Inducing Permutation Invariant Priors in Bayesian Optimization for Carbon Capture and Storage Applications

Sofianos Panagiotis Fotias[a,*], Vassilis Gaganis[a]

[a]*School of Mining and Metallurgical Engineering, National Technical University of Athens, Athens, 15773, Greece*

Abstract

Bayesian Optimization is an iterative method, tailored to optimizing expensive black box objective functions. Surrogate models like Gaussian Processes, which are the gold standard in Bayesian Optimization, can be inefficient for inputs with permutation symmetries, as the most common kernels employed are better suited for vector inputs rather than unordered sets of items. Motivated by this issue, we turn to permutation invariant Bayesian Optimization for well placement in Carbon Capture and Storage projects. The high fidelity black box simulator is instructed to operate wells under group control, giving rise to permutation symmetries within injector and producer groups that cannot be exploited with standard GP kernels. In this work, our main contribution is a novel Gaussian Process kernel (GP-Perm) that encodes permutation invariance by comparing sets through a stable divergence between their induced empirical representations, and can be combined with standard kernels for additional vector-valued inputs. As a learned invariant baseline, we also consider a Deep Kernel Learning model (DKL-DS) using the Deep Sets architecture to learn a permutation-invariant embedding. We evaluate the proposed methodology across 8 use cases, comprising seven synthetic benchmarks and one realistic CCS case study (Johansen formation).

Keywords: Bayesian optimization, Permutation invariance, Gaussian processes, Carbon capture and storage, Deep kernel learning

1. Introduction

Optimizing the design and operation of complex physical systems is a critical challenge in science and engineering, from drug discovery to aerospace design. A salient example is the strategic deployment of Carbon Capture and Storage (CCS), a key technology for mitigating climate change [1]. Optimizing a CCS project, for instance, involves determining the locations of injection and production wells and their operational settings to maximize the total volume of securely stored CO_2, while simultaneously minimizing operational costs and long-term leakage risks. Finding such an optimum requires navigating a challenging design space where each evaluation entails a computationally expensive, high-fidelity numerical simulation [2]. Such problems

*Corresponding author

Email addresses: sfotias@metal.ntua.gr (Sofianos Panagiotis Fotias), vgaganis@metal.ntua.gr (Vassilis Gaganis)

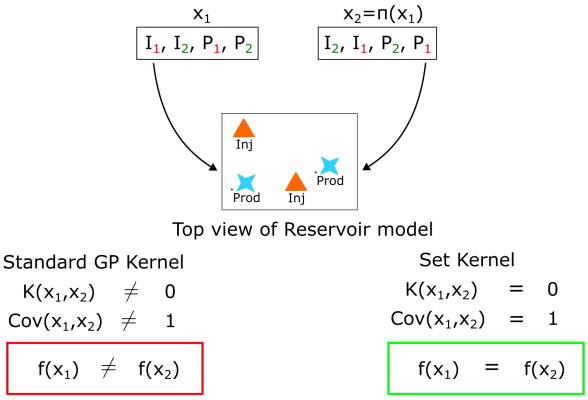


Figure 1: Permutation invariance in physical configuration: a standard GP sees x_1 and x_2 as different inputs, but since $x_w = \pi(x_q)$ for some permutation π, we have $f(x_1) = f(x_2)$. This can be inefficient during BO.

are prime candidates for Bayesian Optimization (BO), a sample-efficient methodology for global optimization of black-box functions that has proven effective in a wide range of scientific applications [3].

While powerful, standard BO surrogates like Gaussian Processes (GPs) [4] often assume a vector input space, which can be a critical limitation when the objective is invariant under a known group action and the surrogate does not encode this symmetry [5]. In many real-world problems, the input is not an unstructured vector but possesses inherent symmetries. For instance, in our CCS well-placement problem, the simulator output, and by extension the objective function, is invariant to the order in which injector well coordinates are listed within the injector group, and likewise invariant to permutations within the producer group. Equivalently, the objective is invariant under the action of the permutation group $S_m \times S_n$ acting separately on the m injectors and n producers. We claim that the Gaussian Process surrogate should exploit this information directly. A non-invariant GP treats permuted representations of the same physical design as distinct inputs, which can distort posterior geometry and lead to unnecessary exploration and reduced sample efficiency (Figure 1).

Several approaches can encode permutation symmetries in BO surrogates. One direction is to engineer permutation-invariant set kernels, for example by aggregating pairwise element similarities (double-sum kernels) or by radializing the induced RKHS distance between sets, yielding analytic invariant GP kernels [6]. A second direction is to learn an invariant representation with a set encoder such as Deep Sets [7] and then place a GP on the learned embedding via Deep Kernel Learning (DKL) [8]. The former offers an explicit inductive bias with no representation learning inside the BO loop, while the latter offers flexibility but introduces repeated nonconvex training from small, non-i.i.d. BO datasets.

We introduce **GP-Perm**, a permutation-invariant GP surrogate tailored to CCS well design under group control. GP-Perm enforces within-group exchangeability directly at the kernel level by comparing injector and producer configurations through a stable set discrepancy, while remaining compatible with standard vector-valued kernels for auxiliary continuous controls. Unlike pooled-similarity set kernels (e.g., double-sum / RKHS-distance constructions), GP-Perm is

built around an explicit geometry-sensitive discrepancy between point sets. For completeness, we benchmark GP-Perm against (i) a learned-invariant baseline formed by combining DKL with a Deep Sets encoder (DKL-DS) [8, 7], (ii) engineered invariant set-kernel baselines (DS/DE) [6, 9], and (iii) standard non-invariant GP/DKL surrogates. We evaluate these surrogates under a unified BO protocol across synthetic benchmarks (for controlled symmetry stress-tests) and simulation-based CCS tasks for realistic assessment.

The remainder of the paper is organized as follows: Section 3.1 formalizes the CCS design problem and its symmetry structure; Section 3 presents GP-Perm and DKL-DS; Sections 4–5 report benchmark and CCS results; and Section 6 discusses implications and limitations.

2. Background

Bayesian optimization is well established for continuous vector spaces, but adapting its surrogate models to inputs with non-Euclidean or structured geometry is still an active area of research. Prominent examples include BO over strings [10] and combinatorial/graph-structured inputs [11]. Our focus is on inputs that include unordered sets, where the objective is invariant to permutations of the set elements. This setting arises naturally in CCS well design under group control, where injector and producer well configurations form two unordered groups, accompanied by auxiliary continuous parameters. Equivalently, the objective is invariant under within-group permutations, i.e., under the action of $S_m \times S_n$ acting separately on injectors and producers. Accordingly, we develop an explicit kernel-based invariant GP surrogate (GP-Perm) and, for completeness, benchmark against a learned-invariant alternative implemented via Deep Sets within deep kernel learning (DKL-DS). These alternatives make different trade-offs in how faithfully they preserve geometry, how stable they are under the small, non-i.i.d. datasets encountered during BO, and how naturally they accommodate cross-set injector–producer interactions.

Engineered GP Kernels. One approach to encoding set symmetry is to engineer permutation-invariant kernels. In CCS well placement, Fotias et al. [12] solve the assignment problem with the Hungarian method to define a kernel distance. While effective, the resulting hard assignment is non-differentiable with respect to the inputs, which complicates gradient-based learning of kernel hyperparameters. More broadly, kernels based on hard matchings or raw distances can be brittle as small geometric perturbations may induce abrupt changes in correspondences or distances, and the induced notion of similarity can be difficult to tune beyond global length-scale parameters. Related transport-based constructions have been explored for set inputs in other domains, e.g., an Earth Mover's Distance (Wasserstein-1) kernel for sensor-set selection [13] and an optimal-transport distance on neural architectures in NASBOT [14]. This motivates using smooth transport-based discrepancies that retain geometric sensitivity while remaining amenable to stable hyperparameter learning inside BO. In contrast to using a raw set–set distance, the present work builds GP-Perm from a stable divergence between set representations that can be computed efficiently (e.g., via entropic transport solvers [15]), and composed with standard vector-valued kernels. This enables permutation-invariant BO for inputs containing multiple set components (injectors and producers) alongside auxiliary continuous variables.

Learned Permutation-Invariant Encoders within DKL. A parallel line of work enforces invariance through neural architectures. Deep Sets [7] shows that any continuous permutation-invariant function on a set admits the form $\rho\left(\sum_{x\in S} \phi(x)\right)$, and the Set Transformer [16] adds attention for richer inter-element interactions. For a recent overview of permutation-invariant neural networks

and their variants (including how design choices such as the aggregation operator affect behavior), see [17]. Our DKL-DS model uses a Deep Sets encoder as a feature extractor within the Deep Kernel Learning (DKL) framework [8], which combines the representation power of deep neural networks with the probabilistic uncertainty quantification of GPs. In this paper, DKL-DS is included strictly as a learned invariant baseline against which we assess the benefits of encoding permutation invariance directly in the GP kernel. This baseline is especially relevant in expensive CCS BO, where the encoder is retrained repeatedly from small datasets and representation drift can alter posterior geometry and acquisition behavior across iterations.

Set and Distribution Kernels. Several methods define kernels for non-vector inputs such as distributions or point clouds. A classical route is Maximum Mean Discrepancy (MMD) [18], which embeds distributions into an RKHS. For set-valued inputs, RKHS-based set kernels provide permutation-invariant GPs: the Double-Sum kernel averages pairwise base-kernel similarities across elements and the Deep-Embedding (DE) kernel applies a radial kernel to the RKHS mean-embedding distance. DE is often strictly positive-definite and numerically stable [9]. Computationally efficient approximations make such set kernels practical for BO via unbiased subsampling and set-aware acquisition optimization [6]. However, these RKHS-based constructions often induce similarity through averaged pairwise comparisons or mean embeddings, which can smooth away fine geometric distinctions between layouts. In CCS design, where injector-producer placement depends on detailed spatial geometry and cross-set structure, this motivates complementary kernels that compare configurations through more geometry-sensitive discrepancies. Recent work also connects GP modeling on measures to entropic transport constructions [19]. In our experiments, we compare GP-Perm to both non-invariant surrogates and to these alternative set-kernel baselines.

3. Methodology

3.1. Problem setup and Notation

We consider Bayesian Optimization of an expensive black-box objective function

$$\max_{\boldsymbol{x}\in\mathcal{X}} \; f(\boldsymbol{x}), \tag{1}$$

where each evaluation of f requires a high-fidelity reservoir simulation. The decision variable $\boldsymbol{x}$ has a structured form that reflects the physics and control of CCS well design. In particular, $\boldsymbol{x}$ contains (i) auxiliary continuous inputs that describe operational or contextual parameters and (ii) two unordered sets that describe the injector and producer well configurations.

We represent each design as

$$\boldsymbol{x} = (\boldsymbol{v}^{\boldsymbol{x}},\ \mathcal{I}^{\boldsymbol{x}},\ \mathcal{P}^{\boldsymbol{x}}), \tag{2}$$

where $\boldsymbol{v}^{\boldsymbol{x}} \in \mathbb{R}^{d_v}$ denotes a vector of auxiliary continuous parameters (e.g., group control targets or other scalar settings). In our experiments, $\boldsymbol{v}^{\boldsymbol{x}}$ consists of the target rates of the injector and producer groups, so $d_v = 2$. Furthermore, $\mathcal{I}^{\boldsymbol{x}} = \{\boldsymbol{i}_1, \ldots, \boldsymbol{i}_{n_{\text{inj}}}\}$ is the set of injector well descriptors, and $\mathcal{P}^{\boldsymbol{x}} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n_{\text{prod}}}\}$ is the set of producer well descriptors. In this work, each well descriptor is a 2D spatial coordinate, i.e., $\boldsymbol{i}_j, \boldsymbol{p}_j \in \mathbb{R}^2$, but the formulation naturally extends to higher-dimensional well features (e.g., depth or completion parameters).

Under group control, the simulator output and the resulting objective depend on the injector and producer configurations only through the *sets* $\mathcal{I}^{\boldsymbol{x}}$ and $\mathcal{P}^{\boldsymbol{x}}$, not on their internal ordering. Formally, for any permutations π of $\{1, \ldots, n_{\text{inj}}\}$ and σ of $\{1, \ldots, n_{\text{prod}}\}$,

$$f\Big(\boldsymbol{v}^{\boldsymbol{x}}, \{\boldsymbol{i}_1, \ldots, \boldsymbol{i}_{n_{\text{inj}}}\}, \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n_{\text{prod}}}\}\Big) = f\Big(\boldsymbol{v}^{\boldsymbol{x}}, \{\boldsymbol{i}_{\pi(1)}, \ldots, \boldsymbol{i}_{\pi(n_{\text{inj}})}\}, \{\boldsymbol{p}_{\sigma(1)}, \ldots, \boldsymbol{p}_{\sigma(n_{\text{prod}})}\}\Big). \tag{3}$$

A surrogate model intended for BO should respect these symmetries to avoid redundant learning across equivalent permutations of the same physical design. We seek a probabilistic surrogate $p(f \mid \mathcal{D})$ that can be trained on observations $\mathcal{D} = \{(\boldsymbol{x}_t, y_t)\}_{t=1}^{N}$ with $y_t = f(\boldsymbol{x}_t) + \epsilon_t$, and that supports BO via an acquisition function. In the following subsections we construct (i) a permutation-invariant Gaussian Process surrogate by defining kernels that operate on the structured input $(\boldsymbol{v}, \mathcal{I}, \mathcal{P})$, and (ii) a learned invariant baseline using Deep Kernel Learning with Deep Sets.

