The Convex Connection: Bridging Deep Learning and Optimal Transport

Mouad ID SOUGOU
mouad.id_sougou@telecom-sudparis.eu

Omar ARBI
omar.arbi@student-cs.fr

Abstract

In this report, we examine Montone Gradient Networks (MGN) as an effective method for approximating gradients of convex functions. We explore the core principles, mathematical foundations, and practical applications of MGN, highlighting its effectiveness in modeling gradients within different datasets. Our discussion is based on the approach and formulations introduced mainly in [1].

1. Introduction

Convex optimization is a cornerstone of modern signal processing, machine learning and broadly data science. The gradients of convex functions are particularly ubiquitous, as they are at the core of gradient-based optimization algorithms, optimal transport and domain adaptation tasks. However, manually crafting and designing convex functions and their gradients for complex problems is often impractical. Recent advances in deep learning have enabled data-driven approaches to learn convex functions but learning their gradients directly has for long been a challenging and unexplored problem. That's where the paper introduced by Chaudhari et al. gets into play.

In this project, we focus on a comparative analysis between traditional deep learning models and novel architecture proposed in recent work [1] for learning gradients of convex functions.

Our Primary Contributions

1. **Implementation and Evaluation:** We implement the M-MGN and C-MGN architectures and evaluate its performance in approximating the gradients of known convex functions. We compare its accuracy and efficiency to state-of-the-art methods, such as Input Convex Neural Networks (ICNNs) and Input Convex Gradient Networks (ICGNs).
2. **Optimal Transport Applications:** We apply the proposed architectures to solve optimal transport problems, demonstrating its ability to map 2D distributions to Gaussian priors and augment image datasets for autonomous driving.
3. **Empirical Analysis:** We conduct a thorough empirical analysis of the two architectures, examining its strengths, limitations, and scalability to high-dimensional problems.

This report provides a comprehencive examination of MGN, showcasing their potential for a wide range of applications. By implementing and examining this approach, we aim to provide insights into its practical utility and inspire further research in this direction.

2. Learning Gradient functions using Neural Network

Our aim is to learn the gradients of convex functions by harnessing the capabilities of deep learning architectures. Unlike previous methods such as input-convex neural networks (ICNNs) and input-convex gradient networks (ICGNs), which focus on explicitly learning the convex function itself or its Hessian, the proposed architectures directly learn the gradient of the convex function

We want to learn a monotone gradient function $g(x) : \mathbb{R}^n \to \mathbb{R}^n$ such that $g(x) = \nabla f(x)$ for some convex, twice differentiable function $f(x) : \mathbb{R}^n \to \mathbb{R}$. Now a function $f(x)$ is convex if and only if its Hessian is PSD $H_f(x) = J_g(x) \succeq 0, \quad \forall x \in \mathbb{R}^n$ Therefore, when parameterizing $g(x)$ using a neural network, its Jacobian must be PSD with respect to the input to guarantee convexity.

2.1 Cascaded Network `C-MGN`

The first proposed architecture is a Cascaded Monotone Gradient Network (C-MGN) formulated as follows :

$$z_0 = Wx + b_0 \tag{1}$$

$$z_\ell = Wx + \sigma_\ell(z_{\ell-1}) + b_\ell \tag{2}$$

$$\texttt{C-MGN}(x) = W^\top \sigma_L(z_{L-1}) + V^\top Vx + b_L \tag{3}$$

where the layer outputs z_ℓ, biases b_ℓ, and activation functions σ_ℓ may vary across layers ℓ, but **all** L **layers share the weight matrix** W. For common activation functions, this Jacobian of this architecture is positive semi-definite (PSD).

2.2 Modular Network `M-MGN`

The second architecture introduced by Chaudhari et al [1] is the Modular Monotone Gradient Network (M-MGN). This architecture is formulated as follows:

$$z_k = \mathbf{W}_k x + b_k \tag{4}$$

$$\texttt{M-MGN}(x) = a + V^\top Vx + \sum_{k=1}^{K} s_k(z_k)\mathbf{W}_k^\top \sigma_k(z_k) \tag{5}$$

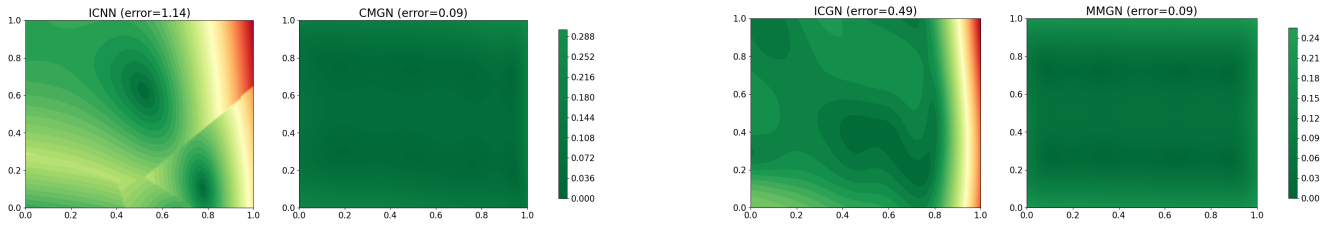
where a is a bias parameter, and the number of modules K can be adjusted based on the application. The activation function σ_k, weight matrix $\mathbf{W}_k$, and bias b_k may vary for each network module. If the scalar-value function s_k is convex, twice differentiable, and nonnegative and can be expressed by $\sigma_k(\cdot) = \nabla s_k(\cdot)$, then the Jacobian of the M-MGN wit respect to its input is PSD.

