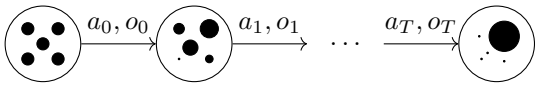
Conditional Deep Generative Models for Belief State Planning

Abstract

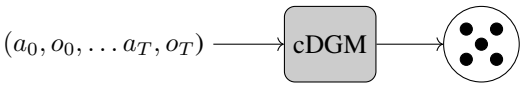
Partially observable Markov decision processes (POMDPs) are used to model a wide range of applications, including robotics, autonomous vehicles, and subsurface problems. However, accurately representing the belief is difficult for POMDPs with high-dimensional states. In this paper, we propose a novel approach that uses conditional deep generative models (cDGMs) to represent the belief. Unlike traditional belief representations, cDGMs are well-suited for high-dimensional states and large numbers of observations, and they can generate an arbitrary number of samples from the posterior belief. We train the cDGMs on data produced by random rollout trajectories and show their effectiveness in solving a mineral exploration POMDP with a large and continuous state space. The cDGMs outperform particle filter baselines in both task-agnostic measures of belief accuracy as well as in planning performance.

1 INTRODUCTION

Partially observable Markov decision processes (POMDPs) can model a wide range of applications such as robotics [Pineau and Gordon, 2007], autonomous vehicles [Wray et al., 2021], and, more recently, subsurface problems such as ground water [Wang et al., 2022], carbon capture and storage [Corso et al., 2022], and mineral exploration [Mern and Caers, 2023]. A major challenge for solving POMDPs with high dimensional states is the difficulty of accurately representing the belief, which is the probability distribution over the state of the system given the history of actions and observations. In this paper, we propose a novel approach to address this challenge using conditional deep generative models (cDGMs) to represent the belief.



(a) Particle Filter Belief Update



(b) cDGM Belief Update

Figure 1: Summary of proposed approach

Traditional belief representations, such as particle filters, are often not suitable for high-dimensional states and large numbers of observations. They often suffer from *particle depletion*, where, after successive iterations of re-weighting, only a small number of particles retain most of the weight, leading to a low quality approximation of the posterior belief (depicted in fig. 1a). Our approach (depicted in fig. 1b) trains a deep generative model that is conditioned directly on a sequence of actions and observations, and can generate an arbitrary number of samples from the posterior belief. The cDGMs are trained from data produced by random rollout trajectories, which allows for the complex Bayesian inversion to be learned directly from the training data.

We apply our proposed approach to solve a mineral exploration POMDP [Mern and Caers, 2023] that has a large and continuous state space as well as continuous, noise-free observations. We investigate two types of cDGMs: generative adversarial networks (GANs) and denoising diffusion probabilistic models (DDPMs). We find that both model classes outperform a particle filter baseline on task-agnostic metrics, and that the DDPM models enable significanlty improved planning performance.

Prior work [Mosser et al., 2018, Laloy et al., 2017, Mosser et al., 2017, Song et al., 2021] has investigated the use of deep generative models for geoscience applications, but does not condition the generation of samples on observed data. To condition to observations, traditional Bayesian in-

Submitted to the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023). **To be used for reviewing only**.

version approaches are combined with unconditioned deep generative models. Laloy et al. [2017] use MCMC with the generator of a variational autoencoder while Liu et al. [2019] use a form of principle component analysis to produce conditioned samples and use deep style transfer to refine the samples. In contrast to prior work, we condition to observations directly in the generative model, and use the model for POMDP planning.

In summary, our contributions in this paper are as follows

- We propose the use of conditional deep generative models to represent beliefs for belief-state planning solvers of POMDPs.
- We experiment with a wide range of state-of-the-art deep generative models, and identity architectures that are well suited for observation conditioning.
- We show that cDGM belief representations outperform other representations when planning under uncertainty in a mineral exploration problem.

2 BACKGROUND

This section introduces the topics of POMDPs, particle filter approximations to Bayesian inference and deep generative models. Then, we provide a summary of our motivating application domain: mineral exploration.

2.1 POMDPS

A partially observable Markov Decision Process (POMDP) is a model for sequential decision making under state uncertainty [Kochenderfer et al., 2022]. A POMDP is defined by the tuple $(S, A, T, R, O, Z, \gamma)$ where S is the set of states, A is the set of actions, T is a transition model, R is the reward function, O is the observation space, Z is the observation model, and γ is the discount factor. At each timestep, an agent takes an action $a \in A$ in a state s and transitions to the next state $s' \sim T(\cdot \mid s, a)$, receiving a reward $r = R(s, a)$ and an observation $o \sim Z(\cdot \mid s, a, s')$. Since states are not fully observed, the agent maintains a probability distribution over states or belief $b(s)$ that is updated based on the action-observation trajectory $\tau = \{a_1, o_1, a_2, o_2, \ldots\}$. The agent's policy $a = \pi(b)$ maps beliefs to actions with the goal of maximizing the expected discounted sum of future rewards.