3.2. GP-Perm: a divergence-based permutation-invariant Matérn kernel

To build a GP surrogate that is invariant to the ordering of wells, we define a kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ whose notion of similarity is itself permutation-invariant with respect to the injector and producer sets. Standard metrics such as Euclidean distance on the full input vector are sensitive to the ordering of set elements and are therefore unsuitable. Our approach is to construct a composite distance $D(\boldsymbol{x}, \boldsymbol{x}')$ that respects the structured input $\boldsymbol{x} = (\boldsymbol{v}^{\boldsymbol{x}}, \mathcal{I}^{\boldsymbol{x}}, \mathcal{P}^{\boldsymbol{x}})$ introduced in Section 3.1.

We define a squared distance as an ANOVA-style sum of contributions from the auxiliary vector component, the injector and producer set geometries, and an injector-producer interaction term (Figure 2):

$$D^2(\boldsymbol{x}, \boldsymbol{x}') = \sum_{j=1}^{d_v} \frac{(v_j - v_j')^2}{\ell_{v,j}^2} + \frac{S_\varepsilon(\mathcal{I}, \mathcal{I}')}{\ell_I^2} + \frac{S_\varepsilon(\mathcal{P}, \mathcal{P}')}{\ell_P^2} + \frac{S_\varepsilon(\mathcal{R}, \mathcal{R}')}{\ell_{IP}^2}, \tag{4}$$

where $\boldsymbol{v} \in \mathbb{R}^{d_v}$ denotes the vector-valued design variables, and $\mathcal{I} = \{\boldsymbol{i}_a\}_{a=1}^{n_{\text{inj}}} \subset \mathbb{R}^2$ and $\mathcal{P} = \{\boldsymbol{p}_b\}_{b=1}^{n_{\text{prod}}} \subset \mathbb{R}^2$ denote the injector and producer sets, respectively. The additional interaction set is defined as

$$\mathcal{R} = \{\boldsymbol{r}_{ab}\}_{a=1,b=1}^{n_{\text{inj}},n_{\text{prod}}} = \{\boldsymbol{p}_b - \boldsymbol{i}_a \ : \ \boldsymbol{i}_a \in \mathcal{I},\ \boldsymbol{p}_b \in \mathcal{P}\} \subset \mathbb{R}^2, \tag{5}$$

This interaction set encodes cross-group spacing and relative positioning, which are central to CCS performance. Injectors and producers must be arranged to balance pressure management, plume front expansion and operational feasibility. The marginal geometries of $\boldsymbol{I}$ and $\boldsymbol{P}$ alone do not uniquely determine injector-producer spacing (e.g., two designs can share similar injector and producer layouts yet differ substantially in cross-group alignment), motivating the explicit inclusion of $S_\varepsilon(\mathcal{R}, \mathcal{R}')$ in equation 4.

In our experiments, $\boldsymbol{v}^{\boldsymbol{x}}$ consists of group injector and group producer control targets (so $d_v = 2$), but the formulation directly supports higher-dimensional auxiliary inputs.

The set terms are computed using the *Sinkhorn divergence* S_ε between the injector sets and between the producer sets. Concretely, for two injector sets $\mathcal{I} = \{\boldsymbol{i}_a\}_{a=1}^{n_{\text{inj}}}$ and $\mathcal{I}' = \{\boldsymbol{i}_a'\}_{a=1}^{n_{\text{inj}}}$ (and analogously for $\mathcal{P}, \mathcal{P}'$ and $\mathcal{R}, \mathcal{R}'$), we form the pairwise cost matrix

$$C_{ab} = \|\boldsymbol{s}_a - \boldsymbol{t}_b\|_2, \tag{6}$$

and compute the entropically regularized OT cost $W_\varepsilon(S, T)$ between the associated empirical measures with uniform weights, where $S = \{\boldsymbol{s}_a\}_{a=1}^{m}$ and $T = \{\boldsymbol{t}_b\}_{b=1}^{m}$ denote the two sets being

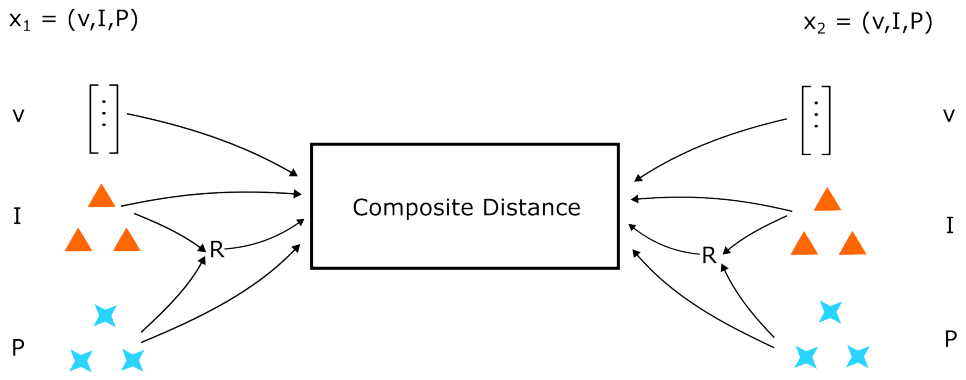


Figure 2: GP-Perm kernel construction. The left and right panels show two inputs, each decomposed into an auxiliary vector component (e.g., group injection and production targets) and two unordered sets of well locations: injectors (orange) and producers (cyan). GP-Perm measures similarity by combining a conventional ARD distance on the auxiliary inputs with permutation-invariant Sinkhorn divergences between injector sets, producer sets, and the interaction sets $\boldsymbol{R}$ and $\boldsymbol{R}'$ defined in equation 5. These components form a single composite distance used by the GP; no ordering information within a set is used.

compared (e.g., $S = \mathcal{I}$ and $T = \mathcal{I}'$). We then debias this quantity via

$$S_\varepsilon(S,T) \;=\; W_\varepsilon(S,T) - \frac{1}{2}W_\varepsilon(S,S) - \frac{1}{2}W_\varepsilon(T,T), \tag{7}$$

which enforces $S_\varepsilon(S,S) = 0$ and empirically improves numerical behavior compared to using W_ε directly. In practice we clamp S_ε to be non-negative to guard against small numerical violations. Since our application involves small sets of well locations, the cost of computing these set divergences is negligible compared to a single reservoir simulation. The entropic regularization parameter ε controls the geometry of the discrepancy, interpolating between OT-like matching at small ε and an MMD-like, pairwise-kernel regime at large ε, thereby linking GP-Perm to the behavior of classical double-sum set kernels in the high-regularization limit [20].

Finally, we construct GP-Perm by placing a Matérn kernel on the composite distance:

$$k_{\text{GP-Perm}}(\boldsymbol{x},\boldsymbol{x}') = \sigma^2\, \mathcal{M}_\nu(D(\boldsymbol{x},\boldsymbol{x}'))\,, \tag{8}$$

where ν controls smoothness and σ^2 is an output scale. Let $\pi \in S_{n_{\text{inj}}}$ and $\tau \in S_{n_{\text{prod}}}$, and define $\pi(\mathcal{I}) = \{\boldsymbol{i}_{\pi(a)}\}_{a=1}^{n_{\text{inj}}}$ and $\tau(\mathcal{P}) = \{\boldsymbol{p}_{\tau(b)}\}_{b=1}^{n_{\text{prod}}}$. Since S_ε depends only on the induced empirical measures, we have $S_\varepsilon(\mathcal{I},\mathcal{I}') = S_\varepsilon(\pi(\mathcal{I}),\mathcal{I}')$ and $S_\varepsilon(\mathcal{P},\mathcal{P}') = S_\varepsilon(\tau(\mathcal{P}),\mathcal{P}')$ (and likewise for $\mathcal{R}$).

Hence $D(\boldsymbol{x},\boldsymbol{x}')$ and therefore $k_{\text{GP-Perm}}(\boldsymbol{x},\boldsymbol{x}')$ are invariant under $S_{n_{\text{inj}}} \times S_{n_{\text{prod}}}$.

Optimizing hyperparameters of composite kernels that include set divergences can lead to ill-conditioned covariance matrices during marginal likelihood evaluation. We therefore adopt a staged jitter strategy: if Cholesky factorization fails at a nominal jitter level, we retry optimization under progressively larger diagonal jitter values. In our experiments this procedure was sufficient to obtain stable fits across all runs.

Even after successful fitting, posterior covariance computations can occasionally be sensitive to numerical roundoff when the kernel matrix is nearly singular. As a minimal safeguard, we

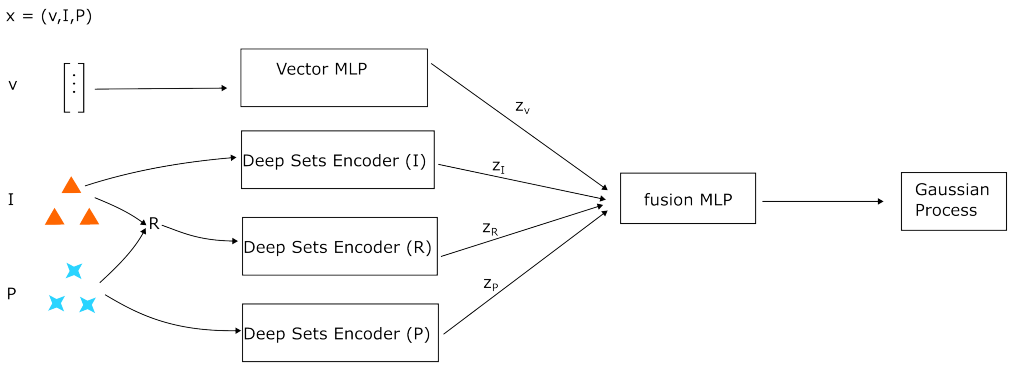


Figure 3: DKL-DS architecture. An MLP embeds the auxiliary vector inputs, while Deep Sets encoders embed the injector, producer, and interaction sets using shared element-wise maps and invariant pooling; the fused latent representation is modeled by a GP in the latent space.

clamp small negative eigenvalues that may arise from numerical error, ensuring a valid positive semidefinite covariance and hence a well-defined predictive distribution for acquisition optimization.

3.3. DKL-DS: a learned permutation-invariant surrogate baseline

As a learned-invariant baseline to complement explicit kernel design, we consider a permutation-invariant surrogate based on Deep Kernel Learning (DKL) [8]. In DKL, a neural feature extractor f_θ maps inputs to a latent space, and a GP is placed on the latent representation. Concretely, for $\boldsymbol{z} = f_\theta(\boldsymbol{x})$, the GP kernel operates as

$$k(\boldsymbol{x}, \boldsymbol{x}') \;=\; k_{\text{lat}}(f_\theta(\boldsymbol{x}), f_\theta(\boldsymbol{x}')), \tag{9}$$

where k_{lat} is a standard kernel (e.g., Matérn or RBF) defined on the latent space. This yields a GP model in a learned latent space, with uncertainty inherited from the GP posterior. Unlike GP-Perm, which enforces invariance directly through the kernel geometry, DKL-DS learns an invariant representation from data and places a standard GP on the resulting latent space.

To ensure invariance to permutations within injector and producer groups, we instantiate f_θ using Deep Sets [7]. Given the structured input $\boldsymbol{x} = (\boldsymbol{v}^{\boldsymbol{x}}, \mathcal{I}^{\boldsymbol{x}}, \mathcal{P}^{\boldsymbol{x}})$ from Section 3.1, our encoder processes the components separately and then fuses them (Figure 3). In addition to $\mathcal{I}^{\boldsymbol{x}}$ and $\mathcal{P}^{\boldsymbol{x}}$, we construct the injector–producer interaction set $\mathcal{R}^{\boldsymbol{x}}$ as in equation 5, providing a permutation-invariant representation of cross-group spatial relationships.

1. The auxiliary vector input $\boldsymbol{v}^{\boldsymbol{x}}$ is mapped through an MLP to a fixed-size embedding.
2. The injector set $\mathcal{I}^{\boldsymbol{x}}$, producer set $\mathcal{P}^{\boldsymbol{x}}$, and interaction set $\mathcal{R}^{\boldsymbol{x}}$ are each processed by a permutation-invariant Deep Sets module. Each module first applies a shared element-wise map $\phi(\cdot)$ and then aggregates across the set using order-invariant pooling. In our implementation, we enrich the basic Deep Sets pooling by concatenating several invariant aggregations (mean, max, standard deviation, and power-means) before applying a final map $\rho(\cdot)$.

3. The vector embedding and the three set embeddings are concatenated and passed through a fusion MLP to produce the final latent representation $\boldsymbol{z} = f_\theta(\boldsymbol{x})$.

