Discrete-Continuous Variational Optimization with Local Gradients

Jonathan Warrell[1,2*] JWARRELL@NEC-LABS.COM
Francesco Alesiani[3*] FRANCESCO.ALESIANI@NECLAB.EU
Cameron Smith[4] CAMERON.SMITH@MGH.HARVARD.EDU
Anja Mösch[3] ANJA.MOESCH@NECLAB.EU
Martin Renqiang Min[1] RENQIANG@NEC-LABS.COM
[1] *NEC Laboratories America, Princeton, NJ, USA*
[2] *Yale University, New Haven, CT, USA*
[3] *NEC Laboratories Europe, Heidelberg, Germany*
[4] *Massachusetts General Hospital, Harvard Medical School, Boston, MA, USA*
**Equal Contribution*

Abstract

Variational optimization (VO) offers a general approach for handling objectives which may involve discontinuities, or whose gradients are difficult to calculate. By introducing a variational distribution over the parameter space, such objectives are smoothed, and rendered amenable to VO methods. Local gradient information, though, may be available in certain problems, which is neglected by such an approach. We therefore consider a general method for incorporating local information via an augmented VO objective function to accelerate convergence and improve accuracy. We show how our augmented objective can be viewed as an instance of multilevel optimization. Finally, we show our method can train a genetic algorithm simulator, using a recursive Wasserstein distance objective.
Keywords: Variation methods, Black-box optimization, Multilevel optimization

1. Introduction

There has been much attention recently in developing efficient methods to optimize objectives which may contain function or gradient discontinuities, or whose gradients are difficult to calculate. Such situations arise, for instance, in black box optimization methods [1], objectives over mixed discrete and continuous variables [2], optimization of simulator parameters, and hyper-parameter optimization [3]. An interesting general class of methods for such problems are Variational Optimization (VO) approaches, which involve placing an auxiliary variational distribution over the parameter space, which may be optimized in place of directly optimizing the parameters [4]–[6]. Such an approach may be classed as a type of smoothing-based optimization [5], and has been related to scale-space methods in computer vision [7] and Gaussian continuation methods [3], [8].

Adopting a VO approach can be used to render almost any objective function differentiable. Methods such as score-function gradients with control variates [1], or stochastic gradient descent (SGD) [6] may then be applied to optimize the resulting VO objective. However, the need to stabilize score function gradients means that this approach may not be efficient for all problems if appropriate control variates cannot be provided or learned, and SGD approaches such as [6] will only be applicable when an approximate gradient or sub-gradient is straightforwardly calculated.

© J. Warrell, F. Alesiani, C. Smith, A. Mösch & M.R. Min.

The smoothing-based VO approach of [5] offers advantages, in light of the above issues, since it requires only that the objective function be efficiently evaluable. Further, the gradient-free updates are guaranteed to improve the scale-space objective in expectation. However, due to the lack of gradient information, such updates may be inefficient. We note that, while many objectives include function and gradient discontinuities globally, there will often be regions or subspaces of parameter space over which gradients may be calculated. We therefore propose an augmented form of the VO approach of [5], which may make use of such local gradient information. In doing so, we are influenced by recent approaches to mixed discrete-continuous multilevel optimization [9], and we show that our method may also be formulated as solving a multilevel optimization problem.

We briefly summarize the contributions and significance of our work below. (a) Sec. 2 outlines our approach to smoothing-based VO augmented with local gradients, and details its relationship with multilevel optimization (Appendix A). (b) In Sec. 3 we then demonstrate our approach on the problem of learning the parameters of an arbitrary genetic algorithm (GA), using a multilevel Wasserstein objective, where output populations of the GA are available as training data. (c) Finally, we note that this problem, while complex, has extensive biological applications, where such approaches may be used for instance to infer phylogenetic structure in large-scale genetics analyses. A number of Variational Inference (VI) methods have recently been proposed in related contexts [10]–[12], and our method offers an alternative VO-based approach which may offer advantages in terms of its scalability. We conclude with a brief discussion of such directions in Sec. 4.

