Approximate Bayesian Geophysical Inversion using Generative Modeling and Subset Simulation

Eliane Maalouf
University of Neuchâtel
Neuchâtel, CH 2000
eliane.maalouf@unine.ch

David Ginsbourger
University of Bern
Bern, CH 3012
david.ginsbourger@stat.unibe.ch

Niklas Linde
University of Lausanne
Lausanne, CH 1015
niklas.Linde@unil.ch

Abstract

We present preliminary work on solving geophysical inverse problems by exploring the latent space of a joint Generative Neural Network (GNN) model by Approximate Bayesian Computation (ABC) based on Subset Simulation (SuS). Given pre-generated subsurface domains and their corresponding solver outputs, the GNN surrogates the forward solver during inversion to quickly explore the input space through SuS and locate regions of credible solutions. Akin to ABC methods, our methodology allows to tune the similarity threshold between observed and candidate outputs. We explore how tuning this threshold influences the uncertainty in the solutions, allowing to sample solutions with a selected diversity level. Our initial tests were carried out with data from straight-ray (linear) tomography with Gaussian priors on slowness fields and Gaussian versus Gumbel observation noise distributions. We are presently testing the methodology on non-linear physics to demonstrate its applicability in more general inversion settings.

1 Introduction

In geophysics, cross-hole Ground Penetrating Radar (GPR) sources are used to emit high-frequency electromagnetic waves at different locations in one borehole that are recorded by receivers at different locations in another borehole. The first-arrival travel times of waves that travel through the subsurface between the boreholes are then determined from the recorded traces. Cross-hole GPR tomography relies on such noise-contaminated measurements to retrieve the slowness field (i.e. inverse of the velocity field) of the geophysical domain (hereafter denoted by “domain”).

Bayesian inference treats the unknown field as random and provides access to its posterior distribution given the observed measurements and a priori assumptions. Markov Chain Monte Carlo (MCMC) methods are widely used to explore analytically intractable posterior distributions through sampling [12]. One crucial aspect to guarantee the convergence of MCMC to the true posterior distribution is to use a likelihood function reflecting correct assumptions on both the considered physical phenomenon and the observation noise distribution. Gathering realistic information on the noise can be a hard task, and for convenience, it often appears as a default choice to employ likelihood functions relying on Gaussian or other simple noise distribution assumptions. Likelihood free methods such as Approximate Bayesian Computation (ABC) algorithms avoid the specification of the likelihood function by means of simulations from an approximate posterior [5]. Given the simulation nature of ABC, it requires sampling from the domain space followed by forward simulations on those samples.

Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020), Vancouver, Canada.

A sample is accepted if the simulator output differs from the measurement of interest by a distance limited by a tolerance parameter. Hence, standard ABC becomes very expensive when the forward simulator is expensive to evaluate and the tolerance threshold is small. Furthermore, the efficiency of ABC algorithms depends on the proposal distribution as we want more mass around samples that yield simulator outputs close to the measurements. In the absence of a priori information about these locations in the domain space, adaptive sequential ABC methods [11] [2] allow to sequentially improve the proposal distribution to better locate and target these regions.

In this context, we apply a new simulation-based method, which avoids expensive frequent calls to the simulator. The methodology leverages Generative Neural Networks (GNN) to learn the joint distribution between the subsurface domains and their simulator outputs. We explore the solutions to an inverse problem given a noisy measurement vector by sampling in the latent space of the GNN. The ABC sampling scheme is based on Subset Simulation (SuS) [1]. We replace all calls to the simulator during inversion by evaluations from the GNN model. Moreover, we illustrate how tuning the tolerance parameter influences the diversity/uncertainty in the solutions.

Related works have used GNN for dimensionality reduction of the domain space through the latent space [3], [6], [10] followed by MCMC sampling on the latent space. Others aimed to learn a direct conditional map from the measurements space to the domain space [4], [7]. The first approach still requires frequent calls to the expensive forward solver. The second, if not generative, performs uncertainty estimation in an ad-hoc manner.

