IMPERIAL

Master of Science in Computing
(Artificial Intelligence and Machine Learning)

Imperial College London
Department of Computing

Bayesian Causal Discovery with Preference-Guided Normalising Flows

Author:
Cesare Bergossi

Supervisor:
Prof. Sonali Parbhoo

Second Marker:
Prof. Yingzhen Li

September 2025

Abstract

Causal discovery aims to identify the underlying structure of cause–effect relations, which is essential for reliable decisions in science, medicine, and policy. Classical methods assume access to large datasets or explicit numerical outcomes of interventions, but in practice such information is often scarce, noisy, or impractical to obtain. What is easier to elicit are preferences: comparative judgements about which intervention is more beneficial.

This thesis introduces Preferential Causal Bayesian Optimisation (PCBO), a framework that learns causal structure from preference signals rather than raw outcomes. PCBO combines Bayesian updates of local parent sets with Preferential Normalising Flows (PNFs), which flexibly model latent utilities behind observed choices. An adaptive acquisition strategy balances structural discovery and preference refinement, guiding interventions toward those expected to be most informative for both structure and utility.

The framework is evaluated on a spectrum of settings: small illustrative graphs, larger Erdős–Rényi families, and a medical-inspired case study. Across these tests, PCBO consistently recovers meaningful and often near-accurate causal structure and remains competitive with outcome-based baselines, despite relying only on comparative feedback.

These results demonstrate that preference data, though weaker than direct measurements, can support robust causal discovery when coupled with Bayesian reasoning and principled intervention design. PCBO therefore opens a practical path for structure learning in domains where conventional data requirements cannot be met.

Contents

Chapter 1

Introduction

1.1 Motivation

Decisions that matter rarely depend on correlation. In medicine, it is not enough to know that patients receiving a treatment *tend* to recover; what truly matters is whether prescribing that treatment would *cause* recovery for a given patient. The same distinction holds in economics: a policymaker is not satisfied that unemployment *moves with* taxation; they must anticipate what *will happen* if a specific reform is enacted. Likewise, an engineer deploying a control policy in a safety–critical system must reason about interventions rather than past correlations. Across such domains, reliable decisions depend on causal reasoning rather than statistical association.

Causal models encode directional relations and the consequences of hypothetical interventions, which allows to predict not only what is likely, but what would change if we were to act differently. This capacity to guide interventions and support counterfactual reasoning is what gives causality its unique value in science and high-stakes decision-making.

1.1.1 The Challenge of Learning Causality from Limited Information

Discovering causal structure is notoriously difficult in modern settings. Real-world datasets are high-dimensional, noisy, and often small relative to the complexity of the system. Controlled experiments (the gold standard of causal inference) are costly, ethically restricted, or logistically infeasible. It would be unrealistic, for example, to test every possible drug combination in clinical medicine, or to perform nationwide economic interventions at will. Consequently, many studies rely on observational data, where separating genuine causation from spurious correlation requires strong assumptions and leaves substantial uncertainty.

Even advanced approaches such as score-based search, constraint-based tests, or differentiable formulations struggle to scale as complexity grows. Bayesian methods provide a logical way to represent uncertainty and integrate prior knowledge, but exact posterior inference quickly becomes intractable for larger graphs.

These challenges motivate the search for approaches that are both robust to scarce, noisy data and scalable to complex environments.

1.1.2 Why Preference Signals?

One promising direction is to rethink the type of data we use. Traditional approaches assume precise numerical outcomes: probabilities of recovery, exact growth rates, or measured phenotypic effects. In practice, such information is often unavailable or unreliable. What experts *can* provide more naturally are comparative judgements. A clinician may not know the exact success rate of two treatments, but can often say which is more likely to help. Such preferences are easier to elicit, less sensitive to noise, and often more realistic to obtain than cardinal measurements.

Preference-based causal discovery builds on this idea. Instead of requiring full outcome distributions, it treats pairwise or ordinal comparisons as the fundamental input. These comparisons, while potentially noisy, still carry valuable information about the underlying causal mechanisms. Using preferences, we can exploit expert intuition in a flexible way, without demanding unrealistic data collection.

Recent methodological advances make this particularly powerful. Preferential Normalising Flows (PNFs) are models that learn a hidden "utility function" capable of explaining why one intervention is preferred over another. Instead of assuming preferences follow a rule, PNFs can represent very flexible, non-linear patterns: they can bend and adapt to match complex expert judgements. At the same time, because they are built on normalising flows, they remain mathematically tractable: we can train them efficiently and evaluate how likely different preferences are. When embedded in a Bayesian framework, PNFs do more than just fit past data: they continually update our uncertainty about the causal graph as new comparisons are observed.

1.1.3 Towards a New Approach

This thesis introduces a new framework, *Preferential Causal Bayesian Optimisation* (PCBO), designed to make effective use of limited and noisy information. The central idea is intuitive: when we cannot observe everything, we must choose carefully what to ask. PCBO achieves this by (i) turning preference judgements into Bayesian updates of the causal graph, (ii) actively selecting interventions that are most likely to improve our understanding of the system.

The process is iterative. Each new comparison refines both our beliefs about the underlying structure and our estimate of which interventions are promising. At the same time, acquisition strategies guide exploration towards interventions that are maximally informative, rather than wasteful. Integrating preferences, Bayesian uncertainty quantification, and targeted intervention design is what gives PCBO its strength: a clear path to causal discovery in settings where traditional methods – reliant on large, clean datasets – quickly reach their limits.

1.2 Problem Statement

The central aim is to test whether preferences can provide a solid basis for causal discovery. The challenge is not simply to fit a model of pairwise choices, but to understand whether such feedback can truly guide the recovery of causal structure when observations are limited, noisy, or costly to obtain.

The task naturally splits into two parts. First, we seek to maintain a posterior distribution over possible graphs, to represent what is known and what remains uncertain. Second, we learn a latent utility function that explains why one intervention is preferred to another. Each new comparison should move both fronts forward: refining our beliefs about the underlying structure, and clarifying which interventions are likely to be useful.

This leads to four guiding questions:

1. *Structure learning.* How accurately and reliably does PCBO recover the true causal graph across different settings?
2. *Baselines.* How does PCBO compare to outcome-based and existing preference-based methods?
3. *Thresholding.* How sensitive is structural recovery to edge-probability threshold choice?
4. *Flow architecture.* How do different flow families affect preference modelling and overall performance?

The working hypothesis is that, although preferences may appear weaker than numerical outcomes, when paired with Bayesian reasoning and careful intervention choice they can drive scalable and effective causal discovery.

1.3 Contributions

The original contributions of this work fall into four areas, each addressing a core challenge in bringing together preference-based learning and causal discovery.

A new framework for preference-based causal discovery. The thesis introduces *Preferential Causal Bayesian Optimisation* (PCBO), a unified framework that combines Preferential Normalising Flows (PNFs) with Bayesian structure learning. This makes it possible to infer causal graphs directly from pairwise or ordinal comparisons, extending causal discovery to domains such as medicine, biology, or policy analysis where calibrated numerical measurements are costly, noisy, or unavailable.

Acquisition functions tailored to preferences. Two acquisition strategies are developed to make preference queries maximally informative. *Preference Information Gain* (PIG) reduces uncertainty about latent utilities, while *Expected Edge Information Gain* (EEIG) reduces uncertainty about the causal graph. These objectives are blended through an adaptive schedule that prioritises structure in early iterations and utility refinement later, ensuring that each query is used effectively.

Scalable Bayesian learning under uncertainty. To make the framework feasible beyond small graphs, PCBO implements both an exact local parent posterior and a new *Scalable Parent Posterior* based on MCMC approximations. A greedy DAG projection is incorporated to guarantee acyclicity, making posterior edge probabilities consistent with valid graph structures. These components extend Bayesian causal discovery to settings with many nodes while keeping computation tractable.

Implementation refinements and evaluation. The system was engineered for stability and robustness. Flow training was stabilised with early stopping, learning-rate scheduling, gradient clipping, and a short warm-up on synthetic preferences; numerical routines (e.g. Cholesky-based updates and vectorised statistics) improved reliability and speed. Evaluation spans a range of synthetic datasets: hand-crafted three- and six-node graphs, a medical toy model, and Erdős–Rényi families at increasing sizes. Results are reported on both structural metrics (SHD, edge precision/recall, calibration) and utility metrics (preference accuracy, likelihood), together with runtime diagnostics and ablations of design choices.

Code availability. The full implementation is openly available at this GitHub repository. Instructions for reproducing the results are provided in Appendix 6.2.4.

Research environment. This thesis was carried out within the AI for Actionable Impact (AI4AI) Lab at Imperial College London, led by Prof. Sonali Parbhoo. The lab creates an environment of open discussion and curiosity, which proved extremely valuable in questioning assumptions and refining the ideas that shaped PCBO. Contributions also came from PhD student Marcello Negri, whose support and feedback helped strenghten the work at several stages.

1.4 Thesis Organisation

The remainder of this thesis is organised as follows.

Chapter 2 reviews the background on causal discovery, preference learning, and normalising flows, and situates the work within related literature.

Chapter 3 develops the theoretical framework: formal setting, modelling assumptions, and the probabilistic objects (utility from preferences, local parent posteriors, information objectives).

Chapter 4 presents the methodology: the PCBO pipeline (utility learning, local Bayesian updates, greedy DAG projection, acquisition), along with implementation details.

Chapter 5 reports the empirical evaluation on synthetic datasets and a medical case study, comparing to baselines, and analysing design choices.

Chapter 6 concludes with a summary of contributions and directions for future work.

References and declarations follow the main chapters, with appendices containing additional figures and extended results.

Chapter 2

Background and Literature Review

2.1 Causal Graphs and Structural Causal Models (SCMs)

At the heart of causal inference lies a simple question: how can we represent *mechanisms*, not just correlations? Structural Causal Models (SCMs), introduced by Pearl [1], provide a precise answer. They combine a graphical representation with functional equations, giving us a language for reasoning not only about associations, but also about interventions.

2.1.1 Directed Acyclic Graphs (DAGs)

An SCM is indexed by a *Directed Acyclic Graph* (DAG) [1]. A DAG $G = (V, E)$ consists of nodes $V = \{X_1, \ldots, X_d\}$ representing random variables, and directed edges E encoding possible causal relations. The acyclicity condition (that no directed path loops back to its start) guarantees a consistent causal ordering: each variable can only depend on its predecessors.

Concretely, every variable X_i is determined by a structural equation

$$X_i := f_i(\mathrm{Pa}(X_i), U_i),$$

where $\mathrm{Pa}(X_i)$ denotes its parents in the graph, f_i is a deterministic function, and U_i is an exogenous noise term. Together, these equations and the noise variables specify the data-generating process.

A particularly useful implication of this structure is the *Markov factorisation* [1]:

$$p(X_1, \ldots, X_d) = \prod_{i=1}^{d} p(X_i \mid \mathrm{Pa}(X_i)).$$

This tells us that each variable depends only on its parents, and once those are known, the rest of the system provides no additional information. In other words, the DAG encodes conditional independencies. This property makes DAGs expressive, capable of representing complex dependencies, while also computationally convenient, since modelling the full joint distribution reduces to modelling a set of local conditionals.

2.1.2 Interventions and the Do-Operator

Observational data describe correlations, but these may reflect confounding or common causes rather than genuine causation. To reason about cause and effect, we need to model interventions. Pearl's *do-operator* formalises this [1].

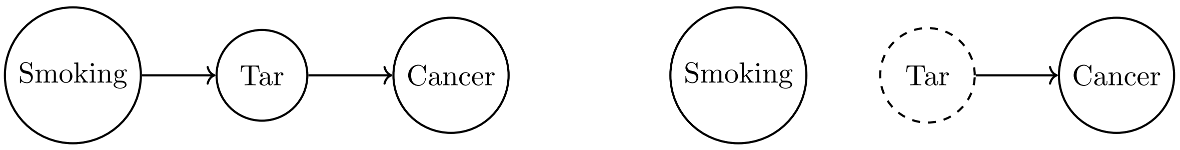
An intervention $\mathrm{do}(X_j = x)$ replaces the structural equation for X_j with a constant x, removing all incoming edges to X_j in the graph. The resulting distribution is written as

$$p(X_1, \dots, X_d \mid \mathrm{do}(X_j = x)).$$

This is fundamentally different from conditioning on $X_j = x$, which selects cases where X_j happens to take value x, but leaves the mechanisms generating X_j unchanged. The do-operator, by contrast, alters the mechanism itself, and as a result,

$$p(Y \mid X = x) \neq p(Y \mid \mathrm{do}(X = x))$$

in general. This distinction is at the core of causal reasoning [1].



(a) Original DAG: Smoking → Tar → Cancer. **(b)** Mutilated DAG under do(Tar).

Figure 2.1: Illustration of an intervention. (a) In the original DAG, Smoking influences Cancer indirectly through Tar. (b) Under the intervention do(Tar $= t$), incoming edges to Tar are cut, showing how interventions alter the graph *locally* while preserving other mechanisms.

Truncated factorisation. Interventions act like surgical edits to the model. Recall that in a DAG the observational distribution obeys the Markov factorisation. When we apply an intervention $\mathrm{do}(X_j = x)$, the conditional for X_j is overwritten and replaced by a point mass at x, while all other local conditionals remain untouched:

$$p(x_1, \dots, x_d \mid \mathrm{do}(X_j = x)) = \left[\prod_{i \neq j} p(x_i \mid x_{\mathrm{Pa}(X_i)})\right] \delta(x_j - x).$$

This is known as the *truncated factorisation formula* [1].

Soft interventions. Not all interventions set variables to constants; some modify the mechanism. A *soft intervention* replaces $p(x_j \mid x_{\mathrm{Pa}(X_j)})$ with a new conditional $q(x_j \mid x_{\mathrm{Pa}(X_j)})$ [2, 3]:

$$p(x_1, \dots, x_d \mid \mathrm{do}_q(X_j)) = q(x_j \mid x_{\mathrm{Pa}(X_j)}) \prod_{i \neq j} p(x_i \mid x_{\mathrm{Pa}(X_i)}).$$

This covers, for example, randomised treatment assignment or policy changes that shift behaviour without clamping a variable to a constant.

2.1.3 Interventional Distributions and Identifiability

A natural question is whether interventional effects can be estimated from observational data alone. The causal effect $p(Y \mid \mathrm{do}(X = x))$ is said to be *identifiable* if it can be expressed purely in terms of the observational distribution and the known graph, without needing new experiments [1]. Identifiability therefore tells us whether the data we already have suffice to answer a causal query.

The classic *back-door criterion* provides one such condition [1, 4]. Suppose we want the effect of X on Y. If we can find a set Z of variables that blocks all back-door paths (paths entering X through an incoming edge), then adjusting for Z gives

$$p\big(Y \mid \mathrm{do}(X = x)\big) \;=\; \mathbb{E}_Z\big[\,p\big(Y \mid X = x, Z\big)\,\big]\,.$$

This adjustment allows causal effects to be estimated from purely observational data, provided the right set Z is known.

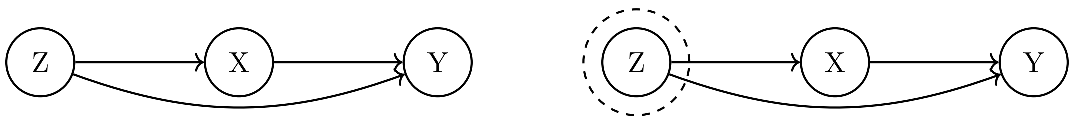


Figure 2.2: Illustration of the back-door criterion. Left: a confounder Z opens a spurious path between X and Y. Right: conditioning on Z blocks the back-door path, allowing identification of the causal effect.

Identifiability marks the boundary of what observations can tell us; beyond it, we need additional information – typically interventional data, and in our case preferences – to recover causal effects.

2.2 Bayesian Structure Learning

Bayesian methods have long been central to causal discovery [5, 6]. Instead of selecting a single "best" graph, they treat the graph G itself as a random variable and infer a posterior distribution over possible structures. This perspective has two main advantages: it allows to quantify uncertainty explicitly, and it provides a natural way to incorporate prior knowledge about the system. In data-limited or noisy settings, this ability to represent uncertainty is especially important.

2.2.1 Local Decomposition of the Likelihood

A central idea in Bayesian structure learning is the *local decomposition* of the likelihood [6, 7]. Suppose we have data D consisting of n i.i.d. samples of variables $X = \{X_1, \dots, X_d\}$. The likelihood of the data given a DAG G with parameters Θ factorises according to the parent sets of the graph:

$$p(D \mid G, \Theta) = \prod_{i=1}^{d} \prod_{j=1}^{n} p\big(x_i^{(j)} \mid \mathrm{pa}_i^{(j)}, \theta_i\big),$$

where $\mathrm{pa}_i^{(j)}$ are the observed values of $\mathrm{Pa}(X_i)$ in sample j. This means that the contribution of each variable depends only on its parents. The global problem of learning the entire DAG thus reduces to deciding, for each node, which parent set provides the best explanation of the data.

This property is not just mathematically neat, but computationally crucial. Without decomposition, evaluating every possible DAG would quickly become infeasible, since the number of candidate graphs grows super-exponentially with the number of nodes. Local factorisation instead allows Bayesian methods to evaluate structures one parent set at a time. This same principle motivates the *parent posterior* developed in Chapter 3, where each node's parents are inferred locally before being combined into a global graph.

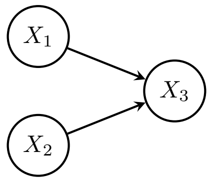


Figure 2.3: Illustration of local decomposition: X_1 and X_2 are parents of X_3. The graph score factorises as $\mathrm{Score}(X_1) \cdot \mathrm{Score}(X_2) \cdot \mathrm{Score}(X_3 \mid X_1, X_2)$.

2.2.2 Conjugate Updates in the Linear–Gaussian Case

A particularly elegant property of Bayesian structure learning emerges when each node is assumed to follow a linear–Gaussian conditional distribution. For a variable X_i with parents $\mathrm{Pa}(X_i)$,

$$X_i = \beta_i^\top \mathrm{Pa}(X_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_i^2).$$

This is simply Bayesian linear regression embedded inside a DAG.

With a Normal–Inverse-Gamma prior on the regression parameters and noise variance,

$$(\beta_i \mid \sigma_i^2) \sim \mathcal{N}(\mu_0, \sigma_i^2 \Lambda_0^{-1}), \quad \sigma_i^2 \sim \text{Inv-Gamma}(\alpha_0, \beta_0),$$

posterior updates admit a closed form. This conjugacy implies that the marginal likelihood of a parent set can be computed exactly:

$$p(x_i^{1:n} \mid \mathrm{Pa}(X_i)) = \int p(x_i^{1:n} \mid \mathrm{pa}_i^{1:n}, \beta_i, \sigma_i^2)\, p(\beta_i, \sigma_i^2)\, d\beta_i\, d\sigma_i^2.$$

This closed-form result, first formalised by Geiger and Heckerman [8], is known as the *Bayesian Gaussian equivalent* (BGe) score. It remains widely used because of its efficiency and mathematical clarity: once data are observed, we can integrate out the parameters and directly compare how well different parent sets explain the variability of a node. Although real-world data often depart from linearity or Gaussian noise, this model provides a tractable and interpretable foundation that continues to serve as a benchmark for more general approaches. In this thesis, it forms the basis of the local parent posterior model used within PCBO.

Uniform prior
Sparse prior
X_1
X_2
X_3
X_1
X_2
X_3

Figure 2.4: Effect of different priors: a uniform prior implicitly favours dense graphs, while a sparsity-promoting prior leads to simpler structures.

2.2.3 Priors and Edge Posteriors

A Bayesian treatment of structure learning also requires specifying a prior distribution over graphs. The simplest choice is a *uniform prior*, which assigns equal weight to all DAGs with d nodes [6]. While convenient, this prior implicitly favours dense graphs, since there are far more of them. A more realistic approach is to use sparsity-promoting priors, which penalise large parent sets. A common formulation assigns independent Bernoulli probabilities to each edge:

$$p(G) \propto \prod_{i=1}^{d} \rho^{|\mathrm{Pa}(X_i)|}(1-\rho)^{(d-1)-|\mathrm{Pa}(X_i)|},$$

where ρ controls the expected edge density [9]. Smaller values of ρ encourage sparser graphs, which aligns with the intuition that most variables in science or medicine have only a few direct causes.

The posterior over graphs then combines this prior with the marginal likelihoods of parent sets:

$$p(G \mid D) \propto p(G) \prod_{i=1}^{d} p(X_i \mid \mathrm{Pa}(X_i)).$$

Rather than committing to a single graph, it is often more informative to summarise results in terms of *posterior edge probabilities* [10]:

$$p\big((X_j \to X_i) \in E \mid D\big) = \sum_{G:(X_j \to X_i) \in G} p(G \mid D).$$

These values represent our degree of confidence in each edge given the data. For instance, a posterior probability of 0.8 for the edge $X_j \to X_i$ means that, under the model and the observed data, we are 80% confident that X_j is a direct cause of X_i.

2.3 Active Learning in Causal Discovery

Most classical approaches to causal discovery are studied in a *passive* setting: the learner observes samples from an unknown system and attempts to reconstruct its structure. This faces a fundamental obstacle: different graphs can encode exactly the same observational distribution, forming what are known as *Markov equivalence classes* [1]. No matter how much observational data we collect, these equivalences cannot be broken by statistics alone.

Active learning offers a way forward. Instead of waiting for more data, the learner deliberately selects interventions to resolve these ambiguities. By perturbing variables and observing the consequences, we can distinguish between competing causal hypotheses and reduce uncertainty in a principled way. Such strategies have become increasingly relevant in domains where experiments are costly and must be chosen with care, such as biology (e.g. targeted gene knockouts) and economics (e.g. policy interventions) [11, 12, 13].

