Counterfactual Explanations via Riemannian Latent Space Traversal

Anonymous Authors ANONYMOUS@EMAIL.COM
Anonymous Institutions

Editors: List of editors' names

Abstract

The adoption of increasingly complex deep models has fueled an urgent need for insight into how these models make predictions. Counterfactual explanations form a powerful tool for providing actionable explanations to practitioners. Previously, counterfactual explanation methods have been designed by traversing the latent space of generative models. Yet, these latent spaces are usually greatly simplified, with most of the data distribution complexity contained in the decoder rather than the latent embedding. Thus, traversing the latent space naively without taking the nonlinear decoder into account can lead to unnatural counterfactual trajectories. We introduce counterfactual explanations obtained using a Riemannian metric pulled back via the decoder and the classifier under scrutiny. This metric encodes information about the complex geometric structure of the data and the learned representation, enabling us to obtain robust counterfactual trajectories with high fidelity, as demonstrated by our experiments in real-world tabular datasets.

Keywords: Counterfactual Explanations, Riemannian Geometry, Deep Generative Models

1. Introduction

Counterfactual explanations are useful to gain insights into classifier behavior, as they can be seen as answering the question *"how should I transform my input data to, as efficiently as possible, change the prediction of the classifier while retaining its identity features?"*. In situations where machine learning algorithms are used to make decisions that affect our lives, counterfactual explanations could be used to give individuals feedback on how they might most easily change themselves to alter the algorithm's decision. Wachter et al. (2017) proposed to optimize the objective function,

$$\mathbf{x}_{\text{CE}} = \arg\min_{\mathbf{x}\in\mathcal{X}} \ell(c(\mathbf{x}), y), \tag{1}$$

where $\mathbf{x}$ is the input data of interest, y is the desired class, c is the classifier, and ℓ is a loss function. However, since a simple stochastic gradient descent (SGD) optimization scheme does not take the geometry of the data manifold into account, the resulting counterfactual $\mathbf{x}_{\text{CE}}$ is not guaranteed to belong to the data manifold.

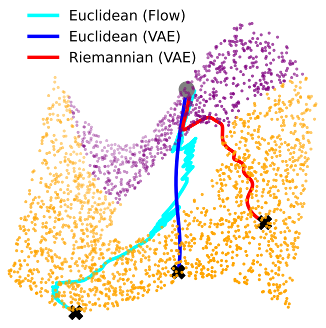


Figure 1: Optimization in the latent space of generative models endowed with Euclidean metric cannot guarantee paths stay on the data manifold. Our Riemannian approach produces realistic trajectories.

As a consequence, $\mathbf{x}_{\mathrm{CE}}$ could represent data that is very different from (i) the original input and (ii) *any* realistic data point. Both issues make the counterfactual less useful for interpretation and for the ability of the individual to take action. Thus, a common strategy is to employ latent generative models such as Variational Autoencoders (VAEs) (Kingma and Welling, 2014) or Normalizing Flows (Papamakarios et al., 2021) to consider counterfactuals onto the data manifold by applying Equation (1) in the latent space endowed with the standard Euclidean metric (Dombrowski et al., 2023). Yet, standard decoders do not preserve the topology of the data, and thus, cannot guarantee that the counterfactual trajectories stay on the data manifold, or even lead to meaningful, smooth transitions (see Figure 1).

We propose: To generate counterfactual explanations via Riemannian latent space traversal. To this end, we introduce a Riemannian geometry in the latent space, induced by a well-calibrated stochastic decoder, to respect the topology of the data manifold. By using a Riemannian Stochastic Gradient Descent (RSGD) optimizer with respect to this geometry, we ensure counterfactuals stay on the data manifold and remain realistic. On top of that, we propose a Riemannian metric which also includes the classifier under observation. Here, we aim to address potentially unstable trajectories caused by complex decision boundaries of the classifier. This allows us to create a latent geometry that not only encodes the data topology but also encourages smoother classification boundaries, yielding more realistic and actionable counterfactual trajectories. Our approach is illustrated in Figure 2.

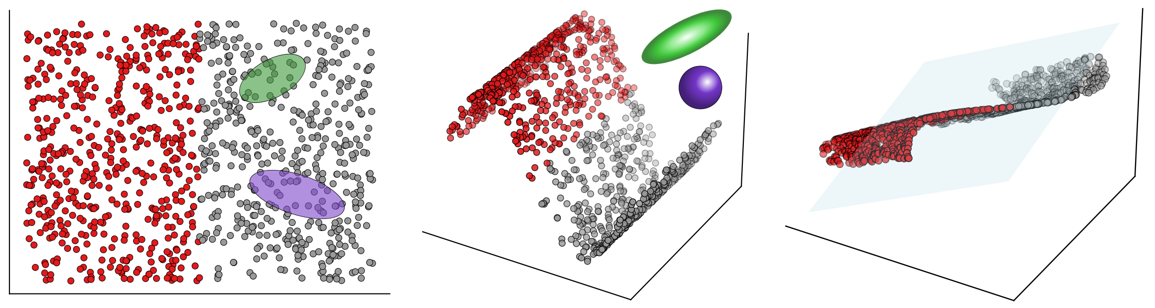
2. Background and Related work

2.1. Riemannian latent space geometry

A Riemannian manifold (do Carmo, 1992; Lee, 2019) is a well-defined metric space, where we can compute the shortest path between two points. A manifold $\mathcal{M}$ can be seen as a smooth d-dimensional surface embedded within a higher dimensional ambient space $\mathcal{X}$, for example $\mathbb{R}^D$. We assume that there exists a parametrization $g : \mathcal{Z} \subseteq \mathbb{R}^d \to \mathcal{U}$, where $\mathcal{Z}$ is the parameter space. The columns of the Jacobian $\mathbf{J}_g(\mathbf{z}) \in \mathbb{R}^{D\times d}$ of $g(\cdot)$ span the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ at a point $\mathbf{x} \in \mathcal{M}$, so a tangent vector can be written as $\mathbf{v} = \mathbf{J}_g(\mathbf{z})\tilde{\mathbf{v}}$, where $\tilde{\mathbf{v}} \in \mathbb{R}^d$ are the *intrinsic coordinates* of the tangent vector on the associated tangent space.

