23^{rd} Computer Vision Winter Workshop
Zuzana Kúkelová and Júlia Škovierová (eds.)
Český Krumlov, Czech Republic, February 5–7, 2018

What is optimized in Wasserstein GANs?

Thomas Pinetz, Daniel Soukup
Austrian Institute of Technology
1220 Vienna
{thomas.pinetz, daniel.soukup}@ait.ac.at

Thomas Pock
Graz University of Technology
8010 Graz
pock@icg.tugraz.at

Abstract. *Wasserstein Generative Adversarial Networks (GANs) work by approximating the Wasserstein distance using large Neural Networks (NNs) with specific architectures, hyperparameter and regularization techniques. We show that those networks severely underestimate the Wasserstein distance even on toy problems. For this reason a new algorithm based on convex optimization is proposed to calculate the Wasserstein distance optimally in a discretized space. By comparing the output to the state-of-the-art, we show that current regularization techniques are too restrictive to allow for rich structure in the learned critic functions. Using our critic function as part of a new algorithm called Sequential Convex Optimization (SCO) we achieve reliable convergence on our GAN games for 1D and 2D distributions. This is compared to the current state-of-the-art in training Wasserstein GANs, which do not converge to an optimal solution on any single testcase.*

1. Introduction

In its vanilla form GANs consist of a generator and a discriminator [9], denoted as critic in Wasserstein GANs [4], which are trained in turn adversarially. The solution to a GAN game between a generator and a critic is a saddle point, where the weights of the generator are minimal and the weights of the critic are maximal with respect to the combined value function [19]. While there might exist multiple saddle points the optimal solution to such a problem is referred to as a Nash Equilibrium [9].

GANs are used to sample from an unknown distribution [10], for anomaly detection [13] or for semi supervised learning [9]. However, the mathematical theory underlying GANs has not been as well researched [5]. It is still unclear, if an optimal solution to a GAN game must exist [5]. Also no provable convergence rates or performance guarantees exist for GAN games [5]. Current state-of-the-art methods rely on regularization, carefully crafted architectures and initialization to produce their results [4].

It has been shown that the vanilla GAN [9] minimizes the Jensen-Shannon divergence and converges as soon as the distribution of the generator sample is the same as the distribution of the data samples [9]. The problem with this framework is that the gradient norm of the discriminator approaches zero as the discriminator improves [3]. On one hand a good discriminator is desirable to show the correct improvement direction, but on the other hand a perfect discriminator has gradient zero everywhere and it is not possible to learn the generator from it [3]. To combat this behaviour, non-saturating GANs [16] or Wasserstein GANs [4] are used. In this work, the focus is on the Wasserstein GAN, which minimizes the dual of the Wasserstein distance instead of the Jensen-Shannon divergence. Let μ, ν be two probability distributions in the compact space $\Omega \subset \mathbb{R}^d$, then the dual of the Wasserstein distance on those probability distributions is defined as follows:

$$W(\mu,\nu) = \max_{Lip(f)\leq 1} \mathbb{E}_{x\sim\mu}[f(x)] - \mathbb{E}_{x\sim\nu}[f(x)] \quad (1)$$

The goal of a GAN game is to approximate the target distribution μ with a parametric model ν_θ. To learn the parametric distribution using the Wasserstein distance the following equation has been proposed [4]:

$$\min_{\theta} \max_{Lip(f)\leq 1} \mathbb{E}_{x\sim\mu}[f(x)] - \mathbb{E}_{x\sim\nu_\theta}[f(x)], \quad (2)$$

where f denotes the critic function and is required to have at most a Lipschitz constant of 1 and therefore:

$$|f(x) - f(y)| \leq Lip(f)||x - y|| \quad (3)$$

To sample from a parmetric distribution we use a known distribution ν', e.g. $\mathcal{N}(\mathbf{0}, I)$, and transform this distribution with a function $g_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^d$ giving us the Wasserstein GAN formulation:

$$\min_{\theta} \max_{Lip(f) \leq 1} \mathbb{E}_{x \sim \mu}[f(x)] - \mathbb{E}_{z \sim \nu'}[f(g_\theta(z))] \quad (4)$$

For the optimal critic function f, Equation (5) holds and is used to learn the unknown distribution μ [4].

$$\nabla_\theta W(\mu, \nu_\theta) = -\mathbb{E}_{z \sim \nu'}[\nabla_\theta f(g_\theta(z))] \quad (5)$$

In Wasserstein GANs, the function f is approximated using a NN with various regularization schemes [4, 10, 2, 1]. Given this intuition, the following contributions are made as part of this work:

1. We show that the NN approximations are not optimal even on toy datasets after 10^6 iterations and therefore the mathematical justification does not hold.
2. We propose an algorithm based on convex optimization [6], which does provide an optimal solution in a discretized space.
3. We use this critic function and propose an algorithm called SCO for the Wasserstein GAN problem for small d.

