


DATA DRIVEN TECHNIQUES

Lecture 7

Realization of Random Variables - Generative Models



Generative Modeling

- Realizations of random variables
- Classical Methods
 - Inverse cdf
 - Acceptance-rejection
- Using transformations
 - Generative models
 - Design with adversarial methods
 - Design with non-adversarial methods
- Probability density vs Generative model

Realizations of Random Variables

Realizations of random variables are needed in Monte-Carlo simulations
Generate synthetic data (images, music, videos)

When densities of interest are usual then classical methods

In case of modern datasets classical methods fail **miserably** in producing realizations

Need alternative techniques that can handle data that are multi-dimensional and can lie in lower dimensional manifolds

Regarding random variables that lie on lower dimensional manifolds description using probability densities is not efficient

Classical Methods

For given density $f(x)$ we like to generate realizations

Assume we have probability density $h(z)$ for which we can easily generate realizations, for example Uniform or Gaussian

Inverse CDF

If z_1, z_2, \dots independent realizations, uniform in $[0,1]$, apply transformation

$$x_i = F^{-1}(z_i)$$

where $F(x) = \int_{-\infty}^x f(w) dw$ is the cumulative distribution function (cdf)

and $F^{-1}(z)$ its inverse function ($F^{-1}(z) = x \Leftrightarrow z = F(x)$)

x_i follows $f(x)$

Every z_i generates x_i

$$\mathbb{P}(x \leq x) = \mathbb{P}(F^{-1}(z) \leq x) = \mathbb{P}(z \leq F(x)) = F(x)$$

Can we extend this to multi-dimensional densities?

If we have joint density $f(x_1, x_2, x_3)$ can we generate triplets?

Bayes Rule: $f(x_1, x_2, x_3) = f(x_3|x_2, x_1)f(x_2|x_1)f(x_1)$

Conditionally independent

Generate independent uniform realizations z_1, z_2, z_3

$$f(x_1) \Rightarrow F(x_1) \Rightarrow x_1 = F^{-1}(z_1)$$

$$f(x_2|x_1) \Rightarrow F(x_2|x_1) \Rightarrow x_2 = F^{-1}(z_2|x_1)$$

$$f(x_3|x_2, x_1) \Rightarrow F(x_3|x_2, x_1) \Rightarrow x_3 = F^{-1}(z_3|x_2, x_1)$$

then x_1, x_2, x_3 follow $f(x_1, x_2, x_3)$

Not practically convenient !!!

Acceptance/Rejection

We are given $f(x)$, cannot compute $F(x)$ or $F^{-1}(z)$, can evaluate $f(x)$

Assume another density $h(z)$ for which we can generate realizations (e.g. Gaussian)

Assume we know L such that $\frac{f(x)}{h(x)} \leq L < \infty$ for all x (L necessarily > 1)

Generate pair (z_i, t_i) , independent with $z_i \sim h(z)$ and $t_i \sim U([0, 1])$ ← uniform

If $\frac{f(z_i)}{h(z_i)} \geq L t_i$ then accept and set $x_i = z_i$

x_i follows $f(x)$

Otherwise reject pair and try again with a new one

On average only one every L realizations is accepted.

True for vector densities $f(X)$ when can generate $h(Z)$

Using Transformations

Generative Models

Want realizations X_i that follow $f(X)$. Common method using **transformations**

Start with $Z \sim h(Z)$, find $G(Z)$ deterministic so that $X = G(Z)$ follows $f(X)$

Does $G(Z)$ exist ?

THEOREM: Under general conditions

YES IT EXISTS !!!