2.3 Experiment : Gradient Field in 2D dimension

Here we will test our models on the standard problem mentioned in the paper which attempts to estimate the gradient field (see A) of $f : \mathbb{R}^2 \to \mathbb{R}$ over the unit square such that: $f(x) = x_1^4 + \frac{x_2}{2} + \frac{x_1 x_2}{2} + \frac{3x_2^3}{2} - \frac{x_2^3}{3}$

We reproduced the same experiment in the paper using 10000 data points sampled, 50 epochs and with an Adam Optimizer with 0.001 as learning rate.

We can indeed confirm that **C-MGN and M-MGN** are able to learn more precisely the gradient of f with **fewer parameters**



(a) C-MGN and I-CGN

(b) M-MGN and I-CNN

Figure 1: l_2 error maps between the learned gradient and true one for the 4 architectures

	C-MGN	M-MGN	I-CGN	I-CNN
Number of Parameters	14	24	15	76
RMSE	0.09	0.09	0.49	1.14

Table 1: Comparison of models with their parameters and their RMSE

3. Optimal Transport and Brenier Map

3.1 Leveraging Optimal Transport and Brenier Maps in Gradient Neural Networks

In optimal transport (OT), we are usually interested in finding the most efficient way to move mass from one distribution to another, minimizing the cost of transport. In its classical formulation given by Monge [2], OT aims to find a transport plan T that pushes a probability measure α onto another one β, while minimizing the total transport cost c. This can be expressed as :

$$\inf_T \int_{\mathcal{X}} c(x, T(x)) d\alpha(x), \quad T_{\#}\alpha = \beta. \tag{6}$$

A fundamental result in OT theory is *Brenier's Theorem* [3], which states that in the case of **quadratic cost**, the optimal transport map T) is uniquely given by the gradient of a convex function.

Theorem 1 (Brenier). *Let $\mathcal{X} = \mathcal{Y} = \mathbb{R}^d$ and consider the quadratic cost function $c(x, y) = \|x - y\|^2$. If the probability measure α has a density with respect to the Lebesgue measure, then there exists a unique optimal transport map $T : \mathbb{R}^d \to \mathbb{R}^d$ solving Monge's problem. Moreover, this map is the gradient of a convex function φ, meaning $T(x) = \nabla\varphi(x)$.*

This naturally aligns with our context. In fact, by leveraging the properties of monotone gradient networks, we can structure the learning task as an optimal transport problem. Thus, they serve as a practical framework and are particularly well-suited for applications inspired by optimal transport theory.

3.2 Experiments using a gaussian setup

We begin by generating 10000 samples from a standard normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I}_2)$. Then, we apply the transformation $Ax + \mu$ using Cholesky decomposition to obtain the desired source distribution. Specifically, we generate samples from the source distribution $\mathbf{x}_{\text{source}} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where:

$$\boldsymbol{\mu} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}.$$

The target distribution remains a standard normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I}_2)$.

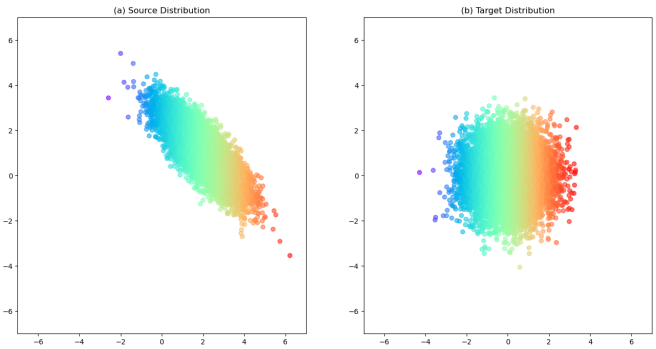


Figure 2: Source and Target Distribution

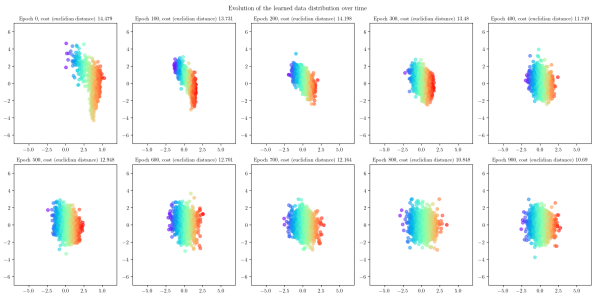
We trained the two proposed architectures, C-MGN and M-MGN, using the squared euclidean cost and minimizing the negative log-likelihood (NLL).