This construction makes f_θ permutation-invariant within each set by design, while the interaction set $\mathcal{R}^x$ provides a dedicated channel for learning cross-group injector–producer geometric structure. The hybrid model is trained end-to-end by maximizing the GP marginal log-likelihood, allowing gradients to backpropagate through both the GP hyperparameters (e.g., latent-space lengthscales, optionally with ARD) and the network parameters θ. We include DKL-DS to represent a strong AI engineered learned-invariance baseline, but note that it introduces repeated nonconvex training inside the BO loop, which can affect stability in the small-data regime. For improved optimization in the BO loop, we optionally warm-start f_θ using a short MSE pretraining phase with a linear prediction head, and then switch to joint DKL training. We also employ a staged jitter schedule to ensure robust Cholesky-based marginal likelihood evaluation.

3.4. Bayesian Optimization loop

We optimize the expensive black-box objective using a standard Bayesian Optimization loop. Starting from an initial dataset $\mathcal{D}_0 = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_0}$, BO iterates between (i) fitting a probabilistic surrogate model to the accumulated data and (ii) selecting new candidate designs by maximizing an acquisition function.

We generate an initial design set by sampling the auxiliary continuous inputs uniformly within their bounds and sampling well locations from the feasible grid without replacement. When available, we bias the sampling towards interior feasible cells (Appendix B) using a precomputed interior score, to reduce the likelihood of selecting wells close to invalid regions. Each initial design is evaluated with the reservoir simulator to obtain y_i.

At BO iteration t, we rescale each continuous input dimension to $[0, 1]$ before fitting and acquisition optimization, which improves numerical conditioning for surrogate training and gradient-based acquisition maximization. We refit the surrogate on all available observations $\mathcal{D}_t$. If the simulator returns an invalid value (e.g., failed run) or a sentinel near-zero value indicating infeasibility, we assign it a constant penalty value chosen to be worse than the worst feasible observation observed so far. This ensures a well-defined training target while preserving the ordering among feasible outcomes.

We use the Monte Carlo acquisition function qLogEI [21] with a stochastic sampler. At each iteration, we select a batch of q candidates by approximately maximizing qLogEI in the normalized continuous space using multi-start gradient-based optimization. Acquisition maximization is performed in the continuous, normalized space using multi-start gradient-based optimization. To promote feasible well placements during acquisition maximization, we augment the acquisition objective with a differentiable barrier penalty derived from a signed distance field over the feasible grid, with a user-controlled safety margin and penalty strength.

The optimized candidates are post-processed to produce valid discrete well locations. Specifically, each candidate well coordinate is snapped to a feasible grid cell by selecting among nearby feasible cells, favoring interior locations while avoiding duplicate wells within the same design. The resulting discrete designs are then unnormalized back to the original bounds and evaluated with the simulator. The new observations are appended to the dataset, $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(\boldsymbol{x}_{t,j}, y_{t,j})\}_{j=1}^{q}$, and the process repeats for a fixed budget of BO iterations. The post-processing step enforces feasibility and uniqueness (no duplicate wells within a design) while remaining close to the optimized continuous candidate.

3.5. Baseline surrogate models

To contextualize the performance of GP-Perm and DKL-DS, we compare against four baselines that isolate the role of set-aware modeling:

- **Non-invariant GP.** A standard Gaussian Process surrogate with a Matérn kernel applied directly to the flattened input vector. This baseline does not encode permutation invariance and therefore treats different permutations of the same physical configuration as distinct inputs.
- **Non-invariant DKL.** A Deep Kernel Learning surrogate in which an MLP feature extractor maps the flattened input to a latent representation and a GP with a Matérn kernel is placed on the learned features. Unlike DKL-DS, the encoder is not permutation-invariant.
- **Double-Sum (DS) set-kernel GP.** A classical permutation-invariant set kernel that aggregates a base kernel across all cross-element pairs between two sets (equivalently, the average pairwise similarity). This provides a strong set-aware GP baseline and is commonly used in set-valued BO comparisons [9, 6]. For grouped inputs, we apply DS separately to injectors and producers and combine the resulting terms additively with the auxiliary-vector kernel.
- **Deep-Embedding (DE) set-kernel GP.** A permutation-invariant set kernel based on RKHS mean embeddings: each set is mapped to its empirical kernel mean embedding under a chosen base kernel, and set similarity is computed by applying a radial kernel to the induced RKHS distance between embeddings. In this work, DE uses a fixed kernel mean embedding (not a learned neural embedding), serving as a stable and widely used set-kernel baseline [9, 6]. As above, DE is applied per set component and combined with the auxiliary-vector kernel using the same composite structure across experiments.

For inputs with multiple components (auxiliary vectors plus injector and producer sets), DS and DE are applied independently to the injector and producer sets and combined with a conventional kernel on the auxiliary inputs using the same additive composition used in equation 4. All baselines are trained by maximizing the marginal log-likelihood and evaluated under the BO protocol in Section 3.4.

4. Benchmark study for model selection

Before running expensive CCS case studies, we perform a benchmark-driven model selection study on a suite of synthetic permutation-invariant objectives. This stage serves two purposes: (i) to verify under controlled conditions that the proposed surrogates exploit permutation symmetry, and (ii) to select stable model variants and hyperparameters that are carried forward unchanged to the CCS case studies in Section 5. Full definitions of the benchmark objectives are provided in Appendix Appendix A.

4.1. Synthetic benchmark suite

The synthetic benchmarks are constructed to be permutation-invariant by design and to stress different aspects of set reasoning (geometric structure, coverage, distributional matching, and graph-based objectives). Each benchmark takes as input a single unordered set of n_{points} two-dimensional locations, represented in code as a flattened vector in $\mathbb{R}^{2n_{\text{points}}}$ but interpreted by the

Table 1: Synthetic benchmarks and BO protocol used for model selection. All benchmarks operate on normalized coordinates in $[0, 1]^2$. Here n_{points} is the set cardinality and Dim = $2\,n_{\text{points}}$ is the flattened decision-space dimension.

Benchmark	$\mathbf{n}_{\text{points}}$	Dim	$\mathbf{n}_{\text{trials}}$	**T**
Particle Physics	10	20	10	20
Max Area Coverage	16	32	10	20
Distribution Matching (MMD)	20	40	10	10
Max Spanning Tree	15	30	10	10
Facility Location (Soft Coverage)	12	24	8	10
Soft k-Medoids	12	24	10	10

models as a set. Importantly, unlike the CCS setting, these benchmarks do not include auxiliary continuous inputs and do not include separate injector and producer sets.

Accordingly, during the benchmark stage we use reduced versions of our surrogate families that operate on a single set input. This isolates the effect of permutation-invariant modeling from CCS-specific components (auxiliary controls and multiple well groups) and allows us to compare kernel/encoder variants in a controlled regime. The selected configurations are then transferred to the CCS input structure (auxiliary vectors + two sets) without additional tuning.

4.2. BO protocol and evaluation metrics

For each synthetic benchmark objective, we run BO for a fixed evaluation budget with multiple independent trials. All benchmarks follow the BO loop described in Section 3.4, specialized to a single-set input: each trial begins with an initial design of size N_{init}, followed by T BO iterations with batch size q. At each iteration, the surrogate is refit on all collected data and the acquisition function is optimized to propose the next batch of candidate sets. Benchmark-specific budgets are used to keep overall difficulty comparable across tasks.

We evaluate performance using best-so-far trajectories and two scalar summaries computed per trial: (i) the area under the best-so-far curve (AUC) and (ii) the final best value. Model selection within each family (GP-Perm and DKL-DS) is performed via a sensitivity study over hyperparameters; the full selection procedure and any score normalization used for variant selection are reported in Appendix E. Table 1 summarizes the benchmark settings used throughout this section.

4.3. Benchmark comparison of selected models against baselines

Beyond model selection, the synthetic benchmark suite provides a controlled setting to compare permutation-invariant surrogates against strong non-invariant and set-kernel baselines under identical BO budgets and protocols (Section 4.2). We first perform a hyperparameter ablation for GP-Perm and DKL-DS and select a single configuration of each based on benchmark performance; the complete sensitivity analysis and all variant results are reported in Appendix E. The two selected models (denoted **GP-Perm** and **DKL-DS** below) are then compared against (i) non-invariant baselines (**GP**, **DKL**) and (ii) set-kernel baselines (**DS-GP**, **DE-GP**), all evaluated with the same BO setup.

Figure 4 shows the mean best-so-far objective across trials (shaded: ±1 std). Tables 2–3 summarize the two scalar metrics used throughout the study: the AUC of the best-so-far curve and the final best value, reported as mean ± std across trials (best per benchmark in bold). All objectives are maximized; for benchmarks whose range is negative, values closer to zero correspond

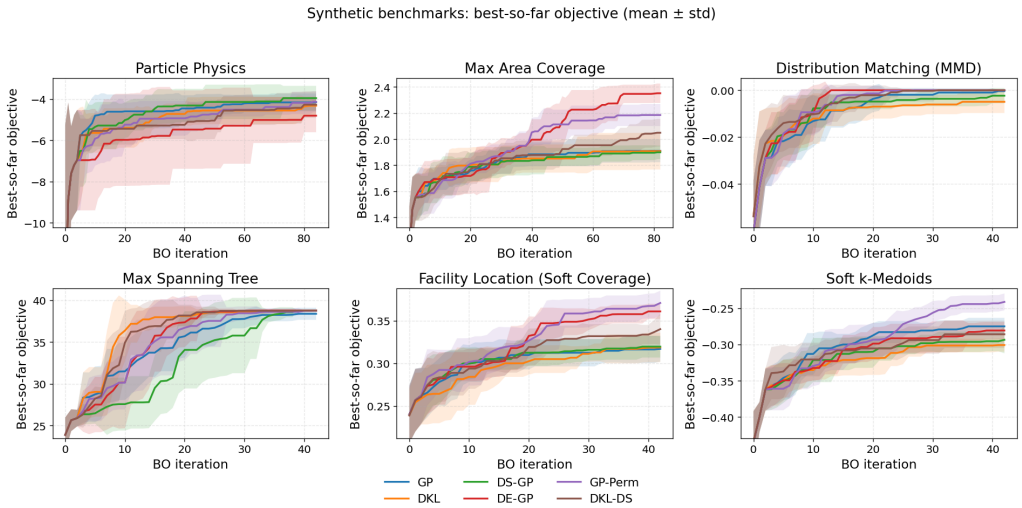


Figure 4: Synthetic benchmarks: best-so-far objective versus BO iteration (mean ± std across trials).

Table 2: Benchmark summary using AUC of the best-so-far curve (mean ± std across trials).

Benchmark	**GP**	**DKL**	**DS-GP**	**DE-GP**	**GP-Perm**	**DKL-DS**
Distribution Matching (MMD)	-0.371 ± 0.096	-0.425 ± 0.242	-0.369 ± 0.112	**-0.238 ± 0.104**	-0.272 ± 0.126	-0.243 ± 0.141
Facility Location (Soft Coverage)	12.720 ± 0.712	12.478 ± 0.703	12.806 ± 0.547	13.607 ± 0.522	**13.826 ± 0.635**	13.071 ± 0.512
Max Area Coverage	149.450 ± 5.887	149.373 ± 7.542	148.171 ± 4.986	**163.894 ± 5.954**	160.621 ± 5.660	151.845 ± 4.758
Max Spanning Tree	1452.410 ± 62.506	**1524.105 ± 39.065**	1368.808 ± 87.785	1474.793 ± 65.525	1465.257 ± 76.264	1507.203 ± 34.486
Particle Physics	**-393.629 ± 48.427**	-425.810 ± 52.625	-395.263 ± 64.242	-482.277 ± 148.410	-427.180 ± 48.192	-434.479 ± 54.414
Soft k-Medoids	-12.663 ± 0.469	-13.509 ± 0.425	-13.298 ± 0.446	-13.057 ± 0.493	**-12.365 ± 0.515**	-12.990 ± 0.591

Table 3: Benchmark summary using final best value at the end of BO (mean ± std across trials; higher is better).

Benchmark	**GP**	**DKL**	**DS-GP**	**DE-GP**	**GP-Perm**	**DKL-DS**
Distribution Matching (MMD)	-0.000 ± 0.001	-0.005 ± 0.005	-0.002 ± 0.003	**0.000 ± 0.000**	**0.000 ± 0.000**	**0.000 ± 0.000**
Facility Location (Soft Coverage)	0.317 ± 0.015	0.319 ± 0.017	0.320 ± 0.013	0.361 ± 0.009	**0.371 ± 0.014**	0.340 ± 0.018
Max Area Coverage	1.911 ± 0.088	1.908 ± 0.141	1.899 ± 0.051	**2.352 ± 0.074**	2.185 ± 0.084	2.049 ± 0.105
Max Spanning Tree	38.384 ± 0.722	38.734 ± 0.073	38.770 ± 0.000	**38.772 ± 0.009**	38.751 ± 0.042	38.769 ± 0.002
Particle Physics	-4.153 ± 0.417	-4.324 ± 0.451	**-3.957 ± 0.594**	-4.807 ± 0.802	-4.133 ± 0.510	-4.274 ± 0.640
Soft k-Medoids	-0.275 ± 0.012	-0.301 ± 0.009	-0.294 ± 0.019	-0.281 ± 0.012	**-0.241 ± 0.012**	-0.286 ± 0.019

to better performance. The selected GP-Perm and DKL-DS configurations are used *unchanged* in the CCS case studies in Section 5.