2. Variational Optimization Augmented with Local Gradients

2.1. Smoothing-based Variational Optimization

We briefly summarize the smoothing-based VO framework of [5]. We assume that we desire to optimize maximize a non-negative function, $F(\boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \mathbb{R}^d$, and F may be discontinuous or contain discontinuities in its gradients. Hence, F can model an output value which depends on a discrete deterministic latent variable; for example F may depend on $\boldsymbol{z} = g(\boldsymbol{\theta})$, with $\boldsymbol{z}$ a categorical variable. Additional examples are included in Appendix B. To optimize F, we introduce a variational distribution $Q_{\boldsymbol{\mu},\sigma}(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \sigma^2\mathbf{I})$, with $\mathbf{I}$ the identity matrix, where $\mathcal{N}(.|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a multivariate Gaussian distribution, and an associated smoothed variational objective:

$$\mathcal{L}^{smooth}(\boldsymbol{\mu}, \sigma) \quad = \quad \mathbb{E}_{\boldsymbol{\theta}\sim Q_{\boldsymbol{\mu},\sigma}}[F(\boldsymbol{\theta})]. \tag{1}$$

By optimizing $\mathcal{L}^{smooth}$ we are optimizing a lower-bound on $\mathcal{L}$, since:

$$\mathbb{E}_{\boldsymbol{\theta}\sim Q_{\boldsymbol{\mu},\sigma}}[F(\boldsymbol{\theta})] \leq \max_{\boldsymbol{\mu},\sigma} \mathbb{E}_{\boldsymbol{\theta}\sim Q_{\boldsymbol{\mu},\sigma}}[F(\boldsymbol{\theta})] \leq \max_{\boldsymbol{\theta}} F(\boldsymbol{\theta}). \tag{2}$$

In [5], it is shown that the following updates are guaranteed to increase $\mathcal{L}^{smooth}$ in expectation:

$$\boldsymbol{\mu}_{t+1} \quad = \quad \frac{\sum_s f_s \boldsymbol{\theta}_s^t}{\sum_s f_s}, \quad \sigma_{t+1} = \sqrt{\frac{\sum_s f_s |\boldsymbol{\theta}_s^t - \boldsymbol{\mu}_t|_2^2}{d \sum_s f_s}},$$

where $\boldsymbol{\theta}_s^t$ for $s \in \{1...S\}$ are samples drawn from $Q_t = \mathcal{N}(.|\boldsymbol{\mu}_t, \sigma_t^2\mathbf{I})$, and $f_s = F(\boldsymbol{\theta}_s^t)$. As noted in [5], introducing the variational distribution Q allows us to navigate complex optimization landscapes, potentially with many local minima, by smoothing the underlying objective. Further, any nonnegative function with compact support has a single global maximum for a sufficiently large value of σ, although this may not hold for nonnegative functions with infinite support [13].

2.2. Augmented Objective with Local Gradients

We now consider that, in addition to the function F, we have a local differentiable region selector $R(\boldsymbol{\theta})$, defined as:

Definition 1 (Local Differentiable Region Selector): A function $R(\boldsymbol{\theta})$ is a *local differentiable region selector* for F, with $\boldsymbol{\theta}$ and F defined as above, iff for all $\boldsymbol{\theta} \in \mathbb{R}^d$, we have that $R(\boldsymbol{\theta}) \subseteq \mathbb{R}^d$, and either $R(\boldsymbol{\theta}) = \emptyset$, or there exists a linear subspace $R'(\boldsymbol{\theta}) \subseteq \mathbb{R}^d$ such that $R(\boldsymbol{\theta})$ is an open neighborhood of $\boldsymbol{\theta}$ in $R'(\boldsymbol{\theta})$ (hence $\boldsymbol{\theta} \in R(\boldsymbol{\theta}) \subseteq \mathbb{R}'(\boldsymbol{\theta})$), and the restriction of F to R' is differentiable over R. Further, we require that $\boldsymbol{\theta}' \in R(\boldsymbol{\theta}) \implies R(\boldsymbol{\theta}') = R(\boldsymbol{\theta})$. $\square$