$$c(x, y) = \|x - y\|^2,$$

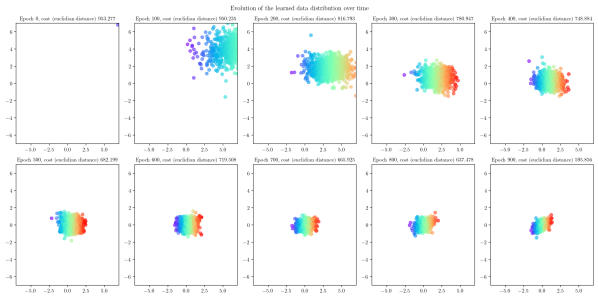
It is known that the optimal cost in this case corresponds to the **Wasserstein distance** (more generally when $c(x, y) = \|x - y\|^p$ defined as :

$$W_2(\alpha, \beta) = \sqrt{\inf_T \int_{\mathcal{X}} \|x - T(x)\|^2 d\alpha(x)} \tag{7}$$

Also, in the case of gaussian distributions, the Kullback-Leibler has a simple formula that can be computed using only the caracteristics of the two distributions. (see Appendix C)



(a) C-MGN



(b) M-MGN

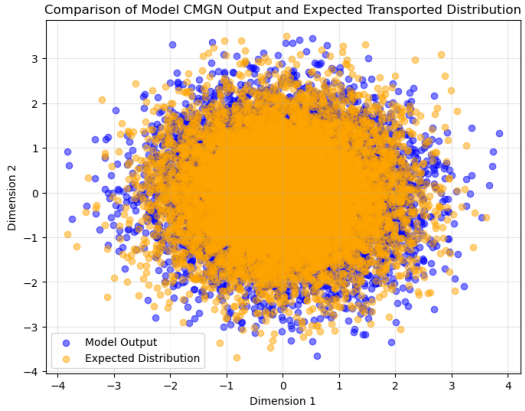
Figure 3: Evolution of the learned data distribution over time

We compare the learned transport maps $g(x)$ from C-MGN and M-MGN to the **closed-form optimal transport map** for Gaussian distributions, given by:

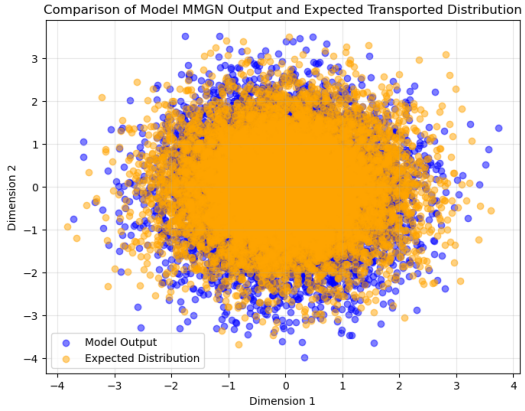
$$g^*(x) = \Sigma_X^{-1/2}(x - \mu_X).$$

For a detailed derivation of this closed-form solution, see Appendix B.

The results are visualized below:



(a) C-MGN vs. Closed-Form Solution



(b) M-MGN vs. Closed-Form Solution

Figure 4: Comparison of learned transport maps to the closed-form solution

From the results, we observe that:

- Both architectures approximate the closed-form solution well with slightly the same **Wasserstein** cost, but **C-MGN** requires fewer parameters and converges faster.
- The learned transport maps $g(x)$ closely match the theoretical optimal map $g^*(x)$, as shown in the comparison plots.

These findings highlight the efficiency of the proposed architectures in addressing the optimal transport problem, especially in the Gaussian scenario where a closed-form solution is available. We will further investigate in this report their ability to extend to multivariate Gaussian distributions and mixtures of Gaussian distributions.

3.3 Color Domain Adaptation

In this section, we discuss the process of transporting a color distribution from one domain to another using Bernier Maps and the networks proposed in the paper. Specifically, we focus on adapting the color distribution of an input image to match that of a target image.

(a) Input Image: Dark Zurich Dataset

(b) Sunset Image: Target Color Distribution

Figure 5: Model Inputs for Optimal Transport

We begin by analyzing the color distribution of the target image, as shown in Figure 6. The target distribution does not perfectly follow a multivariate Gaussian distribution. However, as a first approximation, we assume it can be modeled as a multivariate Gaussian. Later in the report, we will explore more complex distributions, such as mixtures of multivariate Gaussians.