Overall, no single surrogate dominates every benchmark. DE-GP is strongest on Distribution Matching and Max Area Coverage, while DKL attains the best AUC on Max Spanning Tree; for Facility Location and Soft k-Medoids, the selected GP-Perm achieves the best performance under both AUC and final best. These results highlight that the benefit of permutation invariance is task dependent: enforcing the correct symmetry can yield clear gains, but the best-performing configuration within a model family can vary across objectives. In particular, the full sensitivity results in Appendix E show that some GP-Perm (and DKL-DS) variants outperform all baselines on specific benchmarks even when the single configuration selected for transfer to CCS is not the per-benchmark winner. We therefore interpret the synthetic suite both as (i) evidence that permutation-invariant surrogates can be competitive or superior under controlled symmetries,

and (ii) motivation for careful hyperparameter selection when targeting a specific task.

5. CCS case study

5.1. CCS-like two-set synthetic benchmark

Before turning to expensive reservoir simulation, we validate our permutation-invariant surrogate on a controlled two-set benchmark that captures key geometric features of CCS well design under grouped injectors and producers. The input is composed of two unordered sets, injector and producer locations and the objective is constructed so that each producer is encouraged to have (at least) one injector at a preferred spacing and along a preferred migration direction, while within-set repulsion discourages collapse of injector–injector and producer–producer locations.

Concretely, for n_{inj} injectors and n_{prod} producers in a bounded 2D domain, a candidate design is a flattened vector of coordinates $\boldsymbol{x} = [(\boldsymbol{i}_1, \ldots, \boldsymbol{i}_{n_{\text{inj}}}), (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n_{\text{prod}}})]$, with $\boldsymbol{i}_k, \boldsymbol{p}_\ell \in [-1, 1]^2$. For each producer $\boldsymbol{p}_\ell$, we define a pairwise mismatch cost to each injector $\boldsymbol{i}_k$ that penalizes deviation from a preferred distance $d^\star$ and misalignment with a preferred direction $\boldsymbol{u}$ (normalized):

$$c_{\ell k} = \left(\frac{\|\boldsymbol{p}_\ell - \boldsymbol{i}_k\| - d^\star}{\sigma_d} \right)^2 + \left(\frac{1 - \cos(\angle(\boldsymbol{p}_\ell - \boldsymbol{i}_k), \boldsymbol{u})}{\sigma_\theta} \right)^2, \tag{10}$$

and we aggregate over injectors via a soft minimum (temperature τ) to reflect that each producer mainly depends on its most effective injector:

$$C_{\text{cross}}(\boldsymbol{x}) = \frac{1}{n_{\text{prod}}} \sum_{\ell=1}^{n_{\text{prod}}} \Big(-\tau \log \sum_{k=1}^{n_{\text{inj}}} \exp(-c_{\ell k}/\tau) \Big). \tag{11}$$

Finally, we add within-set repulsion terms for injectors and producers (inverse squared distances with small ε) to discourage degenerate clustering:

$$C_{\text{rep}}(\boldsymbol{x}) = w_{\text{inj}} \sum_{k<k'} \frac{1}{\|\boldsymbol{i}_k - \boldsymbol{i}_{k'}\|^2 + \varepsilon} + w_{\text{prod}} \sum_{\ell<\ell'} \frac{1}{\|\boldsymbol{p}_\ell - \boldsymbol{p}_{\ell'}\|^2 + \varepsilon}. \tag{12}$$

The benchmark returns a scalar objective $f(\boldsymbol{x}) = -(C_{\text{cross}}(\boldsymbol{x}) + C_{\text{rep}}(\boldsymbol{x}))$ (higher is better), and is permutation-invariant within the injector set and within the producer set by construction. This mirrors the symmetry induced by grouped well control in CCS, where only the unordered configuration of injectors and producers matters.

Table 4: Surrogate configurations carried forward to CCS after benchmark selection.

Model	**Key settings**
GP (non-invariant)	kernel:matern52.
DKL (non-invariant)	latent_dim:64; mlp_hidden:64; kernel:matern52; ARD:true.
GP-DS (set-kernel baseline)	vec_kernel:matern52; set_kernel:DS (RBF base) applied to $\mathcal{I}^{\boldsymbol{x}}$, $\mathcal{P}^{\boldsymbol{x}}$, and $\mathbb{R}^{\boldsymbol{x}}$.
GP-DE (set-kernel baseline)	vec_kernel:matern52; set_kernel:DE (RBF base + RBF outer) applied to $\mathcal{I}^{\boldsymbol{x}}$, $\mathcal{P}^{\boldsymbol{x}}$, and $\mathbb{R}^{\boldsymbol{x}}$.
GP-Perm (selected)	kernel:matern52; ε: benchmark-selected (Table E.11).
DKL-DS (selected)	latent_dim:64; ϕ:64; ρ:128; pools: benchmark-selected (Table E.11); kernel:matern52; +ARD if selected; pretrain_epochs:100.

5.2. Reservoir simulator case study: Johansen formation

We next evaluate the same surrogate models on a realistic CCS optimization task derived from one of the only large-scale, publicly available geological models for prospective CO_2 storage: the Johansen formation [22]. We use the structured design variable from Section 3.1, i.e., $\boldsymbol{x} = (\boldsymbol{v}^{\boldsymbol{x}}, \mathcal{I}^{\boldsymbol{x}}, \mathcal{P}^{\boldsymbol{x}})$, where $\boldsymbol{v}^{\boldsymbol{x}}$ denotes group control targets and $\mathcal{I}^{\boldsymbol{x}}, \mathcal{P}^{\boldsymbol{x}}$ are unordered well-location sets.

A candidate design consists of (i) the (x, y) locations of each injector and producer well and (ii) auxiliary continuous operational controls given by group target rates (injection and production). The objective is to maximize a regularized Net Present Value (NPV) that incorporates penalties for undesirable operational outcomes such as CO_2 recycling and rate inconsistency. Each evaluation requires a computationally expensive run of the OPM-Flow reservoir simulator [23]. In addition to the active injection/production period, each simulation includes a post-operation monitoring period of 50 years:

$$f(\boldsymbol{x}) \;=\; \mathrm{NPV}(\mathrm{Sim}(\boldsymbol{x})) \;-\; P_L(\mathrm{Sim}(\boldsymbol{x})) \;-\; P_S(\mathrm{Sim}(\boldsymbol{x})). \tag{13}$$

5.3. Feasible well-placement regions and candidate generation

A key challenge in CCS well placement is defining a realistic search space. Valid well locations form a discrete, non-convex region constrained by the reservoir geometry and operational restrictions. For Johansen, we exclude the thin western part of the formation to mitigate potential leakage risks [24, 25]. Visualizations of the feasible region before and after masking are provided in Appendix Appendix B.

The resulting mask defines the feasible grid cells used during Bayesian optimization. Candidate generation and feasibility handling follow the procedure described in Subsection 3.4: acquisition optimization is performed in the normalized continuous design space, and proposed well coordinates are post-processed onto valid grid cells before simulator evaluation. This ensures that all evaluated designs satisfy the geological feasibility constraints while keeping the acquisition-optimization step compatible with gradient-based BO.

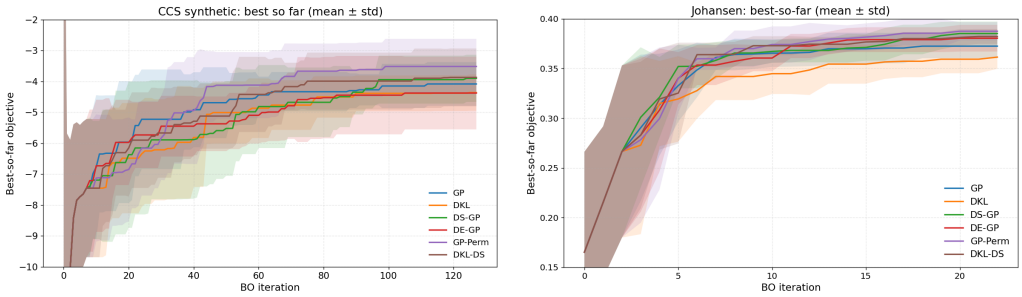
5.4. Evaluation protocol

Synthetic benchmark. We evaluate the models in Table 4 on the CCS-like two-set synthetic function with $n_{\mathrm{inj}} = 3$, $n_{\mathrm{prod}} = 5$, $n_{\mathrm{trials}} = 9$ independent trials, $N_{\mathrm{init}} = 8$ initial Sobol designs, $T = 30$ BO iterations, and batch size $q = 4$ per iteration. We report mean best-so-far objective values across trials, with variability bands given by mean ± one standard deviation.

Johansen case study. We then evaluate the same surrogate models on the Johansen formation under a tighter evaluation budget due to the cost of OPM-Flow simulations: $n_{\mathrm{trials}} = 7$ independent trials, $N_{\mathrm{init}} = 3$ initial Sobol designs, $T = 10$ BO iterations, and batch size $q = 2$ per iteration. In preliminary experiments we observed persistent simulator instability in the S-meaheia setup under repeated BO evaluations, preventing reliable trial-based comparisons under the same protocol; we therefore focus on Johansen for the realistic CCS study, while retaining feasibility construction and visualizations for both formations in Appendix Appendix B.

5.5. Results: CCS-like synthetic and Johansen

Figure 5 reports best-so-far trajectories for the CCS-like two-set synthetic benchmark (left) and the Johansen CCS case study (right). The main takeaway across both settings is that explicitly encoding within-set permutation invariance yields consistently better sample-efficiency than



(a) CCS-like two-set synthetic benchmark.

(b) Johansen CCS case study.

Figure 5: BO results on (left) CCS-like two-set synthetic benchmark and (right) Johansen CCS case study: mean best-so-far trajectories (shaded: ±1 std).

Table 5: AUC of best-so-far curves (mean ± std across trials) for the CCS-like synthetic benchmark and the Johansen case study.

Model	Synthetic AUC	Johansen AUC
GP	-633.905 ± 99.069	7.563 ± 0.467
DKL	-688.067 ± 113.562	7.256 ± 0.486
DS-GP	-674.499 ± 124.006	7.672 ± 0.336
DE-GP	-673.955 ± 115.208	7.626 ± 0.267
GP-Perm	**-603.991 ± 120.954**	**7.717 ± 0.479**
DKL-DS	-636.862 ± 89.690	7.675 ± 0.277

Table 6: Final best-so-far objective value (mean ± std across trials) for the CCS-like synthetic benchmark and the Johansen case study.

Model	Synthetic FINAL	Johansen FINAL
GP	-4.082 ± 0.704	0.372 ± 0.012
DKL	-4.362 ± 0.587	0.361 ± 0.012
DS-GP	-3.900 ± 0.762	0.385 ± 0.012
DE-GP	-4.376 ± 1.177	0.380 ± 0.013
GP-Perm	**-3.515 ± 0.893**	**0.387 ± 0.009**
DKL-DS	-3.869 ± 0.936	0.382 ± 0.007

non-invariant surrogates, and that our proposed GP-Perm provides the strongest overall performance among the tested approaches.

On the CCS-like synthetic benchmark, GP-Perm attains the best mean AUC (Table 5), improving from −633.905 (GP) to −603.991, and also achieves the best mean final objective (Table 6), improving from −4.082 (GP) to −3.515 and also performs better than set kernels (DS-GP/DE-GP). This setting isolates the core structure targeted by GP-Perm: the objective depends on relative geometry between two unordered sets while being insensitive to within-set permutations. In this regime, GP-Perm consistently converts fewer evaluations into larger best-so-far gains.

Crucially, this advantage carries over to the realistic simulator-based CCS objective on Johansen, despite a substantially tighter BO budget (T=10, q=2) than in the synthetic study (T=30, q=4). GP-Perm again achieves the best mean AUC (7.717) and best mean final objective (0.387), outperforming both non-invariant baselines and competitive set-based alternatives (Tables 5–6). Taken together, the results support that GP-Perm's permutation-invariant inductive bias improves sample-efficiency in a CCS-like setting and remains beneficial when transferred to a high-fidelity CCS simulator where evaluations are scarce.

6. Discussion

This work studies Bayesian optimization for CCS well design when the input combines unordered well-location sets with auxiliary continuous controls. Under group control, the objective is invariant to within-group permutations of wells, so non-invariant surrogates waste capacity modeling distinctions that are physically meaningless and can distort posterior geometry under tight evaluation budgets. Our primary contribution is **GP-Perm**: a permutation-invariant Gaussian process surrogate that enforces this symmetry directly through an explicit set-based kernel built from a stable divergence between well configurations. For completeness, we also evaluate a learned-invariant baseline, **DKL-DS**, which implements the same symmetry indirectly by training a Deep Sets encoder and placing a standard GP in the induced latent space. Finally, we include non-invariant GP/DKL and set-kernel baselines (DS/DE) to isolate the effect of explicit invariance versus pooled representations. Model configurations are selected via a benchmark-driven stage and then transferred unchanged to the CCS setting.

