left\{\frac{p(x)}{q(x)}\log\frac{p(x)}{q(x)} - \left(\frac{p(x)}{q(x)}+1\right)\log\left(\frac{p(x)}{q(x)}+1\right)\right\} dx$$

    - $f(u) = u\log u - (u+1)\log(u+1),\ f'(u) = 1 + \log u - \log(u+1) - (u+1)\frac{1}{u+1} = \log\frac{u}{u+1}$
    - Optimal setting of $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right) = \log\frac{p(x)}{p(x)+q(x)}$

- $D_f(P||Q) \geq \sup_{\tau\in T}\{E_{x\sim p(x)}[\tau(x)] - E_{x\sim q(x)}[f^*(\tau(x))]\}$
  - The domain of f is $\frac{p(x)}{q(x)}$
  - $f(u) = sup_{t\in T}\{tu - f^*(t)\}$
    - $f^*(t) \coloneqq \sup\{<t,u> - f(u)\}$
    - Fenchel conjugate of the generator function, $f^*(t), t \in T$
  - Optimal setting of $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right)$

Variational Divergence Minimization

- $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx \geq \sup_{\tau \in \mathrm{T}} \{E_{x\sim p(x)}[\tau(x)] - E_{x\sim q(x)}[f^*(\tau(x))]\}$
  - $f(u) = u \log u - (u+1)\log(u+1)$, $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right) = \log \frac{p(x)}{p(x)+q(x)}$

> - The domain of f is $\frac{p(x)}{q(x)}$
> - $\boldsymbol{f(u) = sup_{t\in T}\{tu - f^*(t)\}}$
>   - $f^*(t) := \sup\{< t, u > - f(u)\}$
>   - Fenchel conjugate of the generator function, $f^*(t), t \in T$

- We cannot optimize the f-divergence, directly, so we optimize the lower bound
  - Any functions that we do not need to approximate
    - $p(x)$: distribution sampled as the dataset
    - $f^*(t)$: Fenchel conjugate of $f(u)$, already determined by setting a certain f-divergence
  - Any functions that we need to approximate
    - $q(x)$: distribution to be approximated. generator function!
      - $z \sim p(z), x_{gen} = G(z)$
    - $\tau(x)$: a function as changing t by x, a function to select out of T given $\tau \in \mathrm{T}$
      - T : a set of functions that can be approximated by a neural network
      - optimal $\tau(x)$ is set to be $\log \frac{p(x)}{p(x)+q(x)}$, so we need to learn τ to be the classifier between $p(x)$ and $q(x)$

- Eventually, we can provide the parameterized version of the lower bound
  - $F(\theta, \omega) = E_{x\sim P}[T_\omega(x)] - E_{x\sim Q_\theta}[f^*(T_\omega(x))]$
    - minimize $F(\theta, \omega)$ to reduce the divergence by θ
    - maximize $F(\theta, \omega)$ to tighten the inequality, or finding optimal τ, by ω

Instantiation of Variational Divergence Minimization

- $F(\theta,\omega) = E_{x\sim P}[T_\omega(x)] - E_{x\sim Q_\theta}\left[f^*\left(T_\omega(x)\right)\right]$
  - minimize $F(\theta,\omega)$ to reduce the divergence by θ
  - maximize $F(\theta,\omega)$ to tighten the inequality, or finding optimal τ, by ω

- Under the instantiation of GAN divergence, a type of f-divergence
  - $f(u) = u\log u - (u+1)\log(u+1)$, $\tau(x) = f'\left(\frac{p(x)}{q(x)}\right) = \log\frac{p(x)}{p(x)+q(x)}$
  - Now, $T_\omega(x)$ is a neural network without a restriction
    - However, we are providing inputs to the Fenchel conjugate of $f^*(t)$
    - So, we need to make sure that $T_\omega(x)$ provides an input fall into the range of t of $f^*(t)$

- $f^*(t) = \sup_{u\in U}\{ut - f(u)\} = \sup_{u\in U}\{ut - u\log u + (u+1)\log(u+1)\}$

  - Let's say $g(t,u) = ut - u\log u + (u+1)\log(u+1)$
  - $\frac{dg(t,u)}{du} = t - \log u - u\frac{1}{u} + (u+1)\frac{1}{u+1} + \log(u+1) = t + \log\frac{u+1}{u}$
    - $\frac{d^2 g(t,u)}{(du)^2} = \frac{1}{u+1} - \frac{1}{u} < 0$, $g(t,u)$ is concave with respect to u
  - $\frac{dg(t,u)}{du} = 0 \rightarrow t = \log\frac{u}{u+1} \rightarrow t < 0$
  - Given $t = \log\frac{u}{u+1}$, $g(t,u) = ut - u\log\frac{u}{u+1} + \log(u+1) = ut - ut + \log(u+1)$