Pair $\{G(Z), h(Z)\}$ called **Generative Model**

Theorem proves **existence**

Scalar $z \sim h(z)$, strictly increasing $G(z)$, then density of $x = G(z)$ can be found

$$\mathbb{P}(x \leq x) = \mathbb{P}(G(z) \leq x) = \mathbb{P}(z \leq G^{-1}(x)) = H(G^{-1}(x))$$

$z(x) = G^{-1}(x)$ is the unique solution to $G(z) = x$

$$\Rightarrow f(x) = h(G^{-1}(x)) \times \frac{1}{G'(G^{-1}(x))} = h(z(x)) \times \frac{1}{|G'(z(x))|}$$

If $G(z)$ **not monotone** then $G(z) = x$ may have multiple solutions:

$z_1(x), \dots, z_p(x)$ *← even p can be a function of x.*

$$f(x) = h(z_1(x)) \times \frac{1}{|G'(z_1(x))|} + \dots + h(z_p(x)) \times \frac{1}{|G'(z_p(x))|}$$

Vector $Z \sim h(Z)$ and vector transformation $G(Z)$

Random vector $X = G(Z)$, how to compute density?

Equation $G(Z) = X$ has multiple solutions $Z_1(X), \dots, Z_p(X)$

$$f(X) = \frac{h(Z_1(X))}{|\det\{\mathbb{J}_Z G(Z_1(X))\}|} + \dots + \frac{h(Z_p(X))}{|\det\{\mathbb{J}_Z G(Z_p(X))\}|}$$

Target

$$G(Z) = \begin{bmatrix} g_1(z_1, \dots, z_k) \\ \vdots \\ g_k(z_1, \dots, z_k) \end{bmatrix}, \quad \mathbb{J}_Z G(Z) = \begin{bmatrix} \frac{\partial g_1}{\partial z_1} & \dots & \frac{\partial g_1}{\partial z_k} \\ \vdots & \dots & \vdots \\ \frac{\partial g_k}{\partial z_1} & \dots & \frac{\partial g_k}{\partial z_k} \end{bmatrix}$$

To design

Why?? To generate realizations of X following $f(X)$ by generating realizations of Z following $h(Z)$ and then transforming $X = G(Z)$

Adversarial Methods (GANs)

Goal: Start with Z following $h(Z)$. Find generative transformation $G(Z)$ so that $X=G(Z)$ follows $f(X)$. **Solve problem without knowing $f(X)$!!!**

Let us develop our method step-by-step

Suppose we have two probability densities $f(X)$ and $g(X)$

How can we test $f(X) = g(X)$?

$$\text{Sufficient: } \frac{g(X)}{f(X)} = 1 \quad \equiv \quad \omega \left(\frac{g(X)}{f(X)} \right) = \omega(1)$$

for ANY $\omega(\cdot)$ strictly increasing

Suppose **we do not have** $f(X)$, $g(X)$ but still desire to test $f(X) = g(X)$

For every function $\delta(X)$ ^{there is} mechanism ^{that} provides $\mathbb{E}_f[\delta(X)], \mathbb{E}_g[\delta(X)]$

Can we compute $D(X) = \omega \left(\frac{g(X)}{f(X)} \right)$ using averages ? **YES !!!**

THEOREM:

Specify **strictly increasing** function $\omega(r)$ and **positive** function $\rho(z)$

Define: $\phi(z), \psi(z) :$ $\psi'(z) = \rho(z), \quad \phi'(z) = -\omega^{-1}(z)\rho(z)$

and for any scalar function $D(X)$ the cost

$$J(D) = \mathbb{E}_f[\phi(D(X))] + \mathbb{E}_g[\psi(D(X))]$$

Then the optimum solution to the maximization problem $\max_{D(X)} J(D)$ satisfies

$$D_o(X) = \omega \left(\frac{g(X)}{f(X)} \right)$$

Why insist on averages ?

If we do not have $f(X)$, $g(Y)$ but two sets of realizations

$$\begin{array}{ll} X_1, X_2, \dots, X_n & \sim f(X) \longrightarrow \mathbb{E}_f[\delta(X)] \approx \frac{1}{n} \sum_{t=1}^n \delta(X_t) \\ Y_1, Y_2, \dots, Y_m & \sim g(Y) \longrightarrow \mathbb{E}_g[\delta(Y)] \approx \frac{1}{m} \sum_{t=1}^m \delta(Y_t) \end{array}$$

How can we optimize with respect to unknown function $D(X)$?