2. Related Work

GANs have been proposed by Goodfellow et. al. [9]. In their vanilla form, GANs have been shown to be a minimization problem on the Jensen-Shannon divergence between two distributions [9]. This has been extended to arbitrary f-divergences [15]. While the authors in [7] argued, that it is not necessary to decrease the divergence in every step, we show that doing so leads to reliable convergence to the optimal solution. The only current performance and generalization guarantee for training GANs has been proposed in settings with multiple generators and discriminators [5]. The prediction step used in our algorithm has also been used in regular GAN training to stabilize the training [19].

One divergence in particular, namely the Wasserstein Distance, has been used to combat mode collapse and vanishing gradients as the discriminator approaches optimality [4]. Based on this work, multiple other works have proposed regularization techniques to improve performance [10, 2, 1]. Another related work replaced the discriminator with its dual and solved a maximization problem instead of a max-min problem [12].

In this work, the Wasserstein distance is calculated using a primal-dual algorithm [6] using the dual of the Wasserstein distance proposed by Kantorovich and Rubinstein [18]. While other algorithms to compute the Wasserstein distance exist [8], in the context of Wasserstein GANs, only NN have been used.

3. Preliminaries

In this section, some common terms are introduced. The dual of the Wasserstein distance follows from Equation (1):

$$\begin{aligned} W(\mu, \nu) &= \max_{Lip(f) \leq 1} \int_\Omega f(x) d\mu(x) - \int_\Omega f(x) d\nu(x) \\ &= \max_{Lip(f) \leq 1} \int_\Omega f(x)(d\mu(x) - d\nu(x)) \quad (6) \end{aligned}$$

In Wasserstein GANs, the function f is approximated using a NN. However, the NN does not necessarily output a function, which satisfies the Lipschitz constraint. However, the Lipschitz constraint can be scaled as follows:

$$W(\mu, \nu) = \frac{1}{K} \max_{Lip(f) \leq K} \mathbb{E}_{x \sim \mu}(f(x)) - \mathbb{E}_{x \sim \nu}(f(x)) \quad (7)$$

The indicator function on a convex set X is defined as:

$$\delta_X(x) = \begin{cases} 0 & if\ x \in X, \\ \infty & else, \end{cases} \quad (8)$$

To obtain the dual of the critic function the convex conjugate is used. The convex-conjugate of a function $F(x)$ is defined as:

$$F^*(x) = \sup_y \langle y, x \rangle - F(y). \quad (9)$$

The primal-dual algorithm requires the calculation of the proximal map. The proximal map of a function

F is calculated for any $\tau > 0$:

$$x = (I+\tau\partial F)^{-1}(y) = \arg\min_x\{\frac{||x-y||^2}{2\tau}+F(x)\} \quad (10)$$

In order to compute the critic f, the input space needs to be discretized. The input domain is given by $\Omega \subset \mathbb{R}^d$ for $d \geq 0, d \in \mathbb{Z}$, which is discretized at a scale $d_x > 0$. Then Ω_{d_x} denotes the discretized space which is given by

$$\Omega_{d_x} = \text{int}(\bigcup\{S_{\boldsymbol{i}} : \boldsymbol{i} \in \mathbb{Z}^d, S_{\boldsymbol{i}} \subset \Omega\}), \quad (11)$$

where

$$S_{\boldsymbol{i}} = [(i_1 - \tfrac{1}{2})d_x, (i_1 + \tfrac{1}{2})d_x)\times \\ ... \times [(i_d - \tfrac{1}{2})d_x, (i_d + \tfrac{1}{2})d_x) \quad (12)$$

and the corresponding index set $\mathcal{I}_d$ is defined as:

$$\mathcal{I}_d = \\ \Big\{\boldsymbol{i} = (i_1, ..., i_d) : 1 \leq i_j \leq N_j, \forall j : 1 \leq j \leq d\Big\}, \quad (13)$$

In this paper, we set $d\mu(S_{\boldsymbol{i}}) = \int_{S_i} d\mu(x)$. For known distributions, this expression can be calculated exactly. For a dataset $x \subset \mathbb{R}^{m\times d}$ with an underlying unknown data distribution μ, we use $d\mu(S_{\boldsymbol{i}}) = \frac{1}{m}\sum_j^m I_{S_i}(p_j)$, where $I_{S_i}(p_j) = \begin{cases} 1 & if\ p_j \in S_{\boldsymbol{i}}, \\ 0 & else. \end{cases}$

4. Wasserstein Critic based on Convex Optimization

In this section, the generator is fixed to an arbitrary distribution and only the critic is solved to optimality. This solution is obtained using convex optimization [6] and compared to a NN trained with either weight clipping [4] or gradient penalty [10], which are currently used in GAN games. The architecure of the NN was chosen to have enough capacity to model the function produced using convex optimization.