2.3.1 Information-Theoretic Acquisition Functions

How should we decide which intervention is most valuable? The dominant principle comes from information theory: choose the action expected to reduce uncertainty about the causal graph the most.

Formally, this is expressed through an *acquisition function* $\alpha(x)$, which assigns a score to each candidate intervention on variable x. The score is often defined in terms of expected entropy reduction (a measure of how much uncertainty, or disorder, is reduced) or mutual information:

$$\alpha(x) = I(G; Y \mid \mathrm{do}(x), D), \quad \text{with} \quad I(A; B) = H(A) - H(A \mid B),$$

where entropy is $H(Z) = -\sum_z p(z) \log p(z)$. Here, G is the causal graph, D the data collected so far, and Y the outcome of the intervention. Intuitively, the best intervention is the one that, on average, rules out the largest number of currently plausible structures.

A simple example illustrates the idea. Suppose the current evidence cannot distinguish between $X \to Y$ and $Y \to X$. Observing correlations alone leaves both options equally plausible. But if we intervene on X and see whether changes propagate to Y, the ambiguity collapses. The mutual-information criterion captures exactly this: it rewards interventions expected to resolve structural uncertainty.

Several acquisition strategies have been proposed, differing in what uncertainties they target and how they approximate information gain.

Edge Entropy. Early work by Tong and Koller [11] proposed *edge entropy*, a simpler heuristic that targets uncertainty at the edge level. The idea is to intervene where the posterior distribution of an edge is maximally uncertain:

$$\alpha_{\mathrm{EE}}(x) = \sum_{(i,j)} H\big[p(G_{ij} \mid D, \mathrm{do}(x))\big],$$

where G_{ij} indicates whether an edge from X_j to X_i is present. Here $H[p(G_{ij} \mid D, \mathrm{do}(x))]$ is read as a tractable proxy for the expected posterior edge entropy after an intervention on x. While less principled than full mutual-information methods, edge entropy is cheap to compute and has shown competitive empirical performance.

Expected Information Gain (EIG). A more general approach formalises acquisition in terms of *expected information gain*, which evaluates interventions by how much they are expected to shift the posterior over graphs [14, 15, 16]. Formally,

$$\alpha_{\text{EIG}}(x) = \mathbb{E}_{y\sim p(y|\text{do}(x),D)}\Big[\text{KL}(p(G \mid D \cup \{(\text{do}(x), y)\}) \,\|\, p(G \mid D))\Big],$$

where the expectation is over possible outcomes y of the intervention. This captures the principle that an experiment is valuable not for its raw outcome, but for how much it sharpens our beliefs about the causal structure.

Approximations. In practice, exact computation of EIG is intractable: it requires averaging over all graphs and outcomes. Recent work addresses this through approximations. Choi et al. [15] use variational bounds to approximate the mutual-information objective, while Wang et al. [16] exploit the factorisation of Bayesian scores into local parent sets. These methods preserve the spirit of information-theoretic selection while keeping computations feasible in larger systems.

These approaches point to a simple principle: interventions should not be chosen arbitrarily, but because they are expected to most reduce our uncertainty about the structure. This naturally sets the stage for a sequential, optimisation-oriented view of causal discovery.

2.3.2 Causal Bayesian Optimisation (CBO)

The ideas of active learning can be cast in a broader optimisation perspective. *Causal Bayesian Optimisation (CBO)* was formalised by Aglietti et al. [17], and treats discovery not as isolated interventions, but as a sequential decision process: each experiment is chosen because it is expected to be maximally informative, beliefs are updated, and the cycle repeats.

The connection to Bayesian optimisation (BO) is straightforward. In classical BO, the task is to optimise an unknown function f by sequentially querying it at points chosen by an acquisition strategy. Each query updates a surrogate model (often a Gaussian process), which balances exploration with exploitation.

In the causal setting, the objective changes. The "function" is the causal graph itself, and the learner intervenes on variables rather than black-box inputs. At round t, a posterior distribution $p(G \mid \mathcal{D}_t)$ represents current beliefs about the graph. The learner selects an intervention a_t (for example, fixing a variable to a chosen value), observes the outcome y_t, and updates its posterior through Bayes' rule:

$$p(G \mid \mathcal{D}_{t+1}) \;\propto\; p(y_t \mid \text{do}(a_t), G)\, p(G \mid \mathcal{D}_t).$$

This cycle, illustrated in Figure 2.5, makes causal discovery an explicitly iterative optimisation problem. In a biological network, for instance, silencing a gene is not performed arbitrarily, but because it offers evidence to distinguish genuine causal effects from spurious ones.

2.3.3 Limits of Numerical Outcomes

Most existing formulations of CBO assume that outcomes of interventions can be measured as precise numerical values: expression levels in gene networks, continuous rewards in reinforcement learning, or sensor readings in physical systems. This makes the mathematics convenient, since such outcomes fit neatly into parametric models or flexible surrogates like Gaussian processes.

However, this assumption is restrictive. In practice, experts often express outcomes comparatively rather than numerically. A doctor may judge one treatment as preferable to another

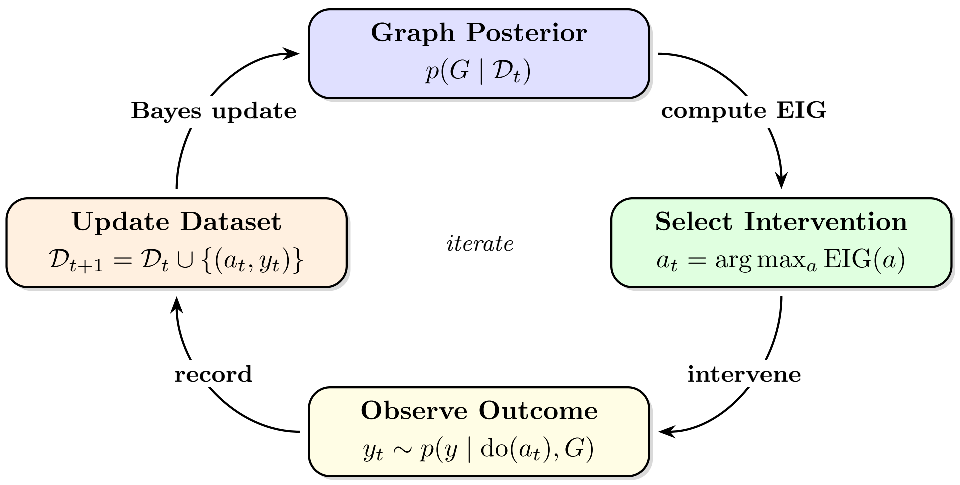

Graph Posterior
$p(G \mid \mathcal{D}_t)$
compute EIG
Select Intervention
$a_t = \arg\max_a \mathrm{EIG}(a)$
intervene
Observe Outcome
$y_t \sim p(y \mid \mathrm{do}(a_t), G)$
record
Update Dataset
$\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(a_t, y_t)\}$
Bayes update
iterate

Figure 2.5: The iterative loop of CBO. Graph beliefs are updated through the posterior, interventions are chosen by maximising expected information gain, outcomes are observed, and the dataset grows for the next round.

without assigning exact probabilities of recovery. In behavioural science, or human-in-the-loop systems, responses are naturally ordinal or pairwise.

This observation motivates a shift: replacing numeric outcomes with relative feedback. Preferences are easier to elicit, more robust to noise, and align better with how decisions are actually made in many domains. The next chapter builds directly on this idea, extending CBO into a preference-guided framework that forms the core of this thesis.

2.4 Preference Learning

In many problems we do not observe precise numerical outcomes; we observe *comparisons*. A clinician may say that treatment A worked better than B, a user may click one recommendation over another, and a domain expert may rank several interventions by their perceived impact. Such relative judgements are often easier to elicit, more robust to noise, and – in settings where calibrated measurements are costly or ill-defined – more realistic than absolute scores. *Preference learning* provides the statistical tools to turn these qualitative signals into quantitative models.

2.4.1 Random Utility Models (RUMs)

Random Utility Models (RUMs) formalise the idea that each option i has an unobserved *utility*

$$U_i \;=\; \theta_i \;+\; \varepsilon_i,$$

where $\theta_i \in \mathbb{R}$ reflects its intrinsic value and ε_i is a noise term [18, 19]. An observed preference $i \succ j$ occurs precisely when $U_i > U_j$. Different assumptions on the noise distribution yield different models: Gaussian noise gives rise to Thurstone's (probit) model [18], while Gumbel noise leads to the Bradley–Terry–Luce family [20, 21]. This simple construction connects noisy qualitative judgments to tractable probabilistic inference, and has become the foundation of much of discrete choice modelling.

2.4.2 Pairwise Comparisons: Logistic and Probit Links

In the pairwise case, the probability of preferring i over j depends only on the difference in their utilities. Two classic special cases are:

Bradley–Terry / Luce (logistic). If utility differences follow a logistic distribution (equivalently, each ε is Gumbel), then

$$p(i \succ j \mid \theta) \;=\; \frac{\exp(\theta_i)}{\exp(\theta_i) + \exp(\theta_j)} \;=\; \sigma(\theta_i - \theta_j),$$

the logistic sigmoid of the utility gap [20]. Large margins result in near-certain preferences, while small gaps produce near-random outcomes, which indicates uncertainty in choice.

Thurstone / Mosteller (probit). If noise terms are Gaussian, then $U_i - U_j$ is normally distributed and

$$p(i \succ j \mid \theta) \;=\; \Phi\!\left(\frac{\theta_i - \theta_j}{\sigma}\right),$$

where Φ is the standard normal CDF [18].

Both formulations emphasise the same idea: what matters is the *margin* $\theta_i - \theta_j$, translated into a probability of preference by a smooth link function. This margin-based form makes estimation convenient, since θ can be fitted by maximising the likelihood of observed comparisons with standard optimisers.

2.4.3 Beyond Pairs: k-wise Choices and Rankings

Many applications involve more complex signals than pairs. Users might pick one item from a shortlist, or experts might provide partial or full rankings of interventions. The Plackett–Luce (PL) model extends the Gumbel-based RUM to these settings [21]. For a choice set S, the probability of selecting $i \in S$ is

$$p(i \mid S, \theta) \;=\; \frac{\exp(\theta_i)}{\sum_{j \in S} \exp(\theta_j)},$$

a softmax over utilities. Full rankings are decomposed into a sequence of such choices: pick the top item, then the best among the remaining, and so forth. This decomposition yields a tractable likelihood for ranking data, which explains the model's long-standing use in economics, psychology, and modern machine learning.

2.4.4 Where Preferences Already Matter

Preference learning is central in several areas of machine learning. In recommender systems, relative clicks and implicit choices often substitute for calibrated ratings, with discrete choice models and their neural extensions turning such signals into ranked recommendations [20]. In reinforcement learning from human feedback (RLHF), annotators compare trajectories or responses, and models are trained to prefer those judged better [22]. These examples illustrate a common principle: even when precise labels are unavailable, relative comparisons can provide strong information, as long as we have models that connect them to latent values.

In the causal setting, interventions may yield outcomes that are hard to calibrate numerically but are easily comparable. Treating these comparisons as data opens a path to structure learning from qualitative feedback, a path we develop in Chapter 3.

2.5 Normalising Flows

Normalising Flows (NFs) are a family of generative models designed to combine two properties that are rarely achieved together: the flexibility of neural networks and the exact tractability of probabilistic models. The main idea is to start from a simple, well-understood distribution, such as a multivariate Gaussian, and then transform it through a sequence of *invertible, differentiable mappings*. If these mappings are chosen carefully, the transformed distribution can approximate arbitrarily complex data, while still allowing us to compute exact likelihoods via the change-of-variables formula.

This dual property, expressive modelling with exact probabilities, sets NFs apart from other modern generative approaches such as variational autoencoders or GANs, which typically trade tractability for flexibility, or vice versa [23, 24].

2.5.1 Intuition: Reshaping Simple Distributions

A simple way to picture NFs is through geometry. Imagine samples drawn from a two-dimensional Gaussian: plotted, they form a circular cloud. Now apply a series of smooth, reversible transformations that stretch, bend, and twist the cloud. After enough steps, the points could form almost any target distribution: spirals in finance returns, multimodal clusters in biology, or the irregular landscapes of latent utilities in preference data.

What makes NFs powerful is that these transformations are not arbitrary. Each one is invertible and differentiable, which means we can move both *forward* (sample from the model) and *backward* (evaluate exact densities). In practice, the mappings are parameterised by neural networks, making NFs interpretable as invertible neural networks for probability distributions.

2.5.2 Mathematical Formulation

Formally, let $z_0 \sim p_0(z_0)$ be drawn from a simple base distribution, typically $\mathcal{N}(0, I)$. A sequence of K invertible transformations

$$z_k = f_k(z_{k-1}), \quad k = 1, \dots, K,$$

yields the final output $x = z_K$. The change-of-variables formula gives the resulting density as

$$p(x) = p_0(z_0) \prod_{k=1}^{K} \left| \det \frac{\partial f_k}{\partial z_{k-1}} \right|^{-1}.$$

Each Jacobian determinant quantifies how the transformation stretches or compresses space locally. Flow architectures are designed so that these determinants are efficient to compute, which keeps exact likelihoods tractable even in high dimensions. This is what allows NFs to be both flexible and statistically rigorous: unlike most deep generative models, they never lose track of the probability mass they reshape.

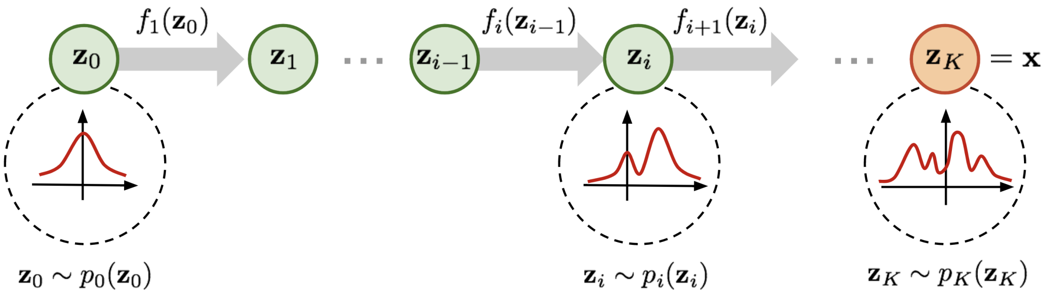

$f_1(\mathbf{z}_0)$
$\mathbf{z}_0$
$\mathbf{z}_1$
$\mathbf{z}_{i-1}$
$f_i(\mathbf{z}_{i-1})$
$\mathbf{z}_i$
$f_{i+1}(\mathbf{z}_i)$
$\mathbf{z}_K = \mathbf{x}$
$\mathbf{z}_0 \sim p_0(\mathbf{z}_0)$
$\mathbf{z}_i \sim p_i(\mathbf{z}_i)$
$\mathbf{z}_K \sim p_K(\mathbf{z}_K)$

Figure 2.6: Illustration of a normalising flow. A simple base distribution $z_0 \sim p_0(z_0)$ is transformed through a sequence of invertible mappings $f_1, \ldots, f_K$ into a complex target distribution $p(x)$.

2.5.3 Preferential Normalising Flows (PNFs)

Preferential Normalising Flows (PNFs) were introduced only recently [25], as one of the first attempts to unify preference learning with deep generative modelling. The motivation is simple: Random Utility Models provide a clean probabilistic framework for preferences but remain limited in flexibility, while normalising flows can capture highly complex distributions yet do not, by themselves, explain comparative data. PNFs combine these strengths by learning flexible latent utility distributions that are directly aligned with observed preferences.

The construction is straightforward. Each item or intervention x is mapped through a flow transformation into a latent utility $u \sim p_\phi(u \mid x)$, with parameters ϕ learned from data. These utilities are never observed directly; instead, we see comparisons. A logistic random utility model links the latent utilities to preference outcomes:

$$p(i \succ j) \;=\; \sigma\!\left(\tfrac{u_i - u_j}{s}\right),$$

where s is a noise scale and $\sigma(\cdot)$ is the sigmoid function.

This design makes PNFs a natural drop-in replacement for classical discrete choice models. Unlike the Bradley–Terry or Thurstone formulations, PNFs can represent non-linear, multi-modal, and highly structured utility landscapes, while still producing probabilistic likelihoods

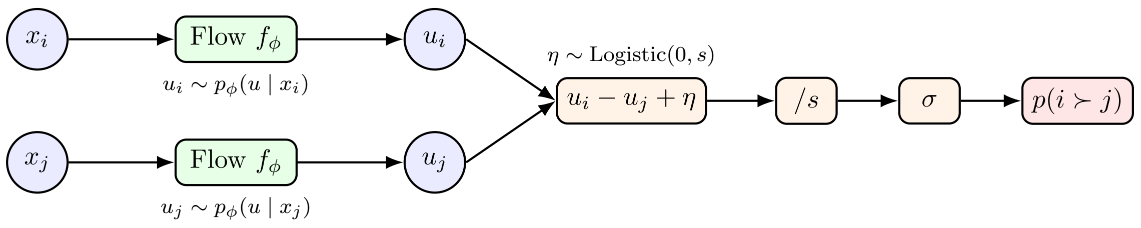

x_i
Flow f_ϕ
$u_i \sim p_\phi(u \mid x_i)$
u_i
x_j
Flow f_ϕ
$u_j \sim p_\phi(u \mid x_j)$
u_j
$\eta \sim \text{Logistic}(0, s)$
$u_i - u_j + \eta$
$/s$
σ
$p(i \succ j)$

Figure 2.7: A Preferential Normalising Flow (PNF). Items are mapped to latent utilities u via a flow, and preferences are modelled by a logistic random utility model.

for comparisons. In other words, even qualitative, noisy preferences can be turned into rigorous statistical inference, but now with the flexibility of deep generative modelling.

In this thesis, PNFs serve as the modelling backbone. They provide calibrated uncertainty over latent utilities, which propagates into Bayesian updates of causal structures and drives the acquisition strategies that decide which interventions to query. Without PNFs, preference-based causal discovery would be restricted to overly simplistic assumptions about utilities. With them, it becomes possible to learn from comparative feedback in complex, data-limited environments.

2.6 Related Work and Positioning within Literature

Causal discovery has been studied for decades, producing methods that differ in assumptions, scalability, and data needs. Broadly, prior work falls into three families. First are *constraint-based* and *score-based* algorithms, which infer graphs from observational data by testing conditional independences or searching for high-scoring structures. A second line rethinks causal discovery as *continuous optimisation*, enabling gradient-based methods. More recent approaches connect causal discovery to *Bayesian optimisation* and *generative modelling*, treating interventions as sequential decisions or using flexible density estimators such as normalising flows.

2.6.1 Constraint-Based, Score-Based, and Differentiable Approaches

The earliest causal discovery algorithms work directly on observational data, assuming access to conditional independences among variables. *Constraint-based* methods such as the PC algorithm [26] and Fast Causal Inference (FCI) [27] build graphs by performing a sequence of independence tests, eliminating edges that contradict the observed dependencies. Under assumptions of faithfulness and infinite samples, these methods are provably consistent. In practice, however, they are fragile: statistical tests become unreliable in high-dimensional settings, and small errors can cascade into large structural mistakes [28].

Score-based approaches instead assign each candidate graph a score reflecting how well it explains the data. Greedy Equivalence Search (GES) [29] is a canonical example: it searches through equivalence classes of DAGs using the Bayesian Information Criterion (BIC) or related scores. These methods are often more robust than pure constraint-based ones, but they face severe combinatorial complexity: the number of possible DAGs grows super-exponentially with the number of variables, making exhaustive search infeasible beyond small graphs [30].

To address these bottlenecks, *differentiable causal discovery* methods recast the acyclicity constraint in continuous terms. NOTEARS [31] introduced a smooth characterisation of acyclicity that enabled direct optimisation of a loss function over weighted adjacency matrices. This innovation allowed causal discovery to benefit from gradient-based optimisation and neural architectures, dramatically improving scalability. Subsequent extensions generalised the framework to nonlinear structural equations [32], time series [33], and latent variable models [34].

Together, these three families of methods define the classical backbone of causal discovery. Yet they share some limitations: they rely heavily on abundant, clean observational data; they often struggle with scalability as dimensionality increases; and crucially, they provide no natural mechanism for incorporating qualitative signals such as preferences. These gaps motivate the Bayesian, active, and flow-based approaches reviewed in the following subsections.

2.6.2 Bayesian Optimisation for Structure Learning

A second strand of research approaches causal discovery through *active experimentation.* Rather than relying exclusively on observational data, the learner can design interventions that are expected to be maximally informative about the underlying structure. This connects naturally to *Bayesian optimisation* (BO), where the aim is to sequentially query an unknown function while balancing exploration and exploitation. In the causal case, the goal is not to optimise a function value, but to recover the hidden graph structure as efficiently as possible.

Early work by Tong and Koller [11], as well as He and Geng [12] introduced information-theoretic acquisition rules such as *edge entropy*, selecting interventions on variables where the presence or absence of edges was most uncertain. Later studies developed more principled formulations based on *expected information gain* (EIG), which quantifies how much an intervention outcome is expected to reduce posterior uncertainty over graphs [14, 15, 16]. This view makes causal discovery an explicitly sequential decision process: the learner intervenes, observes outcomes, updates its Bayesian posterior over graphs, and then decides where to intervene next.

More recently, Aghaei et al. [13] extended this framework to preference-based feedback, showing that comparative judgments can be incorporated into acquisition functions such as *Preference Information Gain* (PIG). This extension is important because in many domains, such as medicine, behavioural sciences, or recommender systems, precise numerical outcomes are often noisy, costly, or ill-defined, whereas relative preferences are easier and more natural to obtain. Using preferences as the basis for intervention design allows causal discovery to proceed even when absolute measurements are unavailable.