2 Methodology

We denote by X, Y, defined on spaces $\mathbb{X}$, $\mathbb{Y}$, the variables representing the subsurface domain and the response of interest, respectively. We assume first that X is linked to Y by $Y = F(X) + E$ with E a noise variable with a distribution that is not known a priori, and F expresses the physics of the problem encoded by the simulator. We further assume that both X and Y can be expressed as a function of some latent variable $Z \in \mathbb{Z}$, where $\mathbb{Z}$ is of moderate dimension compared to $\mathbb{X}$ and $\mathbb{Y}$, such that $(X, Y) = G^o(Z) = (g_1^o(Z), g_2^o(Z))$. Setting a prior distribution on $\mathbb{Z}$, G^o is a generative function of couples (X, Y). The considered inverse problem consists in retrieving a distribution of subsurface domains having potentially generated a given vector of measurements y_{obs}.

Phase I - Joint Generative Neural Network training Relying on a sample of couples (X, Y), we estimate the map G^o by training a joint GNN model. We adapted the Wasserstein Auto-Encoder (WAE) [14] and its Sinkhorn Auto-Encoder (SAE) variant [9] to the task of learning the joint distribution $P_{(X,Y)}$. The previously introduced X and Y variables are the inputs to the encoder and we denote by $\tilde{X}$ and $\tilde{Y}$ the outputs of the decoder/generator (cf. Figure 1). The WAE training aims to minimize the Wasserstein distance [15] between $P_{(X,Y)}$ and $P_{(\tilde{X},\tilde{Y})}$. With deterministic encoder and decoder networks, all the randomness in $(\tilde{X}, \tilde{Y})$ is contained in Z.

The training consists in minimizing an empirical version of

$$\min_G \min_Q \mathbb{E}_{XY \sim P_{(X,Y)}} [||X - g_1(Q(X,Y))||_p^p + ||Y - g_2(Q(X,Y))||_p^p]] + \lambda W_p(Q_Z, P_Z),$$

with respect to G and Q, where $||.||_p$ stands for the l_p norm on the considered spaces ($\mathbb{X}$ and $\mathbb{Y}$ typically assumed to be finite-dimensional) and $\lambda > 0$ is a penalty coefficient. This loss function balances the objectives of reconstructing the training data accurately, on the one hand, and minimizing

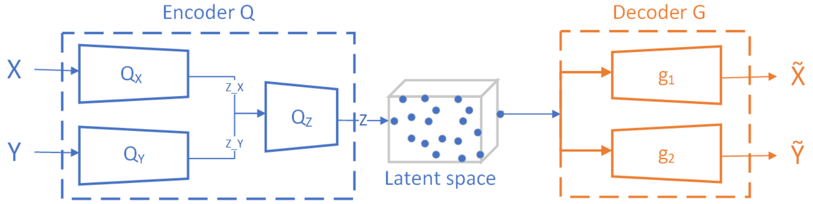

Figure 1: Schematic of the Encoder-Decoder architecture adapted to joint distribution learning

on the other hand a p-Wasserstein distance (i.e. W_p) between the induced distribution on $\mathbb{Z}$ (i.e. Q_Z) by the encoder and some prescribed prior distribution (i.e. P_Z) on $\mathbb{Z}$. Once the GNN is trained, we can cheaply generate couples $(\tilde{X}, \tilde{Y})$ by sampling from the prior on $\mathbb{Z}$ and passing those samples through the decoder networks, denoted for simplicity here and in the following by g_1, g_2 (cf. Figure 1).

Phase II - Inversion by Subset Simulation We now consider a measurement vector y_{obs} from which we want to retrieve a distribution of subsurface domains having potentially generated it. The goal in this phase is to first, explore the region of $\mathbb{Z}$ yielding values of g_2 close to y_{obs} and, second, transform sampled elements from this region back into $\mathbb{X}$ via g_1.