Replace $D(X)$ with neural network $D(X, \vartheta)$. Optimize over network parameters ϑ

$$J(D) = \mathbb{E}_f[\phi(D(X))] + \mathbb{E}_g[\psi(D(X))]$$

$$\hat{J}(\vartheta) = \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(Y_t, \vartheta))$$

Solve $\max_{\vartheta} \hat{J}(\vartheta) \Rightarrow \vartheta_o \Rightarrow D(X, \vartheta_o)$

Expect $D(X, \vartheta_o) \approx D_o(X) = \omega \left(\frac{g(X)}{f(X)} \right)$

Can we use function $D(X, \vartheta_o)$ to examine the two densities $f(X), g(Y)$?

Employ $D(X, \vartheta_o)$ to **discriminate** between $f(X), g(Y)$ using only data

$$X_1, X_2, \dots, X_n \sim f(X) \quad Y_1, Y_2, \dots, Y_m \sim g(Y)$$

If $D(X, \vartheta_o) \not\approx \omega(1) \Rightarrow$ the two datasets have different densities

If $D(X, \vartheta_o) \approx \omega(1) \Rightarrow$ the two datasets have similar densities

$D(X, \vartheta)$ is known as the **Discriminator** function

Examples

A: $\omega(r) = r$

$$\rho(z) = 1, \Rightarrow \phi(z) = -\frac{z^2}{2}, \psi(z) = z$$

(Mean Square)

B: $\omega(r) = \log(r)$

$$\rho(z) = e^{-0.5z} \Rightarrow \phi(z) = -2e^{0.5z}, \psi(z) = -2e^{-0.5z}$$

(Exponential)

C: $\omega(r) = \frac{r}{r+1}$

Goodfellow et al. (2014), NeurIPS

$$\rho(z) = \frac{1}{z}, \Rightarrow \phi(z) = \log(1-z), \psi(z) = \log(z)$$

(Cross Entropy)

What happened to the Generator ?

With $Z \sim h(Z)$ generator $G(Z)$ transforms Z to $Y = G(Z)$

We desire $Y \sim f(.)$ same density as training data $\{X_1, \dots, X_n\}$

Naïve Method: Select a $G(Z)$. **Test** if transformation is the desired
If not, make another selection


How do we test if selection is any good ?

Generate realizations of Z : $\{Z_1, \dots, Z_m\}$

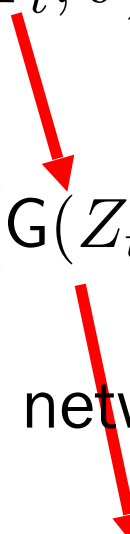
Apply generator to samples, create $\{Y_1, \dots, Y_m\}$ where $Y_i = G(Z_i)$

Test $\{Y_1, \dots, Y_m\}$ against $\{X_1, \dots, X_n\}$


$$\hat{J}(\vartheta) = \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(Y_t, \vartheta))$$


$$\begin{aligned} \max_{\vartheta} \hat{J}(\vartheta) &\Rightarrow \vartheta_o \Rightarrow D(X, \vartheta_o) \\ D(X, \vartheta_o) &\approx \omega \left(\frac{g(X)}{f(X)} \right) \\ D(X, \vartheta_o) &\not\approx \omega(1) \end{aligned}$$

$$\hat{J}(\vartheta) = \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(Y_t, \vartheta))$$

$$\hat{J}(\vartheta) = \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(G(Z_t), \vartheta))$$


We are looking for function $G(Z)$. Use neural network $G(Z, \theta)$

$$\hat{J}(\theta, \vartheta) = \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(G(Z_t, \theta), \vartheta))$$


THEOREM: For fixed θ (generator $G(Z, \theta)$) we have

$$\max_{\vartheta} \hat{J}(\theta, \vartheta) \gtrsim \phi(\omega(1)) + \psi(\omega(1))$$