These studies show that Bayesian optimisation offers a logical way to guide interventions and resolve structural ambiguities, but they also reveal its limits. Most existing approaches assume numerical outcomes and focus on small or controlled environments. This gap motivates the framework developed in this thesis, where Bayesian acquisition is combined with preference-based models to enable scalable, uncertainty-aware causal discovery from qualitative data.

2.6.3 Flow-Based Models

Normalising flows (NFs) have become especially influential because they combine two features that rarely align: flexible density modelling and exact likelihood evaluation [23, 35]. Starting from a simple base distribution and applying a sequence of invertible transformations, NFs can represent multimodal and intricate data distributions while remaining mathematically tractable.

These properties have made flows attractive in causal inference. Khemakhem et al. [36], for example, used flows to achieve identifiable representation learning, while Zheng et al. [32] applied them to model post-intervention distributions. In Bayesian structure learning, flows provide a richer alternative to restrictive parametric families such as Gaussians, enabling conditional distributions and intervention effects to be captured more realistically.

More recently, researchers have started exploring flows for preference learning. Here, the idea is to model distributions over latent utilities, giving a probabilistic account of noisy or qualitative feedback. Applications so far include discrete choice modelling and human preference prediction, though their role in causal discovery has received little attention.

This gap is where PNFs fit. Their aim is to use the expressiveness of flows to model the latent utilities behind comparative judgments. In doing so, PNFs open the door to causal discovery in domains where outcomes are not measured on a numerical scale but are instead expressed as preferences. They form a natural link between flow-based generative modelling and active causal discovery, and provide the modelling foundation for the framework introduced in this thesis.

2.6.4 Positioning within the State of the Art

The strands reviewed above capture both the strengths and the shortcomings of current approaches to causal discovery. Constraint- and score-based algorithms remain foundational, yet they scale poorly and rely on strong assumptions about data quality. Differentiable methods such as NOTEARS [31] bring modern optimisation techniques into the picture, but often at the cost of reliable uncertainty quantification. Bayesian optimisation introduces a principled way to guide interventions, though most existing work assume clean numerical outcomes. Flow-based models, meanwhile, have opened up new possibilities for flexible density estimation, but their use in causal discovery, and particularly in preferential settings, remains limited.

This thesis tackles that gap with *Preferential Causal Bayesian Optimisation (PCBO)*. The main idea is to make causal discovery work when the available feedback is comparative rather than numerical. At the centre of the framework is the *Preferential Normalising Flow (PNF)*, which combines the flexibility of flow-based generative models with the structure of random utility models, so that latent utilities can be inferred directly from pairwise or k-wise comparisons.

Within PCBO, PNFs serve two roles. They capture complex, possibly multimodal patterns in preference data, and they provide well-calibrated uncertainty that carries through to the graph structure. When paired with acquisition strategies such as Preference Information Gain (PIG) and Expected Edge Information Gain (EEIG), they allow interventions to be guided not only by what is observed, but by where uncertainty is most likely to shrink.

The result is a framework that brings together generative modelling, preference learning, and causal inference. It shows that causal discovery can remain feasible (and efficient) even when data are scarce, noisy, or only available in comparative form, which places PCBO as a step towards methods that are both flexible in their modelling and realistic in their assumptions.

Chapter 3

Theoretical Framework

3.1 Setting and Assumptions

The background in Chapter 2 highlighted two persistent challenges in causal discovery. First, classical Bayesian approaches struggle with scalability: exact posterior updates quickly become intractable as the number of variables grows. Second, active learning and causal Bayesian optimisation (CBO) methods typically assume that interventions produce numerical outcomes, which are not always available or reliable in practice. In many domains – medicine, behavioural science, recommender systems – feedback is expressed comparatively: one outcome is judged better than another, without precise calibration.

This motivates the development of *Preferential Causal Bayesian Optimisation* (PCBO): a Bayesian framework that learns causal structure from preference feedback. Conceptually, PCBO couples two views of the problem: structural uncertainty at the graph level and noisy latent utilities at the preference level. Comparative data thus play the same role as numerical outcomes in classical approaches, but with models designed to capture their qualitative nature.

Modelling assumptions. Throughout the chapter we work under the following assumptions, which will be used in the derivations:

- *Local linear–Gaussian conditionals for structure scoring.* For each node X_i, given a candidate parent set $\mathrm{Pa}(X_i)$, $X_i = \beta_i^\top \mathrm{Pa}(X_i) + \epsilon_i$ with $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$. This yields closed-form marginal likelihoods (BGe) for parent-set comparison.

- *Pairwise preference noise.* Preferences arise from latent utilities with logistic noise of scale $s > 0$; for two interventions a, a' with utilities u, u', $p(a \succ a') = \sigma\big((u - u')/s\big)$. Noise terms are i.i.d. and independent of graph noise.

- *Intervention model.* We allow hard interventions $\mathrm{do}(X_j = x)$ as well as soft interventions that replace the conditional of X_j; intervention assignments are under the learner's control, and preference noise is independent of the intervention choice given utilities.

- *Sparsity via bounded indegree.* Graphs are assumed sparse with a fixed maximum indegree

$k_{\max}$ (and an edge-sparsity prior). This regularises the local search over parent sets and supports the scalability claims later on.

Scope of the chapter. We now formalise (i) a Bayesian target over structures, (ii) a factorised approximation that separates local parent posteriors from a flexible utility model, and (iii) information-theoretic criteria for selecting interventions under preference data. Implementation choices and the end-to-end loop appear in Chapter 4.

3.2 Bayesian Formulation

Having introduced the problem setting, we now turn to the Bayesian perspective that backs up our approach. The central idea is to treat both the causal structure and the preference models as random variables, and to infer their joint posterior distribution given observed data. This offers a conceptually elegant way of capturing uncertainty: rather than committing to a single "best" graph or utility model, we maintain a distribution over many possibilities and update it as new preferences arrive.

3.2.1 The Ideal Joint Posterior

Formally, let G denote the causal graph, f the parameters of the preference model (e.g. a normalising flow mapping interventions to latent utilities), $\mathcal{D}$ the observed preference data, and $\mathcal{R}$ prior assumptions. The ideal Bayesian formulation seeks to compute the joint posterior

$$p(G, f \mid \mathcal{D}, \mathcal{R}) \;\propto\; p(\mathcal{D} \mid G, f)\, p(G \mid \mathcal{R})\, p(f \mid \mathcal{R}),$$

where graph priors and flow priors are assumed independent given $\mathcal{R}$.

This distribution encodes all information available to the learner:

- The *likelihood* $p(\mathcal{D} \mid G, f)$ captures how well the observed preferences are explained by a given causal graph and utility model.
- The *graph prior* $p(G \mid \mathcal{R})$ reflects structural assumptions such as sparsity or symmetry.
- The *flow prior* $p(f \mid \mathcal{R})$ regularises the expressiveness of the normalising flow, preventing overfitting to noisy preference data.

In principle, maintaining this joint posterior would allow fully Bayesian causal discovery: we could marginalise over flow parameters to obtain posterior edge probabilities, or marginalise over graphs to obtain posterior predictive distributions for preferences.

Unfortunately, this ideal formulation is *infeasible* in practice. There are two main reasons:

1. The graph space grows super-exponentially with the number of variables d. Even for modest systems, enumerating all possible DAGs is computationally impossible.
2. The flow parameters f typically involve deep neural networks with millions of parameters. Integrating over this high-dimensional space is well beyond the reach of exact Bayesian inference.

As a result, directly computing or sampling from $p(G, f \mid \mathcal{D}, \mathcal{R})$ is unattainable beyond toy settings. This motivates the development of approximations that preserve the Bayesian spirit while enabling practical computation.

3.2.2 A Factorised Approximation: Parent Posteriors and PNFs

Since the joint posterior $p(G, f \mid \mathcal{D}, \mathcal{R})$ is far too complex to work with directly, we adopt a structured approximation that decouples structure learning from preference modelling. The intuition is that causal structure and utilities play complementary roles: the graph G encodes how variables influence one another, while the flow model f explains how interventions translate into noisy preferences. Rather than attempting to update both jointly, we use a mean-field factorisation

$$p(G, f \mid \mathcal{D}, \mathcal{R}) \approx q(G)\, q(f),$$

where $q(G)$ and $q(f)$ are tractable approximations to the marginal posteriors over graphs and flows. This kind of decoupling is standard in variational inference: it breaks strict dependencies to keep inference tractable, while still allowing meaningful uncertainty estimates. In PCBO, the coupling between structure and preferences is reintroduced at the acquisition stage, where preference outcomes guide interventions towards reducing structural uncertainty.

At the *graph level*, inference is local. Each node X_i is associated with a posterior distribution over its possible parent sets $\mathrm{Pa}(X_i)$, often called the *parent posterior*. This reduces the combinatorial problem of reasoning over entire DAGs into a collection of smaller problems, one for each variable. The parent posterior can be updated efficiently under linear–Gaussian assumptions, and structural regularisers (e.g. sparsity-promoting priors, maximum indegree $k_{\max}$) ensure tractability even as the number of variables grows.

At the *utility level*, preferences are modelled using *Preferential Normalising Flows (PNFs)*. The flow f maps interventions into a latent utility space and connect them to observed pairwise comparisons via a logistic noise model. The flexibility of flows allows utility distributions to adapt to nonlinear or multimodal patterns, while still producing exact likelihoods that can be integrated into Bayesian updates.

This approximation sacrifices the elegance of a fully joint posterior, but gains scalability and interpretability. It preserves Bayesian reasoning at the level of edges and utilities, while making inference feasible. The following sections detail each component in turn, beginning with the parent posterior for local structure learning.

3.3 Parent Posterior for Local Structure Learning

With the factorised approximation in place, we turn first to the graph-level component: the *parent posterior*. The main idea is to avoid reasoning over the full space of DAGs, which is combinatorially explosive, and instead decompose the problem into local subproblems. For each node X_i, we maintain a posterior distribution over its possible parent sets $\mathrm{Pa}(X_i)$. This local perspective not only reduces the combinatorial complexity of structure learning but also yields interpretable posterior quantities such as the probability of individual edges.

This section develops the idea in three steps. We begin with the linear–Gaussian assumption that makes closed-form updates possible, then derive the Bayesian update equations for parent sets, and finally interpret the resulting edge posteriors as the quantities that guide preference-driven interventions.

3.3.1 Local Linear–Gaussian Assumption

In the Bayesian network setting, each variable is modelled conditional on its parents. To keep inference tractable, we assume that these conditional distributions are linear–Gaussian. For a variable X_i with candidate parents $\mathrm{Pa}(X_i)$,

$$X_i = \beta_i^\top \mathrm{Pa}(X_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_i^2).$$

Regression weights and noise variance are given conjugate Normal–Inverse-Gamma priors. This choice is mathematically convenient and will allow closed-form Bayesian updates, which we detail in the following subsection.

3.3.2 Bayesian Update Equations

Formally, the posterior over parent sets is defined as

$$p(\mathrm{Pa}(X_i) \mid D) \;\propto\; p(D \mid \mathrm{Pa}(X_i))\, p(\mathrm{Pa}(X_i)),$$

where $p(D \mid \mathrm{Pa}(X_i))$ is the marginal likelihood under the candidate parent set and $p(\mathrm{Pa}(X_i))$ encodes prior beliefs, such as sparsity or bounded indegree.

Under the linear–Gaussian setting introduced above, the likelihood term admits a closed form known as the *Bayesian Gaussian equivalent* (BGe) score [8], which enables direct comparison of alternative parent sets without sampling. We adopt sparsity-promoting priors, for example a Bernoulli prior on candidate edges together with a hard cap on indegree $k_{\max}$, which penalises dense structures and ensures tractable search.

This local Bayesian update forms the backbone of the parent posterior in PCBO: it translates data into edge-level probabilities that later guide preference-driven acquisition.

3.3.3 Edge Posterior Interpretation

Bayesian structure learning does not yield a single "best" graph, but a posterior distribution over possible structures. From this, edge posteriors (probabilities that a directed link $X_j \to X_i$ exists) can be derived by marginalising over all parent sets that contain the edge:

$$p(X_j \to X_i \mid D) = \sum_{\mathrm{Pa}(X_i) \ni X_j} p(\mathrm{Pa}(X_i) \mid D).$$

This local computation makes edge posteriors both interpretable and tractable. In practice they serve two purposes. First, they provide a confidence measure: edges with posterior mass close to one can be regarded as strong discoveries, while those near zero are likely spurious.

Second, they allow us to rank edges by uncertainty, which is especially useful when designing acquisition functions for active learning.

Conceptually, edge posteriors also provide the bridge to point-estimate graphs: in later chapters we will see how a greedy DAG projection can be applied to these marginals to produce an acyclic estimate of the global structure.

In short, edge posteriors translate local Bayesian updates into a structural measure that is both actionable and scalable. Within PCBO they form the link between Bayesian updating and preference-driven acquisition, guiding where interventions should be targeted next.

3.4 Preference Modelling with Preferential Normalising Flows

We now turn to the second component of the factorised approximation: modelling the noisy, comparative feedback that links interventions to latent utilities. Whereas parent posteriors capture structural uncertainty at the graph level, here the task is to provide a likelihood model flexible enough to handle the irregularities of preference data.

3.4.1 From Exponential RUM Likelihood to Flows

As discussed in Chapter 2, classical random utility models such as the Plackett–Luce model assume utilities of the form

$$U_i = \theta_i + \varepsilon_i, \quad \varepsilon_i \sim \text{Gumbel},$$

which yield closed-form logistic or softmax likelihoods for comparisons. These models are elegant and interpretable, but rigid: both the functional form of the utility and the distribution of noise are fixed in advance.

Preferential Normalising Flows (PNFs) generalise this picture. Instead of assuming a parametric utility distribution, they introduce an invertible mapping

$$u = f_\phi(z), \quad z \sim p_0(z),$$

where p_0 is a simple base distribution (typically Gaussian) and f_ϕ is a flow transformation parameterised by neural networks. This construction retains exact log-densities while allowing the utility space to bend into complex, nonlinear, or even multimodal shapes.

Within PCBO, this means that each intervention outcome x is mapped to a latent utility u through f_ϕ, and comparisons are evaluated at the utility level. The result is a model that keeps the clean logistic link to observed comparisons, but endows it with the flexibility needed to capture realistic, noisy judgments. In practice, these flow-based likelihoods supply exactly the quantities required by both Bayesian graph updates and the information-theoretic acquisition criteria introduced later in this chapter.

3.4.2 Pairwise Preference Likelihoods with Flows

In this thesis we restrict attention to pairwise comparisons, which are the most common and practically feasible form of feedback. Formally, given two interventions a and a', the flow maps them into latent utilities $u = f_\phi(x)$ and $u' = f_\phi(x')$. A preference observation is then generated by adding independent logistic noise terms,

$$(a, u) \succ (a', u') \quad \Leftrightarrow \quad u + \eta > u' + \eta', \quad \eta, \eta' \sim \text{Logistic}(0, s),$$

where $s > 0$ is a scale parameter controlling the sharpness of preferences. In practice s can be treated as fixed or learnable, and the independence assumption ensures that noise does not introduce spurious correlations across comparisons.

Marginalising over η, η' yields the familiar logistic likelihood

$$p\big((a, u) \succ (a', u') \mid f_\phi, s\big) = \sigma\Big(\tfrac{u-u'}{s}\Big).$$

This construction mirrors the exponential RUM but replaces its rigid linear utilities with the flexibility of flows: utilities can now represent highly nonlinear or multimodal patterns, while still retaining a closed-form preference likelihood. Importantly, utilities are identifiable only up to monotone transformations, since any monotone rescaling leaves pairwise preferences unchanged. By fixing the base distribution of the flow, we anchor the utility space and make inference well-defined.

In PCBO, these flow-based likelihoods serve as the interface between data and inference: they provide the factors that update parent posteriors, and they supply the predictive distributions that drive acquisition functions such as Preference Information Gain (PIG) and Expected Edge Information Gain (EEIG).

3.4.3 Flow Architectures for Preference Learning

Normalising flows are not a single model but a family of architectures, each designed to balance three needs: expressiveness, tractable Jacobians, and stable training. In the preference setting, these design choices matter because the flow must map intervention features into utilities in a way that is both flexible and computationally efficient. Here we outline three representative families that illustrate the spectrum of approaches.

RealNVP. Real-valued Non-Volume Preserving (RealNVP) flows [37] build transformations through *affine couplings*. The input is split into two parts: one remains unchanged, while the other is scaled and shifted by a function of the first. This triangular structure makes the Jacobian determinant trivial to compute, ensuring exact likelihoods at low cost. RealNVP flows thus prioritise simplicity and efficiency, providing a reliable baseline for modelling preferences.

Residual Flows. Residual flows [38] introduce transformations of the form

$$z_{k+1} = z_k + g_\theta(z_k),$$

with g_θ constrained so that the mapping remains invertible. This residual form allows for deeper and more flexible transformations than affine couplings alone, while still guaranteeing stability. They are particularly suited to modelling curved or highly non-linear utility landscapes.

Neural Spline Flows. Neural Spline Flows (NSFs) [39] replace affine transformations with monotonic splines. Each dimension is warped by a smooth, piecewise rational-quadratic spline, whose parameters are given by neural networks. This yields highly expressive, shape-adaptive transformations that can capture multimodality and heavy tails in the latent utility distribution.

Architectural design choices in flows hence correspond to different trade-offs: efficiency and stability in RealNVP, expressivity and depth in Residual Flows, and fine-grained flexibility in Neural Spline Flows. In this thesis we treat them as interchangeable building blocks for PNFs, with concrete instantiations deferred to Chapter 4.

3.4.4 Training Procedure

Training a Preferential Normalising Flow (PNF) consists of adjusting its parameters ϕ so that the induced distribution over latent utilities is consistent with the observed preference data $\mathcal{D}$. For a single comparison $i \succ j$, the likelihood under the flow is

$$p(i \succ j \mid \phi) = \int \sigma\Big(\tfrac{u_i - u_j}{s}\Big)\, p_\phi(u_i \mid x_i)\, p_\phi(u_j \mid x_j)\, du_i\, du_j,$$

where $p_\phi(u \mid x)$ is the flow-induced density, $\sigma(\cdot)$ is the logistic sigmoid, and s is the noise scale.

This integral is generally intractable, since flows provide tractable densities for individual utilities but not for their differences. It is therefore estimated by Monte Carlo sampling: utilities are drawn from the flow, their differences are passed through the sigmoid, and the results are averaged. Aggregating across a dataset of pairwise comparisons leads to the log-likelihood objective

$$\mathcal{L}(\phi) = \sum_{(i \succ j) \in \mathcal{D}} \log p(i \succ j \mid \phi).$$

Maximising $\mathcal{L}(\phi)$ ensures that the learned flow warps the latent utility space so that preferences are reproduced in expectation. This provides the tractable likelihood terms required for Bayesian graph updates and acquisition scoring. Practical training strategies (optimiser choice, batching, regularisation) are discussed in Chapter 4. Conceptually, this step anchors the flow side of the factorised approximation introduced in Section 3.2.

3.5 Acquisition Functions

In Section 2.3, we introduced acquisition functions as information-theoretic tools for guiding interventions in causal discovery. Here, we refine the concept for the preference-based setting of PCBO. Rather than rederiving general principles, the focus is on how acquisition rules are specialised to handle preference data and how they integrate with the Bayesian framework developed in this thesis.

The central challenge is that in PCBO, outcomes are not numerical but comparative. Acquisition functions must therefore assess the informativeness of an intervention not by predicting exact values, but by anticipating how preference outcomes will shrink uncertainty in the causal graph. We highlight two instantiations that make this idea operational: Preference Information Gain (PIG) and Expected Edge Information Gain (EEIG).

3.5.1 Preference Information Gain (PIG) and Expected Edge Information Gain (EEIG)

In Section 2.3 we introduced *Expected Information Gain* (EIG) as a general principle for active learning: an intervention is informative if its possible outcomes are expected to reduce posterior uncertainty about the causal graph. In the preference-based setting, this principle applies even though outcomes are not numerical but comparative. Relative judgements still constrain which graphs are consistent with the data, and can therefore provide decisive evidence about structural ambiguities. Formally, the goal is to evaluate interventions by how much the distribution of their possible preference outcomes is expected to refine beliefs over graphs.

Preference Information Gain (PIG). PIG [13] adapts this principle to pairwise comparisons. For a candidate intervention a, the score is

$$\alpha_{\mathrm{PIG}}(a) = I(G; \mathrm{Pref}(a) \mid D),$$

the mutual information between the graph G and the next preference outcome. Intuitively, PIG measures how strongly observing the result of a would help discriminate between competing structural hypotheses. It is particularly effective when comparisons can collapse uncertainty between alternative edge directions.

Expected Edge Information Gain (EEIG). Since evaluating PIG over the full graph posterior is intractable, a practical alternative is to approximate at the edge level. EEIG measures the expected entropy reduction in individual edge marginals:

$$\alpha_{\mathrm{EEIG}}(a) = \sum_{(i,j)} \Big[H\big(p(G_{ij} \mid D)\big) - \mathbb{E}_{\mathrm{Pref}(a)} H\big(p(G_{ij} \mid D \cup \{\mathrm{Pref}(a)\})\big) \Big].$$

This sacrifices global consistency but scales efficiently and directs interventions toward the most uncertain links in the graph.

Approximate updates. Both PIG and EEIG are implemented via a one-step "virtual update": for each possible preference outcome, the posterior is hypothetically updated, the resulting entropy change is computed, and the scores are averaged. This approximation keeps the acquisition principle faithful to the Bayesian spirit while remaining computationally feasible.