4.1. Critic in 1D

In 1D the discrete Wasserstein-distance formula rewritten into a linear program is given by:

$$\max_f \sum_{i_1\in\mathcal{I}_1} f_{i_1}(d\mu(S_{i_1}) - d\nu(S_{\boldsymbol{i}})) \quad (14)$$

$$s.t. \quad \frac{|f_{i_1+1} - f_{i_1}|}{d_x} \leq 1$$

By minimizing the negative and setting $c = [d\mu(S_1) - d\nu(S_1), ...]^T$, $(Df)_{i_1} = \frac{f_{i_1+1}-f_{i_1}}{d_x}$, with $(Df)_{N_1} = 0$ and $||Df||_\infty = \max_{1\leq i_1\leq N_1}(|(Df)_{i_1}|)$ for the constraints and solving this problem on the constrained set, the following linear program is obtained:

$$\min_f -c^T f + \delta_{||\cdot||_\infty\leq 1}(Df) \quad (15)$$

Using the convex conjugate of this function results in the following saddle point formulation:

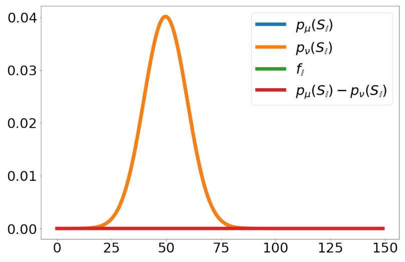
$$\min_f \max_y \langle y, Df\rangle - c^T f - ||y||_1 \quad (16)$$

This problem in saddle point formulation is solved by the primal-dual algorithm [6] with primal variables f and dual variables y given in Algorithm 1. The proximal map of $G(f) = -c^T f$ is given by $(I + \tau\partial G)^{-1}(f) = f + \tau c$ and the proximal map of $F^*(y) = ||y||_1$ is given by the soft shrinkage operator $(I+\sigma\partial F^*)^{-1}(y_i) = \max(0, |y_i| - \sigma)\cdot sgn(y_i)$. The learning rates are chosen in such a way, that equation $\sigma\tau L^2 \leq 1$ holds, where $L = ||D||_2$. It is easily shown, that $||D||_2 = \frac{2}{d_x}$. So by setting $\tau > 0$ and $\sigma = \frac{1}{L^2\tau}$, the condition is satisfied and the algorithm converges in $O(1/N)$, which is optimal for this formulation of the problem [6].

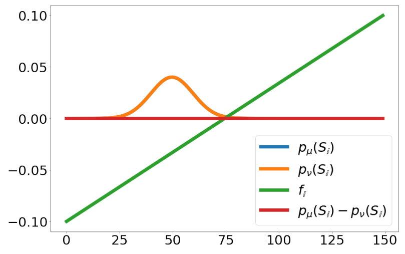
Algorithm 1: Primal-Dual Algorithm [6].
We set $\tau > 0$ and $\sigma = \frac{1}{L^2\tau}$.

```
for $k \geq 0$ do
    $f^{k+1} = (I + \tau\delta G)^{-1}(f^k - \tau D^T y^k)$
    $\bar{f}^{k+1} = 2f^{k+1} - f^k$
    $y^{k+1} = (I + \sigma\delta F^*)^{-1}(y^k + \sigma D\bar{f}^{k+1})$
end
```

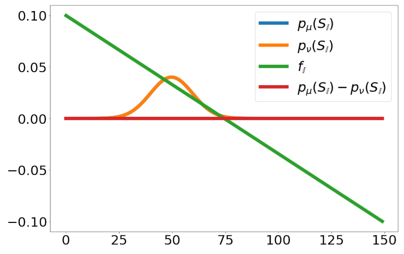
A comparison of our method to NN approximations in multiple gaussian situations is given in Figure 1. For this comparison a NN with 2 hidden layers of 16 neurons and ReLU activation was used. The hyperparameters for the Wasserstein GANs were adapted from the original papers. For weight clipping RMSProp [17] with a learning rate of 10^{-4} and for the gradient penalty Adam [11] with learning rate of 10^{-4}, $\beta_1 = 0$ and $\beta_2 = 0.9$ was used. The NNs were trained for 1.000.000 iterations with a batch size of 128. However, the clipping parameter