Let $\Gamma_\epsilon = \{z \in \mathbb{Z} : d(g_2(z), y_{obs}) \leq \epsilon\}$ with $d(.,.)$ a dissimilarity on $\mathbb{Y}$ (here we take the squared l_2 distance on $\mathbb{Y}$ rescaled by $\dim(\mathbb{Y})$) and $\epsilon > 0$ a tolerance parameter. Furthermore, let π_Z stand for the prior density of Z (with respect to Lebesgue or some other dominating measure on $\mathbb{Z}$). We consider here a surrogate posterior density on $\mathbb{Z}$ (given y_{obs}) defined by $\pi_{Z|Z\in\Gamma_\epsilon}(z) \propto \mathbb{1}_{\Gamma_\epsilon}(z)\pi_{\mathbb{Z}}(z)$. Depending on the choice of ϵ and other problem settings such as the dimension of $\mathbb{Z}$, $\{Z \in \Gamma_\epsilon\}$ may become a rare event to simulate. We use SuS [1], a rare event sampler, as an adaptive sampler in ABC [2] to sample from $\pi_{Z|Z\in\Gamma_\epsilon}$. SuS introduces a decreasing sequence of thresholds $+\infty = t_0 > t_1 > t_2 ... > t_m = \epsilon$ which determines a sequence of nested subsets of $\mathbb{Z}$, $\Gamma_{t_\ell} = \{z \in \mathbb{Z} : d(g_2(z), y_{obs}) \leq t_\ell\}(\ell = 0, ..., m)$. For the sequence of events $\{Z \in \Gamma_{t_\ell}\}$ we have that : $P(Z \in \Gamma_\epsilon) = P(Z \in \Gamma_{t_0}) \prod_{\ell=1}^{m} P(Z \in \Gamma_{t_\ell} | Z \in \Gamma_{t_{\ell-1}})$ with $P(Z \in \Gamma_{t_0}) = 1$ since $\{Z \in \Gamma_{t_0}\}$ is certain. This reduces the problem of estimating the small $p_\epsilon = P(Z \in \Gamma_\epsilon)$ to estimating a sequence of larger conditional probabilities $P(Z \in \Gamma_{t_\ell} | Z \in \Gamma_{t_{\ell-1}})$.

The SuS algorithm starts with an initial sample from the prior of $\mathbb{Z}$, $\{Z_i^{(0)}\}_{i=1}^{N}$, with a predefined size N. The dissimilarity values $\{d(g_2(Z_i^{(0)}), y_{obs})\}_{i=1}^{N}$ are calculated and ordered and the first threshold t_1 is defined as the α-percentile of those values. α is prescribed and typically chosen in the range $[0.1, 0.3]$ [17]. The set Γ_{t_1} is first populated by the observations from this initial sample that yield distances below t_1. Starting from each one of those succeeding observations, sufficiently many states of a Markov chain with stationary distribution $\pi_{Z|Z\in\Gamma_{t_1}}$ are generated to complete the current elements of Γ_{t_1} up to N elements (cf. [8] for specific details on the MCMC sampling methods with SuS). At each subsequent iteration $\ell = 2, ..., m$, the sample $\{Z_i^{(\ell-1)}\}_{i=1}^{N}$ is used to calculate $\{d(g_2(Z_i^{(\ell-1)}), y_{obs})\}_{i=1}^{N}$ and to set t_ℓ as the α-percentile of those distances. New observations in Γ_{t_ℓ} are again sampled starting from the observations that yield distances below t_ℓ. This process stops when ϵ is crossed (if the proposed $t_\ell \leq \epsilon$ then m is defined as ℓ and t_m is set equal to ϵ) or when a prescribed maximum number of iterations is reached. The final elements of Γ_ϵ are used to form candidate solutions in $\mathbb{X}$ via g_1. p_ϵ can be estimated via $\widehat{p_\epsilon} = \alpha^{m-1}\frac{N_{m-1}}{N}$, with N_{m-1} being the number of succeeding particles at the penultimate iteration of SuS. This estimator sheds some light on the diversity/uncertainty in the proposed solutions, from the GNN's latent space perspective.

3 Results

We applied our methodology to cross-hole tomography inversion with linear, that is, straight ray physics. Training and test sets of couples of subsurface domains and their solver outputs (i.e. (X, Y)) were simulated using a linear forward solver (cf. Figure 2). The training set was of size 9000 (sensitivity studies with a focus on smaller training set sizes ought to be performed in future work). The domain (i.e. X) was discretized on a grid of size 50 × 40 with a cell size of 0.1 m. The boreholes were located 4 m apart and placed at the left and the right hand side of the domain (cf. Figure 2(a)). Nine sources and nine receivers were located between 0.5 and 4.5 m depth with 0.5 m spacing leading to a total size of the measurement vector of 81 travel times. A Gaussian prior was used on the slowness field with a mean of 14 ns/m and an isotropic exponential kernel with a length scale of 2.5 m and a variance at the origin of 0.16 (ns/m)^2. We chose a latent space dimension of 30, with a standard Gaussian prior. The data used to train the GNN were not noise-contaminated. However, the measurement vector, y_{obs} was obtained by sampling a solver output from the test set and contaminating it with a noise vector. In our experiments, two noise vectors were sampled; one from independent centred Gaussian-distributed random variables with 0.5 ns standard deviation, and one from independent Gumbel-distributed random variables with 0.5 ns location and 1 ns scale. In both cases, we refer to the rescaled l_2 norm of the noise vector by the noise level.