  $\rightarrow f^*(t) = g(t) = -\log(1-e^t)$ at the optimal t

- Therefore, $T_\omega(x)$ should result in R_-
  - $T_\omega(x)$ can utilize the output rectifier for the negative domain, i.e. softplus with minus
    - softplus : $a(x) = \log(1+e^x) \rightarrow$ negative softplus : $a(x) = -\log(1+e^x)$, here x is the input from the last layer of NN
  - This choice of the output function depends upon the instantiation of f-divergence

- The domain of f is $\frac{p(x)}{q(x)}$
- $\boldsymbol{f(u) = sup_{t\in T}\{tu - f^*(t)\}}$
  - $f^*(t) := \sup\{< t,u > - f(u)\}$
  - Fenchel conjugate of the generator function, $f^*(t), t\in T$

Difference of Two Probability Distributions

- How to compare two sequences of values (or a function)?
  - Ratio or difference
- Is f-divergence the only method in comparing two distributions?
  - $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx$
    - $\frac{p(x)}{q(x)}$ is the likelihood ratio based comparison
      - assuming the support of $q(x)$ will cover the support of $p(x)$
      - What-if $\mathrm{supp}(p) - \mathrm{supp}(q) \neq \phi$?
  - f-divergence requires the generation distribution $q(x)$ to be wider than $p(x)$
    - Numerical instability : $\frac{p(x)}{q(x)}$ can diverge
    - Mode collapse : the ratio in $f\left(\frac{p(x)}{q(x)}\right)$ could be ignored if $q(x)$ becomes 0
- Are there any other comparison method for two probability distributions?
  - (Absolute) difference of two densities over the domain
  - Integral Probability Metrics, or IPM

Integral Probability Metric

- IPM is defined as

$$d_{\mathcal{G}}(\mu,\nu) = \sup_{g\in\mathcal{G}} \left\{ \left| \int g d\mu - \int g d\nu \right| \right\}$$

  - Settings of $\mathcal{G}$ determines the variation of IPM

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

- g could be
  - **Total variation distance** : $\mathcal{G}$ is the class of all measurable functions taking value in $[0,1]$
    - $\delta(P_r, P_g) = \sup_{A\in\Sigma} |\mathrm{P_r(A)} - \mathrm{P_g(A)}|$
  - **Wasserstein metric** : $\mathcal{G}$ is the class of 1-Lipschitz functions
    - Wassertein-1 or Earth-Mover Distance. Norm can be set by the modeler
    - $W(P_r, P_g) = \inf_{\gamma\in\Pi(P_r,P_g)} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$
  - **Maximum Mean Discrepancy** : $\mathcal{G}$ is the unit ball of RKHS
    - Kernel and basis mapping function can be set by the modeler
    - $MMD(P_r, P_g) = \sup_{\|\psi\|_{\mathcal{H}}\le 1} (E_{x\sim P_r}[\psi(x)] - E_{y\sim P_g}[\psi(x)])$

GAN+MMD(1)

Gretton, Arthur, et al. "A kernel two−sample test." *The Journal of Machine Learning Research* 13.1 (2012): 723−773.

- f-GAN optimizes the f-divergence (in spite that the target is the bound)
  - $D_f(P||Q) = \int_x q(x) f\left(\frac{p(x)}{q(x)}\right) dx \geq \sup_{\tau \in \mathrm{T}} \{ E_{x\sim p(x)}[\tau(x)] - E_{x\sim q(x)}[f^*(\tau(x))] \}$
- If we substitute the f-divergence by the IPM
  - Let's exchange $D_f(P||Q)$ with $MMD(P_r, P_g)$
  - $MMD^2(P_r, P_g) = \sup_{\|\psi\|_{\mathcal{H}} \leq 1} (E_{x\sim P_r}[\psi(x)] - E_{y\sim P_g}[\psi(x)]) = \| \mu_p - \mu_q \|_{\mathcal{H}}^2$
    - if ψ(x)=x, then matching the mean
    - if $\psi(x) = (x, x^2)$, then matching the mean and variance
    - $\mu_p = \int k(x,\cdot) p(dx) \in \mathcal{H}$
      - we may not have a direct access to p or q, so $E[f(X)] = \langle f, \mu_p \rangle_{\mathcal{H}}$
    - By following the kernel two-sample test
    - $MMD^2(P_r, P_g) = E_{x,x'}[k(x,x')] - 2E_{x,y}[k(x,y)] + E_{y,y'}[k(y,y')]$