A positive definite matrix $\mathbf{M}_{\mathcal{X}} : \mathcal{X} \to \mathcal{R}^{D\times D}_{\succ 0}$ that changes smoothly across the ambient space $\mathcal{X}$ is a Riemannian metric, and can be used to define the inner product between two tangent vectors $\langle \mathbf{u}, \mathbf{v}\rangle_{\mathbf{x}} = \langle \mathbf{u}, \mathbf{M}_{\mathcal{X}}(\mathbf{x})\mathbf{v}\rangle = \langle \tilde{\mathbf{u}}, \mathbf{J}_g(\mathbf{z})^{\intercal}\mathbf{M}_{\mathcal{X}}(g(\mathbf{z}))\mathbf{J}_g(\mathbf{z})\tilde{\mathbf{v}}\rangle$. Note that the matrix $\mathbf{M}_{\mathcal{Z}}(\mathbf{z}) := \mathbf{J}_g(\mathbf{z})^{\intercal}\mathbf{M}_{\mathcal{X}}(g(\mathbf{z}))\mathbf{J}_g(\mathbf{z}) \in \mathbb{R}^{d\times d}$ is positive definite, and also changes smoothly if $g(\cdot)$ is at least twice differentiable, hence, it is a Riemannian metric in the space $\mathcal{Z}$. This metric $\mathbf{M}_{\mathcal{Z}}(\cdot)$ captures the geometry of $\mathcal{M}$, while respecting the geometry induced by $\mathbf{M}_{\mathcal{X}}(\cdot)$, and intuitively, shortest paths prefer regions where its magnitude is small.

In the ambient space $\mathcal{X}$, we expect the given data $\{\mathbf{x}_n\}_{n=1}^N$ to lie near a low-dimensional manifold $\mathcal{M}$ that has an unknown geometric structure (Bengio et al., 2013). We can approximate the data by considering *latent representations* $\mathbf{z}_n$ and learning a function $\mathbf{x} \approx \hat{g}(\mathbf{z})$. This function does not correspond to the true parametrization $g(\cdot)$ of $\mathcal{M}$, yet it can approximate relatively well the implicit data manifold $\mathcal{M} \approx \hat{g}(\mathcal{Z})$, and hence, induce a *pull-back* metric in $\mathcal{Z}$ which approximates the geometry of $\mathcal{M}$. In practice, $\hat{g}(\cdot)$ can be learned with deep generative models, i.e., stochastic generators (Hauberg, 2018; Arvanitidis et al., 2018)



(a) Latent space and the two distinct Riemannian metrics $\mathbf{M}_{\mathcal{Z}}$ and $\widehat{\mathbf{M}}_{\mathcal{Z}}$.

(b) The data manifold and two ambient metrics $\mathbb{I}_D$ and $\mathbf{M}_{\mathcal{X}}$.

(c) Representation space manifold together with a linear classifier.

Figure 2: Conceptual demonstration: a) the latent space and the two distinct metrics $\mathbf{M}_{\mathcal{Z}}$ and $\widehat{\mathbf{M}}_{\mathcal{Z}}$, b) the data space with the two proposed options for the ambient metrics $\mathbb{I}_D$ and $\mathbf{M}_{\mathcal{X}}$, and c) the representation space and the linear classifier. The ambient metric $\mathbf{M}_{\mathcal{X}}$ in $\mathcal{X}$ captures the geometry of the hidden representation manifold, and we pull it back in $\mathcal{Z}$ to get $\widehat{\mathbf{M}}_{\mathcal{Z}}$, while the metric $\mathbf{M}_{\mathcal{Z}}$ captures only the geometry of the data manifold. See Section 3 for further details.

2.2. Counterfactual explanations

Recently, different counterfactual explanation (CE) methods have been proposed (Verma et al., 2020). Early works address key aspects of counterfactual analysis, such as sparsity (Guidotti et al., 2018; Dandl et al., 2020), actionability (Ustun et al., 2019), diversity (Russell, 2019; Mothilal et al., 2020), and causality (Mahajan et al., 2019; Karimi et al., 2020b). Borisov et al. (2022) underlines two categories of CE methods: One assuming feature independence (Karimi et al., 2020a) and one focusing on feature dependencies, exploring data geometry (Pawelczyk et al., 2020; Downs et al., 2020). Note that CEs are different from counterfactual editing whose goal is *not* to explain changes in classifier's predictions (Melistas et al., 2024). When data lie on a manifold, attempts have been made to maintain CEs close to the data by constructing connective graphs (Poyiadzi et al., 2020) or employing distance measures that can identify feasible paths (Domnich and Vicente, 2024). Other approaches approximate manifold spaces with latent generative models, including VAE- (Dhurandhar et al., 2018; Joshi et al., 2019; Antorán et al., 2020), GAN- (Singla et al., 2019), flow- (Dombrowski et al., 2021; Duong et al., 2023), and diffusion-based methods (Jeanneret et al., 2022; Weng et al., 2023; Pegios et al., 2024; Madaan and Bedathur, 2023). In this work, we employ VAEs to induce a Riemannian latent space geometry, enabling the generation of realistic CEs with trajectories that remain on the data manifold.

3. Theory

Let a dataset $\{\mathbf{x}_n, y_n\}_{n=1}^{N}$ in an ambient space $\mathcal{X} = \mathbb{R}^D$ where $y_n \in \{0, 1\}$. Dombrowski et al. (2023) generate CEs by applying SGD in the latent space $\mathcal{Z} = \mathbb{R}^d$ of a generative model, moving through latent space taking steps with size η of the form: $\mathbf{z}' = \mathbf{z} - \eta \cdot \nabla \ell(c(\hat{g}(\mathbf{z}), y))$.

Yet, this risks "falling off" the data manifold, forcing the generative model to extrapolate beyond its training data, resulting in unrealistic counterfactual paths (see Figure 1).