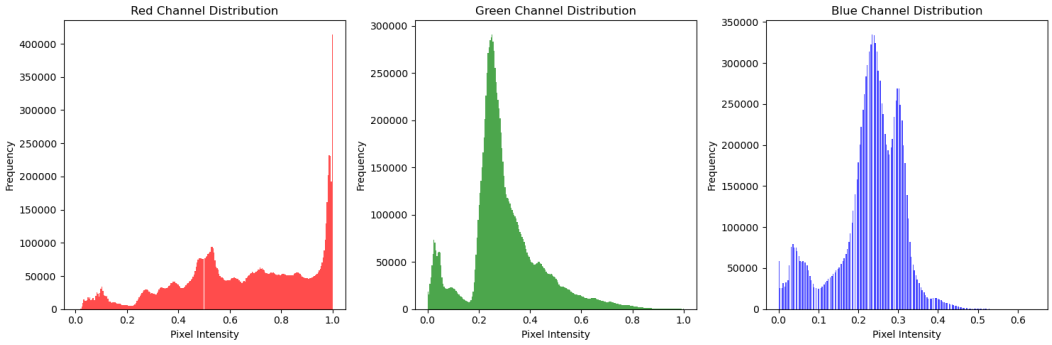


Figure 6: RGB Color Distribution for the Sunset Image

For training, we use a single input image and aim to transport its color distribution to match that of the sunset image. The input data is processed as a single batch, with samples shuffled to improve generalizability. Training is performed on a CPU using both the C-MGN and M-MGN models. We employ the Adam optimizer with a learning rate of $\text{lr} = 0.01$, a weight decay of 0.05, and the Kullback-Leibler Divergence as the loss function. The models are trained for 50 epochs. Below, we present the results for M-MGN, followed by C-MGN, and also demonstrate the application of M-MGN to a novel dataset not discussed in the original paper.

3.4 M-MGN for Color Transport

We train an M-MGN model with 28 parameters. After 50 epochs, the model achieves a training loss of 0.241 and a training cost of 0.179. Visual inspection of the results reveals that the model effectively transports colors even in the early epochs, with satisfactory results by the end of training.

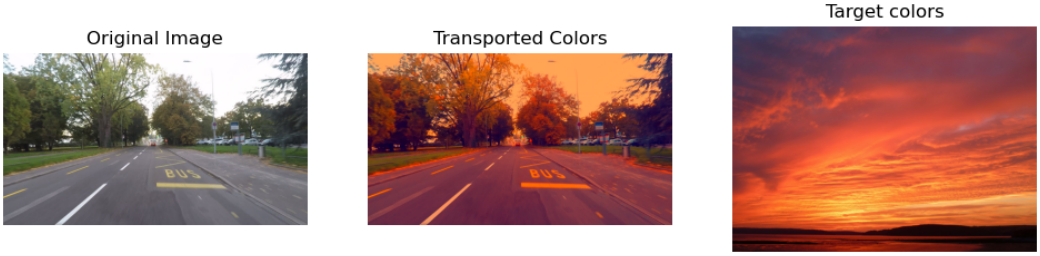


Figure 7: M-MGN Output for the Dark Zurich Input Image at Epoch 50

To better understand the model's performance, we compare the transported colors at the first and last epochs, as shown in Figure 8. Surprisingly, the difference between the two epochs is minimal, suggesting that the model achieves a reasonable color

transport early in the training process.

(a) Transported Colors at Epoch 1

(b) Transported Colors at Epoch 50

Figure 8: Comparison of Model Outputs at Different Training Stages

To further investigate the model's behavior, we analyze the training loss and cost over the course of training, as depicted in Figure 9. The training loss decreases steadily, starting from 3.2 and converging to 0.2 by the final epoch. The cost exhibits oscillations but follows a decreasing trend. The low initial loss value may explain why the visual differences between early and late epochs are minimal, as the model quickly achieves a reasonable approximation of the target distribution.

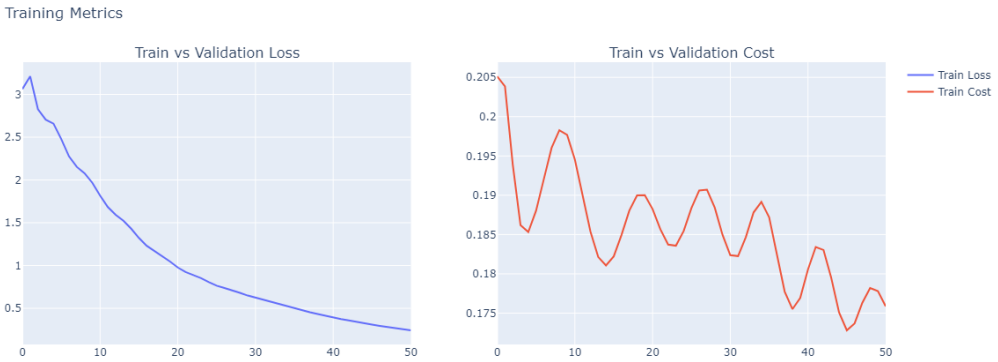


Figure 9: Training Loss and Wasserstein Cost for the M-MGN Model

To evaluate the generalization capability of the learned model, we test it on an unseen image from the Dark Zurich dataset. The results are shown in Figure 10.

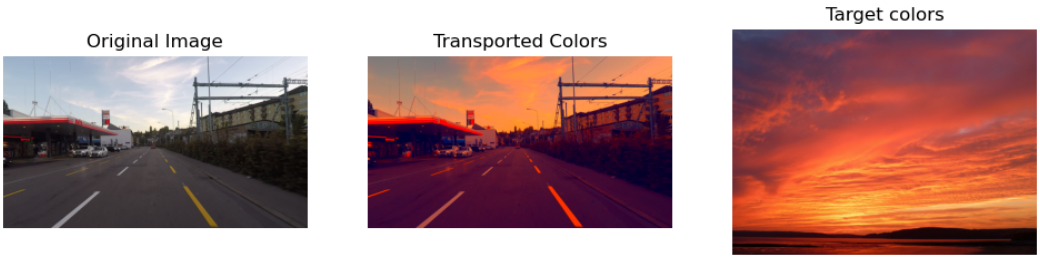


Figure 10: M-MGN Model Output for an Unseen Image from the Dark Zurich Dataset

We observe that the model generalizes effectively to the unseen image, successfully transporting its color distribution to match the target sunset style. This demonstrates the robustness of the M-MGN model and its ability to adapt to new data without requiring additional fine-tuning. The results suggest that the learned mapping is not overfitted to the training data and can be applied to other images within the same domain.