In summary, PIG offers a principled but costly criterion, while EEIG provides a scalable heuristic. Together they give PCBO complementary strategies for selecting interventions under preference feedback.

3.5.2 Balancing Exploration and Exploitation

Information-theoretic acquisition functions specify how to measure informativeness, but they do not resolve a strategic dilemma: should the learner probe areas of the graph that remain uncertain, or reinforce what already seems likely? This tension between *exploration* and *exploitation*, familiar from reinforcement learning and bandit problems [40], also governs causal discovery from preferences.

Exploration. Exploration means testing parts of the structure that are still ambiguous, even if the expected gain is modest. For example, if most edges are well established but a few remain uncertain, exploring them is essential for recovering a complete and reliable graph.

Exploitation. Exploitation directs interventions to places where information gain is already expected to be high. This consolidates strong hypotheses quickly: if evidence strongly suggests $X \to Y$, intervening on X can confirm that link with high probability and accelerate convergence.

The balance. Neither extreme is sufficient. Pure exploration wastes queries, while pure exploitation risks locking onto the wrong structure too soon. In practice, PIG often acts as an exploratory criterion, collapsing major uncertainties, whereas edge-level heuristics like EEIG lean toward targeted exploitation by resolving specific, contested links. Together, they embody the balance required for efficient discovery.

In this thesis, the balance is not enforced by a hard-coded rule but emerges naturally from integrating these two criteria. PIG provides the drive for exploration, while EEIG ensures exploitation of local ambiguities. The precise mechanism for weighting them is discussed in Chapter 4.

3.6 From Building Blocks to PCBO

The pieces introduced in this chapter are designed to fit together. Preferential flows provide a likelihood for comparisons, local posteriors translate that likelihood into structural uncertainty, and information-theoretic scores identify where the next comparison should be made. Individually, each component addresses a distinct modelling challenge; collectively, they form the basis of an iterative algorithm that can turn pairwise preferences into causal knowledge.

While the mechanics of this algorithm are the focus of the next chapter, one point is worth noting here. Under the assumptions made so far — linear–Gaussian conditionals, logistic preference noise with scale s, and a sparsity prior on parent sets — the Bayesian updates are consistent. That is, as more informative comparisons accumulate, edge posteriors are expected to concentrate on the true structure. This theoretical reassurance motivates the methodological design choices that follow, where we show how these components are orchestrated into a practical loop.

Chapter 4

Methodology: The PCBO Algorithm

4.1 Overview of the PCBO Cycle

At its core, Preferential Causal Bayesian Optimisation (PCBO) is an iterative process. Each round begins with what the system already knows, updates its beliefs in light of new comparisons, and then decides which interventions to carry out next. The design principle is simple: when data is scarce and costly, every new query must be chosen carefully to reduce uncertainty as efficiently as possible.

The loop has four main stages:

1. *Preference update.* The flow-based learner is updated using all previously collected comparisons, refining the latent utility model.

2. *Posterior update.* Local Bayesian posteriors over parent sets are refreshed to incorporate new interventional outcomes, maintaining uncertainty over the causal graph.

3. *Acquisition.* An intervention pair is selected according to an information criterion (PIG and EEIG), balancing structural discovery and utility refinement.

4. *Execution.* The chosen interventions are carried out, their outcomes compared, and the new preference added back into the dataset.

The cycle then repeats, progressively sharpening both the causal graph and the utility function. Figure 4.1 shows how these components fit together, while Algorithm 1 provides a step-by-step pseudocode version.

Across iterations, the state is carried forward in several forms: the accumulated dataset of preference pairs, cached statistics for local posteriors, current edge probabilities, and the parameters of the preferential flow. To keep evaluation consistent with global acyclicity, the edge marginals are projected to a DAG at the end of each iteration. This projection is not used for acquisition itself, which relies directly on marginals, but it ensures that evaluation metrics such as structural Hamming distance are always computed against a valid global structure.

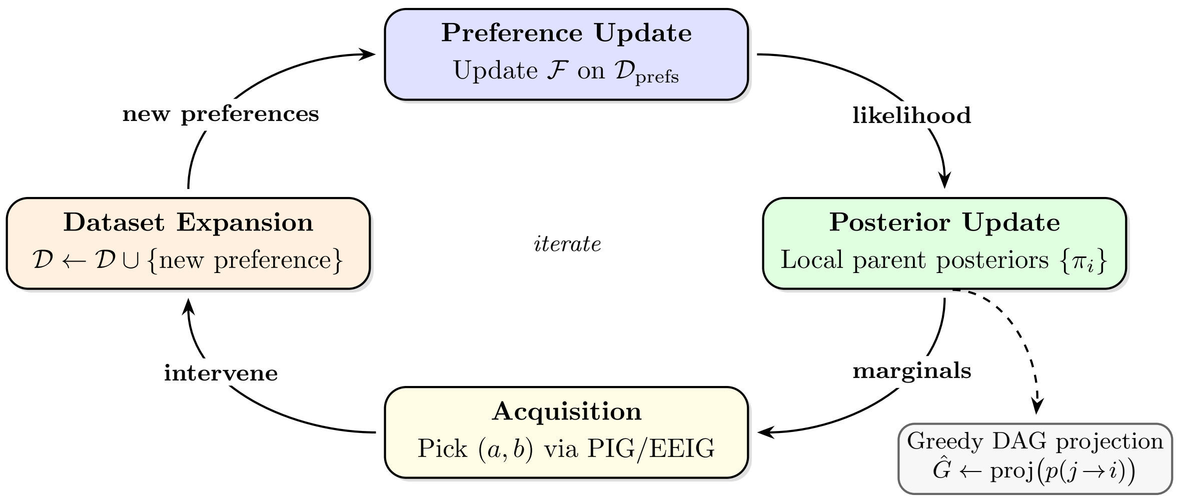

Preference Update
Update $\mathcal{F}$ on $\mathcal{D}_{\text{prefs}}$
likelihood
Posterior Update
Local parent posteriors $\{\pi_i\}$
marginals
Greedy DAG projection
$\hat{G} \leftarrow \text{proj}(p(j \to i))$
Acquisition
Pick (a, b) via PIG/EEIG
intervene
Dataset Expansion
$\mathcal{D} \leftarrow \mathcal{D} \cup \{\text{new preference}\}$
new preferences
iterate

Figure 4.1: PCBO Loop. Each round updates the preference model $\mathcal{F}$, updates local posteriors $\{\pi_i\}$, applies acquisition to select an intervention pair, and executes it to expand the dataset. Edge marginals are periodically projected to a DAG $\hat{G}$.

Algorithm 1 Preferential Causal Bayesian Optimisation (PCBO)

```
1:  **Init:** build local posteriors $\{\pi_i\}$; warm up preference flow $\mathcal{F}$; set $\mathcal{D}_{\text{prefs}} \leftarrow \emptyset$.
2:  **for** $t = 1, \ldots, T$ **do**
3:      *Preference update:* if enough data, train $\mathcal{F}$ on $\mathcal{D}_{\text{prefs}}$.
4:      *Candidate pool:* sample single-node interventions.
5:      *Anchor:* pick $a$ by expected edge-entropy reduction (with light exploration).
6:      *Opponent:* pick $b \neq a$ by $\alpha \cdot \text{PIG}(\mathcal{F}; a, b) + (1-\alpha) \cdot \text{EEIG}(a, b)$.
7:      *Execute & record:* apply $a$ and $b$, record preference, append to $\mathcal{D}_{\text{prefs}}$; add a few extra
    diverse comparisons vs. $a$.
8:      *Posterior update:* update local $\{\pi_i\}$ with new outcomes.
9:      *Monitoring:* form edge marginals $P$ and run greedy DAG projection to get $\hat{G}_t$.
10: **end for**
```

4.2 Graph Beliefs via Local Posteriors

Uncertainty over causal structure in PCBO is represented not by a single posterior over graphs, but by a family of *local parent posteriors*. For each node X_i, the algorithm maintains a distribution over possible parent sets, which is updated whenever new interventional outcomes are observed. This decomposition has two advantages. First, it makes inference tractable: evaluating the probability of a full graph is infeasible in high dimensions, but reasoning about one node at a time with bounded indegree is manageable. Second, it aligns naturally with intervention data: when an outcome is observed under a do-operator, only the conditional distribution of affected nodes needs to be updated, while unrelated local posteriors remain untouched.

Formally, let $\pi_i(S)$ denote the posterior probability that node X_i has parent set $S \subseteq \{1, \ldots, d\} \setminus \{i\}$. The global edge marginals $p(j \to i)$ are then recovered by summing over all parent sets containing j. Maintaining this collection of local objects ensures that structural uncertainty remains calibrated across rounds, while enabling targeted updates and efficient storage of sufficient statistics. Two variants of parent posteriors are used in PCBO: an exact scoring scheme for small graphs and a scalable approximation for larger ones.

4.2.1 Exact Local Scoring

When the number of variables is moderate, local posteriors can be computed exactly. This is done by enumerating all candidate parent sets, scoring each set with a closed-form marginal likelihood, and normalising across alternatives.

Concretely, the score used is a Bayesian Gaussian equivalent (BGe) style marginal likelihood, appropriate under the assumption of linear-Gaussian conditionals with conjugate priors. For a candidate parent set S of node X_i, let X_S denote the design matrix of parent variables and y the outcomes of X_i. The sufficient statistics $(X_S^\top X_S, X_S^\top y, y^\top y)$ are cached once and reused across candidate evaluations, which ensures memory grows only with the number of variables, not the number of samples. The marginal likelihood then takes the form

$$\ell(S) = \int p(y \mid X_S, \beta, \sigma^2)\, p(\beta, \sigma^2)\, d\beta\, d\sigma^2,$$

which admits a closed expression under a Normal–Inverse-Gamma prior. This enables efficient computation of

$$\pi_i(S) \propto \ell(S) \cdot p(S),$$

where $p(S)$ is the sparsity prior favouring smaller parent sets. Normalisation over all valid S yields the posterior distribution for node i.

In practice, this exact scheme is used for graphs with fewer than about ten nodes, where the number of candidate parent sets per node remains small. For each update, the caches of sufficient statistics are refreshed with new outcomes, and the closed-form scores are recomputed only for affected nodes, allowing exact Bayesian updates to remain practical in small graphs.

4.2.2 Scalable Variant: MCMC-Based Search

Exact local scoring becomes infeasible for larger structures: enumerating all 2^{d-1} possible parent sets for a node with d potential parents quickly overwhelms both memory and computation. To handle larger systems, PCBO replaces enumeration with a Markov chain Monte Carlo (MCMC) approximation.

Instead of summing over every subset, we draw samples from the posterior distribution over parent masks $S \subseteq \{1, \ldots, d\}$. Each mask specifies the candidate parents of a target node Y_i, and defines a regression model $Y_i = X_S\beta + \varepsilon$ with closed-form marginal likelihood. Aggregating these samples produces an empirical distribution over edges:

$$p(j \to i \mid \mathcal{D}) \;\approx\; \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}\{j \in S^{(m)}\}.$$

The sampler explores the space of masks through simple bit-flip proposals: an edge is toggled (added or removed) at each step, with acceptance probability proportional to the ratio of marginal likelihoods and subset priors. To avoid getting trapped in local modes, parallel tempering is employed: multiple chains run at different inverse temperatures τ, with occasional swaps to improve mixing when posteriors are sharply peaked. To remain efficient, each chain reuses cached sufficient statistics $(X^\top X, X^\top y, y^\top y)$ so that likelihoods are updated incrementally. Warm starts are also used: when new data are added, chains resume from their previous state rather than restarting.

After burn-in, the sampler produces a stream of masks. The frequency with which each potential edge appears across samples defines its marginal probability. This provides a calibrated posterior over incoming edges to Y_i, analogous to the exact setting but obtained through stochastic approximation rather than full enumeration.

In practice, the system switches to the scalable variant automatically for graphs with more than 10 nodes. This ensures that PCBO uses exact inference where possible, but remains feasible for larger graphs such as Erdős–Rényi networks with tens of nodes.

4.2.3 From Marginals to a Directed Acyclic Graph

Local posterior updates provide, for each node, a distribution over possible parent sets. Aggregating across these gives edge marginals $p(j \to i \mid \mathcal{D})$, a probability for every potential directed link. Taken in isolation, these marginals capture local uncertainty well, but their union does not necessarily form a globally valid graph: accepting all high-probability edges at once can introduce cycles. PCBO therefore *projects to a DAG at every iteration* to keep a coherent global picture as learning proceeds.

The principle is simple: edges are ordered by strength, then added one by one as long as they do not create a cycle. The outcome is a single graph estimate that is consistent with the marginal beliefs while guaranteeing acyclicity.

Formally, edges are ranked based on a logit score

$$w_{j\to i} = \log \frac{P_{ji}}{1 - P_{ji}},$$

which spreads values near $0, 1$ and avoids crowding around 0.5. To avoid overconfident early projections, scores are tempered before ranking:

$$\tilde{P} \;=\; \sigma\!\left(\frac{\operatorname{logit}(P)}{T}\right), \quad T \geq 1.$$

Edges are then processed in descending order of $w_{j\to i}$. The result is a DAG that maximises the total weight $\sum_{(j\to i)\in A} w_{j\to i}$ under acyclicity.

This projection is performed continuously throughout learning. In this way, downstream components (e.g. acquisition that needs a graph view) always see a fresh, acyclic structure. It also provides a stable object for visualisation and for tracking progress against ground truth as learning proceeds. The marginals themselves continue to capture finer uncertainty, while the projected DAG offers the clean summary needed for graph-level metrics.

4.3 Learning Utilities with Preferential Flows

Each observed outcome is mapped to a scalar *utility*; differences between utilities explain why one intervention is preferred over another. To keep this mapping flexible but trainable under small data, we use a normalising flow and read its log-density as the utility. A single scale parameter s controls how sharp or noisy preferences are, letting the model stay cautious when evidence is thin and decisive when it is not.

4.3.1 Model Recap and Parameterisation

Let $x \in \mathbb{R}^D$ denote features of an intervention outcome (optionally including a simple cost term). A normalising flow with parameters θ defines a density $p_\theta(x)$ via an invertible map f_θ. We take the latent utility to be the flow's log-density:

$$u_\theta(x) \;:=\; \log p_\theta(x).$$

Preferences follow a logistic random–utility model. For two outcomes x_a, x_b,

$$p(x_a \succ x_b) = \sigma\big(s\,(u_\theta(x_a) - u_\theta(x_b))\big),$$

where σ is the sigmoid and $s > 0$ is a learnable precision (inverse temperature): larger s yields clearer, less noisy choices; smaller s yields softer, noisier choices. We regularise s with a log-normal prior to keep it positive and to let the model adapt its confidence level as more evidence accumulates.

This parameterisation gives PCBO exactly what it needs: expressive utilities from the flow, and a calibrated precision (inverse temperature) s that adapts to the reliability of the collected comparisons.

4.3.2 Training Objective

Given the model above, learning reduces to maximising the likelihood of the observed comparisons. For each labelled pair (x_a, x_b, y) with $y = 1$ if x_a is preferred and $y = 0$ otherwise, the contribution to the log-likelihood is

$$\log \sigma\Big((2y-1)\,\tfrac{u_\theta(x_a)-u_\theta(x_b)}{s}\Big).$$

Summing over all N comparisons yields the objective

$$\ell(\theta, s) \;=\; \sum_{n=1}^{N} \log \sigma\Big((2y_n-1)\,\tfrac{u_\theta(x_{a,n})-u_\theta(x_{b,n})}{s}\Big).$$

In practice we do not optimise the pure likelihood alone. Following the idea of FS–MAP proposed in the PNF literature [25], we use a MAP-style variant that augments the objective with two small regularisation terms: a mild encouragement for the flow to place density around observed winners, and a prior on the noise scale. Formally, the training loss becomes:

$$\ell_{\text{aug}}(\theta, s) \;=\; \ell(\theta, s) \;+\; \lambda \sum_{\text{winners}} u_\theta(x) \;+\; \log p(s).$$

This does not change the core learning rule, but it stabilises optimisation and improves learning from small preference datasets.

4.3.3 Training Routine

The objective function is optimised with a stochastic gradient descent variant called Adam [41], which adapts the learning rate for each parameter, balancing speed with robustness to noisy gradients. It then backpropagates through both the flow and the logistic link. The noise scale parameter is updated with a slightly higher learning rate than the flow weights, which helps the model adapt its confidence level in step with the utility function. A cosine annealing schedule lowers the rate smoothly across epochs [42], avoiding sharp drops that could destabilise convergence.

Training runs for at most 50 epochs, with early stopping after ten rounds without improvement to prevent overfitting and reduce wasted computation. Several safeguards further improve stability. Gradients are clipped at norm 5.0 to avoid runaway updates, and if optimisation becomes unstable for multiple rounds, the flow is automatically reinitialised with a short synthetic warm-up before training resumes. These measures are what make it possible to keep flows trainable over several iterations.

4.3.4 Caching and Reuse

Since new preference data arrive gradually, the flow is never trained once and for all: it is refined round by round. To make this feasible, the implementation reuses information at three levels:

- *Warm starts.* Parameters are carried forward between rounds, so training continues from an already structured state instead of starting from scratch. This strategy avoids wasting queries on relearning coarse patterns and stabilises optimisation when data are still limited.
- *Mini–batching.* Comparisons are processed in small batches, letting forward passes of the flow serve several pairs at once. This keeps updates computationally tractable and allows stochastic gradients to approximate the full likelihood while retaining efficiency.
- *Cached evaluations.* Intermediate quantities such as flow embeddings and log–densities are stored and reused. Since both posteriors and acquisition repeatedly call the same utilities, caching avoids recomputation. For instance, once $u_\theta(x) = \log p_\theta(x)$ has been evaluated for an outcome x, the same value can be used for preference likelihoods and entropy terms.

These practices do not change the model itself, but they make the iterative loop viable. Without reuse, scoring the many candidate pairs in each round would quickly become intractable.

4.4 Acquisition via Expected Information Gain

Once flows and posteriors are in place, the final ingredient of PCBO is the mechanism that scores and ranks candidate interventions. The acquisition procedure is organised into four elements. First, a candidate set of interventions is defined, specifying which arms can be queried in the current round. Second, each candidate's effect is approximated through a one–step *virtual update*, which imagines how posteriors would change under hypothetical preference outcomes. Third, information–gain estimators are applied, targeting either utility refinement (Preference Information Gain, PIG) or structural discovery (Expected Edge Information Gain, EEIG). Finally, practical parameters such as sampling budgets and pruning heuristics determine how these computations scale to larger pools.

4.4.1 Candidate Set Construction

Before any scoring can take place, the learner requires a pool of possible comparisons. Each candidate is a pair of interventions (a, b) whose outcomes could, in principle, be evaluated by the flow and the posteriors. Formally, at round t the candidate set can be written as

$$\mathcal{C}_t \;=\; \{(a,b) : a,b \in \mathcal{A}_t,\; a \neq b\},$$

where $\mathcal{A}_t$ denotes the feasible interventions at that point in the loop.

Two regimes can be distinguished:

- **Hard interventions.** Each arm corresponds to directly setting a variable (or subset of variables) to a fixed value. Candidate pairs are formed by enumerating feasible assignments subject to any domain constraints.
- **Soft interventions.** Instead of fixing values, these modify the distribution of a node (e.g. shifting its mean). In this case, candidates are parameterised by the intervention strength and location, and pairs are sampled from the resulting distributional space.

In both cases, the pool must balance richness with tractability. Enumerating every possible pair quickly becomes infeasible: with M arms, the number of ordered pairs is $M(M-1)$. To avoid combinatorial explosion, we restrict candidates to subsets using prior entropy or diversity criteria, retaining only those that are likely to be informative for graph or utility refinement.

Once the candidate set is constructed, each pair is evaluated to estimate how much structural or utility information it is expected to provide.

4.4.2 One-Step Virtual Update

Scoring a candidate pair requires more than asking how uncertain the next preference is. The main question is how much the posterior over graphs would change if that comparison were observed. Directly running the full update for every hypothetical outcome would be prohibitively expensive, so PCBO uses a *virtual update*: a light-weight simulation of what would happen if either outcome were to occur.

Suppose the current edge posterior for target node i is represented by a distribution over parent sets, yielding marginals

$$\pi_{j\to i} \;=\; p(j \to i \mid \mathcal{D}_t).$$

For a candidate intervention a against opponent b, the preference flow provides the predictive probability $p(a \succ b \mid \mathcal{D}_t)$, while the complementary event has probability $1 - p(a \succ b \mid \mathcal{D}_t)$.

The virtual update then evaluates two branches: (i) if a wins, apply a *peek* update to obtain temporary edge marginals $\pi^{(a)}_{j\to i}$; (ii) if b wins, apply the analogous update to obtain $\pi^{(b)}_{j\to i}$. Neither branch alters the persistent state — they are rolled back immediately after use — but both provide a glimpse of how the posterior would move.

The expected posterior after the comparison is then

$$\bar{\pi}_{j\to i} \;=\; p(a \succ b \mid \mathcal{D}_t)\, \pi^{(a)}_{j\to i} \;+\; \big(1 - p(a \succ b \mid \mathcal{D}_t)\big)\, \pi^{(b)}_{j\to i}.$$

Information–gain criteria such as PIG or EEIG are based on the difference between the current marginals $\pi_{j\to i}$ and these virtual futures. In this way, PCBO can anticipate the value of a query before spending the budget to actually run it.

4.4.3 Estimators for PIG and EEIG

The one–step virtual update provides the ingredients needed to quantify how informative a candidate comparison is expected to be. PCBO employs two complementary estimators: Preference Information Gain (PIG), which measures refinement of the utility model, and Expected Edge Information Gain (EEIG), which measures refinement of the causal graph.