3.1. Basic pull-back metric

The standard VAE is used for learning stochastic generative models with Gaussian decoders (Kingma and Welling, 2014). Specifically, we consider the stochastic decoder $\mathbf{x} = g_\varepsilon(\mathbf{z}) = \mu(\mathbf{z}) + \sigma(\mathbf{z}) \odot \epsilon$, with $\epsilon \sim \mathcal{N}(0, \mathbb{I}_D)$ and $\odot$ being the point-wise product. When the ambient space $\mathcal{X}$ is endowed with the Euclidean metric $\mathbf{M}_\mathcal{X}(\mathbf{x}) = \mathbb{I}$, $\forall \mathbf{x} \in \mathcal{X}$, the expected pull-back metric in the latent $\mathcal{Z} = \mathbb{R}^d$ space is equal to

$$\mathbf{M}_\mathcal{Z}(\mathbf{z}) = \mathbb{E}_\varepsilon[\mathbf{J}_{g_\varepsilon}(\mathbf{z})^\intercal \mathbf{J}_{g_\varepsilon}(\mathbf{z})] = \mathbf{J}_\mu^\intercal(\mathbf{z})\mathbf{J}_\mu(\mathbf{z}) + \mathbf{J}_\sigma(\mathbf{z})^\intercal \mathbf{J}_\sigma(\mathbf{z}) \tag{2}$$

where $\mu \colon \mathcal{Z} \to \mathcal{X}$ and $\sigma \colon \mathcal{Z} \to \mathbb{R}_{>0}$ are neural networks (Arvanitidis et al., 2018). We induce a meaningful geometric structure in the latent space whether the generator's uncertainty is well-calibrated, i.e, $\sigma(\mathbf{z}) \to 0$ when $\mathbf{z}$ is near the training latent codes, otherwise $\sigma(\mathbf{z}) \to +\infty$ (Hauberg, 2018). We ensure a well-calibrated uncertainty by using $\sigma(\mathbf{z}) = 1/\sqrt{\gamma(\mathbf{z})}$, with $\gamma \colon \mathcal{Z} \to \mathbb{R}_{>0}$ a Radial Basis Function network (Que and Belkin, 2016) based on the latent representation, satisfying the criteria above (Arvanitidis et al., 2018).

Under the mild conditions on the smoothness of $g_\varepsilon(\cdot)$ and the behavior of $\sigma(\cdot)$, $\mathbf{M}_\mathcal{Z}(\mathbf{z}) \in \mathbb{R}^{D\times D}_{\succ 0}$ constitutes a Riemannian metric in $\mathcal{Z}$, which captures the geometry of the data manifold in the ambient space $\mathcal{M} \subset \mathcal{X}$, and hence the shortest paths between points in $\mathcal{Z}$ respect this geometry. Therefore, as the cost of moving off the data manifold increases with the VAE uncertainty, the result is the corresponding shortest paths to prefer staying near the data manifold. Thus, the topology of the data manifold is preserved in a soft sense: It is *possible*, but *expensive*, for shortest paths, and hence counterfactual trajectories to move outside the data manifold, meaning they do not do so unless necessary. This allows us to generate CEs using RSGD w.r.t. the pull-back metric defined in Equation (2),

$$\mathbf{z}' = \mathbf{z} - \eta \cdot \frac{\mathbf{r}}{\|\mathbf{r}\|_2}, \quad \text{where} \quad \mathbf{r} = \mathbf{M}_\mathcal{Z}^{-1}(\mathbf{z})\nabla_\mathbf{z} f_y(\mathbf{z}) \quad \text{and} \quad f_y(\mathbf{z}) = \ell(\mathbb{E}[g_\varepsilon(\mathbf{z})], y). \tag{3}$$

We normalize the Riemannian gradient $\mathbf{r}$ as we are interested in the update direction, and to avoid vanishing gradients near the boundary of the latent data support in $\mathcal{Z}$. Note that this Riemannian metric makes the optimization trajectory invariant under reparametrization, i.e., another latent representation that generates the same density in $\mathcal{X}$ will provide the same trajectory on the data manifold. Hence, using RSGD, we encourage trajectories to align with the geometric structure of the latent codes, staying on the data manifold.

3.2. Enhanced pull-back metric

Equation (2) assumes a Euclidean metric on the ambient space $\mathcal{X}$, but we are able to consider other Riemannian metrics on $\mathcal{X}$ to better capture the desired properties of a problem, enriching the latent space geometry with additional information (Arvanitidis et al., 2021).

To this end, we consider a differentiable ambient space classifier $c \colon \mathcal{X} \to [0, 1]$, which we parameterize with a neural network, and thus can be expressed as $c(\mathbf{x}) = \text{sigmoid}(\mathbf{w}^\intercal h(\mathbf{x}))$, where $h \colon \mathcal{X} \to \mathcal{H} = \mathbb{R}^H$ with typically $H \gg D$, and $\mathbf{w} \in \mathbb{R}^H$ the weights of the last layer.

The mapping $h(\cdot)$ is the learned representation of the final layer, spanning a D-dimensional submanifold of $\mathbb{R}^H$ under mild conditions. By endowing $\mathcal{H}$, where the classification boundary is linear, with an Euclidean metric, we derive the pull-back Riemannian metric in $\mathcal{X}$,

$$\mathbf{M}_{\mathcal{X}}(\mathbf{x}) = \mathbf{J}_h(\mathbf{x})^{\intercal}\mathbf{J}_h(\mathbf{x}). \tag{4}$$

This metric represents in $\mathcal{X}$ the corresponding local geometry of the learned representation $\mathcal{H}$. Assuming well-separated classes in $\mathcal{H}$, we expect $\mathbf{M}_{\mathcal{X}}(\mathbf{x})$ to be small in regions of $\mathcal{X}$ where the classes remain constant and to increase near the decision boundary. In principle, Equation (4) could used for counterfactual optimization directly in $\mathcal{X}$ that respects the geometry of the learned representation. Yet, as $\mathcal{X}$ is typically high-dimensional, it is more advantageous to exploit a generative model and pull $\mathbf{M}_{\mathcal{X}}(\mathbf{x})$ back to the associated latent space $\mathcal{Z}$ using the stochastic generator $\mathbf{x} = g_\varepsilon(\mathbf{z})$ to get the Riemannian metric,

$$\mathbf{M}^{\varepsilon}_{\mathcal{Z},\mathcal{X}}(\mathbf{z}) = \mathbf{J}_{g_\varepsilon}(\mathbf{z})^{\intercal}\mathbf{M}_{\mathcal{X}}(g_\varepsilon(\mathbf{z}))\mathbf{J}_{g_\varepsilon}(\mathbf{z}), \tag{5}$$

which is stochastic due to the random variable ε. Figure 2 illustrates our approach using the classifier-guided enhanced pull-back metric. For practical purposes, we use this metric

$$\widehat{\mathbf{M}}_{\mathcal{Z}}(\mathbf{z}) = \mathbb{E}_\varepsilon[\mathbf{M}^{\varepsilon}_{\mathcal{Z},\mathcal{X}}(\mathbf{z})] \approx \mathbf{J}_\mu(\mathbf{z})^{\intercal}\mathbf{M}_{\mathcal{X}}(\mu(\mathbf{z}))\mathbf{J}_\mu(\mathbf{z}) + \mathbf{J}_\sigma(\mathbf{z})^{\intercal}\mathbf{M}_{\mathcal{X}}(\mu(\mathbf{z}))\mathbf{J}_\sigma(\mathbf{z}). \tag{6}$$

which is a good approximation of the expected metric for well-calibrated decoders (Arvanitidis et al., 2021). We perform CE optimization (Equation (3)) using RSGD w.r.t to the classifier-guided enhanced pull-back metric $\widehat{\mathbf{M}}_{\mathcal{Z}}(\mathbf{z})$, and refer to it as RSDG-C.