In summary, the M-MGN model demonstrates strong performance in color transport tasks, achieving visually appealing results with minimal training. The low initial loss and rapid convergence suggest that the model is well-suited for this task, even when trained on a single image. However, further exploration is needed to evaluate its performance on more complex distributions and diverse datasets.

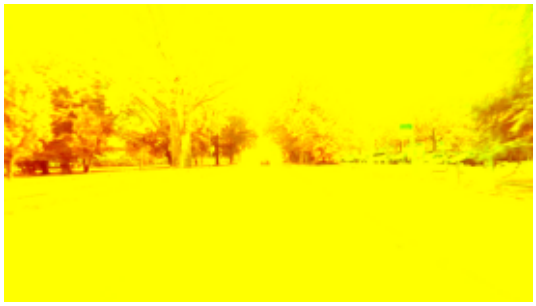
3.5 C-MGN for Color Transport

In this section, we evaluate the performance of the C-MGN model in transporting color distributions. The C-MGN model is trained with 27 parameters, and we conduct two separate training runs with different initializations for the matrix V, as suggested by the authors in reference (3).

The model is trained for 100 epochs, although we note that 50 epochs would have been sufficient for convergence. The results for the two initializations are as follows:

- For an orthogonal initialization of matrix V, the final training loss is 0.06, and the training cost is 0.18.
- For an unconstrained initialization of matrix V, the final training loss is 0.81, and the training cost is 0.20.

Unlike the M-MGN model, the C-MGN model does not achieve optimal color transport in the early stages of training. However, we observe that C-MGN exhibits a faster convergence rate compared to M-MGN. We note a difference of behavior in C-MGN training attributed to the initialization of matrix V. Specifically, an orthogonal initialization leads to more stable training initially, as the gradient eigenvalues are closer to 1. For a detailed mathematical explanation, see Appendix D.



(a) Transported Colors at Epoch 1 with Unconstrained Initialization of V

(b) Transported Colors at Epoch 1 with Orthogonal Initialization of V

Figure 11: Comparison of C-MGN Outputs at Epoch 1 for Different Initializations of V

We further analyze the training dynamics by comparing the loss and cost trajectories for both initializations. These results are provided in Appendix D.2. The orthogonal initialization demonstrates superior performance, achieving lower training loss and cost, which aligns with our observations regarding stability and convergence speed. However both initializations lead to similar results at the end of training and that are similar to those of M-MGN.

After training we test our model on an unseen image from the Dark Zurich Dataset:

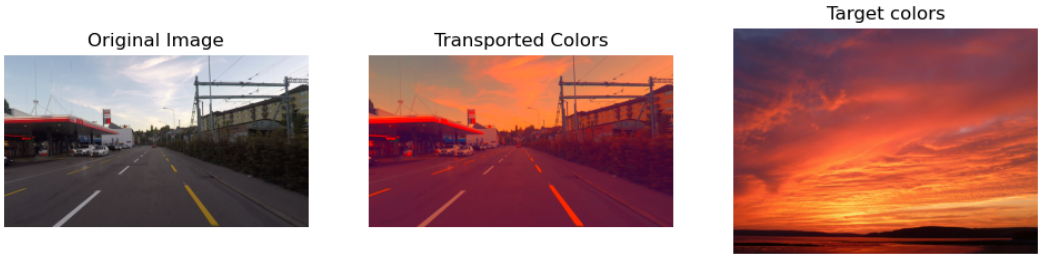


Figure 12: C-MGN Model Output for an Unseen Image from the Dark Zurich Dataset

3.6 M-MGN for Color Transport with a Gaussian Mixture as Target Distribution

In this section, we explore a more complex representation of the target distribution by modeling it as a mixture of multivariate Gaussian distributions. This approach allows us to capture more intricate patterns in the color distribution of the target image. However, the choice of the number of components in the Gaussian mixture model (GMM) is critical, as it directly impacts the model's ability to approximate the target distribution accurately.

To determine the optimal number of components, we analyze the log-likelihood as a function of the number of components, as shown in Figure 13. Using the Elbow method, we select the number of components that balances model complexity and log-likelihood. Based on this analysis, we settle on 5 components, which provides a reasonable trade-off between capturing the target distribution's complexity and avoiding overfitting.