Equality when $Y = G(Z, \theta)$ has density $g(.) = f(.)$ same as $\{X_1, \dots, X_n\}$

Since $\max_{\vartheta} \hat{J}(\theta, \vartheta) \gtrapprox \phi(\omega(1)) + \psi(\omega(1))$

to bring $\max_{\vartheta} \hat{J}(\theta, \vartheta)$ as close as possible to $\phi(\omega(1)) + \psi(\omega(1))$

we must apply **minimization** over θ

$$\min_{\theta} \max_{\vartheta} \hat{J}(\theta, \vartheta)$$

$$= \min_{\theta} \max_{\vartheta} \left\{ \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(G(Z_t, \theta), \vartheta)) \right\}$$

Adversarial
Optimization

G Generative
A Adversarial
N Networks

We have $Z \sim h(Z)$

Design generator $G(Z)$ so that $Y = G(Z)$ has the same density as X for which we have realizations (training set) $\{X_1, \dots, X_n\}$

Approximate generator with neural network $G(Z, \theta)$

Define second neural network the discriminator $D(X, \vartheta)$

For realizations of Z : $\{Z_1, \dots, Z_m\}$ consider adversarial problem

$$\begin{aligned} \min_{\theta} \max_{\vartheta} \hat{J}(\theta, \vartheta) \\ = \min_{\theta} \max_{\vartheta} \left\{ \frac{1}{n} \sum_{t=1}^n \phi(D(X_t, \vartheta)) + \frac{1}{m} \sum_{t=1}^m \psi(D(G(Z_t, \theta), \vartheta)) \right\} \end{aligned}$$

then generator $G(Z, \theta_0)$ when applied to realizations of Z yields samples following closely the density of $\{X_1, \dots, X_n\}$

Example

High definition CelebA (30 000 high-definition images 1024 X 1024 of celebrities)

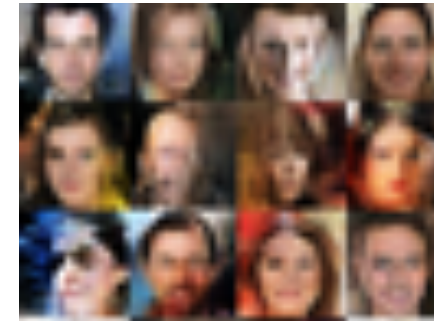
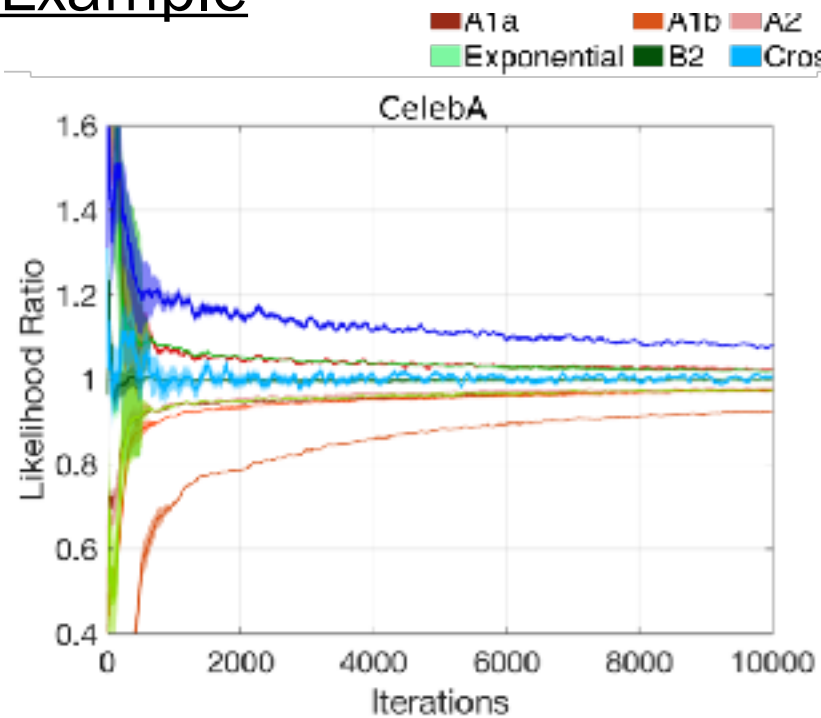