Given such a function $R(\boldsymbol{\theta})$, we now consider a *local optimizer*, $\mathcal{O}$ that performs gradient ascent starting at $\boldsymbol{\theta}$ for U steps, with the update parameter $\boldsymbol{\theta}_{u+1,\alpha}$ constrained to lie within $R(\boldsymbol{\theta}_{u,\alpha})$. More formally, $\mathcal{O}(\boldsymbol{\theta}) = O_{U,\alpha}(\boldsymbol{\theta})$, with :

$$\begin{aligned} O_{0,\alpha}(\boldsymbol{\theta}) &= \boldsymbol{\theta} \\ O_{u+1,\alpha}(\boldsymbol{\theta}) &= \begin{cases} \boldsymbol{\theta}' = O_{u,\alpha}(\boldsymbol{\theta}) + \alpha g(O_{u,\alpha}(\boldsymbol{\theta})), & \text{if } F(\boldsymbol{\theta}') > F(O_{u,\alpha}(\boldsymbol{\theta})) \wedge \boldsymbol{\theta}' \in R(\boldsymbol{\theta}) \\ O_{u,\alpha}(\boldsymbol{\theta}), & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

where U is the maximum number of steps taken by the optimizer, α is the step-size, and $g(\boldsymbol{\theta})$ returns the gradient at $\boldsymbol{\theta}$ in $R'(\boldsymbol{\theta})$ if $R(\boldsymbol{\theta})$ is non-empty, and 0_d otherwise.

Since the updates in Eq. 3 are non-decreasing in F, we have $F(\mathcal{O}(\boldsymbol{\theta})) \geq F(\boldsymbol{\theta})$. We can thus define the auxiliary objective, and the associated augmented variational lower-bound:

$$F'(\boldsymbol{\theta}) = F(\mathcal{O}(\boldsymbol{\theta})), \;\; \mathcal{L}^{aug}(\boldsymbol{\mu}, \sigma) \;\; = \;\; \mathbb{E}_Q[F'(\boldsymbol{\theta})], \tag{4}$$

which provides a tighter bound on the original loss, since:

$$\mathbb{E}_Q[F(\boldsymbol{\theta})] \leq \mathbb{E}_Q[F'(\boldsymbol{\theta})] \leq \max_{\boldsymbol{\mu},\sigma} \mathbb{E}_Q[F'(\boldsymbol{\theta})] \leq \max_{\boldsymbol{\theta}} F(\boldsymbol{\theta}), \tag{5}$$

where $\mathbb{E}_Q[.]$ denotes $\mathbb{E}_{\boldsymbol{\theta} \sim Q_{\boldsymbol{\mu},\sigma}}[.]$. We can optimize $\mathcal{L}^{aug}$ similarly to $\mathcal{L}^{smooth}$, by replacing f_s in Eq. 3 with $f'_s = F'(\boldsymbol{\theta}^t_s)$.

3. Experimental Results

3.1. Learning a Genetic Algorithm using a Multilevel Wasserstein Objective

We design a synthetic problem to test our optimization approach, based on learning the parameters of a fitness function for a discrete Genetic Algorithm (GA), whose genotypes represent solutions to a Traveling Salesman Problem (TSP), with a prior over the parameter space. We use a multilevel Wasserstein distance as our objective, as outlined below.