Beyond CCS, our results support a general lesson for sequential, low-data learning: when inputs contain nuisance symmetries, encoding the symmetry reduces effective complexity by collapsing equivalent designs. In BO, this can translate into a smoother posterior over the quotient space of designs and consequently, more reliable acquisition decisions under small evaluation budgets. In our synthetic suite, permutation-invariant surrogates are consistently superior to non-invariant GP/DKL baselines under identical BO budgets without any one of them comfortably winning on all benchmarks. This is expected because the benchmarks stress different aspects of set reasoning (coverage, geometric structure, distributional matching, and graph-like interactions), and each surrogate family imposes a different inductive bias. To further connect the abstract benchmarks to CCS structure, we also introduced a CCS-like two-set synthetic objective that mimics injector–producer geometric coupling under group control while preserving within-group exchangeability. This benchmark provides a controlled setting in which the symmetry targeted by GP-Perm is exact, while still reflecting the interaction-driven nature of CCS well layouts. Because BO performance depends not only on final objective values but also on posterior conditioning and acquisition stability, we report additional diagnostics that summarize how each surrogate shapes uncertainty and acquisition behavior across iterations on Johansen.

Table 7 provides a complementary view of surrogate posterior and acquisition behavior during the Johansen BO loop. The posterior-standard-deviation diagnostics indicate a potential calibration issue for the representation-learning surrogates in this small-data, sequential BO regime. Both DKL and DKL-DS report very small posterior standard deviations over the fixed probe set Z and at the selected candidates, while also producing large standardized residual diagnostics at the subsequently evaluated points. This combination is consistent with overconfident predictive uncertainty and/or representation drift during repeated retraining. The raw qLogEI quotient reported in the table points in the same qualitative direction, with DKL-based models showing

Table 7: Johansen posterior and acquisition diagnostics (mean ± std across trials; averaged over BO iterations 1–9). Here $\rho_{\text{qLogEI}} = \text{qLogEI}(\boldsymbol{x}_{\text{next}})/\max_{z\in Z}\text{qLogEI}(z)$ is the raw quotient used in the diagnostic script relative to a fixed Sobol probe set Z. Since qLogEI is log-scaled, ρ_{qLogEI} should be interpreted only as an implementation-specific relative diagnostic, not as a calibrated EI-scale acquisition ratio. $\mathbb{E}_Z[\sigma(\boldsymbol{x})]$ is the mean posterior standard deviation over Z, $\sigma(\boldsymbol{x}_{\text{next}})$ is the posterior standard deviation at the selected batch, and $|\bar{r}|$ is the absolute value of the batch-averaged standardized residual at the sampled points.

Model	ρ_{qLogEI}	$\mathbb{E}_Z[\sigma(\boldsymbol{x})]$	$\sigma(\boldsymbol{x}_{\text{next}})$	$\lvert\bar{r}\rvert$
GP	1 ± 0.0369	0.0776 ± 0.0138	0.0626 ± 0.0120	0.459 ± 0.140
DKL	3.22 ± 2.72	0.00291 ± 0.000447	0.00418 ± 0.00124	17.7 ± 4.09
DS-GP	1.03 ± 0.298	0.0193 ± 0.0197	0.0182 ± 0.0196	1.63 ± 0.432
DE-GP	1.51 ± 0.601	0.0142 ± 0.00985	0.0113 ± 0.00531	1.82 ± 0.721
GP-Perm	1.27 ± 0.343	0.0237 ± 0.00700	0.0194 ± 0.00721	1.21 ± 0.528
DKL-DS	6.10 ± 3.86	0.00467 ± 0.00119	0.00568 ± 0.000942	5.82 ± 4.40

larger and more variable relative acquisition scores, although this quantity should not be interpreted as a calibrated acquisition ratio because it is computed directly from log-acquisition values.

The table also highlights a distinction between the analytic invariant set-kernel baselines and GP-Perm. DS-GP and DE-GP generally produce lower posterior uncertainty than GP-Perm, as measured by both $\mathbb{E}_Z[\sigma(\boldsymbol{x})]$ and $\sigma(\boldsymbol{x}_{\text{next}})$. This behavior is plausible given their smoothing structure: DS-GP averages pairwise element similarities across sets, while DE-GP radializes an RKHS-induced set distance through an outer RBF kernel. These operations can increase similarity between distinct layouts and therefore contract posterior variance more rapidly in the few-sample regime. Consistent with this interpretation, DE-GP reports lower average posterior uncertainty than DS-GP in Table 7.

In contrast, GP-Perm compares layouts through a geometry-sensitive Sinkhorn divergence defined directly on the injector, producer, and interaction sets. In these Johansen runs, GP-Perm maintains a more moderate uncertainty profile than DS-GP and DE-GP, while also yielding a smaller standardized residual diagnostic than the other invariant GP baselines. We interpret this as evidence that GP-Perm provides a useful balance between geometric discrimination and uncertainty contraction under the tight evaluation budget. In these experiments, GP-Perm also remained numerically stable under the same standard safeguards used for the other GP models.

Overall, these diagnostics should be viewed as supporting evidence rather than as a standalone explanation of performance. Together with the AUC and final-best results, they suggest that encoding the relevant permutation symmetry improves BO behavior in the Johansen case study. In particular, GP-Perm combines strong optimization performance with moderate posterior uncertainty and less problematic standardized residual behavior than the DKL-based models, which is valuable when simulator evaluations are scarce.

A key design choice in CCS is how to represent injector–producer coupling. Treating injectors and producers as separate exchangeable sets captures within-group structure but may miss cross-group geometry that drives operational outcomes (e.g., spacing effects and recycling risk). Our framework supports an explicit interaction representation via an interaction set constructed from injector–producer relative positions, and Appendix C isolates the effect of including this component in both GP-Perm and DKL-DS. Conceptually, the interaction term is most valuable when the objective depends directly on injector–producer pairwise relationships rather than only

on within-group properties.

Realistic CCS well placement introduces challenges that are orthogonal to permutation invariance. First, feasible well locations lie on a discrete, non-convex region induced by geology and operational constraints. Our pipeline optimizes acquisition functions in a continuous normalized space and then projects candidates to feasible grid cells while enforcing uniqueness. This projection ensures validity but can bias the induced discrete proposal distribution, which may affect exploration. Second, simulator failures can dominate experimental throughput and, in practice, determine which comparisons are possible under a fixed compute budget. The difficulty of obtaining stable trial-based results for Smeaheia in our study reinforces the value of BO pipelines that explicitly model feasibility and failure risk. Finally, while divergence-based set discrepancies are numerically stable in practice, they do not automatically imply strict positive semidefiniteness when embedded inside a radial kernel for all settings. We found standard GP stabilizers (normalization, staged jitter for Cholesky factorization, and additional safeguards reported in the diagnostics appendices) essential for robust BO operation.

7. Conclusion

We studied Bayesian optimization for CCS well design under group control, where the input combines unordered injector and producer location sets with auxiliary continuous controls and the objective is exactly invariant to within-group permutations. Our main contribution, **GP-Perm**, enforces this symmetry directly through an explicit permutation-invariant GP kernel built from a stable set divergence. Across a broad synthetic benchmark suite—including a CCS-like two-set objective designed to mimic injector–producer geometric coupling—and a high-fidelity CCS case study on the Johansen formation, GP-Perm consistently improves sample-efficiency relative to non-invariant GP/DKL baselines and performs competitively against engineered invariant set-kernel baselines (DS/DE) and a learned-invariant DKL baseline (DKL-DS). Diagnostics on Johansen further indicate that these gains are accompanied by more stable posterior and acquisition behavior under tight evaluation budgets.

Our experiments suggest the following usage guidelines: (i) when exchangeability is exact and guaranteed by the problem setup (as under group control), enforcing permutation invariance in the surrogate is typically beneficial under limited BO budgets; (ii) explicit invariant kernels are a strong default when reliability under small data is critical; (iii) learned invariant embeddings can be attractive when sufficient data are available or when strict geometric assumptions are likely misaligned with the objective, but may require careful regularization and training protocols; and (iv) when wells are not truly exchangeable (e.g., well-specific constraints or controls), strict invariance may remove signal and should be relaxed.

This study focuses on fixed numbers of injectors and producers; extending the framework to variable cardinality would better reflect field development decisions. We also evaluate BO on fixed geological models; incorporating geological uncertainty (multiple realizations, calibration, or robust objectives) is needed for decision-making under subsurface uncertainty. Finally, while Deep Sets provide a strong learned-invariant baseline, more expressive set architectures (e.g., attention-based models) may better capture interaction-driven objectives, especially when combined with failure-aware acquisition and constraint handling for large-scale CCS BO campaigns.

Credit author statement

Conceptualization, S.F.; methodology, S.F.; software, S.F.; validation, S.F. and V.G.; resources, S.F.; writing—original draft preparation, S.F.; writing—review and editing, S.F., V.G.; visualization, S.F. All authors have read and agreed to the final version of the manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

The resources were granted with the support of GRNET as part of the project StorageCO_2. The publication of the article in OA mode was financially supported by HEAL-Link.

Data Availability Statement

Project code can be found at https://github.com/flammmes/Permutation-Invariant-Kernels/tree/main

References

[1] M. Bui, G. D. Puxty, M. Gazzani, S. M. Soltani, C. Pozo, The role of carbon capture and storage (ccs) technologies in a net-zero carbon future (2021).

[2] I. Ismail, V. Gaganis, Carbon capture, utilization, and storage in saline aquifers: Subsurface policies, development plans, well control strategies and optimization approaches—a review, Clean Technologies 5 (2023) 609–637.

[3] P. I. Frazier, A tutorial on bayesian optimization, arXiv preprint arXiv:1807.02811 (2018).

[4] C. K. Williams, C. E. Rasmussen, Gaussian processes for machine learning, volume 2, MIT press Cambridge, MA, 2006.

[5] T. Brown, A. Cioba, I. Bogunovic, Sample-efficient bayesian optimisation using known invariances, Advances in Neural Information Processing Systems 37 (2024) 47931–47965.

[6] J. Kim, M. McCourt, T. You, S. Kim, S. Choi, Bayesian optimization with approximate set kernels, Machine Learning 110 (2021) 857–879.

[7] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, A. J. Smola, Deep sets, Advances in neural information processing systems 30 (2017).

[8] A. G. Wilson, Z. Hu, R. Salakhutdinov, E. P. Xing, Deep kernel learning, in: Artificial intelligence and statistics, PMLR, 2016, pp. 370–378.

[9] P. Buathong, D. Ginsbourger, T. Krityakierne, Kernels over sets of finite sets using rkhs embeddings, with application to bayesian (combinatorial) optimization, in: International conference on artificial intelligence and statistics, PMLR, 2020, pp. 2731–2741.

[10] H. Moss, D. Leslie, D. Beck, J. Gonzalez, P. Rayson, Boss: Bayesian optimization over string spaces, Advances in neural information processing systems 33 (2020) 15476–15486.

[11] C. Oh, J. Tomczak, E. Gavves, M. Welling, Combinatorial bayesian optimization using the graph cartesian product, Advances in Neural Information Processing Systems 32 (2019).

[12] S. P. Fotias, I. Ismail, V. Gaganis, Optimization of well placement in carbon capture and storage (ccs): Bayesian optimization framework under permutation invariance, Applied Sciences 14 (2024) 3528.

[13] R. Garnett, M. A. Osborne, S. J. Roberts, Bayesian optimization for sensor set selection, in: Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks, 2010, pp. 209–219.

[14] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos, E. P. Xing, Neural architecture search with bayesian optimisation and optimal transport, Advances in neural information processing systems 31 (2018).

[15] M. Cuturi, Sinkhorn distances: Lightspeed computation of optimal transport, Advances in neural information processing systems 26 (2013).

[16] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, Y. W. Teh, Set transformer: A framework for attention-based permutation-invariant neural networks, in: International conference on machine learning, PMLR, 2019, pp. 3744–3753.

[17] M. Kimura, R. Shimizu, Y. Hirakawa, R. Goto, Y. Saito, On permutation-invariant neural networks, arXiv preprint arXiv:2403.17410 (2024).

[18] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, A. Smola, A kernel two-sample test, The journal of machine learning research 13 (2012) 723–773.

[19] F. Bachoc, L. Béthune, A. Gonzalez-Sanz, J.-M. Loubes, Gaussian processes on distributions based on regularized optimal transport, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2023, pp. 4986–5010.

[20] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouvé, G. Peyré, Interpolating between optimal transport and mmd using sinkhorn divergences, in: The 22nd international conference on artificial intelligence and statistics, PMLR, 2019, pp. 2681–2690.

[21] S. Ament, S. Daulton, D. Eriksson, M. Balandat, E. Bakshy, Unexpected improvements to expected improvement for bayesian optimization, Advances in Neural Information Processing Systems 36 (2023) 20577–20612.

[22] O. Andersen, G. Tangen, P. Ringrose, S. E. Greenberg, Co2 data share: a platform for sharing co2 storage reference datasets from demonstration projects, in: 14th greenhouse gas control technologies conference Melbourne, 2018, pp. 21–26.