$$= \frac{1}{N(N-1)} \sum_{n \neq n'} k(x_n, x_{n'}) + \frac{1}{M(M-1)} \sum_{m \neq m'} k(y_m, y_{m'}) - \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} k(y_m, x_n)$$

GAN+MMD(2)

- We can substitute f-divergence with MMD
  - $MMD^2(P_r, P_g) = \sup_{\|\psi\|_{\mathcal{H}} \le 1} (E_{x\sim P_r}[\psi(x)] - E_{y\sim P_g}[\psi(x)]) = \| \mu_p - \mu_q \|_{\mathcal{H}}^2$
    $= E_{x,x'}[k(x,x')] - 2E_{x,y}[k(x,y)] + E_{y,y'}[k(y,y')]$
    $= \frac{1}{N(N-1)} \sum_{n \ne n'} k(x_n, x_{n'}) + \frac{1}{M(M-1)} \sum_{m \ne m'} k(y_m, y_{m'}) - \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} k(y_m, x_n)$
  - If we optimize the generator of $y = G_\theta(z), z \sim P(z)$
    - $P(z)$ as the base distribution for the stochasticity of $G_\theta(z)$
    - We need to optimize the MMD loss with respect to θ

- $\min_\theta MMD^2(P_r, P_g)$

$$= \min_\theta \frac{1}{N(N-1)} \sum_{n \ne n'} k(x_n, x_{n'}) + \frac{1}{M(M-1)} \sum_{m \ne m'} k(y_m, y_{m'}) - \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} k(y_m, x_n)$$

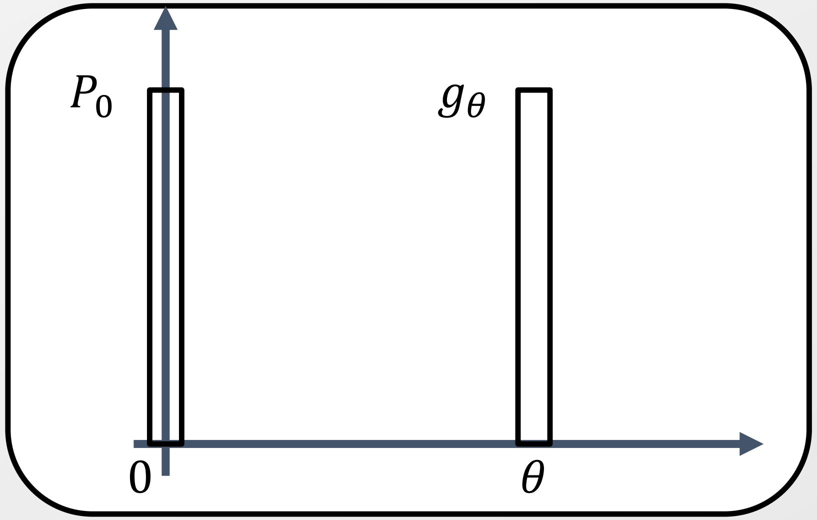
$$= \min_\theta \frac{1}{M(M-1)} \sum_{m \ne m'} k(G_\theta(z_m), G_\theta(z_{m'})) - \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} k(G_\theta(z_m), x_n)$$

$$= \min_\theta \frac{1}{N} \sum_{n=1}^{N} \sum_{m=1}^{M} \left[\frac{1}{M(M-1)} \sum_{m'=1, m' \ne m}^{M} k(G_\theta(z_m), G_\theta(z_{m'})) - \frac{2}{M} \left(k(G_\theta(z_m), x_n) \right) \right]$$

- Still, the gradient method is applicable in optimizing θ
- Then, where is the discriminator learning?
  - f-divergence : optimal τ is required, and the optimality is approached through the optimized Discriminator
  - IPM : MMD requires a good selection of k including its hyperparameter setting, which could be optimized, as well