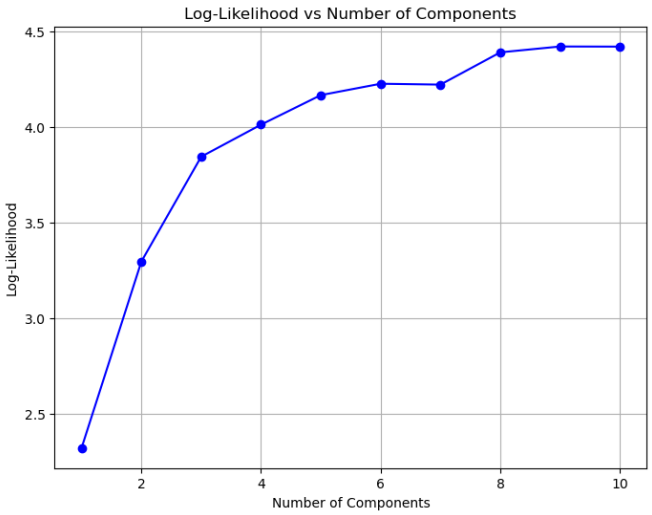


Figure 13: Log-Likelihood vs. Number of Components for the Sunset Image Target Distribution

For training, we use an M-MGN model with 27 parameters. We maintain the same training setup as in the previous experiments, including 50 epochs, an Adam optimizer with a learning rate of $\text{lr} = 0.01$, and the Kullback-Leibler Divergence as the loss function. After 50 epochs, the model achieves a training loss of 24.8 and a training cost of 0.3. Notably, the training loss is significantly higher than that observed for the same model when using a single multivariate Gaussian as the target distribution. This increase in loss is expected, as the Gaussian mixture model introduces additional complexity, making the optimization problem more challenging.

Figure 14: M-MGN Output for a Gaussian Mixture Target Distribution

Despite the higher training loss, the results, as shown in Figure 14, are still visually appealing. However, they are less satisfying compared to the results obtained with a single multivariate Gaussian target distribution. This suggests that while the M-MGN model can handle more complex target distributions, the increased complexity comes at the cost of higher training loss and potentially slower convergence. Further investigation is needed to optimize the model for such complex distributions, possibly by adjusting the architecture or training procedure.

3.7 M-MGN for Novel Dataset: Nighttime Transformation with Northern Lights

To further evaluate the generalization capability of M-MGN, we test it on a novel dataset where the input consists of images of mountains with snow and a clear blue sky, while the target distribution corresponds to nighttime mountain scenes illuminated by northern lights. This setup introduces a significant domain shift, making it a challenging benchmark for the model's adaptability.

For this experiment, we define the target distribution as a simple multivariate Gaussian. The training process remains consistent with previous experiments, maintaining the same model parameters and optimization setup. Specifically, we minimize the Kullback-Leibler (KL) divergence loss between the transformed distribution and the target Gaussian.

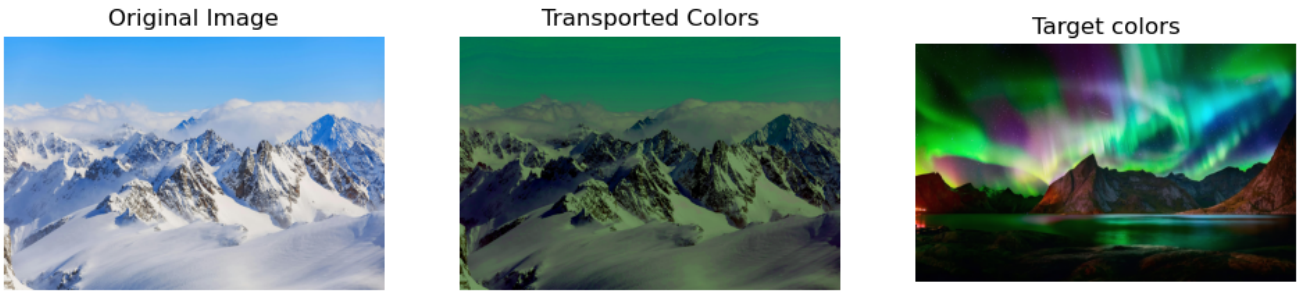


Figure 15: Color transport results using M-MGN.

At the end of training, the model achieves the a train loss of 0.18 and train cost of 0.57

The results demonstrate that M-MGN successfully learns a mapping from daytime mountain landscapes to nighttime scenes, effectively incorporating the overall color transition. However, the transported image still retains some characteristics of the original, particularly in the structural preservation of bright areas. This suggests that while the model effectively captures global color transformations, further refinement may be needed to enhance local texture adaptation and better align with the target domain. Future improvements could include more complex distribution modeling or additional constraints to enforce spatial coherence as done here [4].

4. Conclusion

In this report, we explored the application of Monotone Gradient Networks (MGN) for learning gradients of convex functions. We examined both the Cascaded Monotone Gradient Network (C-MGN) and the Modular Monotone Gradient Network (M-MGN), highlighting their theoretical foundations and practical implementation. Our study demonstrated the efficacy of these architectures in learning convex gradients while preserving monotonicity, ensuring a valid representation of convex function gradients.

Furthermore, we showcased the use of MGN in optimal transport tasks, particularly in image domain adaptation. We evaluated the model on a novel dataset, where we aimed to transform daytime mountain images into nighttime scenes with northern lights. The results indicate that M-MGN successfully captures the target color distribution, though some artifacts persist, suggesting potential areas for improvement in local texture adaptation and structural consistency.