POMDPs can be solved exactly only for problems with small state, action, and observation spaces, but for larger problems, approximate solvers are used [Kochenderfer et al., 2022]. The most scalable approaches often use forward tree search, where hypothetical trajectories are simulated and used to estimate the value of different actions. Some examples of tree-search-based POMDP solvers include POMCP [Silver and Veness, 2010], POMCPOW and PFT [Sunberg and Kochenderfer, 2018].

2.2 PARTICLE APPROXIMATIONS FOR BAYESIAN INFERENCE

Accurate belief representations are an essential element to solving POMDPs, both for planning and execution. After taking action a, the prior belief $b(s)$ is updated with the observation o using Bayes' rule

$$b(s') = O(o \mid a, s') \int_s T(s' \mid s, a) b(s) ds \tag{1}$$

Exact Bayesian inference is often intractable for nonlinear dynamics, and beliefs that are not well represented by simple parametric models. Instead, the belief can be approximated by a weighted set of state samples, or particles. The weight of each particle is computed as the conditional likelihood of the state give the action-observation trajectory $w = p(s \mid \tau)$. To avoid retaining τ, implementations often use a bootstrap filtering algorithm where the belief is represented by N equally weighted samples of the state $b \approx \{s_i\}_{i=1}^N$ that are updated according to the following computational steps:

1. Sample new states from the transition function: $s_i' \sim T(\cdot \mid s_i, a)$.
2. Compute particle weights according to the observation model: $w_i = O(o \mid a, s_i')$.
3. Resample N particles from $\{s_i'\}_{i=1}^N$ according to their weights$\{w_i\}_{i=1}^N$.

Particle filters are flexible belief representations which have favorable convergence properties [Crisan and Doucet, 2002], but they require the observation likelihood $O(\cdot \mid, a, s')$. In problems where the observation likelihood is undefined (as with noiseless observations), or intractable to compute, an additional approximation is needed to compute the likelihood weights. Approximate Bayesian computation (ABC) is a particle filtering approach where the likelihood function is replaced with a approximation $\tilde{O}$. For example, in the case of noiseless observations where $o = Z(s)$, an approximate observation model could be a Gaussian of the form

$$\tilde{O}(o \mid a, s_i) \propto e^{-\|o - Z(s)\|^2 / 2\sigma_{\text{ABC}}^2} \tag{2}$$

for a user-specified value of σ_{ABC}. ABC introduces bias into the belief representation, but is often found to be a useful approximation in practice [Csilléry et al., 2010].

Particle approximations to the belief can suffer from particle depletion, where only a small number of particles retain most of the weight. Particle depletion is exacerbated when the state space is high dimensional and where there are long sequences of actions and observations [Snyder et al., 2008]. It may be mitigated through particle injection, where new particles matching current actions and observations are added to the belief representation [Van Leeuwen et al., 2019]. The tractability of adding new particles, however, depends on the efficiency of generating new state samples.

2.3 DEEP GENERATIVE MODELS

Deep generative models (DGMs) are machine learning models that are trained to represent the underlying distribution of training data, $x \sim P_x$, enabling the generation of novel samples that closely resemble the original data. Well-trained generative models not only capture the salient features of the training data, but also facilitate the generation of diverse outputs. Due to the complexity of the training data distribution, deep neural networks (DNNs) are often used as the model class, but the training approaches can vary widely. In this study, we use two prominent types of DGMs, namely generative adversarial networks (GANs) and denoising diffusion probabilistic models (DDPMs).

Generative Adversarial Networks (GANs) GANs consist of two components: the generator network G which generates new samples $\tilde{x}$, and the discriminator network D, which learns to distinguish the real and generated data. The two networks are trained simultaneously, competing against each other in a minimax game. A GAN has converged once the discriminator is unable to distinguish the generated and real data. Despite their many recent success [Sauer et al., 2022, Kang et al., 2021], GANs often lack stability during training leading to low-quality samples, or mode collapse, where the diversity of samples is compromised [Kynkäänniemi et al., 2019, Saxena and Cao, 2021].

Several loss functions have been proposed for training GANs. Let $z \sim P_z$ be a latent vector sampled from a known probability distribution. The binary cross entropy loss [Goodfellow et al., 2014] treats the discriminator as a classifier model where real images that are labeled 1 and generated images are labeled 0, while the generator is trained to maximize the discriminator's loss:

$$L_D = \mathbb{E}_{\substack{x\sim P_x \\ z\sim P_z}} [\log(D(x)) + \log(1 - D(G(z)))] \quad (3)$$

$$L_G = \mathbb{E}_{z\sim P_z} [\log(1 - D(G(z)))] \quad (4)$$

The Wasserstein loss [Arjovsky et al., 2017] was proposed as a way to stabilize training and is given by

$$L_D = \mathbb{E}_{\substack{x\sim P_x \\ z\sim P_z}} [-D(x) + D(G(z))] \quad (5)$$

$$L_G = \mathbb{E}_{z\sim P_z} [-D(G(z))] \quad (6)$$

The Wasserstein loss is often coupled with a gradient penalty, which limits the norm of the gradient when training the discriminator [Gulrajani et al., 2017].

For our experiments, we consider two types of DNN architectures.