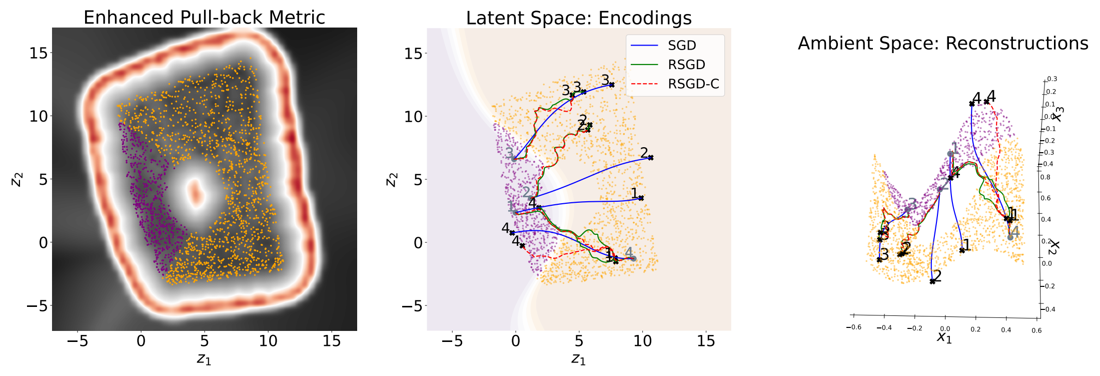
4. Experiments

First, we show that, unlike Normalizing Flows (Dombrowski et al., 2021), our induced geometry softly preserves the dataset topology by heavily penalizing counterfactual trajectories through data-sparse regions. Next, we compare Euclidean SGD on VAE's latent space with our proposed Riemannian optimizers, RSGD and RSGD-C, on real-world tabular datasets. Our source code will be publicly available at `http://ANON-REPO-URL/`.

4.1. Topology preservation and counterfactual trajectories

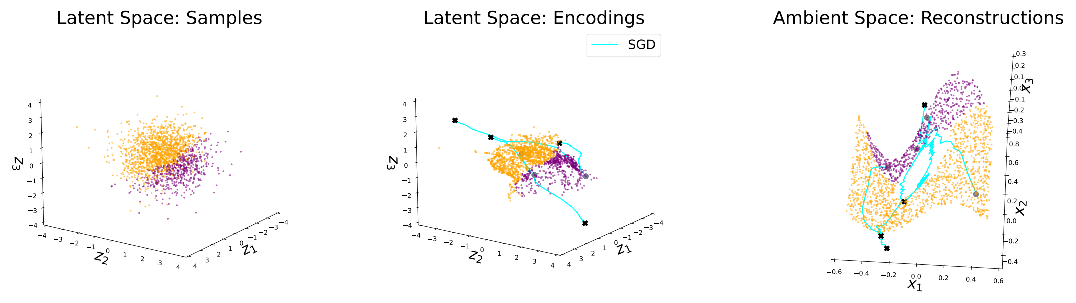
We construct a surface in $\mathcal{X} = \mathbb{R}^3$ as $\mathbf{x} = [\mathbf{z}, 0.25 \cdot \sin(z_1)] + \varepsilon$ where $z_j \sim \mathcal{U}(0, 2\pi),\ j = 1, 2$ and $\varepsilon \sim \mathcal{N}(0, 0.1^2 \cdot \mathbb{I}_3)$, with a hole in the middle by removing the points in the center with radius $||\mathbf{z}||_2 < 0.2$ before the mapping in $\mathbb{R}^3$, and assign labels $y = (\text{sign}(z_2 - 2.5 \cdot z_1^2) + 1) \cdot 0.5$. We train a Normalizing Flow with a latent space of $d = D = 3$, a VAE with $d = 2$, and a classifier with a representation space of $H = 32$ to pull geometry back to the latent space.

The proposed metric is illustrated in Figure 3(*a*), with a black-white-red color scheme representing traversal costs in the latent space, from low to high. The red area in the center indicates the high cost of crossing the data-sparse hole, while the white-and-red boundary highlights the high cost of leaving the data-populated region. Figures 3(*b*) and 3(*c*) show, from the viewpoint of the latent space and ambient space, respectively, that naive Euclidean latent space traversal using standard SGD fails to respect the data manifold's topology as it is neither encoded by the classifier nor the latent space geometry, allowing counterfactuals



(a) Proposed Metric (b) Latent space (c) CE trajectories

Figure 3: Optimization paths in the latent space of VAE using our proposed metric.



(a) Samples from prior (b) Latent space (c) CE trajectories

Figure 4: Optimization paths in the latent space of Normalizing Flow.

to stray from the data distribution. In contrast, using pullback metrics from the ambient space (RSGD) or the target of the classifier (RSGD-C) addresses this by making it costly for counterfactual trajectories to deviate from the data. Next, Figure 4(a) displays the prior distribution of trained Normalizing Flow, which poorly represents the hole. Figures 4(b) and 4(c) show its trajectories in the latent and ambient spaces, respectively, revealing their failure to capture the topology of the dataset and resulting in paths that pass through the hole.