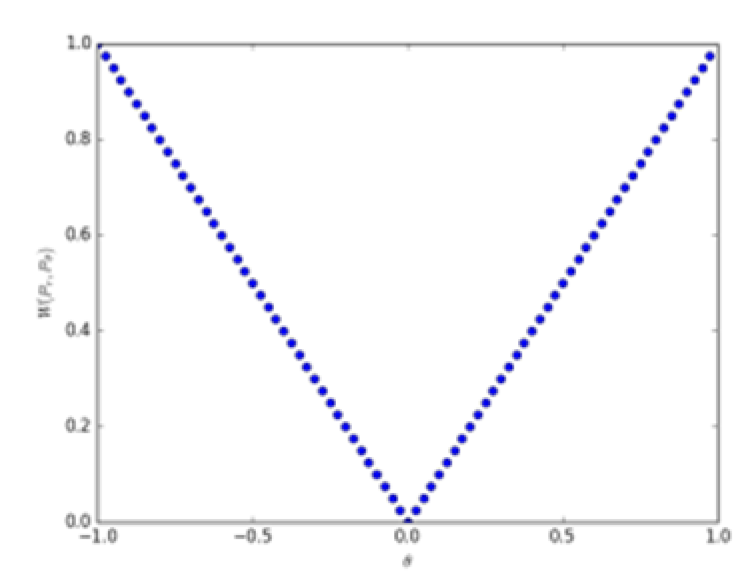
Example of Parallel Line Density

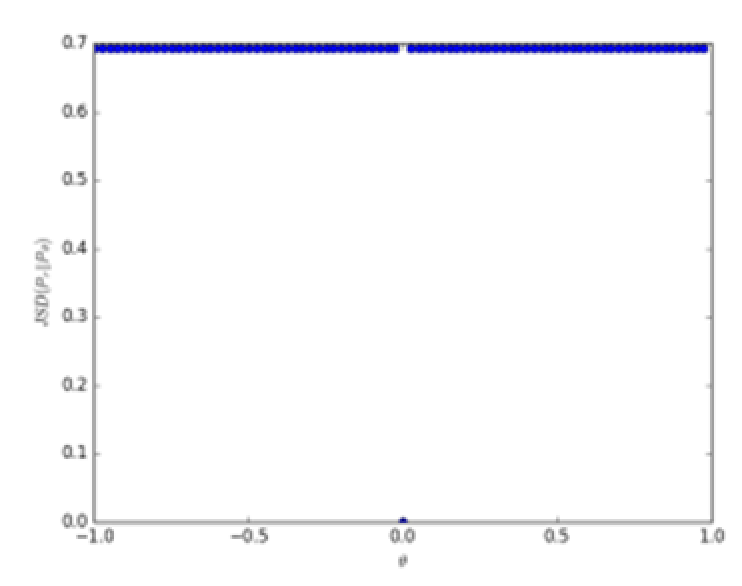
- Let's assume
  - $Z \sim U[0,1]$: the uniform distribution on the unit interval
  - P_0 : the distribution of $(0, Z) \in R^2$, simulation of the data distribution in this example
  - $g_\theta(z) = (\theta, z)$, $\theta \in R$, θ as a parameter of the Generator function
- Then, the Wasserstein metric is the only metric as a continuous function
  - out of total variation, KL, JS, and Wasserstein



- **Total variation distance** : $\mathcal{G}$ is the class of all measurable functions taking value in $[0,1]$
  - $\delta(P_r, P_g) = \sup_{A \in \Sigma} |\mathrm{P_r(A)} - \mathrm{P_g(A)}|$
  - $\delta(P_0, P_g) = \begin{cases} 1, \text{if } \theta \neq 0 \\ 0, \text{if } \theta = 0 \end{cases}$
- **Wasserstein metric** : $\mathcal{G}$ is the class of 1-Lipschitz functions
  - $W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$
  - $W(P_0, P_g) = |\theta|$
- **KL Divergence**
  - $D_{KL}(P_r||P_g) = \int P_r(x) \log \frac{P_r(x)}{P_g(x)} dx$
  - $D_{KL}(P_0||P_g) = D_{KL}(P_g||P_0) = \begin{cases} \infty, \text{if } \theta \neq 0 \\ 0, \text{if } \theta = 0 \end{cases}$
- **JS Divergence**
  - $D_{JS}(P_r||P_g) = \frac{1}{2} D_{KL}(P_r||\frac{P_r + P_g}{2}) + \frac{1}{2} D_{KL}(P_g||\frac{P_r + P_g}{2})$
  - $D_{JS}(P_r||P_g) = \begin{cases} \log 2, \text{if } \theta \neq 0 \\ 0 \quad\ \ , \text{if } \theta = 0 \end{cases}$