The underlying TSP consists of finding the shortest path connecting M cities, whose coordinates sampled from a standard 2d Gaussian distribution. Since the coordinates of the cities implicitly define the fitness of a chosen path, we denote the m'th city's coordinates as $\boldsymbol{\theta}_m = (\boldsymbol{\theta}_{m,1}, \boldsymbol{\theta}_{m,2})$, and our goal in training the GA is to learn $\boldsymbol{\theta}$, hence the coordinates of the cities. Further, we denote the distance between cities m_1 and m_2 as d_{m_1,m_2}. The underlying space of genotypes $\mathcal{X}$ is thus the set of permutations on M elements, and the fitness of genotype $x \in \mathcal{X}$ is defined as:

$$f(x) = \exp(-\beta \sum_{i=1...M-1} d_{x_i,x_{i+1}}) \tag{6}$$

where β is an inverse temperature parameter. A simulation of the GA consists of sampling a population of genotypes of size N uniformly from $\mathcal{X}$, and updating these for T generations using an update rule derived from the Wright-Fisher process with constant population size and selection (see [14]). Hence, at time-step $t > 0$, to generate a new population of size N, N parents are sampled from the population at $t-1$, with each parent being sampled with a probability proportional to its fitness (with replacement). The genotypes at generation t are then generated either by coping the parent's genotype, or with probability p_m, mutating the parents genotype by exchanging two cities (note that reproduction here is asexual, hence each offspring has a single parent).

We run L simulations as above up to time-step T, and a training set is generated by including only the final population state from each simulation. The intention here is to mimic the situation where only genotypes from existing species are available in evolutionary data, or genomics data from a cancer are only available from the time of resection, and the evolutionary history must be inferred. Our ground-truth training set may thus be represented as $X \in \{1...M\}^{LNM}$, where X_{lnm} represents the city $(1...M)$ visited by the n'th population member in simulation l at the m'th location on the route. Similarly, for a given parameter setting $\boldsymbol{\theta}$, we may generate a new set of simulations, $\mathcal{S}(\boldsymbol{\theta}) = \tilde{X}_{\boldsymbol{\theta}} \in \{1...M\}^{LNM}$, of the same dimensions. For convenience, we assume the simulator is deterministic, which may be achieved by fixing the random seed. Further, we provide the parameters β, T, p_m, although in general these may also be inferred. We then perform inference using an objective of the form in Example 2 in Appendix B, where $F_2(\theta, \theta_0) = \mathcal{N}(\theta, \theta_0, I)$, with $\mathcal{N}$ denoting the normal distribution and $\theta_0 = 0$, and F_1 is defined as a multilevel Wasserstein distance as follows:

$$\begin{aligned} F_1(\tilde{X}, X) &= \exp\left(-W'(\tilde{X}, X)\right) \\ W'(\tilde{X}, X) &= W_2\left(\frac{\sum_l \delta(\tilde{X}_l)}{L}, \frac{\sum_l \delta(X_l)}{L}, C\right) \\ C(\rho_a, \rho_b) &= W_2(\rho^a, \rho^b, H) \end{aligned} \tag{7}$$

where $W_2(P_a, P_b, C)$ is the second-order Wasserstein distance between distributions P_a and P_b using cost function C, ρ_a and ρ_b are distributions over genotype space $\mathcal{X}$, $\delta(a)$ is a delta distribution centered at a, and H is the Hamming distance between genotypes, $H(x_a, x_b) = \sum_m [x_{am} \neq x_{bm}]$. We apply the VO approach of Sec. 2.2 to learn the parameter vector θ, and compare this with the stochastic gradient-based SPSA method in Appendix C, where $F(\theta)$ is defined identically. For the simulations, we use $M = 5$, $N = 4$, $L = 4$, $T = 10$, $S = 10$, $\beta = 1$ and $p_m = 0.1$. Further, we use 100 meta-epochs for both VO and SPSA algorithms, which we observed to be sufficient for both algorithms to converge.

3.2. Learning an adaptive Graph Neural Network for Protein Liquid-Liquid Phase Separation

To illustrate the generality of our approach, we also apply the VO approach of Sec. 2.2 to learning an adaptive Graph Neural Network to predict Protein Liquid-Liquid Phase Separation (LLPS) on scaffold proteins, following the problem description in Example 1 of Appendix B, and using the same experimental setup as in [2]. To compare our VO approach, we use a stochastic gradient descent (SGD) optimizer to optimize $F(\theta)$ directly (see [2]), and we use 60 meta-epochs of both VO and SGD optimizers, which was sufficient to ensure convergence.