4.2. Improved fidelity of counterfactual trajectories

4.2.1. Datasets

We evaluate our optimizers, against standard Euclidian SGD on two widely used tabular datasets. **Adult** (Becker and Kohavi, 1996) includes census records with $D = 13$ mixed-type features, intending to predict whether an individual's income exceeds \$50K/year. **Give Me Some Credit** (GMC) (Kaggle-Competition, 2011) includes $D = 10$ continuous features,

#Iter	Constrains	Optimizer	$\mathcal{L}_D \downarrow$	$\mathcal{L}_0 \downarrow$	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_2 \downarrow$	$\mathcal{L}_\infty \downarrow$	$c(\mathbf{x}_{\mathrm{CE}}) \uparrow$	FR $\uparrow$	violation $\downarrow$
50	✗	SGD	0.20±0.29	6.86±0.83	2.97±0.98	2.22±0.97	0.95±0.11	**0.93±0.07**	**0.99**	1.06
		RSGD	**0.04±0.05**	**6.01±0.08**	**1.00±0.43**	**0.40±0.36**	0.48±0.24	0.66±0.22	0.70	**0.97**
		RSGD-C	**0.04±0.07**	6.12±0.33	1.03±0.52	0.42±0.46	**0.47±0.25**	0.63±0.21	0.67	**0.97**
50	✓	SGD	0.12±0.14	6.26±0.54	1.60±0.73	0.87±0.68	0.69±0.21	**0.91±0.07**	**0.99**	1.00
		RSGD	**0.05±0.06**	**6.01±0.09**	**0.91±0.34**	**0.31±0.24**	**0.43±0.17**	0.66±0.21	0.71	**0.96**
		RSGD-C	**0.05±0.06**	6.09±0.30	0.94±0.45	0.35±0.37	0.44±0.22	0.63±0.21	0.67	**0.96**
100	✗	SGD	0.33±0.38	7.08±1.07	3.37±1.21	2.58±1.21	0.96±0.10	**0.94±0.06**	**0.99**	1.10
		RSGD	**0.06±0.06**	**6.02±0.17**	1.44±0.52	0.79±0.50	0.70±0.27	0.85±0.14	0.96	**0.98**
		RSGD-C	0.07±0.07	6.14±0.37	**1.41±0.61**	**0.73±0.60**	**0.63±0.27**	0.83±0.14	0.95	**0.98**
100	✓	SGD	0.13±0.14	6.37±0.66	1.72±0.85	0.98±0.80	0.71±0.22	**0.92±0.06**	**0.99**	1.01
		RSGD	**0.08±0.06**	**6.03±0.18**	**1.17±0.38**	**0.49±0.33**	0.55±0.18	0.85±0.14	0.95	**0.98**
		RSGD-C	0.09±0.07	6.13±0.35	1.20±0.52	0.52±0.48	**0.53±0.22**	0.83±0.14	0.95	**0.98**
150	✗	SGD	0.41±0.49	7.26±1.21	3.65±1.44	2.84±1.42	0.96±0.10	**0.94±0.06**	**0.99**	1.14
		RSGD	**0.05±0.07**	**6.07±0.29**	**1.80±0.47**	**1.14±0.44**	0.88±0.17	0.89±0.13	0.97	0.99
		RSGD-C	**0.05±0.08**	6.20±0.44	1.83±0.61	1.15±0.62	**0.83±0.20**	0.89±0.12	0.97	0.99
150	✓	SGD	0.14±0.15	6.42±0.71	1.79±0.90	1.04±0.84	0.72±0.22	**0.92±0.06**	**0.99**	1.02
		RSGD	**0.09±0.07**	**6.09±0.32**	**1.36±0.45**	**0.65±0.44**	**0.63±0.19**	0.88±0.13	0.97	**0.98**
		RSGD-C	**0.09±0.08**	6.22±0.40	1.46±0.58	0.77±0.59	0.66±0.21	0.88±0.12	0.97	**0.98**

Table 1: Adult dataset. To standardize comparison, we evaluate optimizers with (✓) and without (✗) fidelity constraints *during* optimization for different numbers of steps.

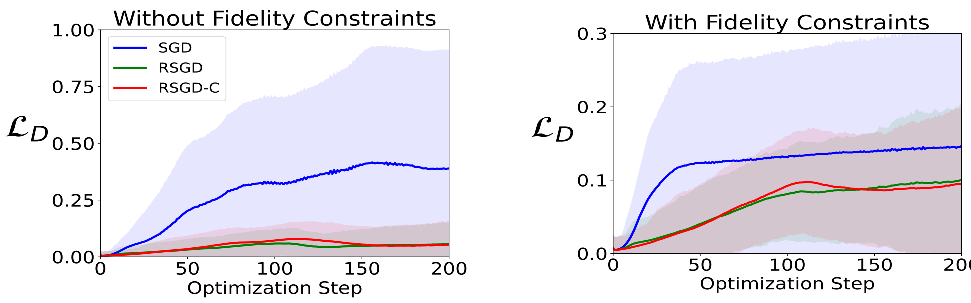


Figure 5: Adult Dataset. Distance to closest training sample as a function of gradient steps.

aiming to predict whether an individual will face financial distress. Both datasets include a subset of features that are considered immutable, e.g., age, as altering them cannot provide actionable feedback. More information can be found in the Appendix.

4.2.2. Evaluation Metrics

We follow the evaluation protocol of Pawelczyk et al. (2021), by measuring *violation*, i.e, how often a CE method breaches user-defined constraints (immutable features), as well as *validity* with Flip Ratio (FR), representing the success rate of CEs classified as the target label, along with their confidence $c(\mathbf{x}_{\mathrm{CE}})$. The quality of valid CEs is measured by their *closeness* to the original factual input using $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$ and $\mathcal{L}_\infty$ distances. To measure *realism*, i.e., how close the CE is to the data manifold, we compute the local Euclidean distance of CE to the closest training data point in the ambient space denoted as $\mathcal{L}_D$.

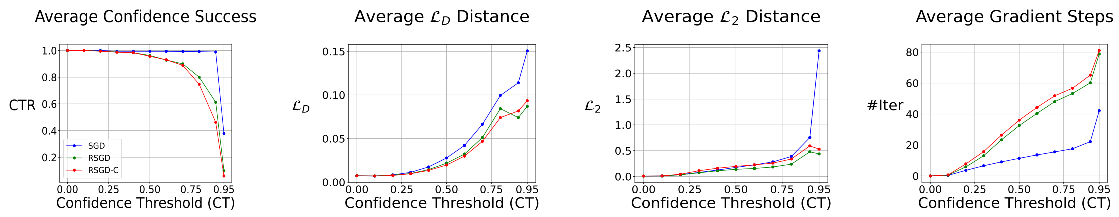


Figure 6: Adult Dataset. Evaluation of CEs as a function of confidence threshold (CT). Confidence Threshold Ratio (CRT) quantifies confidence success, i.e., the proportion of CEs achieving a specified CT. $\mathcal{L}_D$ measures fidelity to the data manifold (realism), $\mathcal{L}_2$ the fidelity to the initial data point (closeness), and #Iter the number of required gradient steps to achieve a specific confidence level.