1. Deep Convolutional GAN (DCGAN): The DCGAN architecture uses feed-forward convolutional neural networks for both the generator and the discriminator. The specific number of convolutional filters and layers may vary, but our architecture is based on prior work [Mirza and Osindero, 2014].
2. StyleGAN: The StyleGAN architecture was proposed by Karras et al. [2019] and obtained state-of-the art-performance on image generation tasks. Contrary to traditional approaches that feed the latent vector only into the first layer, the StyleGAN generator maps the latent vector to an intermediate space, which is used in each layer of the network.

Denoising Diffusion Probabilistic Models (DDPMs) DDPMs are based on the idea of using a diffusion process to iteratively de-noise a noisy input and generate a clean output. During training, a random $t \in [1, T]$ is sampled, where T is the number of denoising steps. Given the original image x_0, t-steps of Gaussian noise can be added by computing

$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon \quad (7)$$

where $\epsilon \sim \mathcal{N}(0, I)$ is Gaussian noise of the same dimension as x and $\overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ for a noise schedule α_t. A network U (typically a UNet model [Ronneberger et al., 2015]) is trained using the mean squared error between its prediction of the image noise and the actual noise

$$L = \|U(x_t, t) - \epsilon\|^2 \quad (8)$$

During inference, we start at step T, with $x_T \sim \mathcal{N}(0, I)$, and iteratively compute x_{t-1} with the following recurrence relation:

$$x_{t-1} = \frac{1}{\alpha_t}\left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \overline{\alpha}_t}} U(x_t, t)\right) + (1 - \alpha_t)\epsilon \quad (9)$$

where $\epsilon \sim \mathcal{N}(0, 1)$ when $t > 1$ and $\epsilon = 0$ otherwise. See Ho et al. [2020] for more details about the UNet architecture and the (de)noising process.

DDPMs are known for both the quality of their outputs and their training stability. However, the major drawback is the time of inference. In order to produce a sample, DDPMs require T inference steps where T is usually between 100 and 1000. In contrast, GANs only require a single inference pass to produce a sample.

2.4 MINERAL EXPLORATION POMDP

Batteries are a key technology enabling the transition from fossil fuels to renewable forms of energy, but require large amount of rare earth elements such as copper, nickel, cobalt and lithium [Sovacool et al., 2020]. The rate of discovery of new mineral deposits has decreased while the demand for these elements has increased [Davies et al., 2021]. Mining operations for these elements are expensive and can have significant environmental and social impacts, so finding ore deposits that are economically viable to mine and in the right geographical regions is essential.

Recent work has formulated the mineral exploration problem as a POMDP [Mern and Caers, 2023]. The states are 2-dimensional 32×32 maps (see fig. 4), with each grid cell having a real value in $[0, 1]$ indicating the local ore concentration. The actions are to drill observational boreholes, or make a final decision to abandon the project or go forward with the mining operation and extract the ore. When a borehole is drilled, the ore density at the drill location is observed directly without any noise. The cost of drilling boreholes is a fixed value $R_{\text{drill}}(s) = -0.1$, and we fix a discrete set of potential borehole locations on a 6×6 grid. If the project is abandoned, then the episode ends with no additional reward $R_{\text{abandon}}(s) = 0$, but if the agent chooses to mine, then the reward is the sum of grid cells where the ore quantity is above a concentration threshold $\rho = 0.7$ minus a fixed capital expenditure $R_{\text{capex}} = -52$,

$$R_{\text{mine}}(s) = R_{\text{capex}} + \sum_{i=1}^{32} \sum_{j=1}^{32} \mathbb{1}_{\{s_{i,j} > \rho\}} \tag{10}$$

where $\mathbb{1}$ is the indicator function and $s_{i,j}$ is the ore value at row i and column j. There are no dynamics to this problem, making it a POMDP-lite [Chen et al., 2016], where information-gathering is the primary goal.

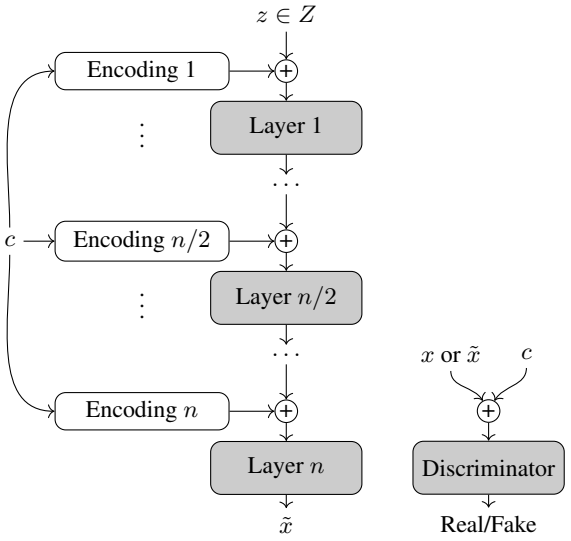
Each mineral deposit is defined by a 100-point polygon that is smoothed with a Gaussian kernel and has Gaussian process noise added on top. The initial state distribution is defined and sampled from using a publicly available repository[1]. The values of ρ and R_{capex} were selected so that the mean return of the initial state distribution is near 0.