Preference Information Gain (PIG). For two outcomes a and b, let the flow predict a win probability $p(a \succ b)$. If this probability is close to 0.5, the result of the comparison is highly uncertain and hence informative about the preference boundary. A simple and effective estimator is the entropy of this Bernoulli outcome:

$$\mathrm{PIG}(a,b) \;\approx\; H\big(p(a \succ b)\big) = -\,p(a \succ b)\, \log p(a \succ b) - \big(1 - p(a \succ b)\big)\, \log\big(1 - p(a \succ b)\big).$$

This score identifies pairs where the preference learner is maximally uncertain. It is also is cheap to compute, making it suitable for wide sweeps.

Expected Edge Information Gain (EEIG). While PIG targets the utility function, EEIG evaluates how much structural uncertainty would shrink after observing the comparison. Let $H(\pi)$ denote the entropy of the edge marginals at the current step, and let $\bar{\pi}$ be the expected posteriors obtained via the virtual update. Then the expected reduction in entropy is

$$\mathrm{EEIG}(a,b) \;=\; H(\pi) - \Big[p(a \succ b)\, H(\pi^{(a)}) \;+\; (1 - p(a \succ b))\, H(\pi^{(b)})\Big].$$

Here $\pi^{(a)}$ and $\pi^{(b)}$ are the temporary edge marginals if a or b were to win. The estimator rewards candidates that would meaningfully reduce uncertainty about the graph regardless of which way the preference falls. It is slower but based on causal posteriors, guiding interventions towards structural discovery.

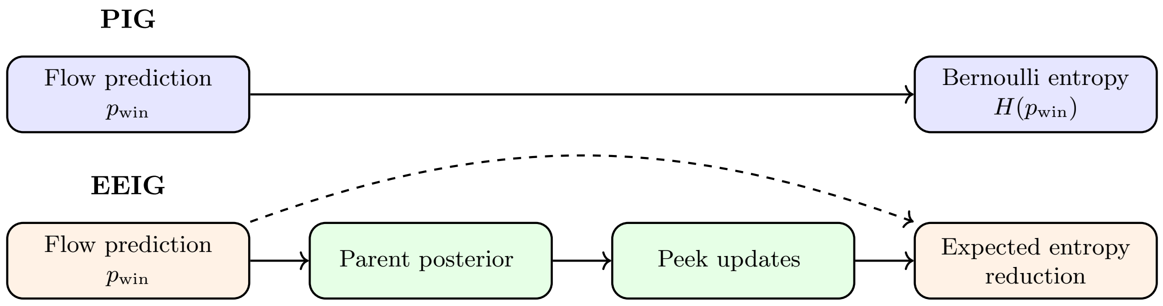

PIG
Flow prediction p_{win}
Bernoulli entropy $H(p_{\text{win}})$
EEIG
Flow prediction p_{win}
Parent posterior
Peek updates
Expected entropy reduction

Figure 4.2: PIG relies only on entropy of the flow's win probability. EEIG uses parent posteriors, applies peek updates, and combines their entropies weighted by the flow's win probability.

4.4.4 Balancing Exploration and Exploitation

Relying exclusively on PIG risks over–fitting to preferences without resolving causal ambiguities, while relying on pure EEIG may explore structural edges endlessly without improving decisions. To reconcile these objectives, PCBO blends the two into a single acquisition score:

$$S_t(a) \;=\; \alpha_t\,\mathrm{PIG}(a) + (1-\alpha_t)\,\mathrm{EEIG}(a),$$

The blending coefficient $\alpha_t \in [0,1]$ controls how much emphasis is placed on preference versus structure. It is not fixed, but evolves during the optimisation, shifting balance between exploration and exploitation. We employ a simple three–phase schedule for α_t based on the iteration index t (normalised to $[0,1]$):

$$\alpha_t = \begin{cases} 0.2 + 0.3\frac{t}{0.2}, & t < 0.2 \\ 0.5, & 0.2 \le t < 0.7 \\ 0.5 + 0.4\frac{t-0.7}{0.3}, & t \ge 0.7\ . \end{cases}$$

Early in the process (0–20% of rounds), data are scarce and structural uncertainty is high. α_t increases linearly from 0.2 to 0.5, prioritising EEIG but keeping a small utility weight so PIG can already shape preferences. It then reaches a middle plateau (20–70%), where $\alpha_t = 0.5$, which balances effort between graph and utility once the initial structure starts to settle. As rounds progress towards the late stages (70–100%) and the graph stabilises, α_t increases linearly from 0.5 to 0.9, shifting weight toward PIG to refine the utility function more precisely.

For each anchor, we score every opponent with the blended acquisition score $S_t(a)$. To avoid one noisy metric from dominating, each score vector is first normalised. We then add a light, history-aware exploration bonus for candidates from under-tested nodes.

4.5 From Components to Workflow

Up to this point we have introduced PCBO piece by piece: the flow for preference learning, the Bayesian update for local structures, the acquisition functions that balance graph and utility, and the DAG projection to keep things coherent. It is useful now to step back and see how these components connect in practice.

At each round, the framework decides how to update the local causal posteriors. For small graphs ($d \leq 10$) this is done exactly, by enumerating all parent sets; for larger graphs, the scalable MCMC-based approximation takes over. In either case the result is an edge probability matrix P that captures current beliefs about the graph. Acquisition then operates directly on P, choosing the interventions most likely to be informative.

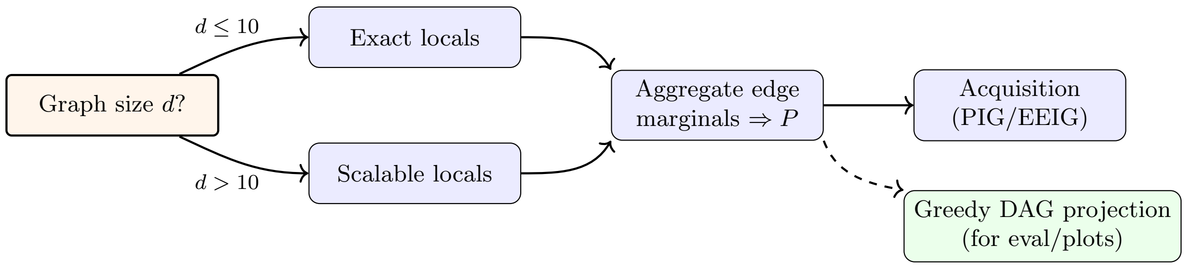


Figure 4.3: Workflow of one PCBO round.

4.5.1 Asymptotics per Iteration

Each PCBO round follows the same rhythm: the flow is updated on fresh preference data, local posteriors are refreshed, and the acquisition function selects the next interventions. Safeguards such as gradient clipping, caching, and warm starts keep runtime manageable in practice, but it is still useful to ask: which parts grow costly as graphs or candidate pools increase?

Table 4.1 gives a breakdown. Flow training scales with batch size and depth of the flow; candidate generation with the number of interventions sampled; and acquisition linearly once flow evaluations are cached. The heavier steps are the exact local updates, which grow exponentially in in-degree, and the greedy DAG projection, which involves sorting and reachability checks. This makes clear why scalable approximations are essential: without them, inference would collapse as graphs move beyond toy scale.

Component	Complexity	Notes
Flow training	$O(B \cdot L)$	Batch size B, flow depth L
Candidate generation	$O(n_{\text{cand}} \cdot d)$	Sample interventions, extract features
PIG/EEIG scoring	$O(n_{\text{cand}})$	With cached flow evaluations
Exact locals	$O(2^{k_{\max}} \cdot k_{\max}^3)$	Parent set enumeration
Scalable locals	$O(N_{\text{mcmc}})$	MCMC proposals per node
DAG projection	$O(d^2 \log d + d^3)$	Sort + reachability updates

Table 4.1: Asymptotic costs per PCBO iteration. Here d is the number of nodes, $k_{\max}$ the maximum in-degree, n_{cand} the number of candidate interventions, B the batch size, L the flow depth, and N_{mcmc} the number of MCMC proposals.

4.6 Baselines and Evaluation Protocol

To interpret the performance of PCBO, it is essential to place it against a set of reference methods and to apply a consistent evaluation protocol. These baselines are not designed as realistic competitors, but as points of contrast that make clear what PCBO is trying to achieve.

The key difference is that baselines see the actual outcome values after interventions, while PCBO only sees which outcome was preferred. Learning from full measurements is a much easier task, yet this is exactly why we built PCBO: in many domains, only comparative judgements are available, not calibrated numbers. The baselines therefore provide a kind of "upper bound" perspective: what can be done when one has much more information than PCBO ever receives?

4.6.1 Baselines

We include six reference learners, each reflecting a different kind of shortcut:

- *Random and fully connected graphs.* Two naive anchors: one samples edges at random, the other assumes every variable is connected. They provide lower and upper bounds of structure, with no statistical reasoning.
- *Lasso regression.* Each variable is regressed on all others with ℓ_1 regularisation, yielding edges wherever coefficients are nonzero. This captures correlations quickly and encourages sparsity, but ignores interventions and acyclicity.
- *PC-lite skeleton.* A stripped-down version of the PC algorithm: starting from a dense graph, edges are pruned by conditional independence tests. The result is an undirected skeleton that reveals strong dependencies, but orientation and cycle checks are not enforced.
- *NOTEARS-lite surrogate.* Inspired by the full NOTEARS algorithm but simplified for efficiency: per-target ElasticNet regressions identify candidate edges, which are then thresholded and greedily projected to a DAG. It captures linear structure under interventions, but lacks the exact smooth acyclicity guarantee of the original method.
- *Causal sufficiency heuristic.* Adds $i \to j$ if intervening on X_i produces a significant shift in X_j (tested with a simple statistical comparison). This picks up strong direct effects, but ignores confounding and indirect paths.

These methods make use of richer signals than PCBO, but lack its preference modelling or information–theoretic design. They serve as benchmarks, not rivals: if PCBO can approach or even surpass their performance while working with strictly weaker feedback, it shows the value of preference modelling in causal discovery.

4.6.2 Thresholding Edge Probabilities

All learners in this work output a dense edge–probability matrix $P \in [0,1]^{d\times d}$, where each entry P_{ij} quantifies the model's confidence in an edge $i \to j$. Taken at face value, such matrices are messy: most entries are small but nonzero, so a naive binarisation would yield graphs cluttered with weak, spurious edges. The solution is thresholding. We introduce an operating point τ, retain only edges with $P_{ij} \geq \tau$, and then apply a greedy projection to ensure acyclicity.

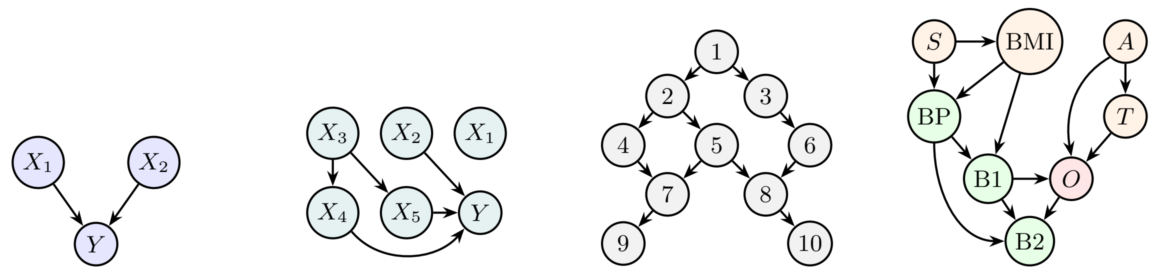
Three strategies are considered:

1. *Label-based.* On synthetic data with known adjacency $A^\star$, we choose τ by grid search to maximise F1, i.e. the harmonic mean of precision and recall computed on $\mathbb{I}\{P_{ij} \geq \tau\}$ versus $A^\star$. This provides an upper bound on achievable performance given oracle access.
2. *Label-free.* For realistic settings without ground truth, we fit a two-component Beta mixture to the off–diagonal entries of P. Let $x = \{P_{ij} : i \neq j\}$. We assume $x \sim \pi \,\mathrm{Beta}(a_1, b_1) + (1-\pi)\,\mathrm{Beta}(a_2, b_2)$, where one component captures "null" edges (concentrated near zero) and the other "signal" edges (concentrated near one). The Beta family is natural here because it flexibly models probabilities in $[0, 1]$ with shapes ranging from uniform to sharply peaked, making it well suited to distinguish weak from strong edges. Parameters are estimated by an expectation–maximisation (EM) routine: responsibilities of each component are updated in the E–step, while new shape parameters are fitted in the M–step using simple moment equations for the mean and variance. The operating threshold $\tau^\star$ is then chosen as the point where the two mixture components are equally likely:
$$\pi f_{\mathrm{Beta}}(\tau^\star; a_2, b_2) = (1-\pi) f_{\mathrm{Beta}}(\tau^\star; a_1, b_1).$$
Intuitively, $\tau^\star$ is the probability level at which an edge is equally likely to be spurious or genuine: the tipping point between noise and structure.
3. *Manual, distribution-based.* In some cases, we simply inspect the edge probability distribution and set τ by eye. If many edges cluster tightly around 0.9 or 1.0, we may enforce a threshold at 0.9. This approach is crude but intuitive: in high–risk domains one might demand near certainty, while in exploratory work a looser cut may be acceptable. Manual thresholding also leaves room for domain experts – their judgement can set the operating point more meaningfully than any automatic rule.

Assuming no access to ground truth, the label-free criterion and manually set threshold are the default in our experiments. This simple step has a decisive impact: without thresholding, PCBO's outputs resemble noisy hairballs; with it, the graphs become sparse, interpretable, and substantially closer to ground truth. Thresholding therefore acts as a filter, and the choice of threshold is not merely technical: it encodes the level of caution we wish to exercise in deciding what counts as a genuine causal relation.

4.6.3 Datasets

To evaluate PCBO we rely on three families of datasets, chosen to test different aspects of the framework. The toy graphs provide transparency and speed: small, nonlinear structures where every edge matters and failures are easy to diagnose. The Erdős–Rényi graphs probe scalability: random structures that grow to dozens of nodes and stress the limits of inference. Finally, the medical case study offers a domain-specific setting with interpretable variables, closer to how preference data might appear in practice.



(a) 3-node toy **(b)** 6-node mixed toy **(c)** ER DAG (10 nodes) **(d)** Medical toy

Figure 4.4: Benchmark DAGs used in experiments: from small hand-crafted toys to larger random and domain-specific graphs.

Small nonlinear graphs. The smallest benchmarks are hand-crafted DAGs with three and six nodes. Their structural equations combine linear and nonlinear functions (e.g. polynomials, tanh links), with additive Gaussian noise. Hard interventions clamp a chosen node, propagate downstream effects, and yield a utility score for comparison. These datasets are deliberately simple: they run in seconds, make the causal structure fully visible, and are designed to check whether the PCBO loop behaves as expected before scaling up.

Erdős–Rényi graphs. To move beyond toys, we generate Erdős–Rényi DAGs [43] with linear–Gaussian structural equations. A random topological order is sampled, and edges are added independently with probability p, giving acyclic graphs of varying size. Interventions again fix a node's value, with outcomes scored by a hidden linear utility that induces pairwise preferences. These graphs scale up to twenty or more nodes, providing a controllable way to stress-test runtime, posterior quality, and acquisition behaviour under growing complexity.

Medical case study. For a more applied test, we use a small medical-style DAG with interpretable nodes such as lifestyle factors, intermediate biomarkers, and a health outcome Y. Structural equations are simple but nonlinear, with noise to mimic variability. Preferences correspond to treatment comparisons: for example, which regimen of diet or exercise produces a healthier simulated patient. This case study is not about scale but about interpretability: it shows how PCBO operates when the variables have real-world meaning, complementing the abstract but scalable ER graphs.

4.6.4 Evaluation Loop

All methods (PCBO and baselines) are run under a fixed query budget. At each checkpoint we project edge posteriors to a DAG via a greedy MAP procedure (ranking edges by logit scores and enforcing acyclicity), so all approaches are compared on a common binary graph. We track three main families of metrics:

- **Graph structure (metrics).** The most direct measure is the *Structural Hamming Distance (SHD)*, which counts edge-level disagreements between the learned graph $\widehat{A}$ and the ground truth $A^\star$:

$$\mathrm{SHD}(\widehat{A}, A^\star) = \sum_{i \neq j} \big|\widehat{A}_{ij} - A^\star_{ij}\big|.$$

An SHD of zero means perfect recovery; higher values indicate extra, missing, or misoriented edges. To tease apart different types of errors, we also track

$$\text{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \quad \text{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \quad \mathrm{F1} = \frac{2\,\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}.$$

Here TP, FP, and FN denote the counts of true positives, false positives, and false negatives respectively, computed over all off–diagonal entries. Precision reflects caution (few false positives), recall reflects coverage (few false negatives), and F1 balances the two. Together, these metrics answer: did the method recover the right edges, and did it avoid hallucinating spurious ones?

- **Graph structure (visualisation).** Beyond a binary prediction, each method produces a dense edge–probability matrix P, where each entry $P_{ij} = p(i \to j)$ is the model's confidence in an edge. These scores let us see not only which edges survive thresholding, but how clearly the signal separates from the noise. Plotting the graph with edge thickness proportional to P_{ij} makes weak edges fade, strong edges stand out, and the overall pattern tells us whether the model is genuinely confident or merely hedging.

  When a concrete graph is required, we set a threshold t (via Beta-mixture or manually) and project to a DAG. But the visualisation gives context that the metrics alone cannot: it shows why certain thresholds work, where the model is decisive, and where ambiguity remains. It answers not only "does the model get the structure right," but also "how does it express its uncertainty along the way?"

- **Preference/utility learning.** Since PCBO is driven by preferences, we test how well the learned utility flow extrapolates to unseen comparisons. For two interventions a, b, the oracle preference is $\mathrm{sign}(u^\star(a) - u^\star(b))$, while the model's is $\mathrm{sign}(\hat{u}(a) - \hat{u}(b))$. Accuracy is the fraction of matches over test pairs:

$$\mathrm{Acc} = \frac{1}{n}\sum_{k=1}^{n} \mathbb{I}[\mathrm{sign}(u^\star(a_k) - u^\star(b_k)) = \mathrm{sign}(\hat{u}(a_k) - \hat{u}(b_k))]\,.$$

  This metric captures whether the flow learns not just the structure but the outcome–relevant utility landscape.

- **Diagnostics.** For interpretability we also log calibration-style edge errors $\frac{1}{d(d-1)}\sum_{i\neq j}|P_{ij} - A^\star_{ij}|$, per-iteration SHD reductions (a proxy for information gain), the flow's noise scale s, and intervention counts per node. These diagnostics help dissect training dynamics but are not used for headline comparisons.

Averaging across seeds reduces incidental randomness, while aligning query budgets ensures that differences reflect modelling choices rather than computational advantage.

Chapter 5

Experimental Evaluation

5.1 Goals & Questions

Having laid out the framework and how we built it, we now ask the crucial question: can PCBO actually recover meaningful causal structure from preferences? The experiments that follow put this to the test across different settings. To structure this evaluation, we focus on four guiding questions:

1. *Structure learning.* Can PCBO reconstruct the true causal graph accurately and consistently, despite being trained on pairwise preferences rather than direct outcome data?
2. *Baselines.* How does PCBO compare to baseline learners that are trained on richer information from full intervention outcomes?
3. *Thresholding.* How does the choice of operating threshold affect the quality of the recovered graphs?
4. *Flow architecture.* Does the choice of normalising flow family (RealNVP, Residual, Neural Spline) influence performance or stability?

The goal is not only to measure accuracy, but to understand how PCBO learns under weak supervision, how design choices shape the results, and how far it can go compared to classical methods that enjoy easier conditions.

5.2 Experimental Setup

To make our evaluation convincing, we need a clear and consistent protocol. Here we focus on how we put PCBO to the test and how the results will be presented.

Datasets. We experiment on the full suite of graphs from Section 4.6.3, from the small and interpretable synthetic settings (3 and 6 nodes), to the more intricate and larger graphs (ER-15 and 20), to the medical case study. Each captures a different challenge: mediators, scaling, nonlinearity, or interpretability.

Synthetic first, then case study. We begin with synthetic graphs, which provide controlled environments where the ground truth is fully known. These experiments establish whether PCBO can recover structure under different conditions of size and complexity. We then move to our case study – a medical graph trained with a Neural Spline Flow – where we analyse in depth the model's behaviour: causal discovery, learning curves, edge posterior probabilities, and preference learning. This serves as the detailed test of how PCBO behaves in a realistic, interpretable setting. Finally, we discuss the influence of flow families in the learning process.

Baselines. We compare against the standard structure-learning methods presented in Section 4.6, trained directly on outcome data: Random graphs, LASSO, the PC algorithm (skeleton), NOTEARS-Lite, a causal sufficiency heuristic, and a fully connected baseline. These are not strict rivals as they have access to stronger supervision, but they provide a useful performance ceiling. If PCBO comes close, it is a strong signal of credibility.

Operating points. Edge probabilities from PCBO are continuous. To turn them into graphs, we evaluate both (i) projected graphs, obtained by greedily enforcing acyclicity directly on probabilities, and (ii) thresholded projected graphs, where we first select a threshold τ (either from a Beta-mixture fit or manually, as mentioned in Section 4.6.2) before projecting.

Metrics. Our primary criterion is Structural Hamming Distance (SHD), complemented by precision, recall, F1, and edge counts. For the in-depth analysis of the medical case study, we also report the area under the SHD-vs-iteration curve (SHD-AUC), as well as diagnostic plots: learning curves, confusion matrices, and predicted causal graphs. These allow to visualise the learning process and the final product of PCBO.