Table 1: Comparing performance of Variational Optimization and Stochastic Gradient-based approaches on Genetic Algorithm (GA) and adaptive GNN (aGNN) optimization. Mean and standard deviation of 10 models are shown. Methods are compared using 2-level Wasserstein distance defined in Eq. 7 and the Euclidean distance of the estimated solution to the TSP to the ground-truth (up to rotation and translation) for the GA problem, and the test set AUC for the aGNN problem.

Problem	Metric	Variational Optimization	SPSA(GA) / SGD(aGNN)
GA	2-level Wasserstein	$\mathbf{0.136 \pm 0.0607}$	0.213 ± 0.0211
GA	Euclidean	$\mathbf{0.735 \pm 0.054}$	0.942 ± 0.0667
aGNN	AUC	$\mathbf{0.726 \pm 0.022}$	0.671 ± 0.03

3.3. Performance Comparison

Table 1 compares the performance of our VO optimizer against stochastic gradient-based approaches for the GA and aGNN optimization problems, comparing with SPSA for the GA problem (Appendix C) and SGD for the aGNN problem. We note that applying SPSA was not found to be advantageous in the aGNN problem, possibly due to the higher dimensionality of the parameter space. As observed, the VO approach achieves better solutions in both problems. For the GA problem, the solutions found achieve lower 2-level Wasserstein distances as measured by Eq. 7, and a lower Euclidean distance between the cities of the estimated TSP to the ground-truth (up to rotation and translation), which determine the underlying fitness function of the genetic algorithm. For the adaptive GNN, VO optimization achieves higher AUC performance on hold-out data for prediction of scaffold proteins.

4. Discussion

We have introduced a method for augmenting smoothing-based Variational Optimization with local gradients, appropriate for a wide range of objectives with discontinuities in function or gradient values, mixed discrete-continuous optimization problems, or problems in which full gradients may be difficult to evaluate. We have shown that our method may be viewed as a type of multi-level optimization approach, and have demonstrated the efficiency of the approach in training a simulator based on a genetic algorithm and an adaptive Graph Neural Network architecture, each using objective functions with function and gradient discontinuities.

In future work, we plan to investigate the potential of our approach in a number of domains. As discussed, we are particularly interested in using a similar approach to Secs. 2 and 3.1 on data from biological evolutionary processes, including phylogenetics and cancer genomics. In the case of the latter, this approach may for instance be used to infer tumor clonal structure (see [15]). Further, we intend to investigate applications to learning discrete structures, for instance, graph inference, in the context of structure learning for energy-based models. Finally, we plan to investigate the potential of our approach in the context of mixed continuous-discrete optimization problems. We note that our division of the parameter space into local differentiable regions has similarities with branch-and-bound-based approaches, and an interesting avenue for further investigation would be to combine VO with branch-and-bound methods. We also plan to investigate integrating implicit gradient methods within our framework to handle multilevel optimization problems.

References

[1] W. Grathwohl, D. Choi, Y. Wu, G. Roeder, and D. Duvenaud, "Backpropagation through the void: Optimizing control variates for black-box gradient estimation," in *International Conference on Learning Representations*, 2018.

[2] G. Wang, J. Warrell, S. Zheng, *et al.*, "A variational graph partitioning approach to modeling protein liquid-liquid phase separation," *bioRxiv preprint doi: https://doi.org/10.1101/2024.01.20.576375*, 2024.

[3] J. Rojas-Delgado, J. Jiménez, R. Bello, and J. Lozano, "Hyper-parameter optimization using continuation algorithms," in *Metaheuristics: 14th International Conference, MIC 2022, Syracuse, Italy, July 11–14, 2022, Proceedings*, Springer, 2023, pp. 365–377.

[4] J. Staines and D. Barber, "Variational optimization," *arXiv preprint arXiv:1212.4507*, 2012.