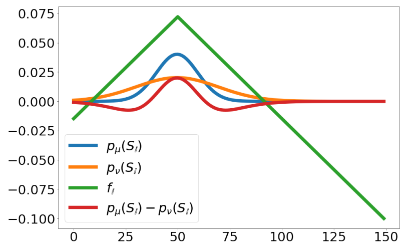
(a) $W(\mu, \nu) = 0$



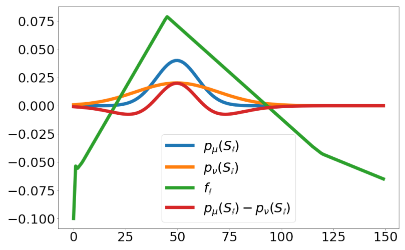
(b) $W(\mu, \nu) = 0$



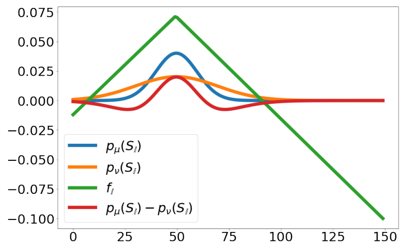
(c) $W(\mu, \nu) = 0$



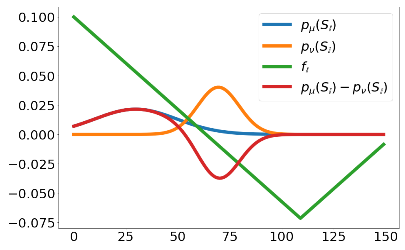
(d) $W(\mu, \nu) = 7.69$



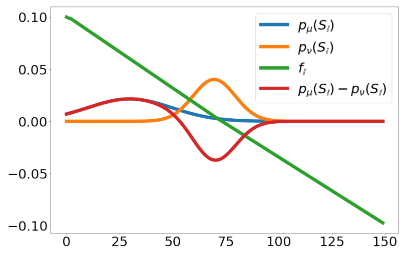
(e) $W(\mu, \nu) = 6.29$



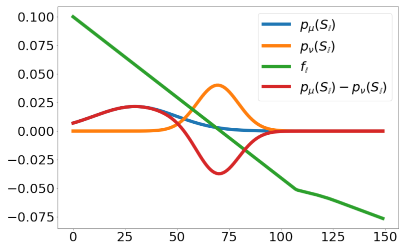
(f) $W(\mu, \nu) = 7.63$



(g) $W(\mu, \nu) = 37.09$



(h) $W(\mu, \nu) = 36.80$



(i) $W(\mu, \nu) = 37.08$

Figure 1: Critic functions in green calculated for two 1D Gaussian distributions using Algorithm 1 (first column), a NN regularized with weight clipping [4] (second column) and a NN regularized with gradient penalty [10] (third column). The critic values are scaled to provide a more pleasent viewing experience using $f_i = \frac{f_i}{10||f||_\infty}$. The critic uses the ground truth probability density function to compute the critic instead of samples.

$c = [-0.1, 0.1]$, because $[-0.01, 0.01]$ was too restrictive for such shallow NN. To visualize the NN approximations, the NNs were evaluated at every point in Ω_{d_x} and interpolated linearly. Even in such a simple case the approximate solutions produced by the NNs reduce non-linear shapes to linear shapes as shown in row 3 of Figure 1. These non-linear shapes actually show, that one distribution has a larger standard deviation than the other distribution and enable learning the σ parameter. Additionally, the approximation critics failed to produce a function outputting 0 for the case, where the same distribution is used twice. Using this critic in a Wasserstein GAN game leads to divergence from the optimal solution. This behavior explains, why Wasserstein GANs do not converge in practice. Still, the Wasserstein distance as shown in Figure 1 is similar to our solution in the one dimensional case.

4.2. Critic in 2D

The program for the 2D case is given by the 2nd order cone problem given in the following equation:

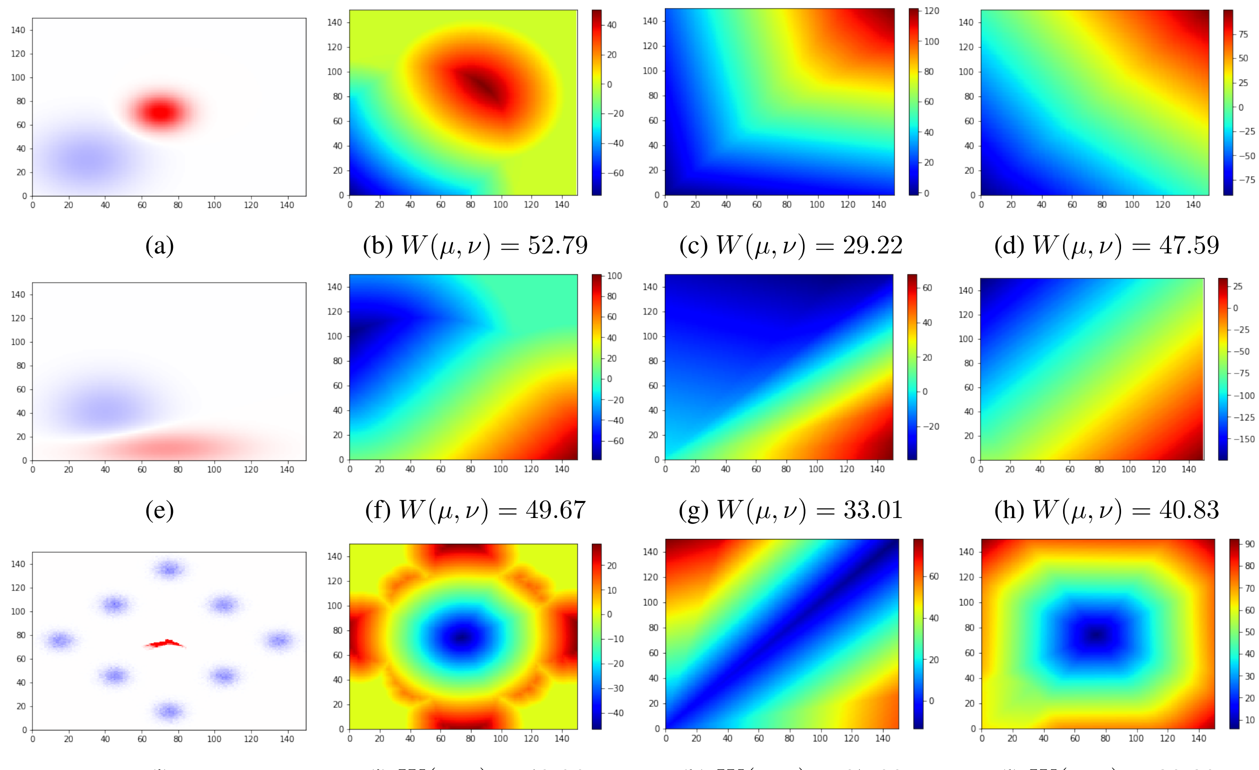
$$\max_f \sum_{\boldsymbol{i} \in \mathcal{I}_2} f_{\boldsymbol{i}}(d\mu(S_{\boldsymbol{i}}) - d\nu(S_{\boldsymbol{i}})) \tag{17}$$

$$s.t. \quad \frac{\sqrt{(f_{i_1+1,i_2} - f_{i_1,i_2})^2 + (f_{i_1,i_2+1} - f_{i_1,i_2})^2}}{d_x} \leq 1$$

This program is convex and is again solved by a primal-dual algorithm in $O(1/N)$ time. In order to use the primal-dual algorithm, Equation 17 is reformulated using matrix D with

$$(Df)_{i_1,i_2,1} = \frac{f_{i_1+1,i_2} - f_{i_1,i_2}}{d_x}, \tag{18}$$

$$(Df)_{i_1,i_2,2} = \frac{f_{i_1,i_2+1} - f_{i_1,i_2}}{d_x}, \tag{19}$$



(a) (b) $W(\mu,\nu) = 52.79$ (c) $W(\mu,\nu) = 29.22$ (d) $W(\mu,\nu) = 47.59$

(e) (f) $W(\mu,\nu) = 49.67$ (g) $W(\mu,\nu) = 33.01$ (h) $W(\mu,\nu) = 40.83$

(i) (j) $W(\mu,\nu) = 49.06$ (k) $W(\mu,\nu) = 27.99$ (l) $W(\mu,\nu) = 39.89$

Figure 2: Comparison of our method to the state-of-the-art. In (a,e,i) the original setting is shown, while the critic based on convex optimization for those settings is shown in (b,f,j). The critic trained using weight clipping is shown in (c,g,k), while the figures in (d,h,l) show the critic trained with gradient penalty. The estimated Wasserstein distance is shown in the caption.