5.3 Synthetic Graphs: Controlled Tests

Synthetic experiments are the natural starting point. They give us full control: the ground truth is known, the structure is transparent, and we can stress-test PCBO under different conditions. From tiny graphs where every edge matters, to random graphs with tens or even a hundred nodes, each setting probes a different aspect of the algorithm.

In these tests we use Neural Spline Flows (NSF), the most expressive family in our framework. The goal here is simple: to check how well PCBO can recover the true structure from preference data. We measure this through the standard structural metrics (SHD, precision, recall, F1) and show representative graph reconstructions where they are most interpretable.

5.3.1 Small Graphs: Three and Six Nodes

We begin with the smallest graphs in our suite: the three- and six-node settings. These are controlled and interpretable cases, where every edge matters and where we can see clearly how PCBO handles dense versus sparse predictions.

Three nodes. Figure 5.1 shows the results. The projected graph is slightly too dense, producing one spurious edge. But thresholding immediately corrects it: both τ^* (Beta-mixture) and $\tau \approx 1.0$ (manually set) recover the exact ground truth, with SHD dropping to zero and precision and recall both at one.

Table 5.1 confirms this picture. Without thresholding, recall is perfect but precision lags because of the extra edge. As soon as a threshold is applied, PCBO locks onto the true structure exactly. Baselines generally struggle more: NOTEARS finds part of the structure but not all, while LASSO overfits with too many false positives. The exact graph structures predicted by the baselines are shown in Appendix 6.1.

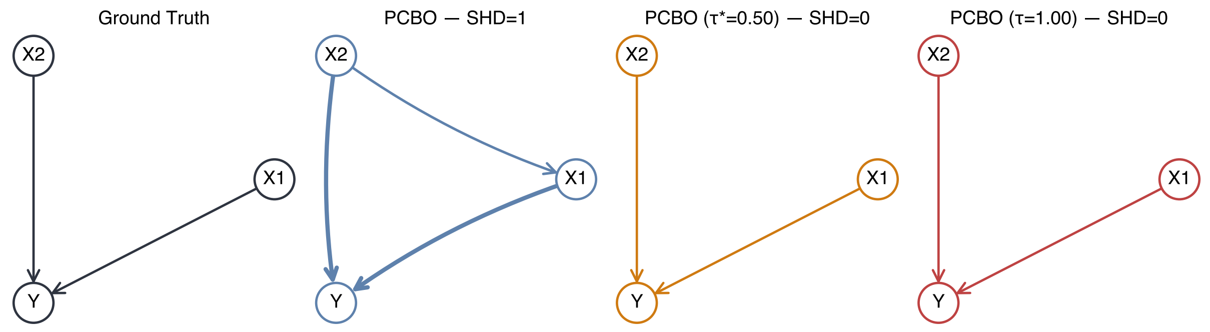


Figure 5.1: Causal graphs on the 3-node dataset. Left: ground truth. Middle-left: PCBO projection (edge thickness is proportional to posterior probability). Middle-right: thresholding with τ^* from the beta-mixture. Right: pushing the threshold to $\tau \approx 1.0$.

Table 5.1: Graph-level performance on the 3-node dataset: PCBO (preference-trained) and baselines (outcome-trained).

Method	Precision	Recall	F1	SHD	#Edges
PCBO	0.667	1.000	0.800	1	3
PCBO ($\tau^* = 0.50$)	1.000	1.000	1.000	0	2
PCBO ($\tau = 1.00$)	1.000	1.000	1.000	0	2
Random	0.000	0.000	0.000	2	0
LASSO	0.333	1.000	0.500	4	6
PC	0.000	0.000	0.000	5	3
NOTEARS-Lite	0.333	0.500	0.400	3	3
Causal Sufficiency	0.000	0.000	0.000	2	0
Fully Connected	0.667	1.000	0.800	1	3

Six nodes. The six-node graph presents a bigger challenge, with more possible edges and mediators in play. Figure 5.2 shows that the projected graph again starts off too dense, carrying nine spurious edges. But here too, thresholding is decisive: both τ^* and $\tau \approx 0.99$ cut away the noise completely, recovering the ground truth with SHD=0 and perfect precision, recall, and F1.

Table 5.2 shows the metrics. Standard PCBO achieves recall of one but with poor precision, due to its over-connectivity. Thresholding flips the balance: spurious edges disappear, precision rises to one, and the recovered graph is exactly right. Baselines, by contrast, vary in behaviour: NOTEARS and PC get some edges right but miss others, and LASSO again produces too many false positives. None reach the perfect recovery achieved by PCBO. The exact graph structures predicted by the baselines are shown in Appendix 6.2.

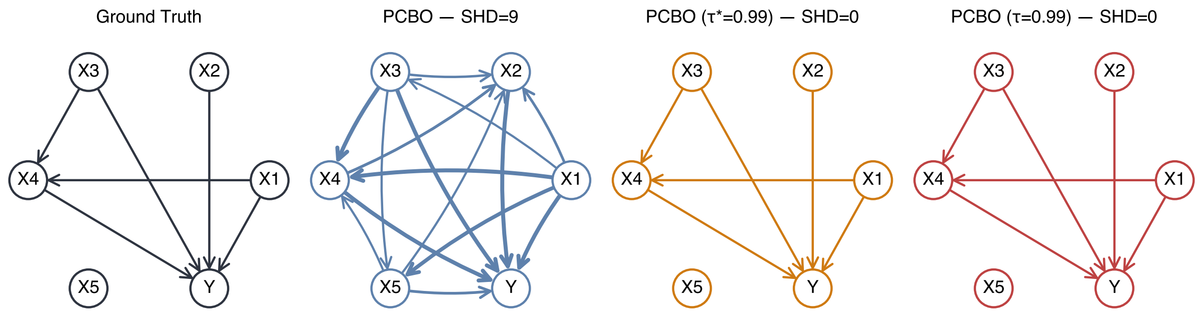


Figure 5.2: Causal graphs on the 6-node dataset with PCBO (standard and thresholded).

Table 5.2: Graph-level performance on the 6-node dataset: PCBO (preference-trained) and baselines (outcome-trained).

Method	Precision	Recall	F1	SHD	#Edges
PCBO	0.400	1.000	0.571	9	15
PCBO ($\tau^* = 0.99$)	1.000	1.000	1.000	0	6
PCBO ($\tau = 0.99$)	1.000	1.000	1.000	0	6
Random	0.500	0.333	0.400	6	4
LASSO	0.316	1.000	0.480	13	19
PC	1.000	0.333	0.500	4	2
NOTEARS-Lite	0.625	0.833	0.714	4	8
Causal Sufficiency	0.000	0.000	0.000	6	0
Fully Connected	0.400	1.000	0.571	9	15

5.3.2 Larger Graphs: ER-15 and ER-20

As we move to Erdős–Rényi graphs with 15 and 20 nodes, the challenge becomes very different from the toy settings. Here the number of possible edges grows quickly, the graphs are denser, and the task is not just to identify a handful of causal links but to explore a large combinatorial space. Moreover, as both graphs have more than 10 nodes, this is the point where we switch to the scalable parent posterior, which keeps inference feasible as the number of candidate parents grows. This is the first real stress test for PCBO: can a method trained only on preferences remain competitive once the search space expands to this scale?

ER-15. The projected graph is heavily over-connected, with more than one hundred edges, far above the true count. As expected, this leads to low precision. Thresholding, however, makes a dramatic difference: once weak edges are cut, PCBO stabilises with SHD = 13, precision above 0.5, and F1 around 0.58. The improvement is clear in both the graph plots (Figure 5.3) and the metrics (Table 5.3).

Comparing to baselines, NOTEARS stands out with near-perfect recovery (F1 = 0.93, SHD = 2), though it benefits from direct outcome supervision. LASSO achieves full recall but pays the price in false positives. Other methods fall short, with poor precision and incomplete structures. In this setting, PCBO is not the top performer, but it shows it can scale up: with only preference data, it finds a credible graph close to the ground truth.

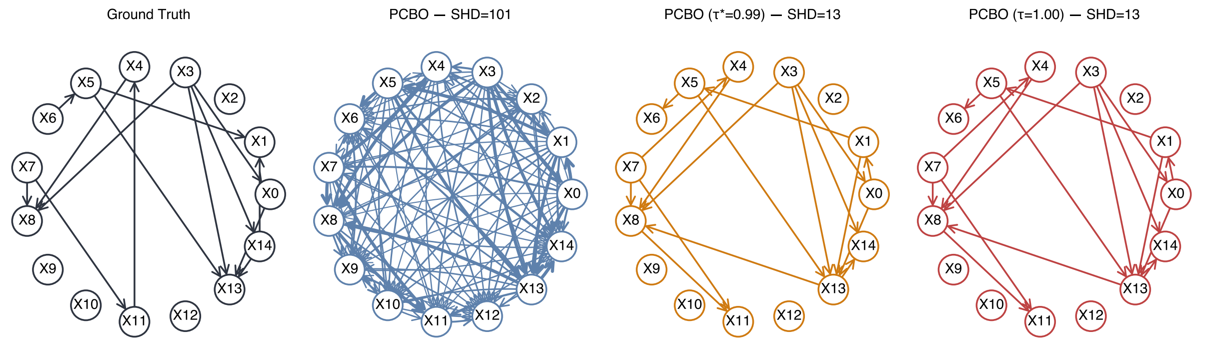


Figure 5.3: Causal graphs on the ER-15 dataset with PCBO (standard and thresholded).

Table 5.3: Graph-level performance on the ER-15 dataset: PCBO (preference-trained) and baselines (outcome-trained).

Method	Precision	Recall	F1	SHD	#Edges
PCBO	0.086	0.643	0.151	101	105
PCBO ($\tau^* = 0.99$)	0.529	0.643	0.581	13	17
PCBO ($\tau = 1.00$)	0.529	0.643	0.581	13	17
Random	0.125	0.286	0.174	38	32
LASSO	0.246	1.000	0.394	43	57
PC	0.000	0.000	0.000	24	10
NOTEARS-Lite	0.929	0.929	0.929	2	14
Causal Sufficiency	0.000	0.000	0.000	14	0
Fully Connected	0.076	0.571	0.134	103	105

ER-20. The twenty-node case pushes this further. Again, the projected graph is very dense, with 190 edges, producing high recall but almost no precision. Thresholding once more makes the difference: SHD drops sharply to 21, and F1 rises to around 0.67. This is far from perfect, but it shows that PCBO still manages to extract a clean and informative structure even in a large space of possibilities.

Similarly to the 15-nodes case, NOTEARS delivers the strongest results (F1 = 0.86, SHD = 8), while LASSO secures recall at the cost of many false positives. The others either fail to identify the structure or collapse into trivial graphs. Again, PCBO, although not matching NOTEARS, still proves robust considering its limited input data.

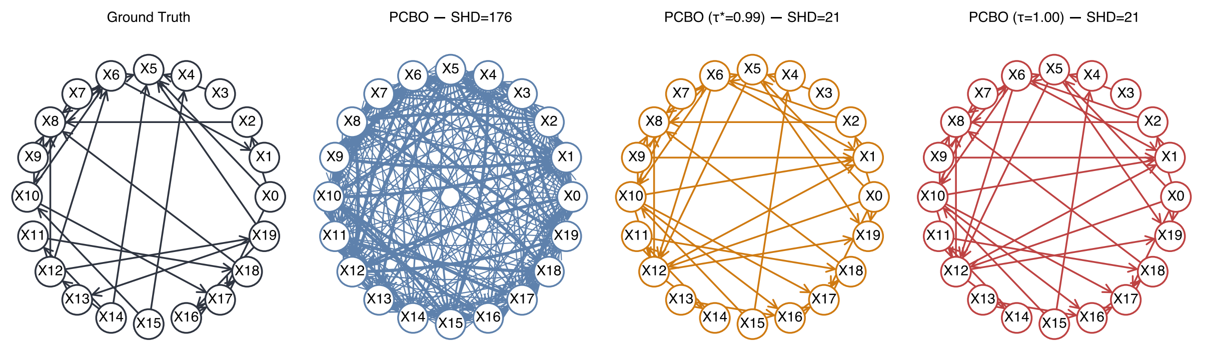


Figure 5.4: Causal graphs on the ER-20 dataset with PCBO (standard and thresholded).

Table 5.4: Graph-level performance on the ER-20 dataset: PCBO (preference-trained) and baselines (outcome-trained).

Method	Precision	Recall	F1	SHD	#Edges
PCBO	0.111	0.750	0.193	176	190
PCBO ($\tau^* = 0.99$)	0.600	0.750	0.667	21	35
PCBO ($\tau = 1.00$)	0.600	0.750	0.667	21	35
Random	0.083	0.179	0.114	78	60
LASSO	0.190	1.000	0.320	119	147
PC	0.000	0.000	0.000	45	17
NOTEARS-Lite	0.857	0.857	0.857	8	28
Causal Sufficiency	0.000	0.000	0.000	28	0
Fully Connected	0.063	0.429	0.110	194	190

5.3.3 Lessons from Synthetic Experiments

The synthetic experiments make one message clear: PCBO is not just a lucky fit for a single dataset. Across different graph sizes and structures, it shows the same core pattern. On the smallest graphs, it recovers the truth exactly, edge by edge, with thresholding removing all noise. As the graphs grow to larger sizes, precision and recall rebalance but the method still produces competitive reconstructions, staying fairly close to the ground truth even with limited supervision. And when the task scales to fifteen and twenty nodes, PCBO remains stable: not perfect, but robust, able to recover a plausible structure in a very large search space.

The real lesson is not only accuracy but consistency. PCBO's thresholding mechanism allows it to cut through the noise and reveal the signal, whether the graph has three nodes or twenty.

Other methods benefit from full outcome supervision, yet PCBO, guided only by preference comparisons, manages to stay credible and often competitive.

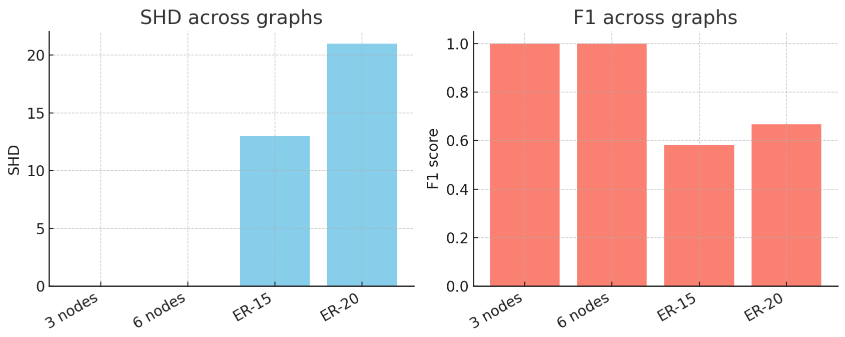


Figure 5.5: Summary of synthetic experiments: SHD (left) and F1 (right) across graph families. PCBO maintains robust performance across different scales.

Another important detail is the balance between precision and recall. Across all settings, recall tends to be higher: PCBO includes most of the true edges, while sometimes carrying extras that later need to be pruned. This is not necessarily a weakness. In many causal discovery scenarios, missing a true cause (false negative) is more damaging than entertaining a few extra candidates (false positives). For instance, overlooking a genuine risk factor in a medical study can be far worse than testing a few additional variables. In the ER-20 graph, PCBO correctly retrieves 21 out of 28 true edges, with only 7 missed – a strong signal in such a large search space. Thresholding then does its job of cleaning up, showing how the framework prioritises inclusivity first, refinement second.

This robustness across controlled tests builds trust: it suggests the framework is solid and not fine-tuned to a single case.

5.4 Case Study: Medical Graph with Neural Spline Flow

The synthetic tests showed that PCBO is not just a toy success: it works across different graph sizes and structures. We now move to the medical case study, where the challenge is more realistic and the variables are interpretable, with nodes such as smoking and blood pressure, and a target that depends on multiple true parents through nonlinear effects. This is the right place to open the black box and look closely at how PCBO learns, combining metrics, plots, and reconstructed graphs to understand its behaviour in detail.

5.4.1 Graph Recovery

The first step, as with the toy graphs, is simply to look at the graphs themselves (Figure 5.6). As expected, the projected graph is overly dense, filled with spurious edges. But thickness encodes probability, and even at this stage the signal is visible: the truly confident edges stand out. Thresholding makes the separation explicit. The beta-mixture threshold τ^* cuts away most of the noise, leaving a structure close to the truth (SHD $= 2$). Pushing further to $\tau \approx 1.0$

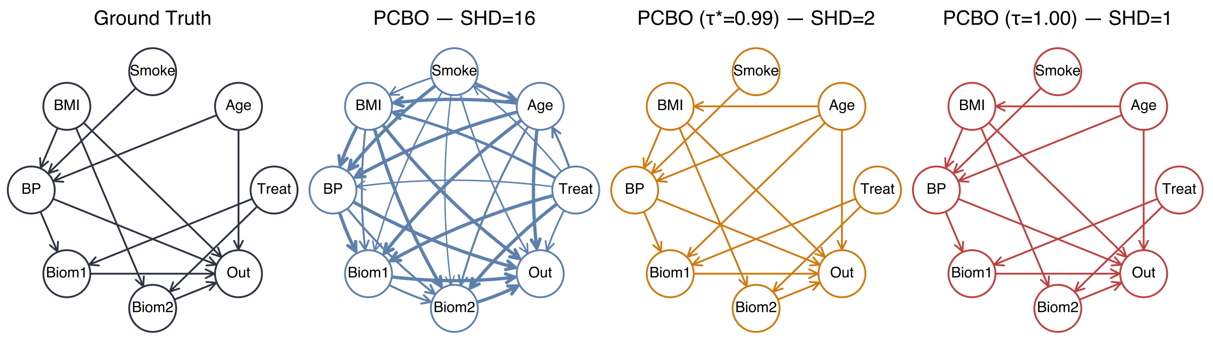


Figure 5.6: Causal graphs on the medical dataset with PCBO (standard and thresholded).

eliminates nearly all residual errors (SHD = 1), an almost perfect recovery out of all possible directed graphs among eight nodes.

The only confident spurious edge that remains is *Age* $\rightarrow$ *BMI*. This is not a completely random error: in reality, age and BMI are often correlated. A plausible explanation is that in our dataset they tend to move together through their shared effect on the outcome. Since the model only sees preferences and never intervenes on the outcome, it cannot fully separate this shared influence from a direct link, that is likely why the edge is kept.

The metrics back up what we saw in the graphs. Table 5.5 reports precision, recall, F1, SHD, and edge counts for the three PCBO variants. The full edge-wise counts (TP/FP/FN/TN) for each variant are reported in Appendix 6.1.

Table 5.5: Graph-level performance of PCBO on the medical dataset.

Method	Precision	Recall	F1	SHD	#Edges
PCBO	0.429	1.000	0.600	16	28
PCBO ($\tau^* = 0.99$)	0.857	1.000	0.923	2	14
PCBO ($\tau = 1.00$)	0.923	1.000	0.960	1	13

The pattern is clear:

- The raw projected graph achieves perfect recall, meaning PCBO keeps all true edges. Still, precision is low (0.429) since it carries a large number of spurious edges (28 in total).
- The adaptive τ^* threshold dramatically cleans up the graph: precision jumps to nearly 0.857, F1 to 0.923, and SHD drops to 2 with 14 edges retained.
- A stricter τ brings the reconstruction within one edge of the truth. As a consequence, precision is near-perfect (0.923) and F1 jumps to 0.960.

This confirms the behaviour we saw in synthetic tests: even in a more realistic setting with nonlinear dependencies, PCBO recovers the true causal structure with just one redundant edge.

5.4.2 Baseline Comparison

To put the medical results in context, we compare PCBO against the baselines. All of these operate on a much richer source of information: the full outcomes of interventions, not just preferences. By design, they should have the advantage.

And yet, the picture that emerges is different. Table 5.6 shows the numbers, and Figure 5.7 provides the side-by-side graph reconstructions. For completeness, the detailed edge-wise counts (TP, FP, FN, TN) for each baseline are reported in Appendix 6.2.

Table 5.6: Graph-level performance of PCBO (τ) vs. outcome-based baselines on the medical dataset.

Method	Precision	Recall	F1	SHD	#Edges
PCBO ($\tau = 1.00$)	0.923	1.000	0.960	1	13
Random	0.600	0.250	0.353	11	5
LASSO	1.000	0.917	0.957	1	11
PC	0.000	0.000	0.000	14	2
NOTEARS-Lite	1.000	0.750	0.857	3	9
Causal Sufficiency	0.636	0.583	0.609	9	11
Fully Connected	0.429	1.000	0.600	16	28

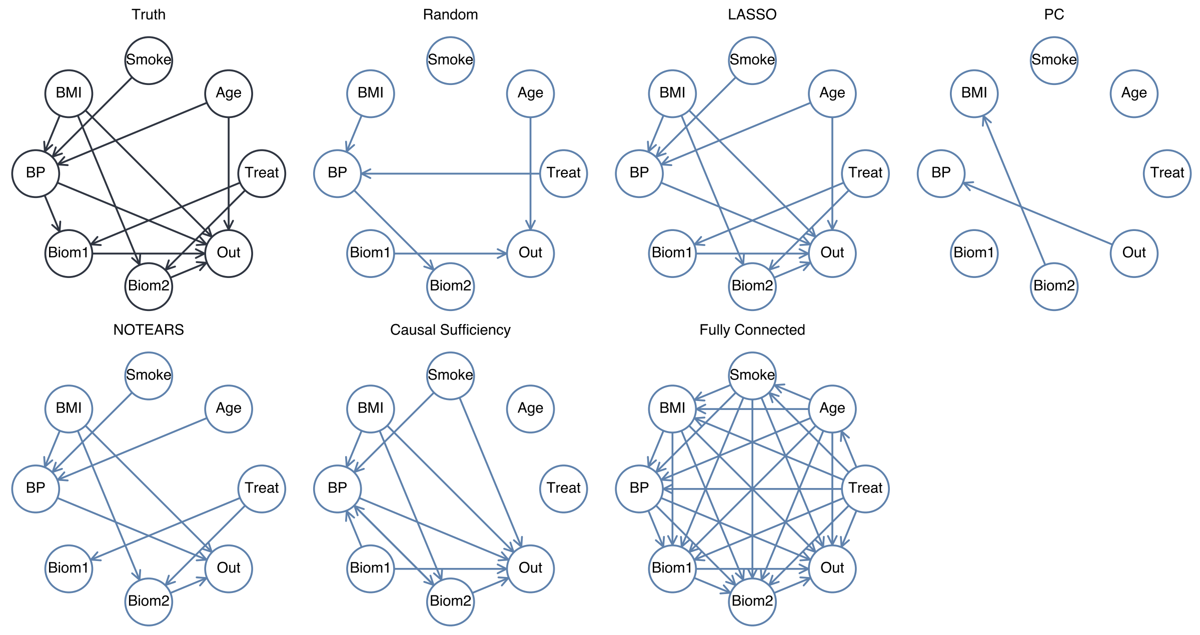


Figure 5.7: Causal graph reconstructions on the medical dataset: truth (left) and baseline methods. While some baselines capture parts of the structure, none achieve the overall fidelity of PCBO.