[23] A. F. Rasmussen, T. H. Sandve, K. Bao, A. Lauser, J. Hove, B. Skaflestad, R. Klöfkorn, M. Blatt, A. B. Rustad, O. Sævareid, et al., The open porous media flow reservoir simulator, Computers & Mathematics with Applications 81 (2021) 159–185.

[24] P. S. Bergmo, E. Lindeberg, F. Riis, W. T. Johansen, Exploring geological storage sites for co2 from norwegian gas power plants: Johansen formation, Energy Procedia 1 (2009) 2945–2952.

[25] P. S. Bergmoa, A.-A. Grimstad, E. Lindeberg, F. Riis, W. T. Johansen, Exploring geological storage sites for co2 from norwegian gas power plants: Utsira south, Energy Procedia 1 (2009) 2953–2959.

Appendix A. Synthetic Test Functions

In all experiments, point coordinates are implemented in a normalized box and can be affinely mapped to $[0, 1]^2$ without changing permutation-invariance; for clarity, the definitions below are given in continuous coordinates.

Appendix A.1. Particle Configuration

This benchmark is inspired by physics problems and tests a model's ability to reason about simple geometric interactions. It involves finding the optimal configuration of a set of N particles in a 2-dimensional plane that minimizes a total energy function. The input is the $2 \cdot N$ dimensional vector representing the flattened (x, y) coordinates of the particles.

$$f(\mathbb{S}) = -\left(A \cdot \|\bar{\boldsymbol{p}} - \boldsymbol{p}_{\text{target}}\|^2 + B \sum_{i<j} \frac{1}{\alpha(x_i - x_j)^2 + \beta(y_i - y_j)^2 + \epsilon} \right) \tag{A.1}$$

Here, $\mathbb{S}$ is the set of particle locations, $\bar{\boldsymbol{p}}$ is the centroid of the set, and $\boldsymbol{p}_{\text{target}}$ is the target location (origin). The first term is a global attractive potential pulling the set's center of mass toward the origin. The second term is an anisotropic repulsive potential that penalizes particles for being too close. Anisotropy ($\alpha \neq \beta$) is introduced to break rotational symmetry, making the problem more challenging. The objective is inherently permutation-invariant as the total energy is independent of the particle ordering. For our experiments, we used $N = 10$ particles, resulting in a 20-dimensional search space.

Appendix A.2. Max Area Coverage

Inspired by sensor placement problems, this benchmark involves placing N circles of a fixed radius r within a $[-1, 1] \times [-1, 1]$ square to maximize their total union area. The objective function, which we approximate via Monte Carlo integration, presents a challenging landscape with large, nearly-flat regions and sharp gradients.

$$f(\mathbb{S}) = \text{Area}\left(\bigcup_{i=1}^{N} B(\boldsymbol{p}_i, r) \right) - C \sum_{i<j} \text{ReLU}\left(k \cdot (2r - \|\boldsymbol{p}_i - \boldsymbol{p}_j\|) \right) \tag{A.2}$$

In this function, $\mathbb{S}$ is the set of the circle centers, $B(\boldsymbol{p}_i, r)$ is the disk of radius r centered at $\boldsymbol{p}_i$, and $\text{Area}(\cdot)$ is the area of the union of these disks. The second term is a soft penalty that discourages overlap. This objective tests a different aspect of geometric reasoning, rewarding configurations that are spread out but remain centrally located to avoid wasting coverage area outside the domain. We use $N = 16$ circles with $r = 0.25$, resulting in a 32-dimensional search space.

Appendix A.3. Distribution Matching

This benchmark is designed to be a rigorous stress test for priors that make strong geometric assumptions. The objective is purely statistical: to arrange a set of N points such that their empirical distribution matches a fixed target distribution as closely as possible. The distance between distributions is measured by the Maximum Mean Discrepancy (MMD) [18].

$$f(\mathbb{S}) = -\max\left(0, \frac{1}{N(N-1)}\sum_{i\neq j} k(\boldsymbol{p}_i, \boldsymbol{p}_j) + \frac{1}{M(M-1)}\sum_{i\neq j} k(\boldsymbol{y}_i, \boldsymbol{y}_j) - \frac{2}{NM}\sum_{i,j} k(\boldsymbol{p}_i, \boldsymbol{y}_j)\right) \quad \text{(A.3)}$$

Here, $\mathbb{S} = \{\boldsymbol{p}_i\}_{i=1}^N$ is the set of candidate points and $\{\boldsymbol{y}_j\}_{j=1}^M$ is a set of M fixed samples from the target distribution. The function $k(\cdot,\cdot)$ is a radial basis function (RBF) kernel. The expression inside the negation is the unbiased estimator for the squared MMD. We clamp it at zero to ensure a well-posed objective for the optimizer. The target distribution is a bimodal Gaussian mixture, requiring the model to learn both the correct locations and relative densities of the modes. For our experiments, we used $N = 20$ points (40-dimensional space) and $M = 5,000$ target samples.

Appendix A.4. Max Spanning Tree

This benchmark tests the models' ability to optimize a combinatorial property of a point set. The objective is to find a configuration of N points in the plane that maximizes the total weight of its maximum spanning tree (MST).

$$f(\mathbb{S}) = \sum_{e\in \mathrm{MST}(\mathbb{S})} w(e) \quad \text{(A.4)}$$

Here, $\mathbb{S}$ is the set of points, which are treated as vertices in a complete graph. The weight $w(e)$ of an edge e between two points is their Euclidean distance. $\mathrm{MST}(\mathbb{S})$ is the set of edges that form the maximum spanning tree for the graph defined by $\mathbb{S}$. While the edge weights are geometric, the objective function is a complex, non-local property of the entire graph structure. It rewards configurations that are spread out in a structured, connected manner, providing a very different optimization landscape from the other benchmarks. For our experiments, we used $N = 15$ points, resulting in a 30-dimensional search space.

Appendix A.5. Facility Location (Soft Coverage)

This benchmark is inspired by facility location and coverage objectives and tests whether the surrogate can learn a smooth *soft-union* notion of coverage. The input is a set $\mathbb{S} = \{\boldsymbol{p}_i\}_{i=1}^N$ of N two-dimensional facility locations. A fixed set of M client locations $\mathbb{Y} = \{\boldsymbol{y}_j\}_{j=1}^M$ is sampled once and held constant throughout optimization.

We define an RBF similarity between a client $\boldsymbol{y}$ and a facility $\boldsymbol{p}$,

$$s(\boldsymbol{y}, \boldsymbol{p}) = \exp\left(-\frac{\|\boldsymbol{y}-\boldsymbol{p}\|_2^2}{2\sigma^2}\right), \quad \text{(A.5)}$$

and a soft coverage probability for each client as a differentiable soft-union,

$$c(\boldsymbol{y}; \mathbb{S}) = 1 - \prod_{i=1}^{N}(1 - s(\boldsymbol{y}, \boldsymbol{p}_i)). \quad \text{(A.6)}$$

The objective maximizes the average coverage across clients, optionally with a mild repulsion term that discourages nearly-collocated facilities:

$$f(\mathbb{S}) = \frac{1}{M}\sum_{j=1}^{M} c(\boldsymbol{y}_j;\mathbb{S}) \;-\; \lambda \sum_{i<k} \mathrm{softplus}(\kappa(r_{\mathrm{rep}} - \|\boldsymbol{p}_i - \boldsymbol{p}_k\|_2))/\kappa. \tag{A.7}$$

This function is permutation-invariant since it depends on $\mathbb{S}$ only through symmetric sums/products over its elements. In our experiments, $\mathbb{Y}$ is sampled either uniformly or from a simple Gaussian-mixture client distribution (fixed per benchmark instance).

Appendix A.6. Soft k-Medoids

This benchmark is a continuous, permutation-invariant relaxation of the k-medoids clustering objective. The input is a set $\mathbb{S} = \{\boldsymbol{p}_i\}_{i=1}^{N}$ of N two-dimensional medoid locations. A fixed dataset $\mathbb{Y} = \{\boldsymbol{y}_j\}_{j=1}^{M}$ is sampled once from a prescribed distribution (e.g., Gaussian, mixture, or ring+blob) and held constant throughout optimization.

For each data point $\boldsymbol{y}$, the (hard) distance to the closest medoid is $\min_i \|\boldsymbol{y} - \boldsymbol{p}_i\|_2$. We replace the minimum with a smooth soft-min using a temperature $\tau > 0$:

$$d_\tau(\boldsymbol{y};\mathbb{S}) = -\tau \log\left(\sum_{i=1}^{N} \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{p}_i\|_2}{\tau}\right)\right). \tag{A.8}$$

The clustering cost is the average soft-min distance over the fixed dataset,

$$\mathcal{L}(\mathbb{S}) = \frac{1}{M}\sum_{j=1}^{M} d_\tau(\boldsymbol{y}_j;\mathbb{S}), \tag{A.9}$$

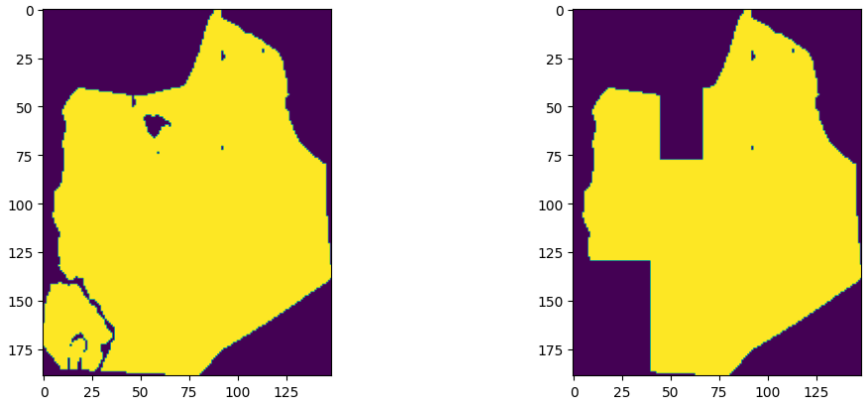
and the benchmark objective is defined to be maximization of the negative cost:

$$f(\mathbb{S}) = -\mathcal{L}(\mathbb{S}). \tag{A.10}$$

As $\tau \to 0$, $d_\tau(\boldsymbol{y};\mathbb{S})$ approaches $\min_i \|\boldsymbol{y} - \boldsymbol{p}_i\|_2$, recovering a sharp k-medoids-like objective while remaining differentiable for $\tau > 0$. The function is permutation-invariant since it depends on $\mathbb{S}$ only through symmetric operations over its elements.

Appendix B. Feasible Regions for CCS Benchmarks

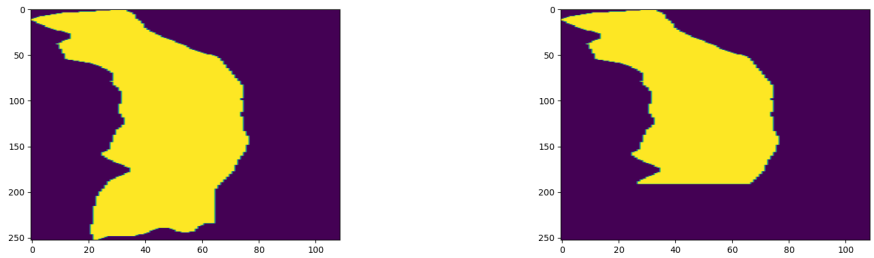
This appendix provides visualizations of the non-convex feasible regions for well placement in CCS formations. Figure B.6 represents the Johansen formation we used in this work and Figure B.7 represents the Smeaheia formation, which was not used in this work due to simulator stability issues. For each formation, we show the initial set of valid grid cells and the final, more constrained search space used by the optimizer after applying binary feasibility masks derived from domain constraints. Such constraint screening is common in CCS planning, where candidate locations are restricted to reduce leakage risk and ensure plume containment. This refinement step defines a realistic and challenging optimization problem, as discussed in Section 3.4.



(a) Initial Johansen Region

(b) Final Johansen Search Space

Figure B.6: **Feasible well placement regions for the Johansen formation.** **(a)** The initial set of all valid grid cells. **(b)** The final, more constrained search space after applying a mask. This mask excludes the thin western part of the formation, a step taken based on domain expertise to mitigate the risk of CO_2 leakage and ensure plume containment.



(a) Initial Smeaheia Region

(b) Final Smeaheia Search Space

Figure B.7: **Feasible well placement regions for the Smeaheia formation.** **(a)** The full extent of the initial valid region. **(b)** The final, highly non-convex search space after applying a mask to exclude cells outside the primary fault-bounded structural trap. This refinement is crucial for ensuring that all candidate well locations are within the primary geological containment structure, preventing leakage.

Appendix C. Ablation: effect of injector–producer interaction terms

This appendix isolates the contribution of the injector–producer interaction representation used in both GP-Perm and DKL-DS. The interaction component is designed to encode cross-group geometry (e.g., spacing, alignment, and relative positioning) that cannot be recovered from the injector and producer sets independently.