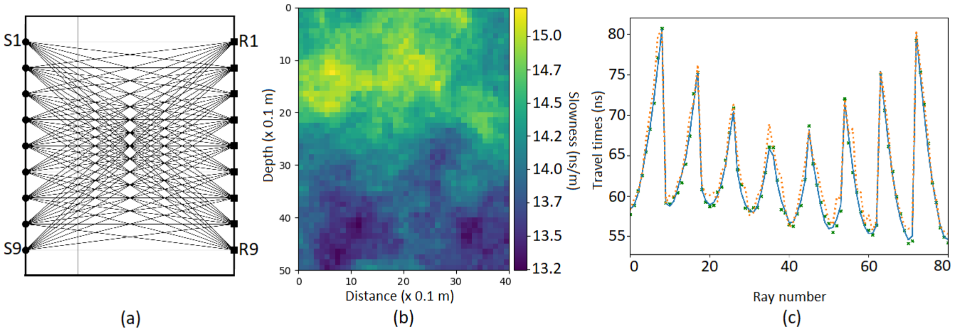

Figure 2: (a): Cross-hole tomography setup, S1-9 GPR sources and R1-9 GPR receivers; (b): slowness field in ns/m (high value corresponds to slow medium); (c): blue full line, travel times without noise, in ns given by the forward solver corresponding to the field on the left; green crosses, travel times contaminated by a standard Gaussian noise realization with a level of 0.52 ns; orange dotted line, travel times contaminated by a Gumbel noise realization with a level of 1.41 ns.

When the noise is Gaussian, given the Gaussian prior on the field and the linear solver, we can compare the proposed candidates obtained by our methodology with samples from the analytical posterior on $\mathbb{X}$ by implementing the analytical solutions available in [13]. Figure 3(b,c) represent a visual comparison of the pixels' means and standard deviations of the proposed solutions against the true Gaussian posterior sample statistics in Figure 3(a). With noise level at 0.52 ns, at tolerance 0.66 ns the proposed solutions approximate the posterior mean model and the true model relatively well with an average RMSE of 0.294 ns/m ([0.145, 0.444]) and 0.336 ns/m ([0.213, 0.460]), respectively. As a reference, the RMSE between samples from the analytical posterior and the true solution have an average RMSE of 0.298 ns/m ([0.220, 0.375]) indicating that our method was able to propose plausible solutions for this test case. Increasing the tolerance to 1.76 ns, more variability exists in the sampled solutions with corresponding average RMSE at 0.454 ns/m ([0.280, 0.629]) with respect to the analytical posterior mean and 0.488 ns/m ([0.323, 0.653]) with respect to the true solution. The pixels' standard deviations in Figures 3(b,c) also show this increase in variability, suggesting more diverse solutions with increasing tolerance that become less informative about the true solution. Under the assumption of an unknown noise distribution, we did not add a noise realization to the GNN output during the inversion phase. With this choice, at the lowest tolerance level reached by