where $(Df)_{N_1,i_2} = \mathbf{0}$ and $(Df)_{i_1,N_2} = \mathbf{0}$. For the learning rate of Algorithm 1, it is important to note, that $||D||_2 \leq \frac{\sqrt{8}}{d_x}$ [6]. Using the constrained set and minimizing the negative instead of maximizing the positive yields:

$$\min_f -c^T f + \delta_{||\cdot||_{2,\infty} \leq \delta_x}(Df) \tag{20}$$

The convex conjugate of this problem is given in the following equation:

$$\min_f \max_y \langle y, Df \rangle - c^T f - ||y||_{2,1} \tag{21}$$

This is solved using the primal-dual algorithm with the proximal map for $G(x) = -c^T f$ given again with $(I + \tau\partial G)^{-1}(f) = f + \tau c$ and the proximal map for $F^*(y) = ||y||_{2,1}$ given as:

$$(I + \sigma\partial F^*)^{-1}(y_i) = y_i\left(1 - \frac{1}{\max(1, \frac{|y_i|_2}{\sigma})}\right) \tag{22}$$

The output of the primal-dual algorithm is compared to NNs trained using weight clipping [4] or gradient penalty [10] as shown in Figure 2. The network architectures for these NNs are two fully connected hidden layers with 128 neurons each, ReLU activation, and a linear final activation. The same algorithm with the same hyperparamters is used to train those networks as in [4, 10].

The estimated Wasserstein distance for the critic functions is obtained with Algorithm 1 and is shown in Figure 2 along with the settings. In the 2D case, the critic takes on non-linear shapes, which is hard for the NN to fit, given the regularization constraints. In essence, the problem is underfit by the NN, even after 10^6 iterations. Therefore, current regularizations appear to be too prohibitive to allow for rich structure, even on toy problems. However, without a correct critic function the mathematical justification for optimizing the Wasserstein GAN does not hold. For the Gaussian Mixture Model in Figure 2(i), the second distribution is a NN with 2 hidden layers with 128 neurons and ReLU activations each. Note, that the Gaussian Mixture Model is a sample based dataset. Again the NNs underestimate the Wasserstein distance significantly.

5. GAN Games

In this section we devise an algorithm based on Algorithm 1 to reliably solve GAN games. Recall that a GAN game searches for a solution to Equation (2). This equation is reformulated as follows:

$$\min_{\theta}\left\{\max_{Lip(f)\leq 1}\int_{\Omega} f(x)(d\mu(x)-d\nu_{\theta}(x))\right\} \quad (23)$$

We use Algorithm 1 to accurately solve for f in the discrete space Ω_{d_x}. Then $f(x)$ is constructed by interpolating between those points using the linear interpolation kernel given as follows:

$$\phi(x)=\begin{cases} x+1 & if\ x\in[-1,0[\\ 1-x & if\ x\in[0,1[\\ 0 & else \end{cases} \quad (24)$$

Then Equation (5) is used to calculate the gradient of the Wasserstein distance between μ and ν_{θ}. Then a gradient descent step is performed on the Wasserstein distance. Due to the descent step the optimal critic between μ and ν_{θ} changes and therefore we resolve Algorithm 1 with the new cost vector c. The SCO algorithm is stated in Algorithm 2.

Algorithm 2: SCO Algorithm. We use the default values of $\alpha = 5\cdot 10^{-3}$ and $m = 128$.

```
Compute $c^0 = [d\mu(S_\mathbf{1}) - d\nu_{\theta^0}(S_\mathbf{1}), ..., d\mu(S_\mathbf{N}) - d\nu_{\theta^0}(S_\mathbf{N})]$
Compute $f^0$ using Algorithmus 1 with $c^0$;
for $l \geq 0$ do
    $z_j \sim \nu', z_j \in R^m$;
    $\theta^{l+1} = \theta^l + \alpha\frac{1}{m}\sum_j^m \nabla_\theta f_i(g_{\theta^l}(z_j))$;
    Compute $c^{l+1} = [d\mu(S_\mathbf{1}) - d\nu_{\theta^{l+1}}(S_\mathbf{1}), ..., d\mu(S_\mathbf{N}) - d\nu_{\theta^{l+1}}(S_\mathbf{N})]$
    Compute $f^{l+1}$ using Algorithm 1 with $c^{l+1}$;
end
```

5.1. 1D Experiments

In this experiment, GAN games are used to find the μ, σ parameters of another 1D distribution. This is accomplished using Algorithm 2. The critic function for the 1D case is given by Equation (25), while the generator function is given by $g_{\sigma,\mu}(z) = \sigma z + \mu$, where $\nu' = \mathcal{N}(0,1)$. Therefore all the necessary components for Algorithm 2 are given.