Extremely **hard to control convergence** of the adversarial problem

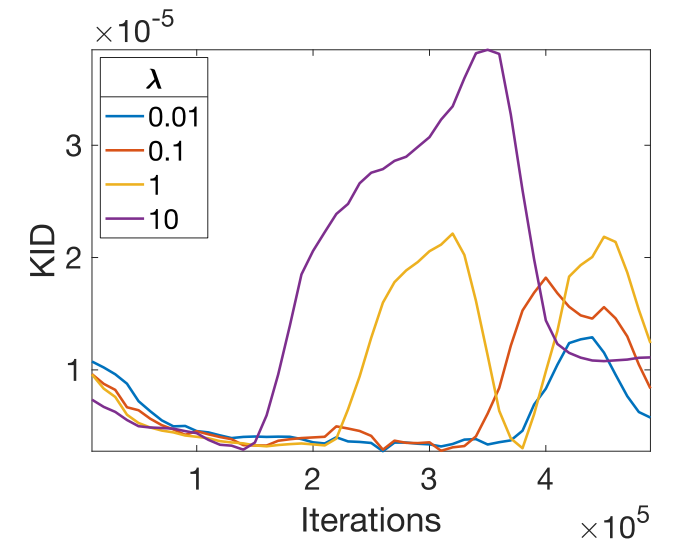
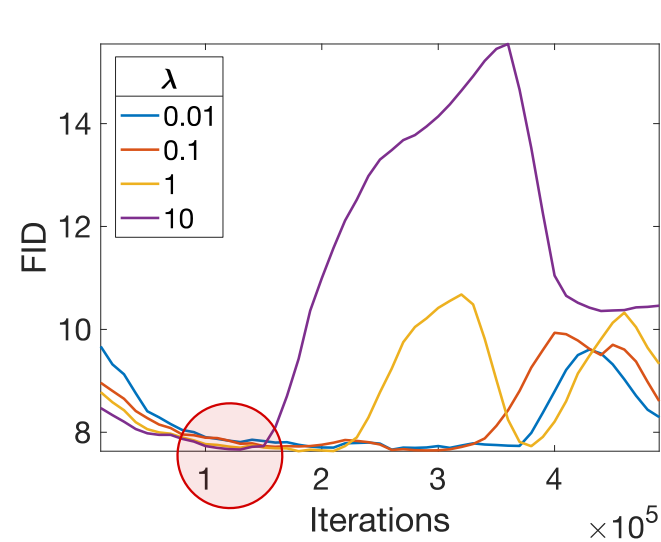
NVIDIA used progressive growing of GANs (4X4), (8X8),..., (1024X1024)



Example



Convergence/Divergence



Non-Adversarial Method

Interested in Generator

Discriminator used during generator design, afterwards useless

Possible (at least in theory) to design Generator **without Discriminator** by **not** employing min-max (adversarial) optimization

We can define optimization criteria involving only maximization or minimization with the help of **positive definite kernels** and design successfully Generators

A symmetric scalar function $K(X, Y)$ will be positive definite if for every nonzero function $\varphi(X)$ it satisfies

$$\iint K(X, Y) \varphi(X) \varphi(Y) dX dY > 0 \quad \text{Gaussian Kernel: } K(X, Y) = e^{-\frac{1}{h^2} \|X - Y\|^2}$$

Validity of the Cauchy-Schwarz inequality

$$\left(\iint \kappa(X, Y) \phi(X) \psi(Y) dX dY \right)^2 \leq \left(\iint \kappa(X, Y) \phi(X) \phi(Y) dX dY \right) \left(\iint \kappa(X, Y) \psi(X) \psi(Y) dX dY \right)$$