3 CONDITIONAL DEEP GENERATIVE MODELS

This section discusses the conditioning of deep generative models and outlines the design choices made for both GANs and DDPMs. We then introduce a variety a metrics that can be used to evaluate conditional generative models and discuss how they can be incorporated into a POMDP solver.

3.1 CONDITIONING THE MODELS

The goal of conditional generative models is to represent the conditional probability distribution $P_{x|c}$ instead of the marginal distribution P_x, where c is a condition that describes x [Mirza and Osindero, 2014, Ho et al., 2020]. In image generation, the conditioning is often done based on a desired class from a discrete set (e.g. conditioning on the digit in MNIST generation). To train a class-conditional generative model, the class label (typically as a one-hot encoding) is passed to the DNN (often after being further encoded via a trainable embedding layer) along with the corresponding training data point. Conditioning on continuous values has been studied in the domain of super-resolution generative models [Ledig et al., 2017] as well as image in-painting [Yu et al., 2018]. The continuous condition is encoded in a matrix that contains data about all or part of the pixels of the image and is concatenated with the latent vector (with or without some additional encoding) [Ledig et al., 2017, Yu et al., 2018].



(a) Generator (b) Discriminator

Figure 2: GAN Conditioning

The mineral exploration problem has continuous conditioning variables (the mineral density observations), but unlike prior work on super-resolution and in-painting, the conditioning variables are sparse: between 1 and 36 points on a 32×32 grid. The mineral density observation is encoded into a tensor c of shape $2 \times 32 \times 32$ where the first channel is a one-hot encoding of where the observations have been made and the second channel contains the corresponding observation value, and is 0 anywhere that has not been observed.

The conditioning strategies for the GAN models are shown in fig. 2. At the first layer the condition c is passed through an encoding layer and concatenated with the latent vector of the generator, and concatenated to the given sample for the discriminator. Motivated by prior work [Karras et al., 2019] and initial experimentation, we found that injecting the condition at additional layers proved useful in producing an accurate belief representation. We therefore consider two additional configurations: *half*, where we inject the condition into each of the first half of the layers of the generator and *all* where it injected at every layer. Each layer that the condition is injected in has its own trainable encoding layer. A similar approach is employed for conditioning the DDPM

[1] https://github.com/sisl/MineralExploration

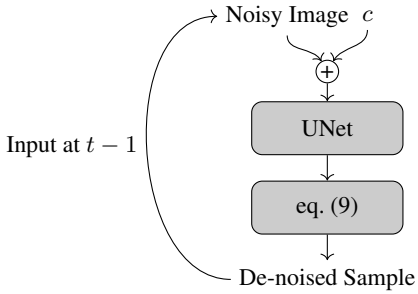


Figure 3: DDPM Conditioning on step t of denoising.

model as shown in fig. 3. The 2-channeled condition is concatenated to each noisy image prior to its input into the network.

3.2 BELIEF EVALUATION METRICS

Evaluating the quality of a conditional generative model is challenging for complex and high dimensional distributions. Prior work has primarily focused on image generation of common objects or human faces, and the corresponding metrics have therefore been designed to correlate with subjective notions of image realism [Heusel et al., 2017]. In this work however, we are primarily concerned with how well the belief representation covers the training distribution and matches the observations/conditions, as well as how the belief representation leads to good planning performance. We therefore study a variety of task-agnostic and task-specific metrics.

To do so, we construct an evaluation set of N_{test} state-history pairs (s_i, τ_i). For each sample in the evaluation set, the posterior is modeled using the chosen belief representation and used to produce 500 posterior samples. From those samples we can compute the following metrics.

Task-agnostic metrics We wish to know how well the generative model distribution covers the posterior distribution over states conditioned on a set of actions and observations. We therefore measure test set recall [Kynkäänniemi et al., 2019] as well as a measure of observation similarity to evaluate the cDGMs in a model agnostic way.

- *Test set recall*: To evaluate recall, we determine how well each belief representation can reproduce samples from the test set. We measure the distance between the closest generated sample in the belief to each of the test set points:

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \min_{\tilde{s} \sim b(\cdot|\tau_i)} \|\tilde{s} - s_i\| \tag{11}$$

  Initial experimentation found that an L_2 norm between samples was sufficient to distinguish high quality models from low quality ones, so we use it for our distance metric.

- *Observation error*: To evaluate the conditioned points we measure the distance between the imposed conditioning and the resulting observations from produced samples:

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathbb{E}_{\tilde{s} \sim b(\cdot|\tau_i)} \|Z(\tilde{s}) - Z(s_i)\| \tag{12}$$

- *Wall-clock time*: To evaluate the computational efficiency of a belief representation, we measure the amount of time it takes to produce samples from the posterior.