Appendix C.1. Ablation design

We compare surrogates that are identical in all hyperparameters and training protocol, except for the inclusion of an interaction term. For GP-Perm, this corresponds to including or removing the interaction-set Sinkhorn term in the composite distance (Eq. 4). For DKL-DS, this corresponds to including or removing the interaction-set Deep Sets encoder branch.

Models compared.. We use the four variants defined in our experiment configuration:

- **GP-Perm (full)**: interaction term enabled with weight `ip_weight` = 1.0.
- **GP-Perm (no interaction)**: interaction contribution disabled by setting `ip_weight` = 0.0 (equivalently $\ell_{IP}^{-2} = 0$ in Eq. 4).
- **DKL-DS (full)**: encoder processes $\{\mathcal{I}, \mathcal{P}, \mathcal{R}\}$ with `rel_weight` = 1.0.
- **DKL-DS (no interaction)**: interaction branch is effectively removed by setting `rel_weight` = 0.0, so the representation depends on $\{\mathcal{I}, \mathcal{P}\}$ only.

Controlled hyperparameters.. All remaining settings (kernel type, optimizer, learning rate, jitter schedule, acquisition function, and BO budget) are held fixed within each model family and match the protocol used for this synthetic two-set experiment. In particular, the injector/producer counts are fixed to $n_{\text{inj}} = 4$ and $n_{\text{prod}} = 6$, and each configuration is evaluated over $n_{\text{trials}} = 10$ independent trials.

Appendix C.2. Synthetic two-set benchmark (explicit objective)

To emphasize cross-group coupling, we use a synthetic objective that rewards *good matching* between producers and injectors while preventing within-group collapse via repulsion. The decision variable is a flattened vector

$$\boldsymbol{x} \in [-1, 1]^{2(n_{\text{inj}}+n_{\text{prod}})},$$

which is reshaped into $(n_{\text{inj}} + n_{\text{prod}})$ planar points. We write $\mathcal{I} = \{\boldsymbol{i}_a\}_{a=1}^{n_{\text{inj}}}$ and $\mathcal{P} = \{\boldsymbol{p}_b\}_{b=1}^{n_{\text{prod}}}$, where $\boldsymbol{i}_a, \boldsymbol{p}_b \in \mathbb{R}^2$. As in Eq. 5, the interaction set is $\mathcal{R} = \{\boldsymbol{p}_b - \boldsymbol{i}_a\}_{a,b}$.

Define squared pairwise distances

$$d_{ba}^2 = \|\boldsymbol{p}_b - \boldsymbol{i}_a\|_2^2.$$

For each producer, we compute a smooth approximation of the minimum distance to an injector using a soft-min with temperature $\tau > 0$:

$$s_b(\mathcal{I}, \mathcal{P}) = -\tau \log \sum_{a=1}^{n_{\text{inj}}} \exp\left(-\frac{d_{ba}^2}{\tau}\right). \tag{C.1}$$

The interaction cost averages this quantity over producers:

$$C_{\mathcal{IP}}(\mathcal{I},\mathcal{P}) = \frac{1}{n_{\text{prod}}} \sum_{b=1}^{n_{\text{prod}}} s_b(\mathcal{I},\mathcal{P}). \tag{C.2}$$

As $\tau \to 0$, s_b approaches $\min_a d_{ba}^2$, i.e., each producer is encouraged to be close to some injector.

To avoid degenerate solutions where all points collapse, we add inverse-distance repulsion within each set:

$$C_{\text{rep}}(\mathcal{I},\mathcal{P}) = \frac{\lambda_{\text{inj}}}{2} \sum_{\substack{a,a'=1 \\ a\neq a'}}^{n_{\text{inj}}} \frac{1}{\|\boldsymbol{i}_a - \boldsymbol{i}_{a'}\|_2^2 + \varepsilon} + \frac{\lambda_{\text{prod}}}{2} \sum_{\substack{b,b'=1 \\ b\neq b'}}^{n_{\text{prod}}} \frac{1}{\|\boldsymbol{p}_b - \boldsymbol{p}_{b'}\|_2^2 + \varepsilon}, \tag{C.3}$$

with a small $\varepsilon > 0$ for numerical stability.

The benchmark defines a total cost

$$C(\mathcal{I},\mathcal{P}) = C_{\mathcal{IP}}(\mathcal{I},\mathcal{P}) + C_{\text{rep}}(\mathcal{I},\mathcal{P}), \tag{C.4}$$

and the scalar objective for BO is *maximization of the negative cost*:

$$f(\mathcal{I},\mathcal{P}) = -C(\mathcal{I},\mathcal{P}). \tag{C.5}$$

We use the defaults $\tau = 0.05$, $\lambda_{\text{inj}} = \lambda_{\text{prod}} = 0.05$, and $\varepsilon = 10^{-4}$, with $n_{\text{inj}} = 4$ and $n_{\text{prod}} = 6$.

Appendix C.3. BO protocol and metrics

We run BO using the same candidate parameterization and best-so-far reporting as in the main experiments, specialized to this two-set synthetic objective.

Each trial starts from $n_{\text{init}} = 6$ randomly drawn points in the bounded domain, followed by $T = 25$ BO iterations with batch size $q = 4$, for a total of $n_{\text{eval}} = n_{\text{init}} + Tq = 106$ function evaluations per trial.

We use qLogExpectedImprovement (qLogEI) with a fixed Monte Carlo sampler and optimize the acquisition function in the normalized domain $[0, 1]^d$ using multi-start optimization (10 restarts and 256 raw samples). The evaluation points are then unnormalized back to $[-1, 1]^d$ and scored by Eq. C.5.

We report (i) mean best-so-far trajectories across trials, (ii) AUC of best-so-far curves, and (iii) final best-so-far values. All metrics are aggregated over the $n_{\text{trials}} = 10$ trials for this benchmark instance.

Appendix C.4. Results

Figure C.8 compares the mean best-so-far trajectories for GP-Perm and DKL-DS with and without explicit interaction modeling. Table C.8 reports AUC and final best-so-far values (mean ± std across trials). Since the objective is $f = -C$ (Eq. C.5), values closer to zero indicate better solutions.

Overall, including the interaction term improves performance in both model families under the fixed BO budget. For GP-Perm, enabling the interaction component yields a small improvement in both AUC and final best-so-far. For DKL-DS, the interaction branch provides a larger mean uplift, consistent with the purpose of the relational representation, although variance across trials remains non-negligible.

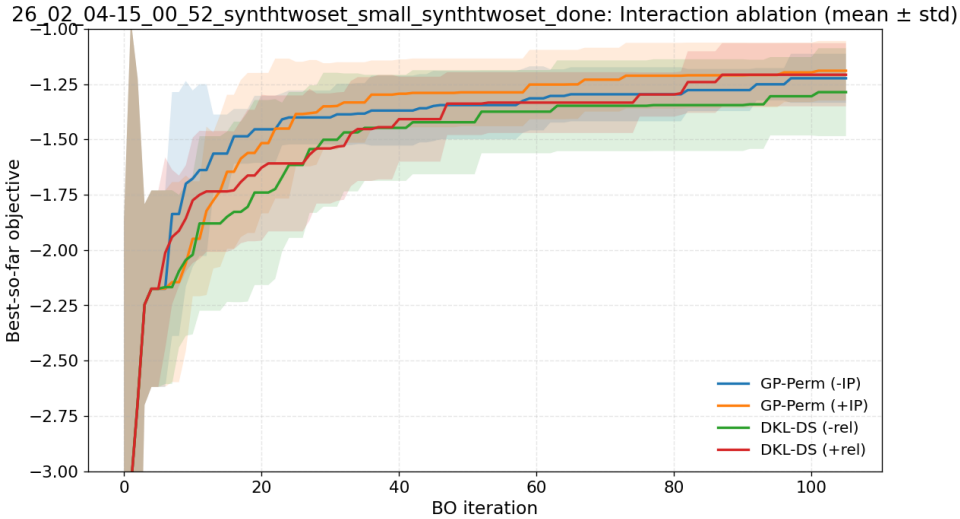


Figure C.8: Interaction-term ablation on the synthetic two-set benchmark (Eq. C.5). Mean best-so-far trajectories across $n_{\text{trials}} = 10$ trials for GP-Perm and DKL-DS, with and without interaction modeling (shaded: ±1 std).

Table C.8: Interaction-term ablation summary on the synthetic two-set benchmark (Eq. C.5). Report mean±std across $n_{\text{trials}} = 10$ trials for AUC of best-so-far curves and final best-so-far (higher is better). Best per metric is bolded.

Model	**AUC (mean±std)**	**Final best-so-far (mean±std)**
GP-Perm (full; +IP)	**-149.052 ± 17.817**	**-1.190 ± 0.134**
GP-Perm (no interaction; -IP)	-150.787 ± 9.013	-1.224 ± 0.111
DKL-DS (full; +rel)	-155.230 ± 12.525	-1.208 ± 0.143
DKL-DS (no interaction; -rel)	-161.654 ± 23.340	-1.286 ± 0.199

This ablation directly tests whether the surrogate needs an explicit cross-group representation to optimize a function whose primary signal is induced by injector–producer coupling (Eqs. C.1–C.5). Improvements in AUC indicate higher sample-efficiency, while improvements in final best-so-far indicate better asymptotic solution quality under the fixed budget.

Appendix D. Numerical diagnostics for divergence-based kernels

This appendix reports numerical diagnostics for the proposed divergence-based kernel construction. While the Sinkhorn divergence is a stable discrepancy, using it inside a radial kernel does not automatically guarantee strict positive semidefiniteness (PSD) for all hyperparameter settings. In practice, GP inference requires that kernel matrices remain numerically PSD (up to standard stabilizers such as jitter).

All diagnostics below are conducted for the synthetic two-set experiment in Appendix Appendix C (dimension $d = 2(n_{\text{inj}} + n_{\text{prod}}) = 20$), using the learned GP-Perm hyperparameters from a representative trained BO run.

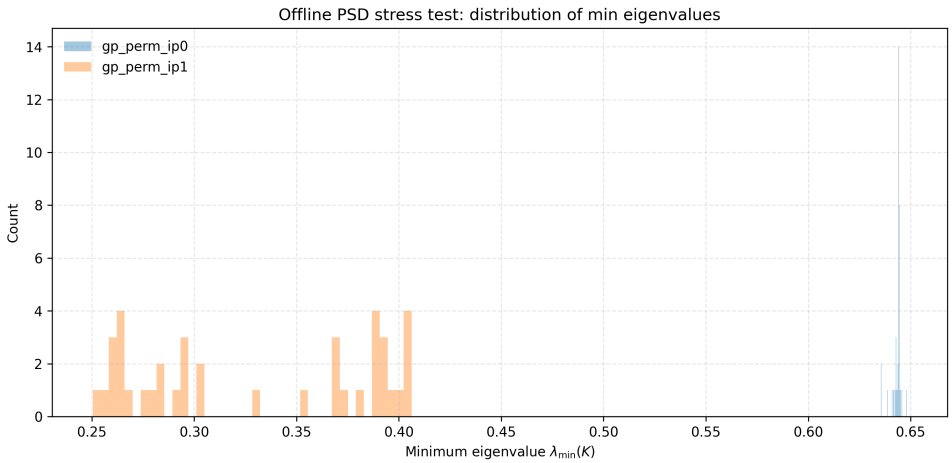


Figure D.9: Offline PSD stress test: histogram of $\lambda_{\min}(K)$ across random designs in $[0,1]^d$. Both GP-Perm variants produce strictly positive minimum eigenvalues in all tested draws.

Table D.9: Offline PSD stress test summary (random designs in $[0,1]^d$). We report median / mean $\lambda_{\min}(K)$ and the fraction of draws where $\lambda_{\min}(K) < -\delta$.

Variant	N	median $\lambda_{\min}$	mean $\lambda_{\min}$	$\delta = 10^{-10}$	$\delta = 10^{-8}$	$\delta = 10^{-6}$
GP-Perm (-IP)	64	0.644	0.644	0.000	0.000	0.000
GP-Perm (-IP)	128	0.644	0.643	0.000	0.000	0.000
GP-Perm (+IP)	64	0.390	0.384	0.000	0.000	0.000
GP-Perm (+IP)	128	0.272	0.276	0.000	0.000	0.000

Appendix D.1. Offline PSD stress test (random designs)

We draw random point sets $\{\boldsymbol{x}_i\}_{i=1}^N$ uniformly in $[0,1]^d$ and compute the kernel matrix $K \in \mathbb{R}^{N\times N}$ with entries $K_{ij} = k_{\text{GP-Perm}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. We report the minimum eigenvalue $\lambda_{\min}(K)$ and the fraction of draws where $\lambda_{\min}(K) < -\delta$ for tolerances $\delta \in \{10^{-10}, 10^{-8}, 10^{-6}\}$.

We test matrix sizes $N \in \{64, 128\}$ with 20 independent draws per N (per variant).

Appendix D.2. Training-set PSD diagnostic during BO

We next evaluate PSD behavior on the BO training sets encountered during optimization. For each trial and BO iteration, we form a subsampled prefix of the training inputs (up to 64 points for efficiency), compute the corresponding kernel matrix, and record $\lambda_{\min}(K)$. Figure D.10 reports the median $\lambda_{\min}(K)$ across trials as a function of BO iteration.