Overall, our findings support the potential of MGNs in various applications, from optimization to generative modeling. Future work could explore refining the architecture to improve its capacity for high-dimensional tasks and extending its application to more complex optimal transport problems.

The full implementation of our work is available on our GitHub repository: https://github.com/Omar-Ar1/Monotone-Gradient-Networks-for-Optimal-Transport

References

[1] S. Chaudhari, S. Pranav, and J. M. Moura, "Learning gradients of convex functions with monotone gradient networks," *Electrical and Computer Engineering*, 2023.

[2] G. Monge, "Mémoire sur la théorie des déblais et des remblais," *Histoire de l'Académie Royale des Sciences de Paris, avec les Mémoires de Mathématique et de Physique pour la même année*, pp. 666–704, 1781.

[3] Y. Brenier, "Polar factorization and monotone rearrangement of vector-valued functions," *Communications on Pure and Applied Mathematics*, vol. 44, no. 4, pp. 375–417, 1991. DOI: 10.1002/cpa.3160440402.

[4] C. McCarter, "Towards backwards-compatible data with confounded domain adaptation," *arXiv preprint arXiv:2301.12720*, 2023. [Online]. Available: https://arxiv.org/abs/2203.12720.

[5] G. Peyré and M. Cuturi, *Computational Optimal Transport* (Foundations and Trends in Machine Learning 5-6). Now Publishers, 2019, vol. 11, pp. 355–607. DOI: 10.1561/2200000073. [Online]. Available: https://arxiv.org/pdf/1803.00567.

A. Gradient Field of the Function

The gradient of the target function is given by:

$$\nabla f(x) = g(x_1, x_2) = \begin{pmatrix} 4x_1^3 + \frac{1}{2}x_2 \\ \frac{1}{2} + \frac{x_1}{2} + 3x_2 - x_2^2 \end{pmatrix}$$

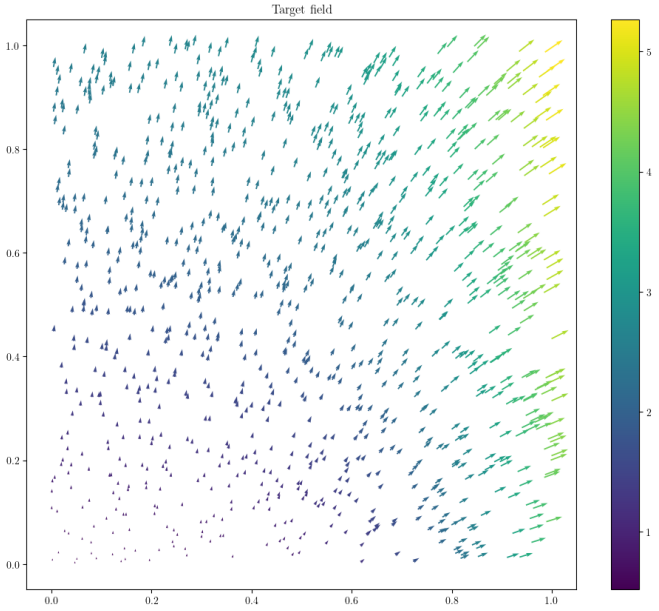


Figure 16: The Gradient Field of $f(x)$

B. Closed-Form Solution for Wasserstein Cost

Given two Gaussian distributions $p_X = \mathcal{N}(\mu_X, \Sigma_X)$ and $p_Y = \mathcal{N}(\mu_Y, \Sigma_Y)$, the **Wasserstein distance** $W_2(p_X, p_Y)$ is defined as:

$$W_2^2(p_X, p_Y) = \|\mu_X - \mu_Y\|^2 + \text{Tr}\left(\Sigma_X + \Sigma_Y - 2(\Sigma_X^{1/2}\Sigma_Y\Sigma_X^{1/2})^{1/2}\right).$$

For the special case where $p_Y = \mathcal{N}(0, I)$, the Wasserstein distance simplifies to:

$$W_2^2(p_X, p_Y) = \|\mu_X\|^2 + \text{Tr}(\Sigma_X) + d - 2\text{Tr}(\Sigma_X^{1/2}),$$

where d is the dimensionality of the space.

The **optimal transport map** $g^*(x)$ that minimizes the Wasserstein cost is given by:

$$g^*(x) = \Sigma_X^{-1/2}(x - \mu_X).$$

This map ensures that the transformed distribution p_g matches the target distribution p_Y.

B.1 Derivation

In the book [5], the optimal transport map between two Multivariate Gaussian distributions μ and ν is given by an affine map:

$$T : x \mapsto m_\nu + A(x - m_\mu),$$

where

$$A = \Sigma_\mu^{-1/2}\left(\Sigma_\mu^{1/2}\Sigma_\nu\Sigma_\mu^{1/2}\right)^{1/2}\Sigma_\mu^{-1/2} = A^T$$

In our case, ν follows a normal distribution and so, $\Sigma_\nu = \mathbf{I}$ and $m_\nu = 0$, hence the result of the optimal transport map.