4.2.3. Experimental Setup

We train a VAE with $d = 5$, and a classifier with a representation space of $H = 24$ to pull geometry back to the latent space, for both datasets. Implementation details can be found in the Appendix. We evaluate different optimizers for generating CEs by traversing the latent space of the VAE, using Equation (2), where standard SGD uses an Euclidean metric with $\mathbf{M}_{\mathcal{Z}}^{-1}(\mathbf{z}) = \mathbb{I}$, and setting $\eta = 0.1$ for all methods. To standardize comparison, we fix the number of iterations across methods. Following Pawelczyk et al. (2021), we generate CEs for correct negative predictions, resulting in 7859 samples from 9268 negatives in the Adult test set and 1220 from 1913 in the GMC test set, with the target label set to the positive class. Joshi et al. (2019) introduce fidelity constraints to keep CEs close to the original input $\mathbf{x}$ during optimization in the VAE's latent space, using the counterfactual loss: $\ell = \text{BCE}(c(\hat{g}(\mathbf{z}))) + \alpha\|\hat{g}(\mathbf{z}) - \mathbf{x}\|_2$, where BCE is binary cross entropy and α regularization parameter. However, our Riemannian optimizers, namely RSGD and RSGD-C, naturally respect the data's topology, eliminating the need for such constraints. To showcase this, we evaluate all optimizers both with ($\alpha = 0.1$) and without ($\alpha = 0$) fidelity regularization.

4.2.4. Results

Table 2 lists results for Adult dataset, with Figure 5 illustrating local Euclidean distance $\mathcal{L}_D$ across gradient steps. Both RSGD and RSGD-C maintain counterfactuals close to the data manifold ($\mathcal{L}_D$) and the original input ($\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_\infty$) without requiring fidelity constraints during optimization. After 100 iterations, our optimizers generate CEs closer to the original input compared to SGD with fidelity constraints and 50 steps while maintaining comparable validity and fewer violations of immutable features. As expected our Riemannian approach requires more gradient steps to achieve high-confidence CEs. To compare optimizers across varying confidence thresholds (CTs), we assess the proportion of CEs successfully generated at each threshold, measured by the Confidence Threshold Ratio (CTR), by selecting the first CE reaching a CT from the counterfactual trajectories. Figure 6 illustrates the average required gradient steps, as well as the fidelity to both the data manifold and the original input, where our approach demonstrates improved fidelity across different confidence levels.

Similarly, we provide the results for the GMC dataset in Table 2 and Figure 7 in the Appendix. RSGD and RSGD-C consistently generate high-fidelity CEs but with the cost of less confident CEs compared to SGD. At the same time, RSGD-C pushes CEs further from an initial factual point to achieve CEs at the same confidence level compared to RSGD, but still it maintains them close to the data manifold, which could potentially be explained by different counterfactual trajectories guided by the geometry of the classifier.

5. Discussion

We generate CEs via Riemannian latent space traversal by employing the Riemannian metric induced by the stochastic generator of a deep generative model. Using RSGD we perform CE optimization in the latent space of a VAE, where the metric acts as a preconditioner that keeps the trajectory near the latent data manifold. In addition, we explore a classifier-guided enhanced latent space Riemannian metric, which pulls back the Riemannian metric from the ambient space and captures the geometry of the learned hidden representation manifold induced by the classifier under observation, yielding our RSGD-C optimizer.

Different from SGD in the latent space of a Normalizing Flow (Dombrowski et al., 2021), we ensure that the CE trajectories stay on the data manifold without crossing unpopulated regions in the data (Figures 3 and 4). Our experiments in real-world datasets demonstrate that our approach results in CEs that indeed stay closer to the data manifold (Figure 5) compared to SGD, giving high fidelity with respect to the original factual input and comparable validity (Table 2 and Figure 6). In general, RSGD and RSGD-C perform similarly but for the GMC dataset, we observe that trajectories for RSGD-C are pushed further from the original input compared to RSGD to achieve comparable confidence levels (Figure 7). However, as RSGD-C is guided by the classifier under observation, we hypothesize that these counterfactual trajectories might represent more realistic and actionable paths. This is also illustrated by case 4 in Figure 3 where RSGD and RSGD-C generate very different paths, with RSGD running more smoothly through the data.

Societal impact. Counterfactual explanations are widely used to analyze classifier behavior. A method that generates more realistic counterfactuals can positively impact society by providing debugging tools and promoting fairness. However, like many AI models, this approach could also be misused for malicious purposes.

Limitations & Outlook. We rely on a well-calibrated stochastic generator to approximate the data and provide reliable uncertainty estimates. Without this, the Riemannian metric in the latent space cannot effectively capture the geometry of the data manifold. While VAEs help induce Riemannian geometry in the latent space, existing methods (Detlefsen et al., 2019; Arvanitidis et al., 2018) for handling uncertainty are limited to low-dimensional spaces. For relatively higher-dimensional data, decoder ensembling (Syrota et al., 2024) or Laplacian Autoencoders (Miani et al., 2022) may be more suitable. Even though frameworks, e.g., Stochman (Detlefsen et al., 2021) support Jacobian computation, the computational complexity increases for complex architectures. Adapting our method to modern models like Diffusion Models (Dhariwal and Nichol, 2021), where understanding Riemannian latent space geometry (Park et al., 2023) or developing Riemannian Diffusion Models (Huang et al., 2022) is an open area of research, remains a topic for future work.

References

Javier Antorán, Umang Bhatt, Tameem Adel, Adrian Weller, and José Miguel Hernández-Lobato. Getting a clue: A method for explaining uncertainty estimates. *arXiv preprint arXiv:2006.06848*, 2020.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations (ICLR)*, 2018.

Georgios Arvanitidis, Søren Hauberg, and Bernhard Schölkopf. Geometrically enriched latent spaces. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In *International Conference on Parallel Problem Solving from Nature*, pages 448–469. Springer, 2020.

Nicki S Detlefsen, Martin Jørgensen, and Søren Hauberg. Reliable training and estimation of variance networks. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Nicki S. Detlefsen, Alison Pouplin, Cilie W. Feldager, Cong Geng, Dimitris Kalatzis, Helene Hauschultz, Miguel González-Duque, Frederik Warburg, Marco Miani, and Søren Hauberg. Stochman. *GitHub. Note: https://github.com/MachineLearningLifeScience/stochman/*, 2021.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in neural information processing systems*, 31, 2018.

M.P. do Carmo. *Riemannian Geometry*. Mathematics (Boston, Mass.). Birkhäuser, 1992.