Takeaway.. Across both offline random designs and BO training prefixes, we observed no PSD violations ($\lambda_{\min}(K)$ remained non-negative in all cases), and all kernel matrices were numerically stable without requiring special treatment beyond standard GP practices.

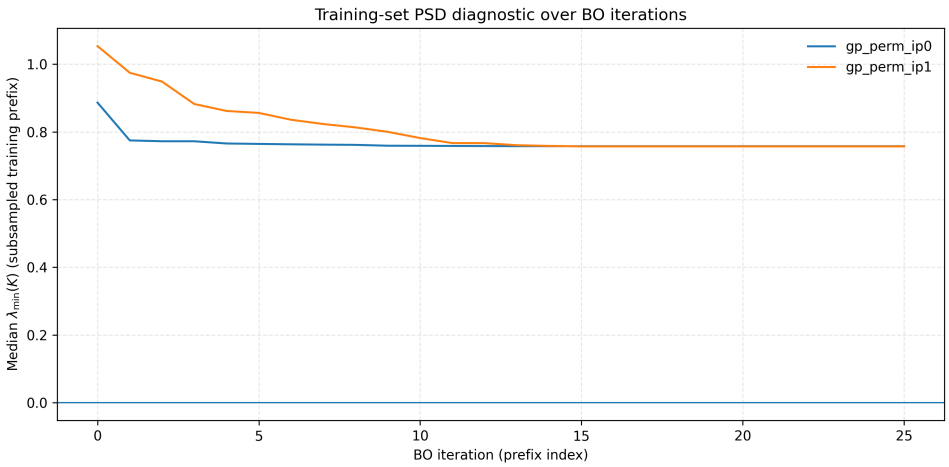


Figure D.10: Training-set PSD diagnostic during BO: median $\lambda_{\min}(K)$ (subsampled training prefixes) versus BO iteration. Both GP-Perm variants remain strictly PSD throughout training.

Table D.10: Training-set PSD summary across all trials and BO iterations (subsampled prefixes). Reported values are the minimum observed $\lambda_{\min}(K)$, the median over all evaluated prefixes, and the fraction of evaluated prefixes where $\lambda_{\min}(K) < -\delta$.

Variant	min $\lambda_{\min}$	median $\lambda_{\min}$	mean $\lambda_{\min}$	$\delta = 10^{-10}$	$\delta = 10^{-8}$	$\delta = 10^{-6}$
GP-Perm (-IP)	0.176	0.764	0.727	0.000	0.000	0.000
GP-Perm (+IP)	5.62×10^{-7}	0.794	0.725	0.000	0.000	0.000

Appendix E. Complete sensitivity analysis results

This appendix reports the benchmark-driven sensitivity analysis used *only* to select GP-Perm and DKL-DS configurations before running CCS case studies. In particular, we introduce here a normalized aggregation score that is used for *variant selection* across heterogeneous synthetic benchmarks. This normalization is *not* used for the main benchmark comparisons between model families (GP-Perm, DKL-DS, and baselines), where results are reported per benchmark using the raw best-so-far trajectories and their benchmark-specific AUC and final best value.

Appendix E.1. Variant selection score (normalized aggregation)

For a given benchmark e, trial r, and model variant m, let $b_t^{(e,r,m)}$ denote the best-so-far value after BO iteration t. We summarize each trial by:

$$\mathrm{AUC}^{(e,r,m)} := \sum_{t=1}^{T_e} b_t^{(e,r,m)}, \tag{E.1}$$

$$b_T^{(e,r,m)} := b_{T_e}^{(e,r,m)}, \tag{E.2}$$

where T_e is the BO horizon for benchmark e.

Table E.11: Model variants explored in the benchmark sensitivity analysis. Left: GP-Perm variants (varying only ε). Right: DKL-DS variants (varying pooling operators).

GP-Perm variants

Variant	ε
GP-Perm-1	10^{-2}
GP-Perm-2	$5 \cdot 10^{-2}$
GP-Perm-3	$5 \cdot 10^{-3}$
GP-Perm-4	10^{-4}
GP-Perm-5	0.5
GP-Perm-6	1
GP-Perm-7	5
GP-Perm-8	10
GP-Perm-9	2
GP-Perm-10	0.1

DKL-DS variants

Variant	Pooling
DKL-DS-1	mean + max + std + GEM ($p = \{0, 2\}$) + ARD
DKL-DS-2	mean + std + ARD
DKL-DS-3	mean + max + LSE + ARD
DKL-DS-4	GEM ($p = \{0, 1, 2\}$) + ARD
DKL-DS-5	mean
DKL-DS-6	mean + max + std + GEM ($p = \{0, 2\}$)
DKL-DS-7	mean + max + std + GEM ($p = \{0, 2\}$) + LSE + ARD
DKL-DS-8	mean + max + std + GEM ($p = \{0, 2\}$) + LSE

Because benchmarks differ in scale, we min–max normalize each metric *within* each benchmark across the compared variants:

$$\widehat{m}_e(m, r) = \frac{m_e(m, r) - \min_{m'} m_e(m', \cdot)}{\max_{m'} m_e(m', \cdot) - \min_{m'} m_e(m', \cdot)} \in [0, 1], \tag{E.3}$$

where m_e denotes either AUC or b_T for benchmark e, and the min/max are taken over all compared variants (pooling trials).

Appendix E.2. Model variants and sensitivity analysis

The benchmark stage includes a targeted sensitivity analysis for the two permutation-invariant surrogate families, GP-Perm and DKL-DS. For GP-Perm, we vary only the Sinkhorn-divergence regularization parameter ε, which following [20], controls a continuum between geometric and kernel-based comparisons: as $\varepsilon \to 0$, S_ε approaches the unregularized OT cost (Wasserstein-type matching), while as $\varepsilon \to \infty$ it converges to an MMD-like discrepancy induced by a Gibbs kernel $k_\varepsilon(\boldsymbol{s}, \boldsymbol{t}) \propto \exp(-C(\boldsymbol{s}, \boldsymbol{t})/\varepsilon)$. For empirical sets with uniform weights, this high-ε regime yields a double-sum form,

$$\mathrm{MMD}^2_{k_\varepsilon}(S, T) = \frac{1}{m^2} \sum_{a,a'} k_\varepsilon(\boldsymbol{s}_a, \boldsymbol{s}_{a'}) + \frac{1}{m^2} \sum_{b,b'} k_\varepsilon(\boldsymbol{t}_b, \boldsymbol{t}_{b'}) - \frac{2}{m^2} \sum_{a,b} k_\varepsilon(\boldsymbol{s}_a, \boldsymbol{t}_b), \tag{E.4}$$

thereby linking GP-Perm to classical double-sum set kernels in the large-ε limit.

For DKL-DS, we evaluate architectural variants that differ primarily in the choice of invariant pooling operators and whether ARD is enabled in the latent GP kernel. The explored variants are summarized in Table E.11. All remaining hyperparameters are held fixed within each family.

We select one GP-Perm variant and one DKL-DS variant by maximizing the selection score $S(m)$ defined in Section Appendix E.1. Detailed sensitivity results are summarized below.

Appendix E.3. GP-Perm: sensitivity to Sinkhorn regularization

Figures E.11–E.12 summarize GP-Perm sensitivity across benchmarks as distributions over trials of (normalized) AUC and final best value b_T, respectively, after per-benchmark normalization (Section Appendix E.1).

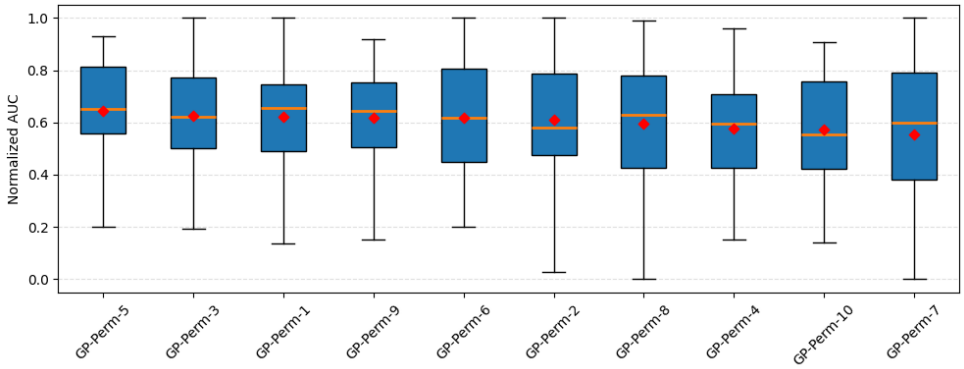


Figure E.11: GP-Perm sensitivity analysis across benchmarks. Distribution of normalized AUC over all trials (after per-benchmark normalization).

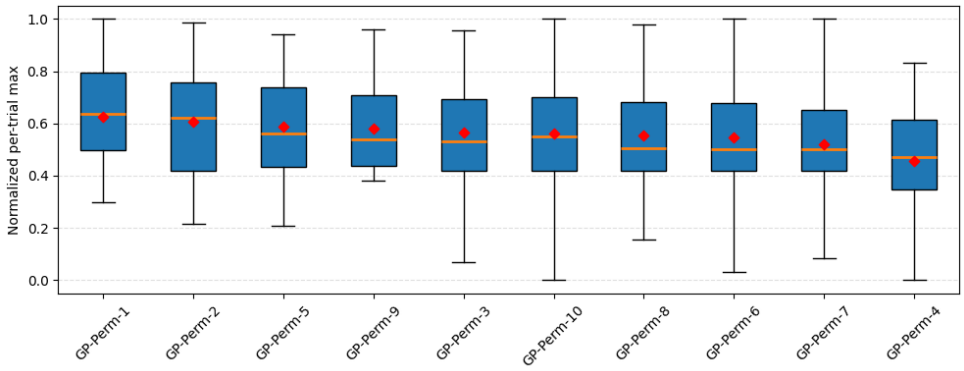


Figure E.12: GP-Perm sensitivity analysis across benchmarks. Distribution of normalized final best value b_T over all trials (after per-benchmark normalization).

Appendix E.4. DKL-DS: sensitivity to pooling design and encoder variants

Figures E.13–E.14 summarize DKL-DS sensitivity across benchmarks as distributions over trials of (normalized) AUC and final best value b_T, respectively, after per-benchmark normalization (Section Appendix E.1).

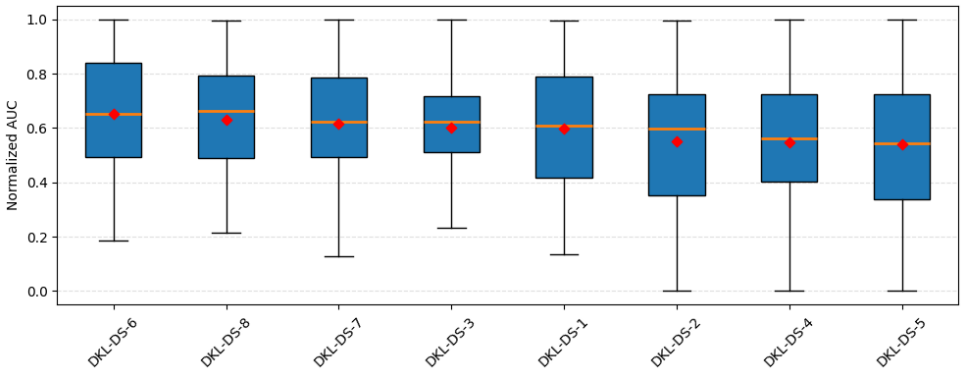


Figure E.13: DKL-DS sensitivity analysis across benchmarks. Distribution of normalized AUC over all trials (after per-benchmark normalization).

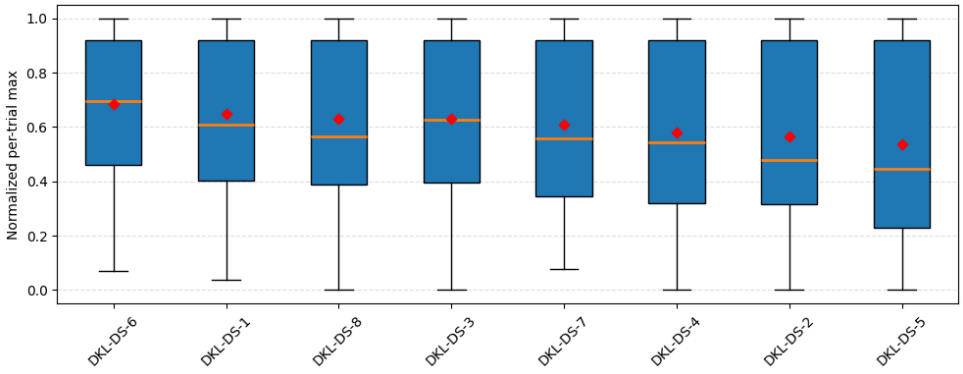


Figure E.14: DKL-DS sensitivity analysis across benchmarks. Distribution of normalized final best value b_T over all trials (after per-benchmark normalization).