Conformal Validity Guarantees Exist for Any Data Distribution (and How to Find Them)

Drew Prinster[*1] **Samuel Stanton**[*2] **Anqi Liu**[1] **Suchi Saria**[1]

Abstract

As artificial intelligence (AI) / machine learning (ML) gain widespread adoption, practitioners are increasingly seeking means to quantify and control the risk these systems incur. This challenge is especially salient when such systems have autonomy to collect their own data, such as in black-box optimization and active learning, where their actions induce sequential feedback-loop shifts in the data distribution. Conformal prediction is a promising approach to uncertainty and risk quantification, but prior variants' validity guarantees have assumed some form of "quasi-exchangeability" on the data distribution, thereby excluding many types of sequential shifts. In this paper we prove that conformal prediction can theoretically be extended to *any* joint data distribution, not just exchangeable or quasi-exchangeable ones. Although the most general case is exceedingly impractical to compute, for concrete practical applications we outline a procedure for deriving specific conformal algorithms for any data distribution, and we use this procedure to derive tractable algorithms for a series of AI/ML-agent-induced covariate shifts. We evaluate the proposed algorithms empirically on synthetic black-box optimization and active learning tasks.

1. Introduction

Quantifying the uncertainty of predictions from artificial intelligence (AI) and machine learning (ML) models is often imperative to managing the risks associated with their downstream use. Principled uncertainty quantification (UQ) is thus especially crucial in real-world scenarios where AI/ML systems are used to inform high-stakes decisions, or where they may even act autonomously. Unfortunately, it is precisely when uncertainty estimation is most vital when it is often most difficult to quantify reliably. For a prime example, consider the increasingly common setting where AI systems are able to take actions of their own, thereby transforming them from passive observers into active agents: common instances include active learning, black-box optimization, and reinforcement learning. In all these cases, merely enabling an AI model to explore by actively selecting the next datapoint induces feedback-loop shifts in the data distribution, which can accumulate over time and cause standard UQ methods to deteriorate or break down entirely.

Conformal prediction (CP) (Vovk et al., 2005) is an increasingly popular UQ framework because it provides coverage guarantees without knowledge of the exact distributional form of the data. Even so, standard CP methods do make one very strong assumption about the data distribution: namely, that the data are *exchangeable* (IID is a special case). Informally, for time-series data, exchangeability means the distribution is "time invariant"; meanwhile in a batch setting, it means the distribution remains constant between training and test batches. Exchangeability thus excludes many real-world settings such as dynamic time series or data generated from (or queried by) active AI agents, which can exhibit stark time-dependent or batch-dependent distribution shifts.

This work builds on that of Tibshirani et al. (2019), who introduced a weighted generalization of CP where the traditional exchangeability requirement is relaxed to a condition called *weighted exchangeability*. For instance, data that are independent but not identically distributed are weighted exchangeable. However, the conventional wisdom is still that conformal prediction methods require some notion of "quasi-exchangeability" to achieve a valid coverage guarantee.[1] We challenge this common perception via the following theoretical and practical contributions:

- Our main theoretical contribution is to present a general view of (weighted) conformal prediction, drawing

[*]Equal contribution [1]Department of Computer Science, Johns Hopkins University, Baltimore, MD, U.S.A. [2]Prescient Design, Genentech, New York City, NY, U.S.A.. Correspondence to: Drew Prinster <drew@cs.jhu.edu>, Samuel Stanton <stanton.samuel@gene.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

[1]For instance, Fannjiang et al. (2022) introduced a similar assumption called "pseudo-exchangeability" to study data shifts with limited dependencies of the test point on the observed data.

heavily upon the analysis of Tibshirani et al. (2019). In particular, we prove that conformal prediction can in theory be extended to *any* joint distribution of data with a valid probability density function f, including non-exchangeable ones with sequential dependencies, although the fully general form is not practically feasible to compute (factorial runtime and requiring knowledge of f). From this general perspective, "quasi-exchangeability" definitions (including exchangeability and weighted exchangeability) are no longer necessary conditions for conformal prediction theory; instead, from this view, these assumptions' only role is to specify conditions for practical tractability.