[5] M. Leordeanu and M. Hebert, "Smoothing-based optimization," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2008, pp. 1–8.

[6] M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava, "Fast and scalable bayesian deep learning by weight-perturbation in adam," in *International conference on machine learning*, PMLR, 2018, pp. 2611–2620.

[7] A. Kuijper, L. M. Florack, and M. A. Viergever, "Scale space hierarchy," *Journal of Mathematical Imaging and Vision*, vol. 18, pp. 169–189, 2003.

[8] H. Mobahi and J. Fisher III, "A theoretical analysis of optimization by gaussian continuation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, 2015.

[9] F. Alesiani, "Implicit bilevel optimization: Differentiating through bilevel optimization programming," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 14 683–14 691.

[10] A. K. Moretti, L. Zhang, C. A. Naesseth, H. Venner, D. Blei, and I. Pe'er, "Variational combinatorial sequential monte carlo methods for bayesian phylogenetic inference," in *Uncertainty in Artificial Intelligence*, PMLR, 2021, pp. 971–981.

[11] C. Zhang and F. A. Matsen IV, "Variational bayesian phylogenetic inference," in *International Conference on Learning Representations*, 2018.

[12] C. Zhang, "Improved variational bayesian phylogenetic inference with normalizing flows," *Advances in neural information processing systems*, vol. 33, pp. 18 760–18 771, 2020.

[13] M. Loog, J. JisseDuistermaat, and L. M. Florack, "On the behavior of spatial critical points under gaussian blurring a folklore theorem and scale-space constraints," in *Scale-Space and Morphology in Computer Vision: Third International Conference, Scale-Space 2001 Vancouver, Canada, July 7–8, 2001 Proceedings 3*, Springer, 2001, pp. 183–192.

[14] J. Felsenstein, "Inferring phylogenies," in *Inferring phylogenies*, 2004, pp. 664–664.

[15] J. Abécassis, F. Reyal, and J.-P. Vert, "Clonesig can jointly infer intra-tumor heterogeneity and mutational signature activity in bulk tumor sequencing data," *Nature communications*, vol. 12, no. 1, p. 5352, 2021.

[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, PMLR, 2017, pp. 1126–1135.

[17] J. C. Spall, "Stochastic optimization, stochastic approximation and simulated annealing," *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2001.

Appendix A. Multilevel Optimization Formulation

The augmented VO approach in Sec. 2.2 may be regarded as an instance of a mixed discrete-continuous bi-level optimization problem (see [9]). Here, for convenience, we assume that, for all non-empty regions $R(\boldsymbol{\theta})$, $F(\boldsymbol{\theta})$ is convex within $R(\boldsymbol{\theta})$, $R(\boldsymbol{\theta})$ is a convex set, and that $\sup_{\boldsymbol{\theta}' \in R(\boldsymbol{\theta})} F(\boldsymbol{\theta}')$ is achieved for some value $\boldsymbol{\theta}' \in R(\boldsymbol{\theta})$. For such problems, we can form the equivalent bi-level optimization problem:

$$\max_{\boldsymbol{\theta} \in \mathbb{R}^d} F_{bilev}(\boldsymbol{\theta}, \boldsymbol{\theta}'), \quad \boldsymbol{\theta}' \in \arg\max_{\boldsymbol{\theta}' \in \mathbb{R}^d} G_{bilev}(\boldsymbol{\theta}, \boldsymbol{\theta}') \tag{8}$$

where:

$$F_{bilev}(\boldsymbol{\theta}, \boldsymbol{\theta}') \quad = \quad F(\boldsymbol{\theta}'), \quad G_{bilev}(\boldsymbol{\theta}, \boldsymbol{\theta}') = F(\boldsymbol{\theta}') - \infty[\boldsymbol{\theta}' \notin R(\boldsymbol{\theta}) \cup \{\boldsymbol{\theta}\}]$$

where $[.]$ denotes the Iverson bracket, which is 1 for a true proposition and 0 otherwise. The augmented VO approach of Sec. 2.2 can thus be viewed as a solution to the bi-level problem in Eq. 8, where the outer optimization problem F_{bilev} is solved via VO, and the inner problem G_{bilev} is solved via the gradient-based optimizer $\mathcal{O}$ (where a sufficiently small step-size α and large U will ensure the optimum is found for each convex inner problem). A similar approach has been proposed for meta-learning [16], where the inner problem is solved approximately by performing a few SGD steps.