For densities $f(X)$, $g(Y)$

$$\frac{\left(\iint \kappa(X, Y) f(X) g(Y) dX dY \right)^2}{\iint \kappa(X, Y) g(X) g(Y) dX dY} \leq \iint \kappa(X, Y) f(X) f(Y) dX dY$$

with equality if and only if $f(X) = g(Y)$

$$\frac{\left(\iint \mathbf{K}(X, Y) f(X) g(Y) dX dY \right)^2}{\iint \mathbf{K}(X, Y) g(X) g(Y) dX dY} \leq \iint \mathbf{K}(X, Y) f(X) f(Y) dX dY$$

$$\frac{\left(\mathbb{E}[\mathbf{K}(X, Y)] \right)^2}{\mathbb{E}[\mathbf{K}(Y^1, Y^2)]} \leq \mathbb{E}[\mathbf{K}(X^1, X^2)]$$

where X^1, X^2 independent with the same density $f(X)$
and Y^1, Y^2 independent with the same density $g(Y)$

$$J(\theta) = \frac{\left(\mathbb{E}[\mathbf{K}(X, G(Z, \theta))] \right)^2}{\mathbb{E}[\mathbf{K}(G(Z^1, \theta), G(Z^2, \theta))]} \leq \mathbb{E}[\mathbf{K}(X^1, X^2)]$$

where Z^1, Z^2 independent and follow $h(Z)$

$$\max_{\theta} J(\theta)$$

We have $Z \sim h(Z)$

Design generator $G(Z)$ so that $Y = G(Z)$ has the same density as X for which we have realizations (training set) $\{X_1, \dots, X_n\}$

Generate $\{Z_1, \dots, Z_m\}$ from $h(Z)$

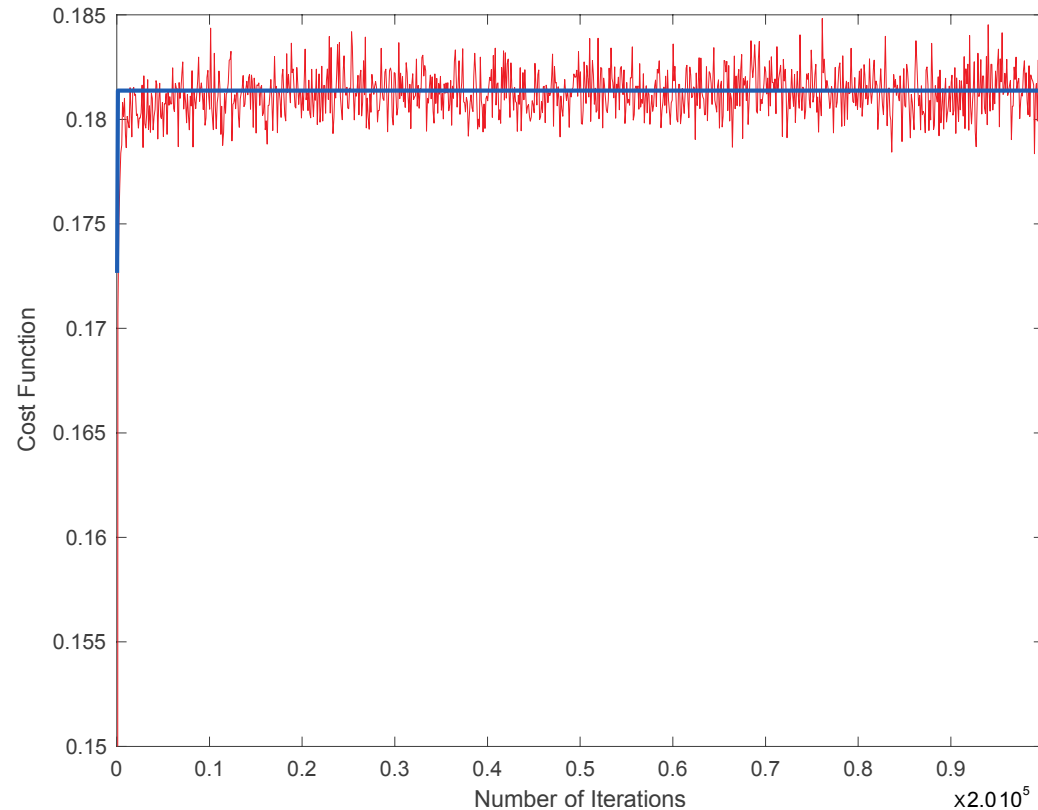
$$\hat{J}(\theta) = \frac{\left(\frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m K(X_i, G(Z_j, \theta)) \right)^2}{\frac{1}{m(m-1)} \sum_{i=1}^m \sum_{j=1, j \neq i}^m K(G(Z_i, \theta), G(Z_j, \theta))}$$
$$\approx \frac{1}{n(n-1)} \sum_{i=1}^n \sum_{j=1, j \neq i}^n K(X_i, X_j)$$