Task-specific metrics Knowing the downstream task that the belief representation will be used for allows us to design task-specific metrics for belief evaluation. For the mineral exploration problem, we can compute the expected value of the ore in the mineral deposit

$$\bar{R}_i = \mathbb{E}_{\tilde{s} \sim b(\cdot|\tau_i)} \left[R_{\text{mine}}(\tilde{s}) \right] \tag{13}$$

and to quantify uncertainty, we can compute the variance of the belief over the value of ore as

$$\sigma_i^2 = \text{Var}_{\tilde{s} \sim b(\cdot|\tau_i)} \left[R_{\text{mine}}(\tilde{s}) \right] \tag{14}$$

We then compute the following metrics:

- *Ore value error*: This metrics measure how well the belief models the true value of the ore. The mean value of the ore is computed from the posterior samples and we measure the distance between the estimated value and the ground truth for each sample:

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \left| R_{\text{mine}}(s_i) - \bar{R}_i \right| \tag{15}$$

- *Decision accuracy* measures how well the belief informs the final mine/abandon decision of the agent. The agent is forced to make a mine or abandon decision based only on the posterior samples. The decision to mine is correct if there is a positive economic return for mining, and the decision to abandon correct if there is a negative economic return. The decision accuracy is computed as

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathbb{1}_{\left\{\text{sign}(R_{\text{mine}}(s_i)) = \text{sign}\left(\bar{R}_i\right)\right\}} \tag{16}$$

- *Probability density of ore value* measures how well the belief representation models uncertainty. We measure the probability density of the ground truth for each

sample, assuming a normal probability distribution for the estimated ore value.

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathcal{N}(R_{\text{mine}}(\tilde{s}) \mid \bar{R}_i, \sigma_i^2) \tag{17}$$

where $\mathcal{N}$ is a normal distribution.

3.3 PLANNING WITH CDGMS

Beliefs represented by cDGMs can replace other implicit belief representations used in POMDP planning algorithms. For example, the particle filter tree (PFT) algorithm [Sunberg and Kochenderfer, 2018] uses Monte Carlo tree search (MCTS) over the belief space approximated with particles, and uses a particle filter to add new belief nodes to the tree. A cDGM could replace this representation directly, however, MCTS is an inherently serial algorithm, requiring only one new posterior sample at each iteration. To get the full benefit of GPU hardware acceleration, we want an algorithm that can batch the generation of posterior samples, so in this work we devise a POMDP solver based on value of information (VOI) [Kochenderfer and Wheeler, 2019], which easily parallelizes the generation of posterior samples. While a policy based on VOI will not recover the optimal POMDP solution, in practice it produces competitive results (see table 2). Also, the VOI policy explicitly reasons about information gain, which better differentiates the quality of the different belief representations.

Algorithm 1 Value of Information Policy

1: **function** VOIPOLICY($\mathcal{P}$, b)
2: $\{s_i\}_{i=1}^N \sim b$
3: $V \leftarrow \emptyset$
4: **for** a in A
5: $V_a \leftarrow \emptyset$
6: **for** i in $\{1, \ldots, N\}$
7: $s_i', o_i, r_i \leftarrow \text{gen}(\mathcal{P}, s_i, a)$
8: $b_i' \leftarrow \text{update}(b, a, o_i)$
9: $\{s_j\}_{j=1}^M \sim b_i'$
10: $\bar{R}_i \leftarrow \text{mean}(\{R_{\text{mine}}(s_j)\}_{j=1}^M)$
11: $V_a \leftarrow V_a \cup r_i + \max(0, \bar{R}_i)$
12: $V \leftarrow V \cup \text{mean}(V_a)$
13: **return** $A[\arg\max V_{\text{drills}}]$

The algorithm (see algorithm 1) takes as input the POMDP $\mathcal{P}$ and the current belief b and returns the next action. The algorithm starts by taking N samples from the belief (line 2) and initializes an array of values (line 3) that will correspond to each action in the action space A. Then, for each action, we loop over the sampled states and for each one, sample a next state, observation, and reward from the POMDP (line 7). The observation is used to update the belief (line 8) from which M new states are sampled (line 9). Those samples are used to compute the estimated value of the belief (line 10). The value for action a in state s_i is computed by assuming the next action is a terminal one (line 11). The expected value for action a is computed by averaging over the N sampled states (line 12), and the action with the highest value is returned (line 13).

4 EXPERIMENTS

We trained a variety of cDGM models on training data consisting of $90\,000$ ore map samples. The action-observation histories were sampled randomly for each batch of training data. The number of actions N_a was sample from an exponential distribution $N_a \sim 1 + \text{Exp}(\lambda = 0.2)$ and the drill locations were sampled uniformly at random from the 32×32 grid. We trained our models for 50 epochs, using a learning rate of 4×10^{-4} for the generator and the discriminator in the GANs and 2×10^{-4} for the DDPMs.

For the GAN models we experimented with the injection strategy (1st, half, all), the number of channels in the convolutions $N_{\text{channels}} \in \{2, 8\}$, the dimension of the latent vector $N_z \in \{32, 64, 128\}$, and the loss function (Wasserstein (W), cross-entropy (CE), and Wasserstein with gradient penalty (W-GP). The GAN models are denoted "GAN (injection, N_{channels}, N_z, loss)". For the DDPM models we experimented with model size (small, medium, large) as well as the number of denoising iterations $N_{\text{iterations}} \in [100, 1000]$ and denote these models "DDPM (size, $N_{\text{iterations}}$)". We compared cDGM belief representations to ABC particle filters with the approximate likelihood given in eq. (2). For the particle filter beliefs, we vary the number of particles ($N_{\text{Particles}} \in \{1\text{k}, 10\text{k}, 100\text{k}\}$) and choice of likelihood width ($\sigma_{\text{ABC}} \in \{0.01, 0.05, 0.1\}$). The particle filter baselines are denoted "PF ($N_{\text{Particles}}$, σ_{ABC})". Additional details and experiments are reported in the supplemental material. In the following analysis we selected the highest performing belief representations from each model type.