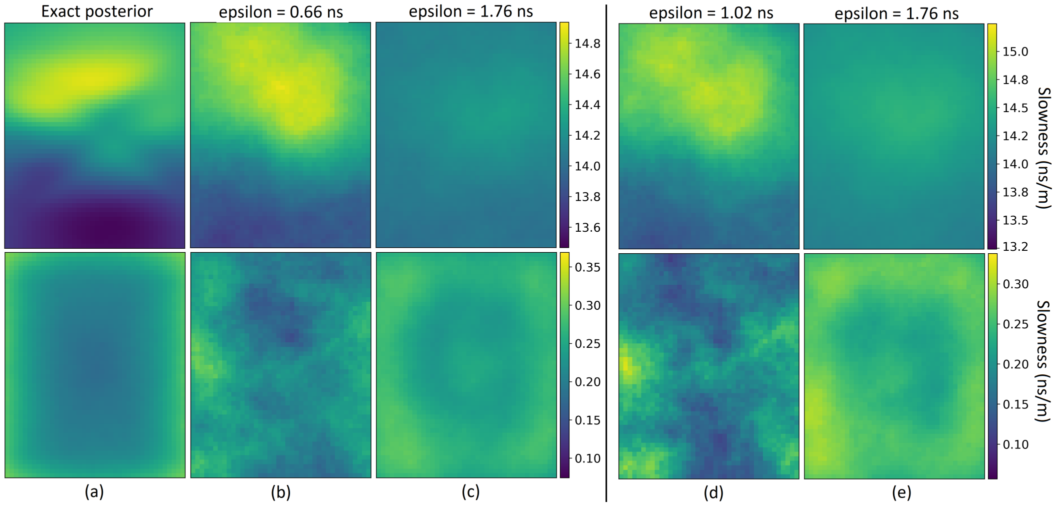

Figure 3: Mean (1st row) and standard deviation (2nd row) pixel wise summary statistics of the proposed solutions against: the reference Gaussian posterior (a) at different tolerance levels (b,c) for a Gaussian noise realization with a level of 0.52 ns; at different tolerance levels (d,e) with a Gumbel noise realization with a level of 1.41 ns.

SuS, the proposed solutions underestimate the uncertainty of the true posterior. Choosing a value for the tolerance higher than this minimum generates solutions with an uncertainty closer to the one in the true posterior, with a tendency to overestimate it (cf. Figures 3(b,c)).

When the noise is Gumbel-distributed with a level of 1.41 ns, an analytical posterior cannot be computed and we can only compare the proposed samples to the true solution from the test set shown in Figure 2(b). At tolerance 1.02 ns, the proposed solutions have an average RMSE of 0.381 ns/m ([0.275, 0.487]) with respect to the true solution. At tolerance 1.76 ns, the average RMSE between the proposed solutions and the true solution increases to 0.464 ([0.295, 0.633]). Reference values for this case are not available, nevertheless, Figure 3(d) visually illustrates that, on average, the proposed solutions are still able to provide information on the true solution at tolerance 1.02 ns with this level of noise.

To further illustrate the impact of tuning the tolerance level ϵ on the variability of the proposed solutions, we ran the algorithm multiple times with different ϵ values and estimated the probability that a randomly sampled Z from the prior belongs to the solution set Γ_ϵ. From Figure 4, we see how the size of the solution set increases with increasing ϵ for the two noise cases presented earlier. We also see that the minimum value of ϵ that is effectively reached by SuS is comparable to the noise level in the Gaussian noise case, or below it for the Gumbel noise case.

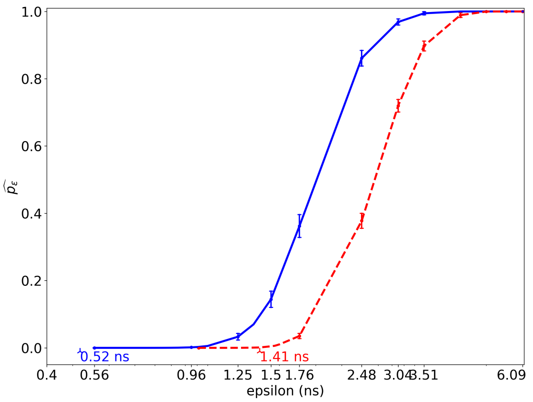

Figure 4: Probability for a randomly sampled latent Z from the prior to belong to the solution set Γ_ϵ vs tolerance level ϵ with 95% confidence intervals. The blue curve illustrates the case with Gaussian noise at level 0.52 ns. The red curve illustrates the case with Gumbel noise at level 1.41 ns. The horizontal scale is logarithmic.

In terms of computational time, the training of the GNN is generally expensive. However, it only needs to be done once, before the actual inversion starts, and is largely accelerated with parallel processing on graphical units. Our inversion procedure is based on the SuS implementation by Willer & Uribe [16]. It requires approximately 4 minutes to propose 1000 new solutions and happens completely offline without calling the forward solver. To establish a reference, we ran direct SuS on the domain space $\mathbb{X}$, sampling directly from the prior on $\mathbb{X}$ and passing those samples through the forward solver for evaluation against the observation y_{obs}. This direct approach required approximately 4 hours to generate 500 solutions on the same hardware. We note here that the direct method was slower because of the need to transform realizations of X from and to the standard space by the SuS algorithm and not due to the linear solver (which only requires 0.002 seconds on average per forward evaluation).These transformations became costly compared to transforming the latent space realizations in our method due to the high dimension of $\mathbb{X}$ compared to $\mathbb{Z}$. Furthermore, the cost of the direct method is expected to further increase with the increasing evaluation time of the forward solver and the increase in the dimension of $\mathbb{X}$ which reduces the acceptance rate of the SuS sampler.