Appendix B. Examples of Continuous-Discrete problems

Example 1 (Adaptive Graph Neural Network): We consider a problem, as discussed in [2], where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$, and F has the form $F(\boldsymbol{\theta}) = -\sum_i L(Y_i|\boldsymbol{\theta}_1, G_i(\boldsymbol{\theta}_2))$. Here, $L(Y_i|\boldsymbol{\theta}_1, G_i)$ is the loss for predicting the label Y_i for datapoint i using a graph neural network (GNN) with parameters $\boldsymbol{\theta}_1$, where the underlying graph has topology G_i. If G_i is fixed, F is differentiable for many GNN parameterizations. Letting G_i depend on $\boldsymbol{\theta}_2$, though, allows the graph to adaptively be learned for each datapoint, causing F to become discontinuous (at values for which the graph changes for some datapoint). Hence, we may choose $R(\boldsymbol{\theta}) = R'(\boldsymbol{\theta}) = \{\boldsymbol{\theta}'|\boldsymbol{\theta}_2 = \boldsymbol{\theta}'_2\}$ to apply the above approach. We therefore perform local gradient descent within the subspace defined by $\boldsymbol{\theta}_1$ while fixing the graph structure for each datapoint by ensuring $\mathcal{O}$ leaves $\boldsymbol{\theta}_2$ constant.

Example 2 (Discrete Simulator with Continuous Prior): We consider that we have a simulator $\mathcal{S}(\theta)$, which generates discrete outputs that we wish to match to ground-truth observations $\mathcal{X}$. Further, we wish to choose θ to be similar to a prior θ_0. We may thus form the objective $F = F_1(\mathcal{S}(\theta), \mathcal{X}) + F_2(\theta, \theta_0)$, where F_1 and F_2 are appropriately defined non-negative similarity functions. In this case, we let $R(\theta)$ be the largest connected open neighborhood of θ over which $F'_1(\theta') = F_1(\mathcal{S}(\theta'), \mathcal{X})$ is constant (that is, $\theta' \in R(\theta) \implies F'_1(\theta') = F'_1(\theta)$). Further, $R'(\theta) = \mathbb{R}^d$. By definition then, we have that $F(\theta)$ is continuous within each region $R(\theta)$ when the latter is non-empty.

Appendix C. Simultaneous Perturbation Stochastic Perturbation (SPSA) optimizer

As an alternative approach, Simultaneous Perturbation Stochastic Perturbation (SPSA) [17] provides a method for performing stochastic gradient descent on a function which is easy to evaluate, but for which an analytic gradient is not available. The method requires hyperparameters a, A, α, c, γ, and for epoch τ we define $a_\tau = (a/(\tau + A))^\alpha$ and $c_\tau = (c/\tau)^\gamma$. Then, θ is initialized randomly at epoch

0, and for each epoch τ we sample a vector $\delta \in \{-1, 1\}^{D_\theta}$, representing randomized perturbations to the current value of θ (where $\delta(i)$ is sampled independently and uniformly from $\{-1, 1\}$). Assuming we wish to minimize the function $F(\theta)$ as above, we then use the following gradient estimator and update at each epoch:

$$\hat{g}_\tau = \frac{F(\theta_\tau + c_\tau \delta) - F(\theta_\tau - c_\tau \delta)}{2c_\tau}$$

and we update the parameters using: $\theta_{\tau+1} = \theta_\tau - a_\tau \hat{g}_\tau$.