4.1 EVALUATING BELIEF REPRESENTATIONS

We first evaluate each belief representation via a qualitative analysis of generated samples (shown in fig. 4). We see that while all samples are able to roughly match the size, shape and location of the ore deposit, there are minor variations that could be important for POMDP planning. Due to particle depletion, the best particle filter samples don't match the fine details of the ground truth sample, and can't improve with additional observations once the particle set is down to a single sample. The GAN model matches the shape of the ore deposit but has artifacts in the space between observations, leading to a lower predicted value of ore in the deposit. The DDPM model is able to match the ore deposit closely, and has the closest agreement with the ground truth with the maximum number of observations.

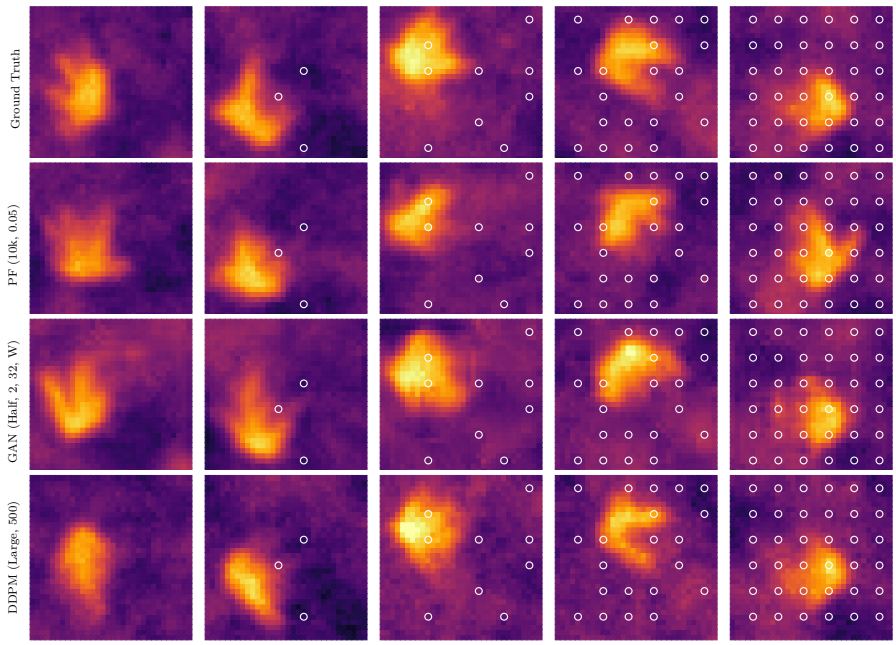


Figure 4: Samples generated from each belief representation. The top row consists of different ground-truth samples, and each other row contains the closest generated sample (out of 500) for each belief representation. The white circles which drilling locations and were used for conditioning the sample. The color inside the white circle shows the ground truth observation.

Table 1: Comparison of belief representations across task-agnostic and task-dependent evaluation metrics. Reporting mean and standard deviation in parentheses.

Belief Representation	Min $L_2(\downarrow)$	Conditioning Error(↓)	Ore Value Error(↓)	Decision Accuracy(↑)	Prob. of Ore Value(↑)	Time(↓)
PF (10k, 0.05)	2.74(0.30)	0.05(0.01)	14.39(11.33)	0.67(0.47)	0.02(0.03)	0.36(0.21)
PF (100k, 0.05)	2.52(0.28)	0.04(0.01)	13.35(10.44)	0.70(0.46)	0.02(0.03)	3.93(2.20)
GAN (Half, 2, 32, W)	2.47(0.56)	0.03(0.01)	15.90(13.71)	0.70(0.46)	0.02(0.02)	**0.22(0.00)**
GAN (All, 2, 128, W)	2.43(0.51)	0.03(0.01)	20.01(16.61)	0.62(0.49)	0.02(0.02)	0.23(0.00)
DDPM (Large, 250)	2.04(0.49)	**0.01(0.00)**	12.72(11.24)	0.71(0.45)	0.02(0.01)	29.17(0.02)
DDPM (Large, 500)	**2.02(0.47)**	**0.01(0.00)**	**12.18(10.25)**	**0.72(0.45)**	0.02(0.01)	58.56(0.03)

We make these qualitative observations more concrete by evaluating each belief representation according to the metrics described in section 3.2 with a test set with $N_{\text{test}} = 1300$. The results are reported in table 1. Across almost all of the metrics the DDPM models outperformed the other belief representations, but at the cost of higher wall-clock time. The GAN models performed well on the task-agnostic metrics, but suffered in the task-specific metrics, though they were by far the most computationally efficient representation.