4 Conclusion & Future work

Those initial results suggest that the proposed methodology could offer a fast and cheap way to explore the solution space of an inverse problem. Furthermore, it allows to tune the exploration strategy and the uncertainty of the solutions by tuning a tolerance parameter. Presently, we are testing this methodology with non-linear physics and more complex noise distributions to demonstrate its ability to generalize. Furthermore, we are investigating how some hyper-parameters influence the inversion, namely the dimension of the latent space, the architecture choices in the GNN, the training sample size and the addition of noise during training or inversion.

Acknowledgments

The authors wish to thank the reviewers for their remarks that helped to improve the paper. EM wishes to thank Shiran Levy for sharing scripts concerning the forward solver and for insightful discussions, and Athénaïs Gautier for proofreading this manuscript. DG's contributions have taken place within the Swiss National Science Foundation project number 178858.

Broader Impact

The presented methodology could play an important role in solving inference problems in geophysics and other domains like cosmology and medical imaging. It provides a rapid exploration of regions of credible solutions to an inverse problem. It also provides a tuning parameter to regulate the uncertainty level that is tolerated in the solutions to be sampled and, hence, allow for a wider or narrower exploration strategy of the space. Furthermore, it can be adapted to different, pre-trained, joint and conditional generative models.

References

[1] Au, S.K., Beck, J.L.: Estimation of small failure probabilities in high dimensions by subset simulation. Probabilistic engineering mechanics **16**(4), 263–277 (2001)

[2] Chiachio, M., Beck, J.L., Chiachio, J., Rus, G.: Approximate bayesian computation by subset simulation. SIAM Journal on Scientific Computing **36**(3), A1339–A1358 (2014)

[3] Laloy, E., Hérault, R., Jacques, D., Linde, N.: Training-Image Based Geostatistical Inversion Using a Spatial Generative Adversarial Neural Network. Water Resources Research **54**, 381–406 (2018)

[4] Laloy, E., Linde, N., Jacques, D.: Approaching geoscientific inverse problems with adversarial vector-to-image domain transfer networks. arXiv preprint arXiv:1912.09954 (2019)

[5] Marjoram, P., Molitor, J., Plagnol, V., Tavaré, S.: Markov chain monte carlo without likelihoods. Proceedings of the National Academy of Sciences **100**(26), 15324–15328 (2003)

[6] Mosser, L., Dubrule, O., Blunt, M.J.: Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. Mathematical Geosciences **52**(1), 53–79 (2020)

[7] Padmanabha, G.A., Zabaras, N.: Solving inverse problems using conditional invertible neural networks. arXiv preprint arXiv:2007.15849 (2020)

[8] Papaioannou, I., Betz, W., Zwirglmaier, K., Straub, D.: Mcmc algorithms for subset simulation. Probabilistic Engineering Mechanics **41**, 89–103 (2015)

[9] Patrini, G., Carioni, M., Forré, P., Bhargav, S., Welling, M., van den Berg, R., Genewein, T., Nielsen, F.: Sinkhorn autoencoders. CoRR **abs/1810.01118** (2018)

[10] Richardson, A.: Generative adversarial networks for model order reduction in seismic full-waveform inversion. arXiv preprint arXiv:1806.00828 (2018)

[11] Robert, C.P., Beaumont, M.A., Marin, J.M., Cornuet, J.M.: Adaptivity for abc algorithms: the abc-pmc scheme. ArXiv e-print **805** (2008)

[12] Robert, C.P., Casella, G.: Monte carlo statistical methods (2004)

[13] Tarantola, A.: Inverse problem theory and methods for model parameter estimation, vol. 89. SIAM (2005)

[14] Tolstikhin, I., Bousquet, O., Gelly, S., Schölkopf, B.: Wasserstein auto-encoders. In: International Conference on Learning Representations (ICLR 2018) (2018)

[15] Villani, C.: Optimal transport: old and new, vol. 338. Springer Science & Business Media (2008)

[16] Willer, Mathias and Uribe, Felipe: Subset simulation (2020), `https://www.bgu.tum.de/era/software/software00/subset-simulation/`, accessed November 29, 2020

[17] Zuev, K.M., Beck, J.L., Au, S.K., Katafygiotis, L.S.: Bayesian post-processor and other enhancements of subset simulation for estimating failure probabilities in high dimensions. Computers & structures **92**, 283–296 (2012)