$$f(x)=\sum_{i_1=1}^{N_1} f_{i_1}\phi(x-i_1) \quad (25)$$

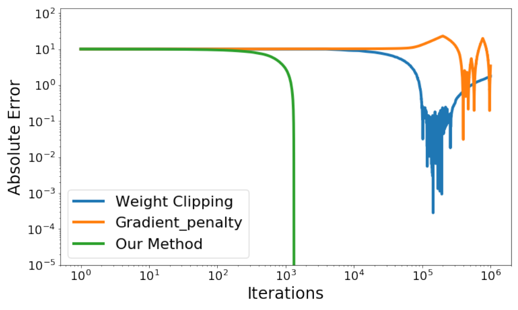
The absolute error between the real μ, σ and the learned ones for various settings is given in Figure 3. Given this setting, the algorithm converges to the correct μ and σ as shown in Figure 3 in 1000 iterations. We stop the algorithm as soon as zero error is obtained. Compared to this training, the Wasserstein GAN games did not converge to the optimal solution on any single test case even for 1D distributions. For this problem, the weight clipping worked better, than the gradient penalty, which we found is due to a faster reaction to overshooting on the target distribution. Still, even after a million iterations the error fluctuates around the optimum. While it is definitely possible to find better hyperparameters for those test cases, we show here that the training of the state-of-the-art is dependent on its hyperparameter and the default parameters do not work well even for the simplest cases. This result also directly follows from the critic shapes and show that we can not converge, due to the non-convergent behavior of the NN critic shown in Figure 1(b,c).

5.2. 2D Experiments

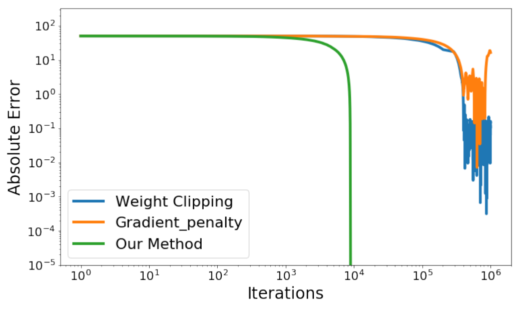
In 2D two μ and σ are used for the generator and therefore the generator function looks as following: $g_{\theta}(z) = (\theta_{\sigma_1}z_1 + \theta_{\mu_1}, \theta_{\sigma_2}z_2 + \theta_{\mu_2})$, where $z \sim \nu'$. In 2D we define $\nu' = \mathcal{N}_2(\mathbf{0}, I)$. The linear interpolation kernel is the same as in 1D and is given in Equation (24). However, the interpolation function in 2D is different and is given as follows:

$$f(x)=\sum_{i_1=1}^{N_1}\sum_{i_2=1}^{N_2} f_{i_1,i_2}\phi(x_1-i_1)\phi(x_2-i_2) \quad (26)$$

This concludes all the necessary components of Algorithm 2. A comparison to the state-of-the-art is shown in Figure 4. Again our algorithm converges rapidly to the optimal solution as shown in Figure 4. Compared to our result the current state-of-the-art fluctuates around the optimal solutions and never actually converges. This time the NN regularized with gradient penalty produced a better output than using weight clipping, as is claimed in their paper [10]. The



(a) $\mu_1 = \mu_2, \sigma_1 \neq \sigma_2$



(b) $\mu_1 \neq \mu_2, \sigma_1 \neq \sigma_2$

Figure 3: Absolute Error plot of GAN games started from the Settings shown in Figure 1(d,g). We compare our method to a NN regularized with weight clipping [4] and a NN regularized with gradient penalty [10]. Note that our algorithm achieves 0 Error.

default parameters seem to be tuned to higher dimensional problems and our results suggest that the current state-of-the-art is reliant on its hyperparameters.

Another experiment using a mixture of 8 Gaussian distributions in 2D is shown in Figure 5. Therein the generator function $g_\theta(z)$ is a NN with 2 hidden layers with 128 neurons, ReLU activation and $\nu' = \mathcal{N}_2(\mathbf{0}, I)$. Many GAN algorithms fail to find all the modes and underfit those modes without extensive hyperparameter and architecture search [14]. For this example $\alpha = 10^{-4}$ and 50.000 samples were used to compute c. While our method, also did not produce 0 distance to the ground truth we did get a superior result to the current state-of-the-art based on the Wasserstein distance of the output and the visual result as shown in Figure 5. Our method used $6 \cdot 10^4$ iterations to train until convergence, while the other algorithms used 10^6 iterations to produce their results.

6. Conclusion

In this work, the inherent problems with Wasserstein GAN optimization are highlighted and new solutions based on convex optimization for low dimensional problems are given. While this approach currently does not scale to high dimensions, it is a step in understanding the inner structure of GAN optimization and how to design critic functions to produce optimal results. The main problem with current approximations is that the critic approximation is far from optimal even on toy problems and therefore it is unclear what we are actually optimizing for. In contrast, by actually using an optimal critic computed using our method we can empirically show rapid convergence. Another advantage of our method is that the Wasserstein distance is calculated optimally and is therefore a useful metric to measure progress. Even on toy examples, the critics learned using the state-of-the-art are far from optimal and better algorithms are needed to tackle this problem.