Visualization of Divergence & Distance





- Wasserstein metric : $\mathcal{G}$ is the class of 1-Lipschitz functions
  - $W\left(P_r, P_g\right) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathrm{E}_{(x,y)\sim\gamma}\left[||x - y||\right]$
  - $W\left(P_0, P_g\right) = |\theta|$

- JS Divergence
  - $D_{JS}(P_r||P_g) = \frac{1}{2} D_{KL}\left(P_r\middle|\middle|\frac{P_r+P_g}{2}\right) + \frac{1}{2} D_{KL}\left(P_g\middle|\middle|\frac{P_r+P_g}{2}\right)$
  - $D_{JS}(P_r||P_g) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$

KAIST

Wassertein Distance with GAN

- Original Wassertein Distance
  - $W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$
- Original GAN objective
  - $\min_G \max_D E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z))]$
- Need to merge the two structure
  - By turning the original GAN objective into the Wassertein distance formula
    - But, there is no common aspect in the original form of the distance
    - Original Wassertein distance defines the mass transport in the joint space, $\Pi(P_r, P_g)$
      - The complexity becomes $O(X \times X)$, which is a huge space given the high dimensionality of X
    - Original GAN objective utilizes the expectation on the marginal distribution of P_r and P_g
      - This becomes the acceptable complexity because P_r is sampled as a dataset, and P_g can be generated multiple times
- Passing-by OR researchers says why not utilize the dual form of the wassertein distance
  - As you can see, $W(P_r, P_g)$ is inherently an optimization problem of infimum

Kantorovich-Rubinstein Duality

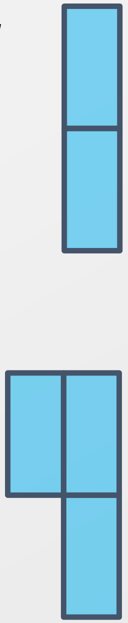
- In linear programming, there is a duality in optimization
  - Primal : Minimize $c^T x$, subject to $Ax = b$, $x \geq 0$
  - Dual : Maximize $b^T y$, subject to $A^T y \leq c$

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)\sim\gamma}[||x - y||]$$

- The optimization becomes choosing an instance of γ from $\Pi(P_r, P_g)$ to minimize $W(P_r, P_g)$

P_r

P_g



1				
	1			
	1	1		
				1

$\gamma \in \Pi(P_r, P_g)$

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

$||x - y||$

Wasserstein as Primal LP

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$$

Primal : Minimize $c^T x$, subject to $Ax = b$, $x \geq 0$
Dual : Maximize $b^T y$, subject to $A^T y \leq c$

- The optimization becomes choosing an instance of γ from $\Pi(P_r, P_g)$ to minimize $W(P_r, P_g)$
- Representing the "earth" movement as a LP problem
  - $Ax = b$: Hard constraint : maintaining the marginal distribution
  - x : Decision variable : each cell value in γ
  - c^T : Objective function coefficient : distance between the earth movement

P_r

P_g

1				
	1			
	1	1		
				1

$\gamma \in \Pi(P_r, P_g)$

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

$||x - y||$

b^T

$P_r(x_1)$	$P_r(x_2)$	$P_r(x_3)$	$P_r(x_4)$	$P_r(x_5)$	$P_g(y_1)$	$P_g(y_2)$	$P_g(y_3)$	$P_g(y_4)$	$P_g(y_5)$
1	2	1	0	1	1	1	0	2	1

x

$\gamma_{ij} = \gamma(x_i, y_j)$

A^T

x										
γ_{11}	1					1				
γ_{12}	1						1			
γ_{13}	1							1		
γ_{14}	1								1	
γ_{15}	1									1
γ_{21}		1				1				
γ_{22}		1					1			
γ_{23}		1						1		
γ_{24}		1							1	
γ_{25}		1								1
γ_{31}			1			1				
γ_{32}			1				1			
γ_{33}			1					1		
γ_{34}			1						1	
γ_{35}			1							1
γ_{41}				1		1				
γ_{42}				1			1			
γ_{43}				1				1		
γ_{44}				1					1	
γ_{45}				1						1
γ_{51}					1	1				
γ_{52}					1		1			
γ_{53}					1			1		
γ_{54}					1				1	
γ_{55}										