C. KL Divergence

The KL divergence between two distributions P and Q of a continuous random variable is given by:

$$D_{KL}(p||q) = \int_x p(x) \log \frac{p(x)}{q(x)}$$

The probability density function of a multivariate Normal distribution is:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Now, let our two Normal distributions be $\mathcal{N}(\boldsymbol{\mu_p}, \Sigma_p)$ and $\mathcal{N}(\boldsymbol{\mu_q}, \Sigma_q)$, both k-dimensional.

$$\begin{aligned} D_{KL}(p||q) &= \mathbb{E}_p\left[\log(p) - \log(q)\right] \\ &= \mathbb{E}_p\left[\frac{1}{2}\log\frac{|\Sigma_q|}{|\Sigma_p|} - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_p})^T\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu_p}) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_q})^T\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu_q})\right] \\ &= \frac{1}{2}\log\frac{|\Sigma_q|}{|\Sigma_p|} - \frac{1}{2}k + \frac{1}{2}\mathbb{E}_p\left[(\mathbf{x}-\boldsymbol{\mu_q})^T\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu_q})\right] \end{aligned}$$

Since $(\mathbf{x}-\boldsymbol{\mu_p})^T\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu_p}) \in \mathbb{R}$, we can use the trace operator:

$$\mathbb{E}_p\left[(\mathbf{x}-\boldsymbol{\mu_p})^T\Sigma_p^{-1}(\mathbf{x}-\boldsymbol{\mu_p})\right] = \operatorname{tr}\left\{\Sigma_p\Sigma_p^{-1}\right\} = k$$

For the third term:

$$\mathbb{E}_p\left[(\mathbf{x}-\boldsymbol{\mu_q})^T\Sigma_q^{-1}(\mathbf{x}-\boldsymbol{\mu_q})\right] = (\boldsymbol{\mu_p}-\boldsymbol{\mu_q})^T\Sigma_q^{-1}(\boldsymbol{\mu_p}-\boldsymbol{\mu_q}) + \operatorname{tr}\left\{\Sigma_q^{-1}\Sigma_p\right\}$$

Combining all terms:

$$D_{KL}(p||q) = \frac{1}{2}\left[\log\frac{|\Sigma_q|}{|\Sigma_p|} - k + (\boldsymbol{\mu_p}-\boldsymbol{\mu_q})^T\Sigma_q^{-1}(\boldsymbol{\mu_p}-\boldsymbol{\mu_q}) + \operatorname{tr}\left\{\Sigma_q^{-1}\Sigma_p\right\}\right]$$

Special Case: $q = \mathcal{N}(0, I)$

$$D_{KL}(p||q) = \frac{1}{2}\left[\boldsymbol{\mu_p}^T\boldsymbol{\mu_p} + \operatorname{tr}\left\{\Sigma_p\right\} - k - \log|\Sigma_p|\right]$$

D. Appendix: Proof and Additional Analysis

D.1 Mathematical Explanation for Initialization of V

The stability and convergence speed of the C-MGN model are influenced by the initialization of matrix V. When V is initialized orthogonally, its eigenvalues are close to 1, which ensures that the gradient updates during training remain stable. This can be shown as follows:

Let V be an orthogonal matrix, such that $V^TV = I$, where I is the identity matrix. The eigenvalues of V are therefore $\lambda_i = 1$ for all i. During gradient descent, the update rule for V is given by:

$$V_{t+1} = V_t - \eta\nabla_V\mathcal{L},$$

where η is the learning rate and $\nabla_V\mathcal{L}$ is the gradient of the loss with respect to V.

For an orthogonal initialization, the condition number of V is 1, which ensures that the gradient updates do not suffer from vanishing or exploding gradients. This leads to more stable training and faster convergence.

In contrast, an unconstrained initialization of V may result in eigenvalues that are not close to 1, leading to a higher condition number. This can cause instability in training, as the gradient updates may vary significantly in magnitude, slowing down convergence.

D.2 Training Loss and Cost Comparison

The training loss and cost trajectories for both initializations of V are shown in Figure 17. The orthogonal initialization achieves lower training loss and cost compared to the unconstrained initialization, supporting our observation that orthogonal initialization leads to more stable and efficient training.

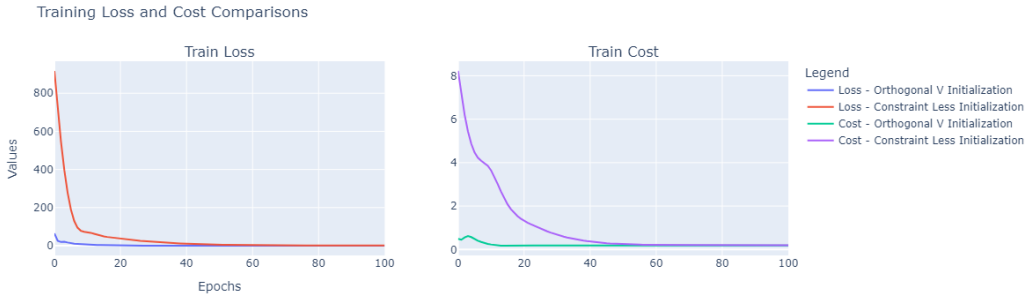


Figure 17: Training Loss and Cost for C-MGN with Different Initializations of V