References

[1] Anonymous. Improving generalization with wasserstein regularization. *International Conference on Learning Representations*, 2018. 2

[2] Anonymous. Improving the improved training of wasserstein gans. *International Conference on Learning Representations*, 2018. 2

[3] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017. 1

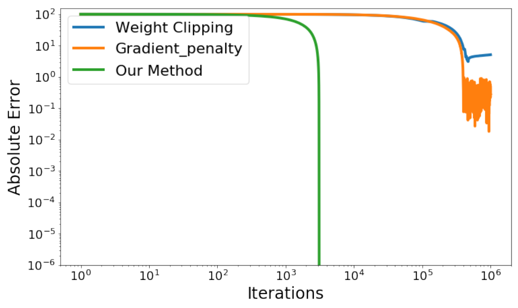
[4] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017. 1, 2, 3, 4, 5, 7, 8

[5] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (gans). *arXiv preprint arXiv:1703.00573*, 2017. 1, 2

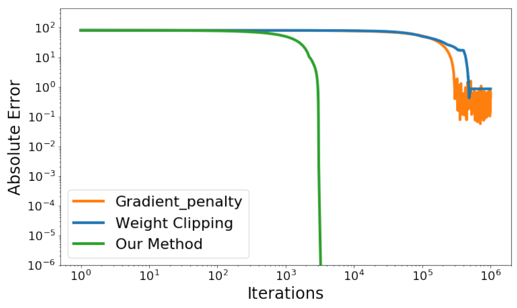
[6] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011. 2, 3, 5

[7] W. Fedus, M. Rosca, B. Lakshminarayanan, A. M. Dai, S. Mohamed, and I. Goodfellow. Many paths to equilibrium: Gans do not need to decrease adivergence at every step. *arXiv preprint arXiv:1710.08446*, 2017. 2

[8] A. Genevay, M. Cuturi, G. Peyré, and F. Bach. Stochastic optimization for large-scale optimal transport. In *Advances in Neural Information Processing Systems*, pages 3440–3448, 2016. 2

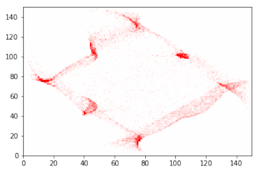


(a) Setting: Figure 2 (a)

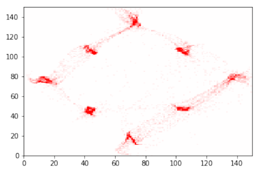


(b) Setting: Figure 2 (e)

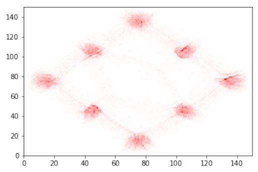
Figure 4: Error plots for fitting 2D distributions showing the sum of absolute errors between the parameters and the real data. The settings correspond to the ones shown in Figure 2(a,e).



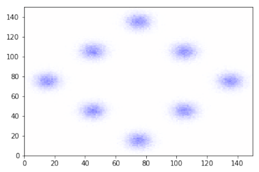
(a) $W(\mu, \nu_\theta) = 5.28$



(b) $W(\mu, \nu_\theta) = 4.03$



(c) $W(\mu, \nu_\theta) = 1.53$



(d) Target Distribution μ

Figure 5: Learning to approximate a sample based distribution μ of 8 gaussians with ν_θ. (a) uses a NN with weight clipping [4], while (b) uses a NN with gradient penalty [10] and (c) uses the SCO algorithm. The Wasserstein distance was calculated using Algorithm 1 on the final output.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1, 2

[10] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017. 1, 2, 3, 4, 5, 6, 7, 8

[11] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[12] Y. Li, A. Schwing, K.-C. Wang, and R. Zemel. Dualing gans. In *Advances in Neural Information Processing Systems*, pages 5609–5619, 2017. 2

[13] A. Makhzani, J. Shlens, N. Jaitly, and I. J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015. 1

[14] L. Mescheder, S. Nowozin, and A. Geiger. The numerics of gans. *arXiv preprint arXiv:1705.10461*, 2017. 7

[15] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016. 2

[16] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 1

[17] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012. 3

[18] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. 2

[19] A. Yadav, S. Shah, Z. Xu, D. Jacobs, and T. Goldstein. Stabilizing adversarial nets with prediction methods. *arXiv preprint arXiv:1705.07364*, 2017. 1, 2