Wasserstein as Dual LP

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathrm{E}_{(x,y)\sim\gamma}[||x - y||]$$

Primal : Minimize $c^T x$, subject to $Ax = b$, $x \geq 0$
Dual : Maximize $b^T y$, subject to $A^T y \leq c$

- The optimization becomes choosing an instance of γ from $\Pi(P_r, P_g)$ to minimize $W(P_r, P_g)$
- Representing the "earth" movement as a LP problem
  - $Ax = b$: Hard constraint : maintaining the marginal distribution
  - x : Decision variable : each cell value in γ
  - c^T : Objective function coefficient : distance between the earth movement

P_r

P_g

1				
	1			
	1	1		
				1

$\gamma \in \Pi(P_r, P_g)$

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

$||x - y||$

$$D_{ij} = ||x_i - y_j||$$

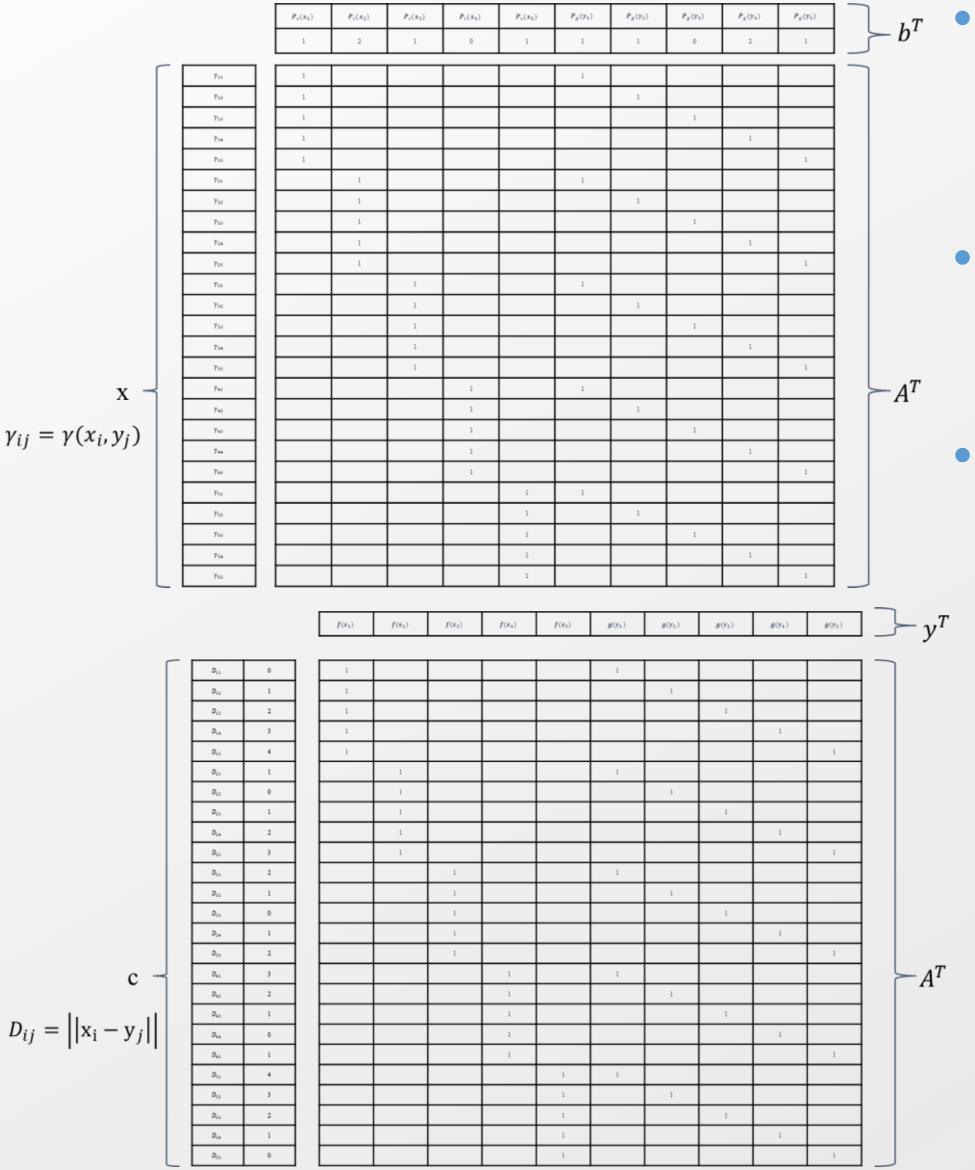
y^T

	c	$f(x_1)$	$f(x_2)$	$f(x_3)$	$f(x_4)$	$f(x_5)$	$g(y_1)$	$g(y_2)$	$g(y_3)$	$g(y_4)$	$g(y_5)$
D_{11}	0	1					1				
D_{12}	1	1						1			
D_{13}	2	1							1		
D_{14}	3	1								1	
D_{15}	4	1									1
D_{21}	1		1				1				
D_{22}	0		1					1			
D_{23}	1		1						1		
D_{24}	2		1							1	
D_{25}	3		1								1
D_{31}	2			1			1				