PCBO not only holds its own, but actually comes out on top. On F1, it beats every baseline:

it combines high precision with strong recall in a way no other method matches. LASSO comes closest: it reaches same SHD of 1, and falls behind PCBO (τ) only marginally in F1 (0.957 vs. 0.960). The trade-off is clear: PCBO achieves perfect recall, while LASSO achieves perfect precision. In practice, that difference matters: LASSO misses blood pressure (BP) as a parent of Biomarker 1, an omission that could be more harmful than including an extra edge. NOTEARS, often seen as a reliable baseline, also attains perfect precision, but misses three true edges, including two direct causes of the outcome. Again, in a medical setting, leaving out genuine drivers of risk is arguably the greater error.

Overall, methods with privileged access to richer data do not manage to outperform PCBO in this medical case study. In fact, they often lag behind. Preferences, when used the right way, are not just enough: they can surpass the performance of outcome-based methods.

5.4.3 Learning in Two Lanes: Structure and Preferences

PCBO learns two things at once: the causal graph and a model of preferences. They interact, but they are not the same objective. Here we examine them side by side to see how the system reaches a stable graph, and what the preference model is doing along the way.

Graph Learning (Figure 5.8)

- *Learning dynamics.* The first row of plots tracks SHD, edge count, and edge-probability error over time. For the projected variant, SHD remains high and the graph dense, which is consistent with the inflated number of edges it produces. The thresholded variant, however, shows the expected pattern: SHD drops initially and then stabilises, while the number of edges first rises and then settles close to the ground truth.

  The edge-probability error curve may look flat at first glance, as if the model were not improving. In reality, this metric is not meant to reflect convergence directly. Standard PCBO will never collapse all mass onto a single graph – it is a Bayesian method, and some uncertainty is always retained. Even for the thresholded version the error cannot vanish: as long as even a single spurious edge remains with very high probability (close to ≈ 1.0 when the truth is 0). This explains why the curve levels off rather than going to zero.

- *Threshold.* The threshold curve rises and then plateaus at a high level, and the histogram of off-diagonal edge probabilities at the end is clearly bimodal: a small set of edges is assigned near-certain probability, the rest near zero. This explains why a high operating threshold works so well here – the signal separates cleanly from the noise.

- *Local analysis.* The parent-posterior panel for the outcome node gives us a clear focus: posterior mass is concentrated on the true parents, while non-parents are pushed down. This is a sign of stability: despite the global uncertainty, when we zoom in on the outcome, PCBO finds the correct causes with high confidence.

- *Interventions.* The intervention-counts panel shows that most interventions are carried out on *BP*, while other variables are touched far less, and the outcome node is never intervened on. This tells us something meaningful about the policy: changing the value of

BP produces the most informative structural feedback, while intervening directly on the outcome would be useless, since outcomes are effects, not causes to manipulate.

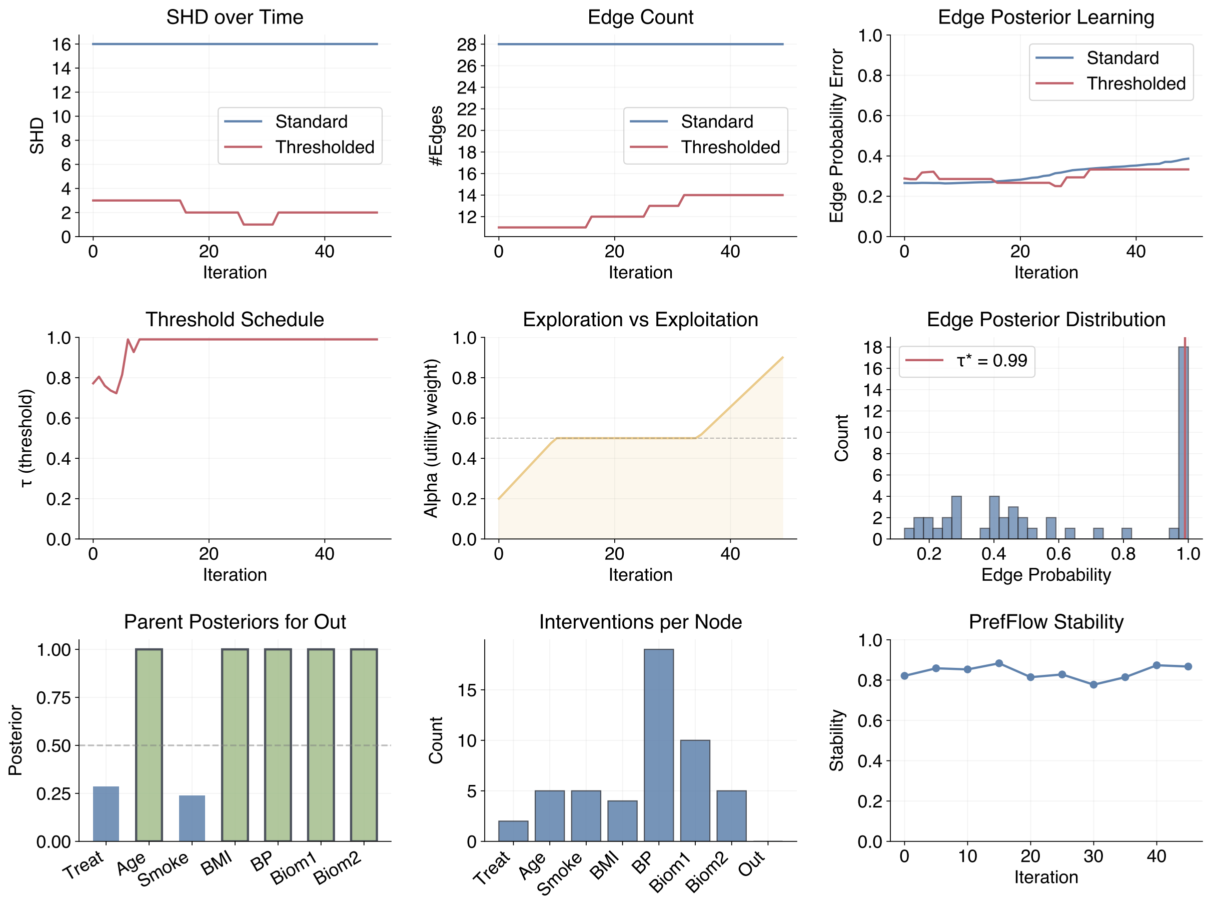


Figure 5.8: Graph learning dynamics on the medical dataset.

Preference Learning (Figure 5.9)

Preference modeling is secondary to graph discovery, and the plots show why it looks less smooth. Prediction accuracy oscillates rather than rising monotonically. This is expected: the algorithm deliberately probes uncertain regions to learn the structure, so many comparisons remain difficult by design. The noise scale s stays close to 1, which reflects good calibration: the model is neither overconfident nor too uncertain. The sampled utility density occupies a coherent region of the 2D feature space, concentrating in plausible outcomes. The data-accumulation plot shows why training remains possible: each iteration produces many preference pairs, giving the flow sufficient data to learn from.

The medical case study shows that PCBO does what it set out to do. With nothing more than preference feedback, it recovers the right causal structure almost perfectly and with high

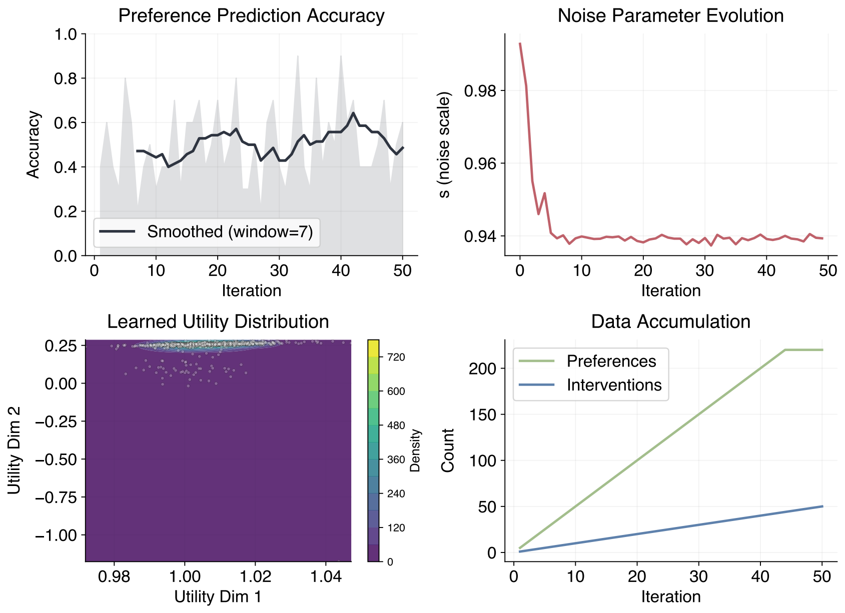


Figure 5.9: Preference learning dynamics on the medical dataset.

accuracy, and holds its own against baselines that rely on richer data. The role of thresholding is clear: it trims away noise and refine the graph. Structure learning carries the main signal, while preference modelling is less smooth, but stays steady enough to support the process. These results give a concrete proof that causal discovery from preferences is not just possible, but effective in a realistic setting.

5.4.4 Do Flow Architectures Matter?

We have shown PCBO with Neural Spline Flows, but what if we swap the backbone — does the story change? To test this, we reran the medical case study with RealNVP and Residual flows, asking whether the choice of flow affects the causal graph, the preferences, or both.

The answer is clear: the graph barely moves, but the preference model does. Structure learning is driven by pairwise comparisons and the exploration policy; as long as the flow fits a reasonable likelihood, edge posteriors separate and thresholding does the rest. By contrast, utility shape and calibration speed depend directly on the flow's inductive bias.

Across NSF, RealNVP, and Residual, the recovered graph is essentially identical under the same budget and thresholds. SHD and edge counts fall in the same narrow range, and the same edges carry high probability (full reconstructions in Appendix 6.3, 6.4). The differences appear only in how the preference model evolves over time.

All prediction accuracy curves are rough (by design we keep querying hard comparisons), but they settle differently. RealNVP climbs early and then sits on a stable plateau. Residual swings wider and takes longer to stabilise. NSF sits between the two: steady, with mild oscillations.

The noise scale s tracks how confident the model is about each comparison. RealNVP spends most of the run near $s \approx 1$, meaning good calibration from early on. Residual starts lower (too "sure" at first), then rises towards 1 as it learns the actual difficulty. NSF stays close to 1 throughout.

Regarding the learned utility distribution (plots in Appendix 6.5, the flows place probability mass in visibly different patterns. RealNVP concentrates density tightly in a compact region, producing crisper preference peaks. Residual keeps broader support, often with gentle ridges, keeping more alternatives viable longer. NSF is tight but slightly more flexible than RealNVP, giving a compact yet smooth shape.

These differences do not affect the final graph, but they matter if preferences themselves are of interest, for instance, when ranking candidates or recommending interventions. In practice, the choice of flow depends on priorities. If the goal is only the causal graph, any of the three backbones suffices. If sharper, stable preference predictions are needed quickly, RealNVP (or NSF) is the safer bet. If broader exploration is valuable before committing, Residual keeps options open for longer before converging.

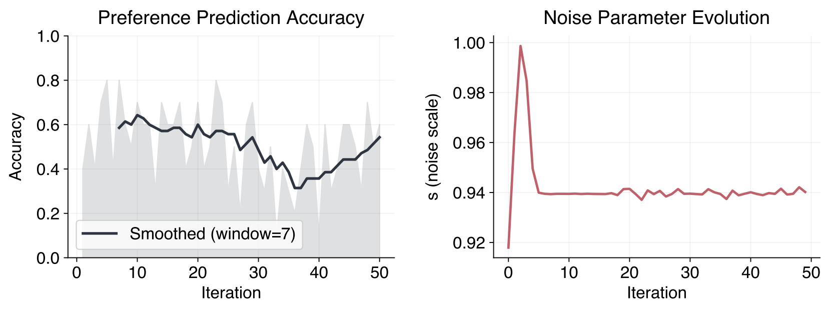


Figure 5.10: Preference learning dynamics on the medical dataset (RealNVP).

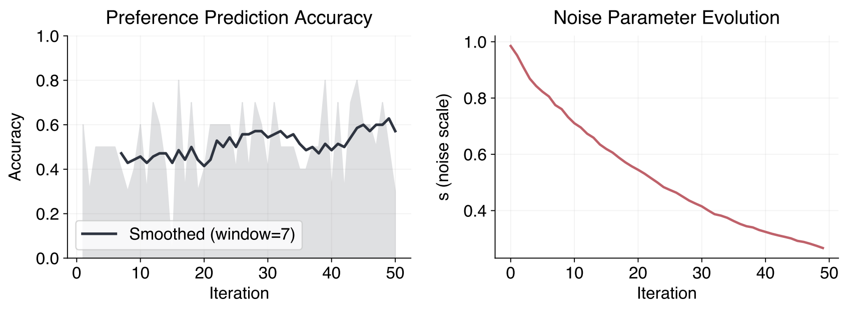


Figure 5.11: Preference learning dynamics on the medical dataset (Residual Flow).

Chapter 6

Conclusions and Future Work

6.1 Contributions

This work began from a simple but ambitious question: can causal discovery be done from preferences alone, without ever observing outcomes directly? That constraint shaped the whole thesis, and is what led us to build a new framework, Preferential Causal Bayesian Optimisation.

Now, at the end of the journey, the natural follow-up is: what has been achieved? The answer is not a single result, but a set of steps that together show that causal discovery from preferences is not only possible, but genuinely attainable.

6.1.1 Technical Innovations

Answering the preference–causality question required more than just applying existing tools. Along the way, we had to make several design choices and develop new components that gave PCBO its shape.

The first was to reframe causal discovery as a preferential problem in the first place. Instead of assuming access to full outcome data, we built everything around comparisons: which intervention leads to a better result? That shift alone changes the landscape, and it forced us to rethink how structure learning can be driven by weaker, noisier signals.

From there, the Bayesian backbone became essential. Rather than chasing a single graph, PCBO maintains distributions: over edges, over parent sets, over preferences. This choice brought robustness, but it also introduced the challenge of scale. To keep inference feasible as the number of nodes grew, we introduced the scalable parent posterior – a mechanism that approximates structural probabilities without collapsing under combinatorics. This is what allowed PCBO to move from toy graphs to settings with twenty nodes or more.

A second line of innovation came from the preference model itself. Using normalising flows gave the system the flexibility to capture nonlinear utilities and calibrated uncertainty. We explored different flow families (RealNVP, Residual, Neural Spline) and showed that while the graphs remain stable, the learned preference distributions reflect the inductive biases of each architecture. This makes PCBO adaptable: it can prioritise sharp predictions, broad exploration, or a balance of both, depending on the flow.

Finally, integration mattered as much as the parts. PCBO is not just a graph learner with a flow attached, but a framework where preference modelling, acquisition policies, and Bayesian updates work together. Each piece supports the others: preferences drive the policy, the policy guides which comparisons are most informative, and the Bayesian update ties it all back into the structure. It is this loop, rather than any single component, that makes the framework work.

6.1.2 Empirical Findings

Building a framework is one thing, showing that it actually works is another. The evaluation was where PCBO had to prove itself – across controlled graphs, larger synthetic structures, and finally a realistic medical case study.

The synthetic experiments were the natural starting point. They gave us control, with ground-truth graphs in hand. The message was clear: PCBO does not just guess. On small graphs it recovers the truth exactly; on medium-sized settings it stays competitive with outcome-based baselines; and even as the scale grows to fifteen and twenty nodes, it remains stable, producing plausible structures where search could collapse. The lesson was less about perfection and more about robustness: PCBO consistently finds its way, even with weaker supervision.

The medical case study then pushed the framework into a more realistic and interpretable setting. The result was remarkable: with nothing more than preference data, PCBO came within one edge of the true causal graph. Thresholding proved decisive, trimming away noise and leaving a structure that matched the ground truth almost perfectly. What mattered most was not just accuracy, but the kind of mistake it made: including an extra edge where two variables are naturally correlated, rather than omitting a genuine cause of the outcome. In practice, that trade-off may be the safer one.

The comparisons to baselines made the story clearer. Methods trained on richer intervention outcomes did not clearly outperform PCBO; in many cases they fell short. LASSO came close, but missed a true causal link. NOTEARS, often a strong general-purpose learner, also failed to recover key parents of the outcome. PCBO, working only from preferences, managed to stay not just competitive but often ahead.

Finally, the ablations gave depth. The choice of flow architecture left the recovered graphs essentially unchanged, but it shaped how preferences were modelled: tighter peaks, broader support, faster or slower calibration. These differences did not shift the structural results, but they matter if preferences themselves are the focus, for instance in ranking or decision support.

Together, the findings show that causal discovery from preferences is not only a theoretical possibility. It is practical, robust, and competitive with methods that rely on stronger supervision. PCBO does not just keep up: it provides an alternative path to learning causality.

6.2 Limitations and Future Directions

No framework comes without its trade–offs, and PCBO is no exception. The work so far has shown that causal discovery from preferences is possible, but it would be naïve to stop here and call the problem solved. Each choice we made — from the Bayesian backbone to the flow family and the thresholding rules — came with assumptions that shaped the outcome. Some worked in our favour, others drew clear boundaries.

At the same time, those boundaries are not dead ends but starting points. Every project leaves behind new questions, and PCBO is no different. Some are technical, about making the method faster, more stable, or more flexible. Others are broader, about where such a framework could matter most once it steps outside controlled experiments and into the messier settings of real data.

6.2.1 Assumptions and Scope

Every method begins with assumptions, and PCBO is no exception. Some of these are deliberate choices, others are simplifications we had to make in order to build a working framework.

The first and most obvious assumption is that preferences are available, reliable, and informative. In practice, that might not always be the case. Preferences can be noisy, biased by how they are collected, or sparse if only a few comparisons exist. PCBO can tolerate some of this noise, but its strength relies on having a signal consistent enough to guide the learning process.

A second assumption is that the separation between strong and weak edges will be clear enough for thresholding to succeed. In our experiments this worked remarkably well: the edge posteriors split into two groups, near-zero and near-one, making it easy to cut through the noise. But this is not guaranteed in every setting. If the probabilities cluster in the middle, deciding where to draw the line becomes harder, and the recovery less certain.

Finally, we assumed causal sufficiency, meaning all relevant variables are observed and no hidden confounders exist. This is a standard simplification in causal discovery, but it rarely holds in reality. Real systems often contain unobserved factors that influence multiple nodes, and handling those requires extensions beyond the current scope of PCBO.

These assumptions do not invalidate the method, but they do set its boundaries. They define the conditions under which PCBO works well, and point out areas where more work is needed to push the framework closer to the messiness of the real world.

6.2.2 Technical Improvements

If assumptions set the stage, the difficulties we faced show where the pressure points appear. Still, these limits point naturally to the next technical steps forward.

One challenge is scalability. With the scalable parent posterior, PCBO already goes beyond toy settings and handles graphs with fifteen or twenty nodes. But the approximation is still heavy, and the cost grows quickly as graphs get denser. Future work needs faster inference strategies – whether sharper approximations, smarter projection rules, or more parallel updates – so that the framework can move into the scale of genomics or social networks without losing accuracy.

A second point is supervision. In its current form, PCBO relies on preferences alone for the utility model. This makes it robust when outcome values are unreliable or costly, but it also means leaving useful information aside when outcomes are available. Blending both signals (comparisons and direct outcomes) could improve stability and efficiency, while keeping the advantages of preference-driven discovery.

Thresholding is another area where improvement is possible. A single hard cut worked well in our experiments, especially with Beta-mixture fits, but in practice the right level depends on the domain. In high–risk medical settings, one might demand near–certainty before accepting an edge, while exploratory science could afford a looser threshold. Automatic, adaptive strategies would let PCBO tune this balance more reliably, especially in high-dimensional systems where visual inspection is not an option.

Finally, robustness remains an open line. Real-world data are rarely clean: preferences may be inconsistent, and confounders may hide in the background. More resilient models – able to tolerate noisy judgments, detect hidden variables, or adapt when assumptions break – would make PCBO not only a proof of concept but a trustworthy tool for practice.

6.2.3 Applications and Impact

If the technical side asks how PCBO can be improved, the applied side asks where it can matter most. The whole point of building such a framework is not only to prove a concept, but to find places where preferences are abundant, outcomes limited, and decisions high-stakes.