$$\max_{\theta} \hat{J}(\theta)$$

No Convergence/Divergence phenomenon!!!!

Example

CelebA images 32 X 32



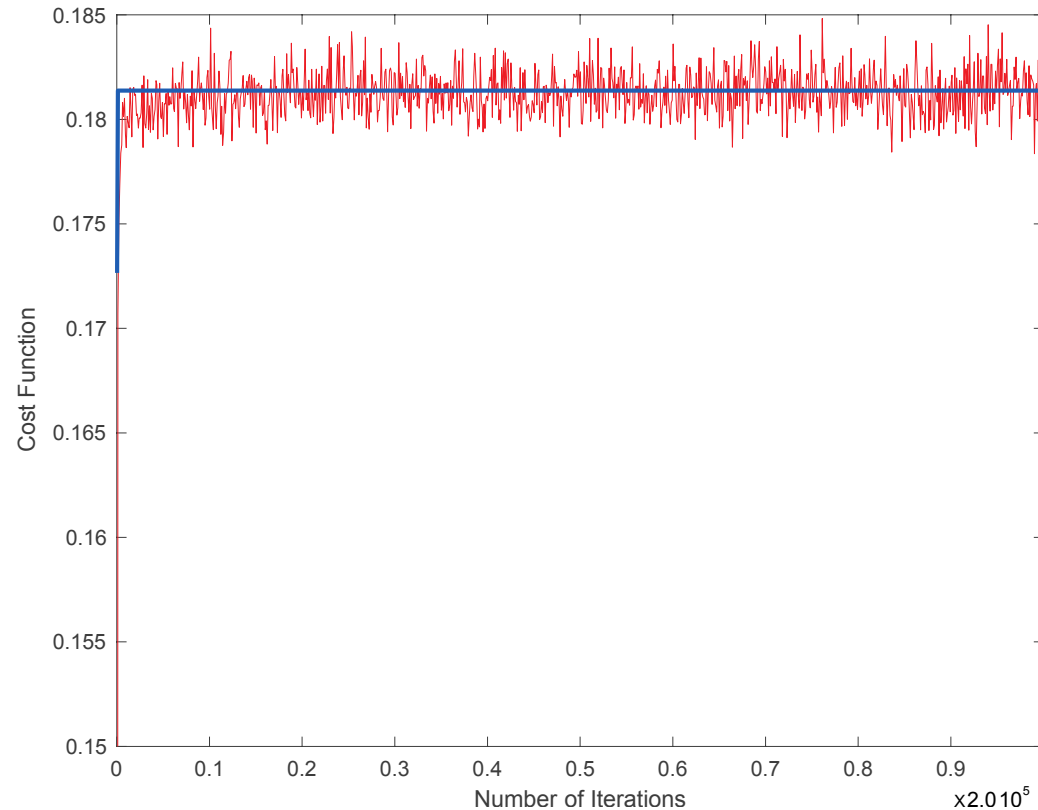
same number of iterations
Kernel
GAN: slow convergence



Suitable for small sized and fully connected networks

Example

CelebA images 32 X 32



Suitable for small sized and fully connected networks

same number of iterations
Kernel
GAN: slow convergence



No improvement!