D_{32}	1			1				1			
D_{33}	0			1					1		
D_{34}	1			1						1	
D_{35}	2			1							1
D_{41}	3				1		1				
D_{42}	2				1			1			
D_{43}	1				1				1		
D_{44}	0				1					1	
D_{45}	1				1						1
D_{51}	4					1	1				
D_{52}	3					1		1			
D_{53}	2					1			1		
D_{54}	1					1				1	
D_{55}	0										

c

A^T

Property of Dual LP on Wasserstein Distance



- Duality in LP
  - Primal : Minimize $c^T x$, subject to $Ax = b$, $x \geq 0$
  - Dual : Maximize $b^T y$, subject to $A^T y \leq c$
- Primal of Wasserstein Distance :

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)\sim\gamma}\left[||x - y||\right]$$

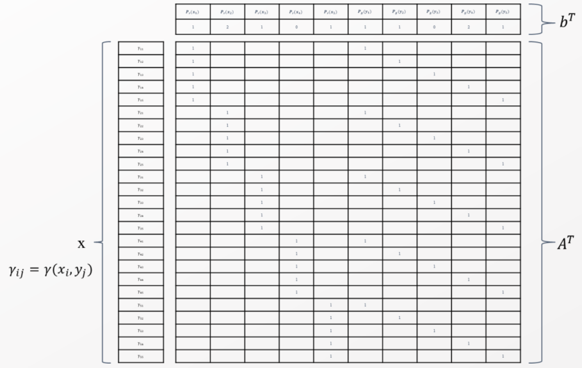
- Dual constraints
  - $f(x_i) + g(y_j) \leq D_{i,j}$: this should be hold for every i, j
    - If $i = j$, $f(x_i) + g(y_i) \leq D_{i,i} = 0$
    - At the optimality, the equality occurs from a diagonal constraint
      - $f(x_i) + g(y_i) = 0 \rightarrow f(x_i) = -g(y_i)$
  - Then, every other constraints need to satisfy
    - $f(x_i) + g(y_j) \leq D_{i,j} \rightarrow f(x_i) - f(y_j) \leq D_{i,j}$
    - This limits the variation of f = Lipschitz constraint on f