Healthcare is perhaps the clearest candidate. Doctors often compare treatments not only on outcomes like survival or recovery time, but on subtler factors, such as side effects, patient comfort, long-term risks. Many of these judgments are inherently comparative: "this drug is easier to tolerate than that one", "this dosage feels safer". PCBO could give structure to those comparisons, revealing the underlying causal web that guides them. Policy is another rich ground. Governments and institutions rarely have clean outcome data when designing interventions, but they do have expert panels, stakeholder opinions, and preference surveys. Instead of discarding those as "soft" data, PCBO suggests they can be organised into a structured framework for causal reasoning. Imagine comparing policy A against policy B not just in theory, but with the aim to find the causes that matter most.

Even in more commercial domains, the idea is powerful. Recommendation systems, for instance, already live on preferences. Today they rank options; tomorrow they might learn the hidden causes behind those preferences, and make the system not just predictive but explanatory.

Anywhere comparisons are easier to gather than clean outcomes, PCBO could find a home. Its promise is not to replace existing methods, but to adapt causal discovery to the many domains where preferences are the only reliable signal.

6.2.4 Methodological Extensions

The version of PCBO presented here lives in the clean world of static DAGs, pairwise comparisons, and controlled datasets. Real applications are rarely that neat. If this framework is to move closer to practice, several extensions naturally suggest themselves.

One obvious direction is to move beyond static DAGs. Many real systems are not one-way streets but have feedback loops, evolve over time, or follow continuous dynamics. Adapting PCBO to cyclic or temporal models would make it relevant to domains such as biology or economics, where change and interaction are constant rather than exceptional.

Another is to place humans directly in the loop. The experiments here assume preferences arrive automatically, but in practice they may come from experts. Doctors comparing treatment plans, policymakers weighing interventions, or users rating options all generate preferences. Making PCBO interactive, with active learning strategies that query experts in the most informative way, could bring the framework into real workflows, where data are scarce but expert judgment is valuable.

Robustness is another natural next step. Preferences are not perfect: people misjudge, data are noisy, and models can be misspecified. A more robust version of PCBO would explicitly account for hidden confounders or inconsistent judgments, to ensure that the conclusions remain stable even when the inputs are messy. This is not only a safeguard but also a step toward making preference-based causal discovery trustworthy outside the lab.

Closing Remarks

PCBO showed that preferences, often seen as weak signals, can be enough to find strong causal structure. The framework is not the final word, it leaves open questions of scale, realism, and robustness, but that is precisely what makes it meaningful. Its value lies in proving that causality can be learned in places where outcomes are noisy or out of reach.

It all comes down to this: if even comparisons alone can guide us to the right causes, then causal discovery may be possible in far more settings than we thought. That possibility is what this work leaves behind.

Bibliography

1. Pearl J. Causality: Models, Reasoning and Inference. 2nd ed. Cambridge University Press, 2009
2. Hauser A and Bühlmann P. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. Journal of Machine Learning Research 2012; 13:2409–64
3. Eberhardt F and Scheines R. Interventions and causal inference. *Proceedings of the 22nd AAAI Conference on Artificial Intelligence.* AAAI Press. 2007 :1157–62
4. Peters J, Janzing D, and Schölkopf B. Elements of Causal Inference: Foundations and Learning Algorithms. Cambridge, MA: MIT Press, 2017
5. Cooper GF and Herskovits E. A Bayesian Method for the Induction of Probabilistic Networks from Data. Machine Learning 1992; 9:309–47
6. Heckerman D. A Tutorial on Learning with Bayesian Networks. Tech. rep. MSR-TR-95-06. Microsoft Research, 1995
7. Koller D and Friedman N. Probabilistic Graphical Models: Principles and Techniques. MIT Press, 2009
8. Geiger D and Heckerman D. Learning Gaussian Networks. *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI).* 1994 :235–43
9. Kuipers J, Moffa G, and Heckerman D. Addendum on the Scoring of Gaussian Directed Acyclic Graph Models. Biometrika 2014; 101:1006–9
10. Friedman N and Koller D. Being Bayesian about Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks. *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI).* 2003 :201–10
11. Tong S and Koller D. Active Learning for Structure in Bayesian Networks. *Proceedings of the 20th International Conference on Machine Learning (ICML).* 2003 :863–70
12. He Y and Geng Z. Active Learning of Causal Networks with Intervention Experiments and Optimal Designs. *Journal of Machine Learning Research Workshop and Conference Proceedings: AISTATS.* 2008 :514–21
13. Aghaei R et al. Active Preference Learning for Causal Structure Discovery. *Proceedings of the 39th International Conference on Machine Learning (ICML) Workshop on CausalML.* 2022

14. Murphy K, Tong S, Koller D, and Jordan MI. Active Learning of Causal Bayesian Network Structure. *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI).* 2001 :378–85

15. Choi J, Chen Y, and Welling M. Active Structure Learning of Bayesian Networks via Mutual Information and Variational Approximations. *Conference on Uncertainty in Artificial Intelligence (UAI).* 2020

16. Wang Y et al. Active Learning for Causal Structure with Score Decompositions. *Advances in Neural Information Processing Systems (NeurIPS).* 2021

17. Aglietti V, Lu X, Paleyes A, and González J. Causal Bayesian Optimization. 2020. arXiv: `2005.11741 [stat.ML]`. Available from: `https://arxiv.org/abs/2005.11741`

18. Thurstone LL. A Law of Comparative Judgment. Psychological Review 1927; 34:273–86

19. McFadden D. Conditional Logit Analysis of Qualitative Choice Behavior. *Frontiers in Econometrics.* Ed. by Zarembka P. Academic Press, 1973 :105–42

20. Luce RD. Individual Choice Behavior: A Theoretical Analysis. John Wiley & Sons, 1959

21. Plackett RL. The Analysis of Permutations. Journal of the Royal Statistical Society: Series C (Applied Statistics) 1975; 24:193–202

22. Christiano PF, Leike J, Brown T, Martic M, Legg S, and Amodei D. Deep Reinforcement Learning from Human Preferences. *Advances in Neural Information Processing Systems (NeurIPS).* 2017

23. Rezende DJ and Mohamed S. Variational Inference with Normalizing Flows. *Proceedings of the 32nd International Conference on Machine Learning (ICML).* 2015 :1530–8

24. Dinh L, Sohl-Dickstein J, and Bengio S. Density Estimation Using Real NVP. *International Conference on Learning Representations (ICLR).* 2017

25. Mikkola P, Acerbi L, and Klami A. Preferential Normalizing Flows. 2024. arXiv: `2410.08710 [cs.LG]`. Available from: `https://arxiv.org/abs/2410.08710`

26. Spirtes P, Glymour CN, and Scheines R. Causation, Prediction, and Search. 2nd ed. MIT Press, 2000

27. Spirtes P, Meek C, and Richardson T. An Algorithm for Fast Recovery of Sparse Causal Graphs. Social Science Computer Review 1999; 17:62–72

28. Kalisch M and Bühlmann P. Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm. Journal of Machine Learning Research 2007; 8:613–36

29. Chickering DM. Optimal Structure Identification with Greedy Search. Journal of Machine Learning Research 2002; 3:507–54

30. Chickering DM. Learning Bayesian Networks is NP-Complete. *Learning from Data: Artificial Intelligence and Statistics V.* Ed. by Fisher D and Lenz HJ. Springer, 1996 :121–30

31. Zheng X, Aragam B, Ravikumar P, and Xing EP. DAGs with NO TEARS: Continuous Optimization for Structure Learning. *Advances in Neural Information Processing Systems (NeurIPS).* 2018

32. Zheng X, Dan C, Aragam B, Ravikumar P, and Xing EP. Learning Sparse Nonlinear Dynamics with Stochastic Interventions. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS).* 2020

33. Pamfil AR, Colombo D, Kuipers J, Maathuis MH, and Saitta S. Dynotears: Structure Learning from Time-Series Data. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS).* 2020

34. Yamada M, Shimodaira H, et al. Causal Discovery with Latent Confounders: A Robust Continuous Optimization Approach. *Advances in Neural Information Processing Systems (NeurIPS).* 2020

35. Papamakarios G, Nalisnick E, Rezende DJ, Mohamed S, and Lakshminarayanan B. Normalizing Flows for Probabilistic Modeling and Inference. Foundations and Trends in Machine Learning 2021; 14:1–236

36. Khemakhem I, Kingma DP, Monti RP, and Hyvärinen A. Variational Autoencoders and Nonlinear ICA: A Unifying Framework. *Proceedings of the 37th International Conference on Machine Learning (ICML).* 2020

37. Dinh L, Sohl-Dickstein J, and Bengio S. Density estimation using Real NVP. 2017. arXiv: `1605.08803 [cs.LG]`. Available from: `https://arxiv.org/abs/1605.08803`

38. Chen RTQ, Behrmann J, Duvenaud D, and Jacobsen JH. Residual Flows for Invertible Generative Modeling. 2020. arXiv: `1906.02735 [stat.ML]`. Available from: `https://arxiv.org/abs/1906.02735`

39. Durkan C, Bekasov A, Murray I, and Papamakarios G. Neural Spline Flows. 2019. arXiv: `1906.04032 [stat.ML]`. Available from: `https://arxiv.org/abs/1906.04032`

40. Sutton RS and Barto AG. Reinforcement Learning: An Introduction. 2nd. MIT Press, 2018

41. Kingma DP and Ba J. Adam: A Method for Stochastic Optimization. 2017. arXiv: `1412.6980 [cs.LG]`. Available from: `https://arxiv.org/abs/1412.6980`

42. Loshchilov I and Hutter F. SGDR: Stochastic Gradient Descent with Warm Restarts. *International Conference on Learning Representations (ICLR).* 2017. Available from: `https://openreview.net/forum?id=Skq89Scxx`

43. Erdős P and Rényi A. On random graphs I. Publicationes Mathematicae 1959; 6:290–7

Declarations

Use of Generative AI

I acknowledge the use of ChatGPT-5 (OpenAI, `https://chat.openai.com/`) to support the writing process of this thesis. The system was used primarily to refine phrasing, improve clarity, and restructure sections for readability. It was not used to generate original research ideas, experiments, or analysis. All technical content, results, and interpretations presented in this thesis are my own work. Any AI-assisted text was carefully reviewed, edited, and integrated to ensure accuracy, originality, and consistency with the research.

Ethical Considerations

This thesis does not involve human participants, personal data, or sensitive information, and therefore raised no ethical risks or requirements for formal approval. All experiments were conducted on synthetic datasets and stylised case studies designed for methodological evaluation. The medical-inspired dataset is entirely simulated and does not use real patient data.

Sustainability

To reduce environmental impact, computations were carried out on local university resources and the Imperial College HPC cluster, making use of shared CPU nodes rather than large-scale cloud deployments. Training was kept to the scale necessary for experimental evaluation, avoiding unnecessary repetition or oversized models. The thesis was written in LaTeX, an efficient typesetting system, to ensure a lightweight and sustainable workflow.

Appendix

Reproducibility Instructions

The full implementation of PCBO, together with experiment scripts and figure generation, is provided at this GitHub repository. The following steps outline how to reproduce the main results presented in this thesis.

Setup.

- Clone the repository:

  ```
  git clone https://github.com/ai4ai-lab/preferential-cbo.git
  cd preferential-cbo
  ```

- Install Python dependencies (Python 3.9+ recommended):

  ```
  pip install -r requirements.txt
  ```

Running experiments.

- The main entry point for experiments is the notebook `demo_pcbo.ipynb`, which contains a complete workflow: data loading, running PCBO, and visualising results.
- Graph families and datasets can be selected by editing the notebook cells. Examples include the 3-node, 6-node, Erdős–Rényi graphs, and medical graphs.
- For larger-scale tests, the Python modules (`acquisition.py`, `parent_posterior.py`, `parent_posterior_scalable.py`, etc.) can be called directly within custom scripts.

Figures and tables.

- Figures generated during experiments are stored under `figures/`, organised by flow family (e.g. `neural_spline/`, `real_nvp/`, `residual/`) and dataset type (e.g. `3_nodes/`, `medical/`).
- The corresponding tables are saved under `figures/tables/`.

Reproducing results. To replicate the main plots in the thesis, run `demo_pcbo.ipynb` with the appropriate dataset and flow family selected. All experiments can be rerun using the provided scripts without modification. Hyperparameters match those reported in Chapter 4.

Table 6.1: Edge-wise counts for PCBO variants (medical dataset, neural spline flow).

Method	TP	FP	FN	TN
PCBO	12	16	0	28
PCBO ($\tau^* = 0.99$)	12	2	0	42
PCBO ($\tau = 1.00$)	12	1	0	43

Table 6.2: Edge-wise counts for baseline methods (medical dataset, neural spline flow).

Method	TP	FP	FN	TN
Random	3	2	9	42
LASSO	11	0	1	44
PC	0	2	12	42
NOTEARS-Lite	9	0	3	44
Causal Sufficiency	7	4	5	40
Fully Connected	12	16	0	28

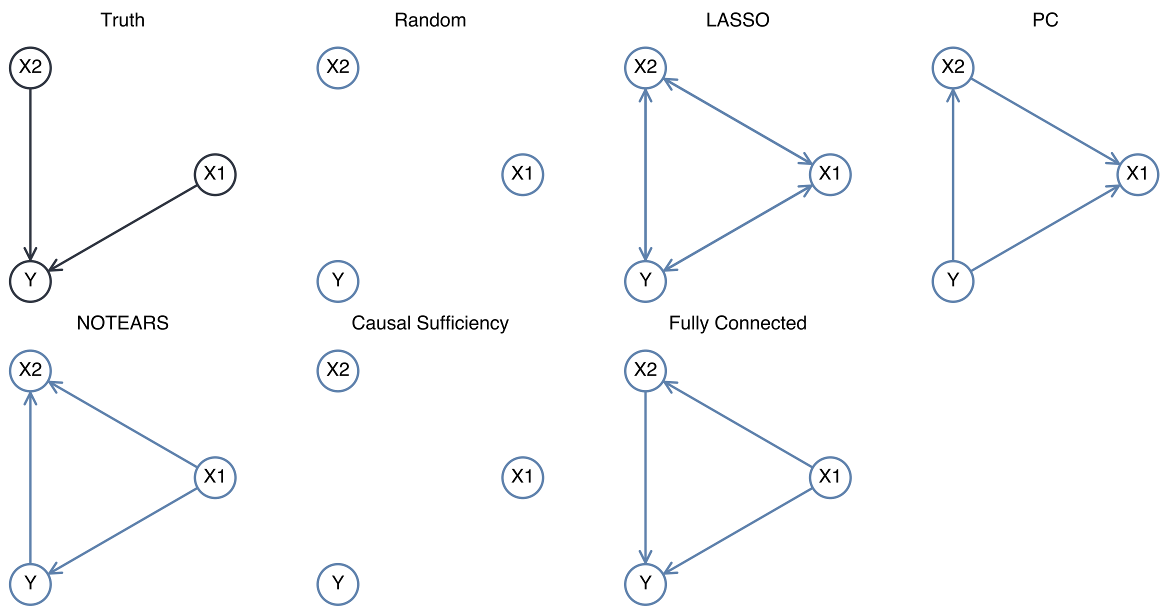


Figure 6.1: Baseline graph reconstructions on the 3-node dataset.

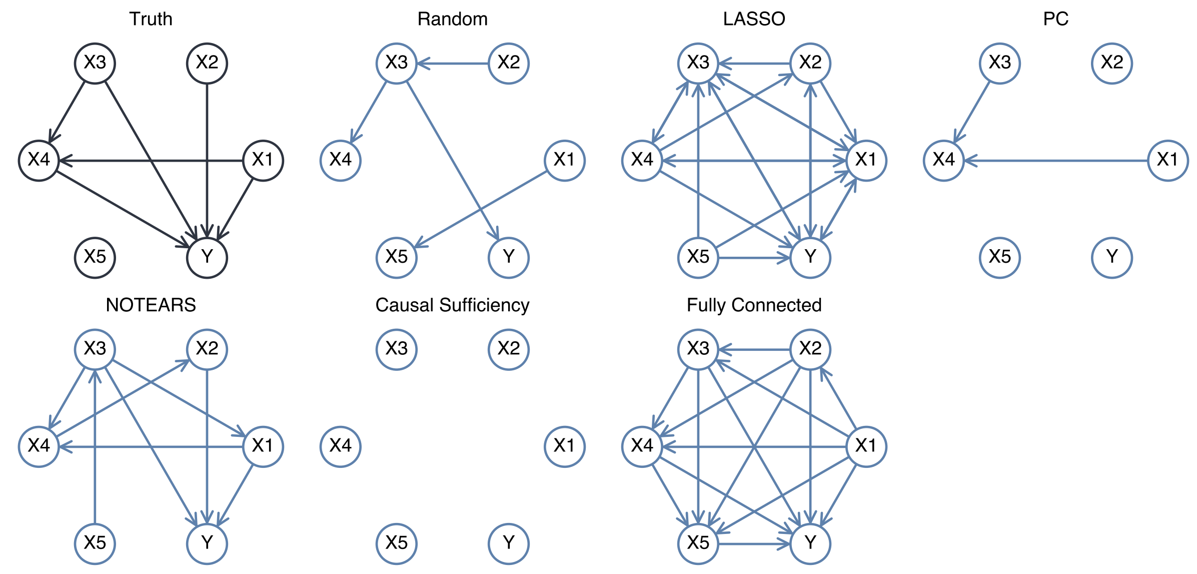


Figure 6.2: Baseline graph reconstructions on the 6-node dataset.

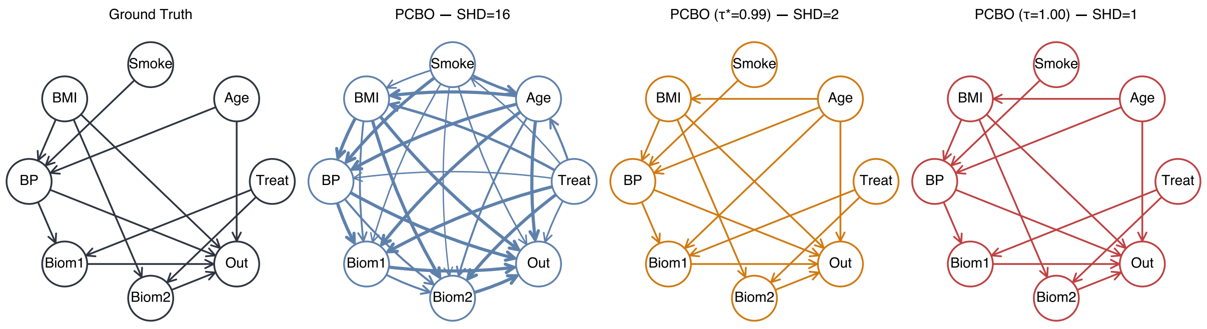


Figure 6.3: Causal graphs on the medical dataset under RealNVP Flows.

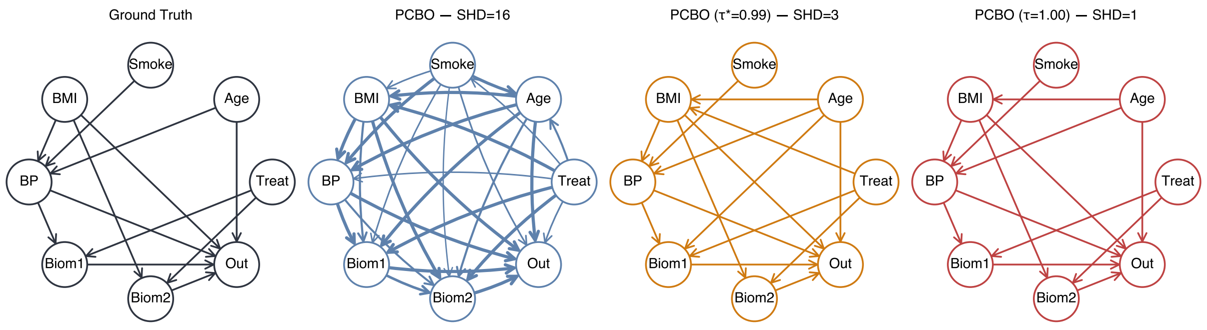
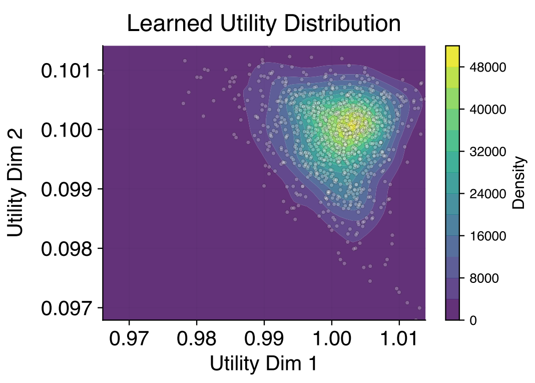
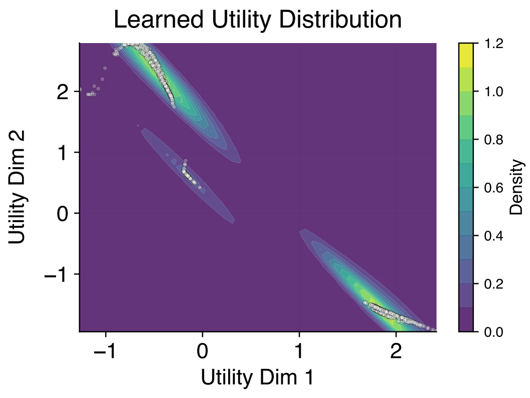


Figure 6.4: Causal graphs on the medical dataset under Residual Flows.



(a) RealNVP flow



(b) Residual flow

Figure 6.5: Preference learning dynamics with different flow backbones on the medical dataset.