In addition to studying the average value of the metrics, we also investigate each metric versus the number of actions and observations used to condition the posterior and plot the results in fig. 5. The cDGMs continue to reduce the minimum L_2 distance with more observations (with DDPMs outperforming GANs) while the particle filters flatten out after about 10, due to particle depletion. Both the particle filter and to a lesser extent, the GANs have increased average observation error with the number of observations while the DDPM models have a consistently low observation error. Over value error decreases and decision accuracy increases with number of observations, but the GAN models are typically lower performing. The probability density of the ground truth increases with observations (as the posterior belief becomes narrower), but this effect is limited for the particle filters, again due to particle depletion. Lastly, we note that the wall clock time for the particle filter representation scales linearly with the number of observations, while the cDGMs have a constant computational cost, albeit much larger for the DDPM models.

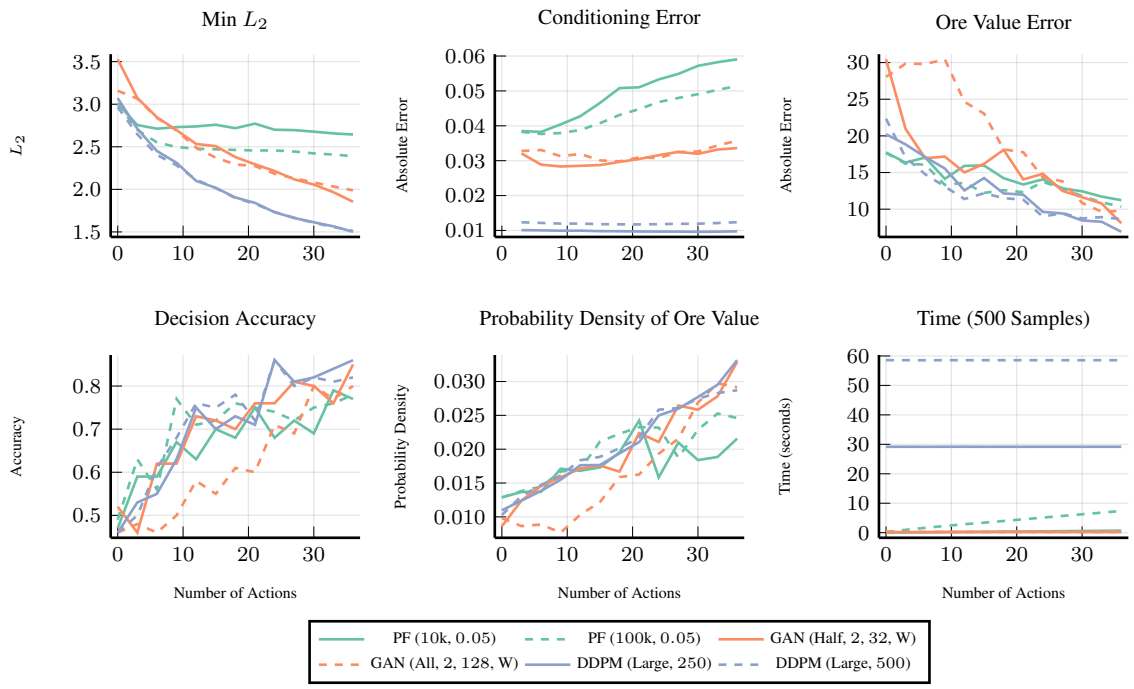


Figure 5: Belief metrics vs. number of actions/observations.

Table 2: Planning performance with various belief representations.

POMDP Solver	Return
POMCPOW w/ PF (10k, 0.05)	4.44
VOI w/ PF (10k, 0.05)	3.37
VOI w/ PF (10k, 0.1)	6.10
VOI w/ GAN (Half, 2, 32, W)	1.78
VOI w/ GAN (All, 2, 128, W)	2.45
VOI w/ DDPM (Large, 250)	6.42
VOI w/ DDPM (Large, 500)	**7.04**

4.2 PLANNING EXPERIMENTS

For the VOI policy we use $N = 50$ observations and $M = 10$ samples to estimate the return. Additionally, we consider a set of $|A| = 50$ actions that include multiple bore-hole locations. The actions are sampled by first selecting the number of bore-hole locations (uniform in $[1, 10]$) and then sampling that number of undrilled locations on the 6 grid. The results of our planning experiments (evaluated on 46 test ore maps) are shown in table 2. The VOI policy with DDPM models obtains the highest return with the 500-iteration model performing better than the 250-iteration model. They outperform the POMCPOW algorithm with a particle filter belief as well as the baseline particle filters used with the same VOI policy. The GAN models perform comparatively poorly, as predicted by the task-dependent measures of the belief representation.

5 CONCLUSION

In this work we found that cDGMs can indeed be useful representations of the belief for belief state planning, even when compared to a particle filter with a similar number of samples used to train the cDGM. We found, however, that the specific algorithm used to train the belief is important to planning performance (i.e. DDPMs significantly outperform GANs). The difference between the quality of belief representations should be evaluated with task-specific metrics, as task agnostic metrics may be misleading (as shown by GANs outperforming particle filters on the task agnostic metrics but ultimately having worse planning performance).