Probability Density vs Generative Model

Points in N-D space can be random and lie on a lower dimensional surface (manifold)

Example red points on sphere (2-D in 3-D space)

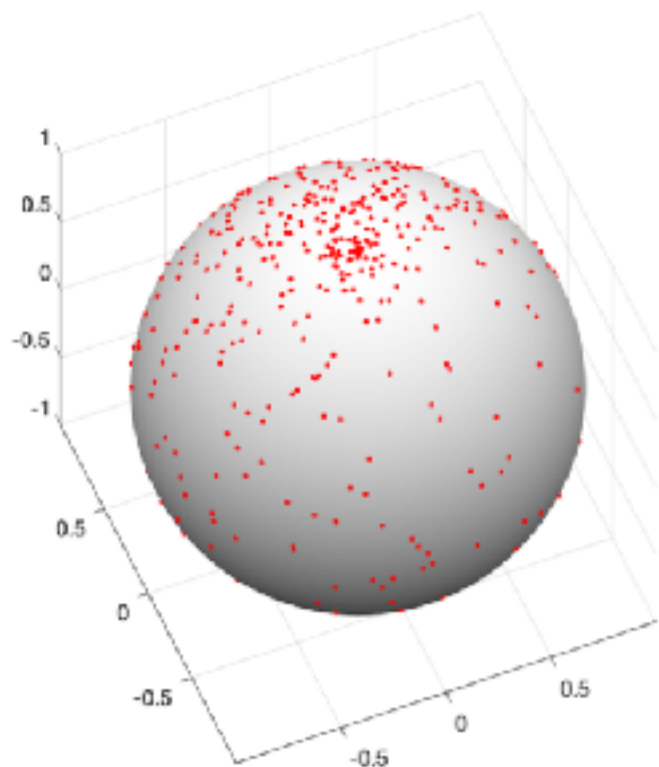
Points are random with coordinates $[x_1, x_2, x_3]$ related through a **deterministic** equation

To lie on a sphere of radius r : $x_1^2 + x_2^2 + x_3^2 = r^2$

$$f(x_1, x_2, x_3) = \delta(x_1^2 + x_2^2 + x_3^2 - r^2) h(x_1, x_2)$$

Dirac $\delta(x)$ **generalized** function is defined as

$$\delta(x) = \begin{cases} 0 & x \neq 0 \\ \infty & x = 0 \end{cases}, \quad \int_{-\epsilon}^{\epsilon} \delta(x) dx = 1$$



Generative model would describe the random data with input density $h(z_1, z_2)$ and generator vector function $G(z_1, z_2)$

$$X = G(z_1, z_2) \Rightarrow \begin{bmatrix} x_1 = G_1(z_1, z_2) \\ x_2 = G_2(z_1, z_2) \\ x_3 = G_3(z_1, z_2) \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 = r \cos(2\pi z_1) \sin(\pi z_2) \\ x_2 = r \sin(2\pi z_1) \sin(\pi z_2) \\ x_3 = r \cos(\pi z_2) \end{bmatrix} \Rightarrow$$

Example: spherical coordinates

$\Rightarrow h(z_1, z_2)$ defined on $[0,1] \times [0,1]$ and $G(z_1, z_2)$ is an ordinary function

Data are representable as $X = G(Z)$, $Z \sim h(Z)$. Many datasets satisfy

$$\dim(Z) \ll \dim(X)$$

In HD CelebA: $\dim(X) = 3 \times 1024 \times 1024 = 3 \times 10^6$

Input to Generator $G(Z)$: $\dim(Z) = 500$ (independent Gaussians)

\rightarrow Instead of estimating X , we first estimate Z , then recover X as $X = G(Z)$

If need to estimate X then