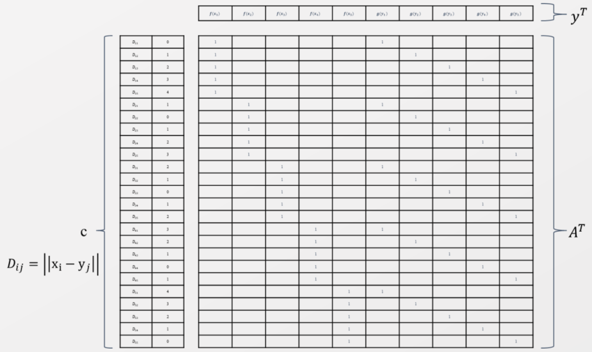
Detour: Lipschitz Continuity

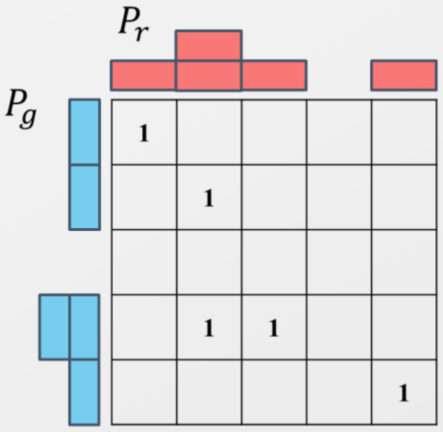
- Lipschitz constraint of the dual problem of Wasserstein distance
  - $f(x_i) + g(y_j) \le D_{i,j} \rightarrow f(x_i) - f(y_j) \le D_{i,j}$
  - This limits the variation of f = Lipschitz constraint on f
- Lipschitz continuity
  - Given two metric spaces (X, d_x) and (Y, d_y) where d_x denotes the metric on the set X and d_y on the set Y
    - a function $f: X \rightarrow Y$ is Lipschitz continuous if there exists a real constant $K \ge 0$
      - such that, for all x_1 and x_2 in X,
    - $d_Y(f(x_1), f(x_2)) \le K d_X(x_1, x_2)$
      - K : Lipschitz constant
- Distance can be the absolute difference in R
  - $|f(x_1) - f(x_2)| \le K|x_1 - x_2|$
- Is a neural network Lipschitz continuous?
  - Exact constant calculation of Neural Network is NP-Hard
    - Most activation functions (ReLU, Softplus, tanh, logistic…) are 1-Lipschitz continuous
    - However, their combinations are difficult to analyze

Scaman, Kevin, and Aladin Virmaux. "Lipschitz regularity of deep neural networks: analysis and efficient estimation." *arXiv preprint arXiv:1805.10965* (2018).

Dual Problem of Wasserstein Distance







$\gamma \in \Pi(P_r, P_g)$

0	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	1
4	3	2	1	0

$||x - y||$

- Primal and dual problem in LP
  - Primal : Minimize $c^T x$, subject to $Ax = b$, $x \geq 0$
    - Primal of Wasserstein Distance : $W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y)\sim\gamma}[||x - y||]$
  - Dual : Maximize $b^T y$, subject to $A^T y \leq c$
    - Dual of Wassersteing Distance : $W(P_r, P_g) = \max_f E_{P_r}[f(x)] + E_{y\sim P_g}[g(y)]$
      - Constrained by $f(x_i) + g(y_j) \leq D_{i,j}$
    - Dual of Wassersteing Distance : $W(P_r, P_g) = \max_f E_{P_r}[f(x)] - E_{y\sim P_g}[f(y)]$
      - Constrained by $f(x_i) + g(y_i) = 0 \rightarrow f(x_i) = -g(y_i)$, $f(x_i) - f(y_j) \leq D_{i,j}$
      - == Constrained by f to be Lipschitz continuous
    - inf → min : continuous function on a compact set by the constraints
  - f and γ : decision variables
  - A : Matrix
    - between $\gamma_{i,j}$ and b
    - between $D_{i,j}$ and y
  - $D_{i,j}$: Distance of the earth movement
  - b : marginal distribution concatenating P_r and P_g

Kantorovich-Rubinstein Duality and Wassertein GAN

- Kantorovich-Rubinstein Theorem
  - $W(p_r, p_g) = \inf_{\gamma \in \Pi(p,q)} E_{(x,y)\sim\gamma}[||x-y||] = \sup_{||f||_L \leq 1} \left[E_{x\sim p_r}[f(x)] - E_{y\sim p_g}[f(x)] \right]$
- Original GAN
  - $\min_G \max_D E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z))]$
- Wassertein GAN
  - Max : To make a Wassertein metric between two distributions
    - $W(P_r, P_g) = \max_f E_{P_r}[f(x)] - E_{y\sim P_g}[f(y)]$
  - Min : To make P_g close to P_r
    - $\min_{P_g} W(P_r, P_g) = \min_{P_g} \max_f E_{P_r}[f(x)] - E_{y\sim P_g}[f(y)]$
  - Constraints to make the Wassertein metric
    - Lipschitz constraint of f
      - Weight clipping : make the gradient of neural network to be bounded
      - Regularization on f (a.k.a. critic)
        - Let's say $x_t = tx + (1-t)y, t \in [0,1], x\sim p_g, y\sim p_r \rightarrow \left||\nabla f^*(x_t)|\right| = 1$ when f^* is the optimal function of f
        - $\min_{P_g} \max_f E_{P_r}[f(x)] - E_{y\sim P_g}[f(y)] - \lambda E_{x''\sim p''}\left[\left(||\nabla_{x''} f^*(x'')||_2 - 1\right)^2\right]$, $x''\sim p''$ is the sampling on the interpolation

Acknowledgements

- This lecture is influenced and adopted materials from 10807 Topics in Deep Learning by Prof. Russ Salakhutdinov, Carnegie Mellon University.
- Some parts are adopted from the slides by
  - Wonsung Lee
  - Kyungwoo Song