Ann-Kathrin Dombrowski, Jan E Gerken, and Pan Kessel. Diffeomorphic explanations with normalizing flows. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021.

Ann-Kathrin Dombrowski, Jan E Gerken, Klaus-Robert Müller, and Pan Kessel. Diffeomorphic counterfactuals with generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Marharyta Domnich and Raul Vicente. Enhancing counterfactual explanation search with diffusion distance and directional coherence. In *World Conference on Explainable Artificial Intelligence*, pages 60–84. Springer, 2024.

Michael Downs, Jonathan L Chu, Yaniv Yacoby, Finale Doshi-Velez, and Weiwei Pan. Cruds: Counterfactual recourse using disentangled subspaces. *ICML WHI*, 2020:1–23, 2020.

Tri Dung Duong, Qian Li, and Guandong Xu. Ceflow: A robust and efficient counterfactual explanation framework for tabular data using normalizing flows. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 133–144. Springer, 2023.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*, 2018.

Søren Hauberg. Only bayes should learn a manifold. 2018.

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3, 2017.

Chin-Wei Huang, Milad Aghajohari, Joey Bose, Prakash Panangaden, and Aaron C Courville. Riemannian diffusion models. *Advances in Neural Information Processing Systems*, 35:2750–2761, 2022.

Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Guillaume Jeanneret, Loïc Simon, and Frédéric Jurie. Diffusion models for counterfactual explanations. In *Proceedings of the Asian Conference on Computer Vision*, pages 858–876, 2022.

Shalmali Joshi, Oluwasanmi Koyejo, Warut Vijitbenjaronk, Been Kim, and Joydeep Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems. *arXiv preprint arXiv:1907.09615*, 2019.

Kaggle-Competition. "give me some credit. improve on the state of the art in credit scoring by predicting the probability that somebody will experience financial distress in the next two years", 2011.

Dimitrios Kalatzis, David Eklund, Georgios Arvanitidis, and Soren Hauberg. Variational autoencoders with riemannian brownian motion priors. In *International Conference on Machine Learning*, pages 5053–5066. PMLR, 2020.

Amir-Hossein Karimi, Gilles Barthe, Borja Balle, and Isabel Valera. Model-agnostic counterfactual explanations for consequential decisions. In *International conference on artificial intelligence and statistics*, pages 895–905. PMLR, 2020a.

Amir-Hossein Karimi, Julius Von Kügelgen, Bernhard Schölkopf, and Isabel Valera. Algorithmic recourse under imperfect causal knowledge: a probabilistic approach. *Advances in neural information processing systems*, 33:265–277, 2020b.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

J.M. Lee. *Introduction to Riemannian Manifolds*. Graduate Texts in Mathematics. Springer International Publishing, 2019.

Nishtha Madaan and Srikanta Bedathur. Diffusion-guided counterfactual generation for model explainability. In *XAI in Action: Past, Present, and Future Applications*, 2023.

Divyat Mahajan, Chenhao Tan, and Amit Sharma. Preserving causal constraints in counterfactual explanations for machine learning classifiers. *arXiv preprint arXiv:1912.03277*, 2019.

Thomas Melistas, Nikos Spyrou, Nefeli Gkouti, Pedro Sanchez, Athanasios Vlontzos, Giorgos Papanastasiou, and Sotirios A Tsaftaris. Benchmarking counterfactual image generation. *arXiv preprint arXiv:2403.20287*, 2024.

Marco Miani, Frederik Warburg, Pablo Moreno-Muñoz, Nicki Skafte, and Søren Hauberg. Laplacian autoencoders for learning stochastic representations. *Advances in Neural Information Processing Systems*, 35:21059–21072, 2022.

Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 607–617, 2020.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research (JMLR)*, 2021.

Yong-Hyun Park, Mingi Kwon, Jaewoong Choi, Junghyo Jo, and Youngjung Uh. Understanding the latent space of diffusion models through the lens of riemannian geometry. *Advances in Neural Information Processing Systems*, 36:24129–24142, 2023.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pages 3126–3132, 2020.

Martin Pawelczyk, Sascha Bielawski, Johan Van den Heuvel, Tobias Richter, and Gjergji Kasneci. Carla: A python library to benchmark algorithmic recourse and counterfactual explanation algorithms. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Paraskevas Pegios, Manxi Lin, Nina Weng, Morten Bo Søndergaard Svendsen, Zahra Bashir, Siavash Bigdeli, Anders Nymark Christensen, Martin Tolsgaard, and Aasa Feragen. Diffusion-based iterative counterfactual explanations for fetal ultrasound image quality assessment. *arXiv preprint arXiv:2403.08700*, 2024.

Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. Face: feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 344–350, 2020.

Qichao Que and Mikhail Belkin. Back to the future: Radial basis function networks revisited. In *Artificial intelligence and statistics*, pages 1375–1383. PMLR, 2016.

Chris Russell. Efficient search for diverse coherent explanations. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 20–28, 2019.

Sumedha Singla, Brian Pollack, Junxiang Chen, and Kayhan Batmanghelich. Explanation by progressive exaggeration. *arXiv preprint arXiv:1911.00483*, 2019.

Stas Syrota, Pablo Moreno-Muñoz, and Søren Hauberg. Decoder ensembling for learned latent geometries. In *ICML 2024 Workshop on Geometry-grounded Representation Learning and Generative Modeling*, 2024.

Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 10–19, 2019.

Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan E Hines, John P Dickerson, and Chirag Shah. Counterfactual explanations and algorithmic recourses for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2020.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

Nina Weng, Paraskevas Pegios, Aasa Feragen, Eike Petersen, and Siavash Bigdeli. Fast diffusion-based counterfactuals for shortcut removal and generation. *arXiv preprint arXiv:2312.14223*, 2023.

Appendix A. Implementation Details

A.1. Datasets

Adult: The dataset includes $\sim$ 49K census records with $D = 13$ features. The goal is to predict whether an individual's income exceeds \$50K/year ($y = 1$, 24% of samples). Following Pawelczyk et al. (2021), we binarize categorical features as aggregated versions of original categories indicating whether the individual works in *private industry*, is *not married*, has an *"other" occupation*, is *not a husband*, identifies as *white* (race), *male* (sex), and is *USA native*. The continuous features are *age, final weight, years of education, weekly work hours,* as well as *capital gain and loss* which we log-transform. The input features sex, race, and *age* are considered immutable.