Limitations and Future Work Our work is subject to a number of limitations that we plan to address in future work:

- The current POMDP does not have a dynamics function, which makes it easier to process sequences of actions and observations. Future work will study POMDPs with a dynamics function, and apply time-series modeling approaches to embedding the action-observation sequence into a fixed sized condition vector.
- The mineral exploration POMDP currently relies on synthetic data that lies on a relatively low-dimensional

manifold. Future work will use higher-fidelity geological data, a setting where particle filtering performs even worse.

- We only compare our cDGM belief representations to particle filtering approaches. More advanced belief representations have been developed for geological data and could be compared to our cDGM approach.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017.

Min Chen, Emilio Frazzoli, David Hsu, and Wee Sun Lee. POMDP-lite for robust robot planning under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

Anthony Corso, Yizheng Wang, Markus Zechner, Jef Caers, and Mykel J. Kochenderfer. A POMDP model for safe geological carbon sequestration. In *NeurIPS Workshop on Tackling Climate Change*, 2022.

Dan Crisan and Arnaud Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746, 2002.

Katalin Csilléry, Michael G.B. Blum, Oscar E. Gaggiotti, and Olivier François. Approximate Bayesian computation (ABC) in practice. *Trends in Ecology & Evolution*, 25(7): 410–418, 2010.

Rhys Samuel Davies, Marianne Julia Davies, David Groves, Keith Davids, Eric Brymer, Allan Trench, John Paul Sykes, and Michael Dentith. Learning and expertise in mineral exploration decision-making: An ecological dynamics perspective. *International Journal of Environmental Research and Public Health*, 18(18):9752, 2021.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of Wasserstein GANs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Minguk Kang, Woohyeon Shim, Minsu Cho, and Jaesik Park. Rebooting ACGAN: Auxiliary classifier GANs with stable training. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Mykel J. Kochenderfer and Tim A. Wheeler. *Algorithms for optimization*. MIT Press, 2019.

Mykel J. Kochenderfer, Tim A. Wheeler, and Kyle H. Wray. *Algorithms for Decision Making*. MIT Press, 2022.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Eric Laloy, Romain Hérault, John Lee, Diederik Jacques, and Niklas Linde. Inversion using a new low-dimensional representation of complex binary geological media based on a deep neural network. *Advances in Water Resources*, 110:387–405, 2017.

Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Yimin Liu, Wenyue Sun, and Louis J. Durlofsky. A deep-learning-based geological parameterization for history matching complex models. *Mathematical Geosciences*, 51:725–766, 2019.

John Mern and Jef Caers. The intelligent prospector v1.0: Geoscientific model development and prediction by sequential data acquisition planning with application to mineral exploration. *Geoscientific Model Development*, 16(1):289–313, 2023.

Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv e-prints*, art. arXiv:1411.1784, 2014.

Lukas Mosser, Olivier Dubrule, and Martin J. Blunt. Reconstruction of three-dimensional porous media using generative adversarial neural networks. *Physical Review E*, 96:043309, 2017.

Lukas Mosser, Olivier Dubrule, and Martin J. Blunt. Stochastic reconstruction of an oolitic limestone by generative adversarial networks. *Transport in Porous Media*, 125(1):81–103, 2018.

Joelle Pineau and Geoffrey J. Gordon. POMDP planning for robust robot control. In *Robotics Research: Results of the 12th International Symposium ISRR*, 2007.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to large diverse datasets. In *ACM SIGGRAPH*, 2022.

Divya Saxena and Jiannong Cao. Generative adversarial networks (gans) challenges, solutions, and future directions. *ACM Computing Surveys (CSUR)*, 54(3):1–42, 2021.

David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2010.

Chris Snyder, Thomas Bengtsson, Peter Bickel, and Jeff Anderson. Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640, 2008.

Suihong Song, Tapan Mukerji, and Jiagen Hou. Geological facies modeling based on progressive growing of generative adversarial networks (gans). *Computational Geosciences*, 25:1251–1273, 2021.

Benjamin K. Sovacool, Saleem H. Ali, Morgan Bazilian, Ben Radley, Benoit Nemery, Julia Okatz, and Dustin Mulvaney. Sustainable minerals and metals for a low-carbon future. *Science*, 367(6473):30–33, 2020.

Zachary Sunberg and Mykel J. Kochenderfer. Online algorithms for POMDPs with continuous state, action, and observation spaces. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 2018.

Peter Jan Van Leeuwen, Hans R. Künsch, Lars Nerger, Roland Potthast, and Sebastian Reich. Particle filters for high-dimensional geoscience applications: A review. *Quarterly Journal of the Royal Meteorological Society*, 145(723):2335–2365, 2019.

Yizheng Wang, Markus Zechner, John M. Mern, Mykel J. Kochenderfer, and Jef K. Caers. A sequential decision-making framework with uncertainty quantification for groundwater management. *Advances in Water Resources*, 166:104266, 2022.

Kyle Hollins Wray, Bernard Lange, Arec Jamgochian, Stefan J. Witwicki, Atsuhide Kobashi, Sachin Hagaribommanahalli, and David Ilstrup. Pomdps for safe visibility reasoning in autonomous vehicles. In *IEEE Conference on Intelligence and Safety for Robotics (ISR)*, 2021.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.