Give Me Some Credit: We use the processed data by Pawelczyk et al. (2021), which includes $\sim$ 116K samples after removing missing data with $D = 10$ continuous features. Furthermore, we log-transform *depht ratio* input feature. The goal is to predict whether an individual will face financial distress within two years ($y = 1$, 93% of samples). The input feature *age* is considered immutable.

We normalize features in $[0, 1]$ and split datasets into 75%/25% for train/test.

A.2. Models

For both datasets, the binary classifier c under observation is a 4-layer MLP with $2 \cdot H$, $2 \cdot H$, H, H neurons including Batch Normalization (Ioffe, 2015) layers, and Tanh activation functions before the last layer, where $H = 24$ is the size of representation space. We trained the classifiers with a batch size of 1024, a learning rate of 10^{-5}, and applied ℓ_2-regularization of 0.05 for 20 epochs using the RMSprop optimizer, resulting in 77.5% (77.6%) and 73.6% (73.8%) *balanced* test (train) accuracy for Adult and GMC datasets, respectively.

Our VAEs employ Gaussian generative models with a 3-layer MLP mirrored encoder-decoder, with D, 512, 256 neurons with Batch Normalization layers and Tanh activations functions, a latent space of $d = 5$, and Sigmoid activation function in decoder's mean output. Following Detlefsen et al. (2019) and Kalatzis et al. (2020), we deterministically warmed up the encoders together with the decoders' mean μ_θ for 100 epochs, with β-regularization (Higgins et al., 2017) of 10^{-4}. Consecutively, we freeze them while optimizing for 300 epochs decoders' uncertainty σ_ϕ, parameterized by Radial Basis Function (Que and Belkin, 2016) networks with a 0.01 bandwidth and 200 and 350 centers for Adult and GMC, respectively. During training, we used a batch size of 512, and the Adam (Kingma, 2014) optimizer, with a learning rate of 10^{-3}, expect for GMC's σ_ϕ where we used 10^{-2}. During inference and CE optimization, we used Stochman (Detlefsen et al., 2021) to track Jacobians through the VAE's decoder and the classifier.

Appendix B. Results for GMC dataset

#Iter	Constraints	Optimizer	$\mathcal{L}_D \downarrow$	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_2 \downarrow$	$\mathcal{L}_\infty \downarrow$	$c(\mathbf{x}_{\mathrm{CE}}) \uparrow$	FR $\uparrow$	violation $\downarrow$
50	✗	SGD	0.024±0.019	1.56±0.45	0.59±0.32	0.56±0.22	**0.90±0.07**	**1.00**	0.97
		RSGD	0.019±0.014	1.21±0.42	0.33±0.23	0.39±0.17	0.78±0.25	0.89	**0.96**
		RSGD-C	**0.015±0.012**	**1.11±0.40**	**0.29±0.23**	**0.37±0.17**	0.69±0.27	0.81	**0.96**
50	✓	SGD	0.023±0.018	1.31±0.44	0.40±0.26	0.44±0.19	**0.86±0.16**	**0.97**	0.97
		RSGD	0.019±0.015	1.10±0.41	0.27±0.20	0.35±0.15	0.77±0.26	0.89	**0.96**
		RSGD-C	**0.015±0.012**	**0.83±0.32**	**0.16±0.13**	**0.26±0.12**	0.61±0.34	0.71	**0.96**
100	✗	SGD	0.026±0.020	1.79±0.57	0.75±0.42	0.62±0.23	**0.92±0.05**	**1.00**	0.98
		RSGD	0.021±0.014	1.52±0.52	0.55±0.36	0.53±0.23	0.82±0.23	0.92	**0.96**
		RSGD-C	**0.020±0.014**	**1.44±0.52**	**0.49±0.35**	**0.48±0.22**	0.79±0.23	0.90	**0.96**
100	✓	SGD	0.025±0.018	1.43±0.49	0.47±0.30	0.48±0.20	**0.88±0.15**	**0.97**	0.98
		RSGD	0.022±0.015	1.30±0.48	0.39±0.27	0.43±0.19	0.81±0.24	0.90	**0.96**
		RSGD-C	**0.020±0.016**	**1.08±0.39**	**0.27±0.21**	**0.36±0.17**	0.73±0.29	0.83	**0.96**
150	✗	SGD	0.026±0.020	1.83±0.58	0.77±0.43	0.63±0.23	**0.92±0.04**	**1.00**	0.99
		RSGD	0.024±0.015	1.62±0.54	0.62±0.40	0.55±0.23	0.83±0.21	0.93	**0.98**
		RSGD-C	**0.022±0.014**	**1.56±0.55**	**0.57±0.39**	**0.52±0.23**	0.80±0.23	0.91	**0.98**
150	✓	SGD	0.026±0.019	1.44±0.50	0.48±0.30	0.49±0.20	**0.88±0.15**	**0.97**	0.98
		RSGD	0.025±0.015	1.36±0.49	0.43±0.30	0.46±0.19	0.82±0.23	0.91	**0.96**
		RSGD-C	**0.021±0.015**	**1.18±0.42**	**0.33±0.25**	**0.39±0.18**	0.76±0.26	0.87	**0.96**

Table 2: GMC dataset. To standardize comparison, we evaluate optimizers with (✓) and without (✗) fidelity constraints *during* optimization for different numbers of steps.

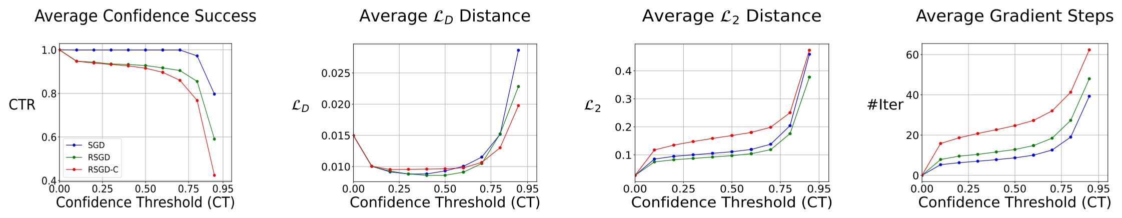


Figure 7: GMC Dataset. Evaluation of CEs as a function of confidence threshold (CT). Confidence Threshold Ratio (CRT) quantifies confidence success, i.e., the proportion of CEs achieving a specified CT. $\mathcal{L}_D$ measures fidelity to the data manifold (realism), $\mathcal{L}_2$ the fidelity to the initial data point (closeness), and #Iter the number of required gradient steps to achieve a specific confidence level.