- For practical applications, we present a general procedure for deriving specific weighted CP algorithms for any given data distribution, and we demonstrate how this procedure can be used to derive weighted CP methods with valid coverage guarantees for sequential feedback-loop covariate shifts that characterize common AI/ML agent scenarios. We moreover show that these methods can be realized in practice via empirical evaluation on *in silico* protein design and active learning tasks, which verify that we maintain coverage when state-of-the-art baselines fail.

2. Background

Assume we are given an initial dataset $Z_{1:n} := \{Z_i\}_{i=1}^n$, where $Z_i := (X_i, Y_i) \in \mathcal{X} \times \mathcal{Y}$ are real-valued ($\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^p \times \mathbb{R}$) feature-label pairs with some distribution function $P_Z^{(0)}$. At each future time t, we observe a test point's features X_{n+t} and are interested in rigorous UQ for an ML estimate of the unknown true label Y_{n+t}. For simplicity we interpret t as a timestep, but in general each datapoint only needs an identifying index; also, we often let $t = 1$ to focus on a single test point indexed $n + 1$. When distinction is needed, capital letters (e.g., Z_i) denote random variables, and lower case (e.g., z_i) denote observed values.

2.1. Standard Conformal Prediction

Conformal prediction (CP) is a framework for *predictive inference*: the problem of computing a predictive confidence set $\widehat{\mathcal{C}}_n(x)$ (in regression, this is often an interval) likely to contain the true label at least at a user-specified target rate $1 - \alpha \in (0, 1)$. This desiderata is called *valid coverage*:[2]

$$\mathbb{P}\big\{Y_{n+1} \in \widehat{\mathcal{C}}_n(X_{n+1})\big\} \geq 1 - \alpha. \tag{1}$$

For an accessible introduction to CP see Angelopoulos et al. (2022) or for the definitive reference see Vovk et al. (2005). Computing conformal predictive sets requires access to a real-valued score function $\mathcal{S}$ that takes as its inputs a point (x, y) and a "bag"[3] of other examples $\bar{Z}$, where $\mathcal{S}\big((x, y), \bar{Z}\big)$ can be thought of as quantifying how "nonconforming" or "strange" the point (x, y) is relative to $\bar{Z}$. For example, the absolute value residual $|\widehat{\mu}_{\bar{Z}}(x) - y|$ is a common score function, where $\widehat{\mu}_{\bar{Z}}$ is an ML predictor trained on the examples $\bar{Z}$. For shorthand, we will sometimes write the fitted score function as $\widehat{\mathcal{S}}(x, y) = \mathcal{S}\big((x, y), \bar{Z}\big)$ and outputted score values as $V_i = \widehat{\mathcal{S}}(Z_i)$. With $Q_{1-\alpha}$ denoting an empirical quantile evaluated at level $1 - \alpha$, a standard CP set $\widehat{\mathcal{C}}_n(x)$ is then constructed by taking a (conservative) quantile on the empirical distribution of score values for each $y \in \mathcal{Y}$:

$$\widehat{\mathcal{C}}_n(x) = \Big\{y \in \mathcal{Y} : \widehat{\mathcal{S}}(x, y) \leq \mathrm{Q}_{1-\alpha}\big(V_{1:n} \cup \{\infty\}\big)\Big\}. \tag{2}$$

Standard CP methods assume that all the data $Z_1, \ldots, Z_{n+1}$ are *exchangeable*—that is, that their joint distribution is invariant to permutations, a special case being independent and identically distributed (IID). Under exchangeability, $\widehat{\mathcal{C}}_n(x)$ in Eq. (2) achieves the valid coverage guarantee in Eq. (1) with finite data samples. This standard CP guarantee is *distribution-free* in the sense of making no assumptions about the specific parametric form of the data distribution—this is not to be confused with robustness to distribution *shift*, which violates exchangeability for example when the distribution of Z_{n+1} differs from that of $Z_{1:n}$. Exchangeability is thus still a strong assumption in its own way. For instance, it excludes many environment-dependent or time-dependent distribution shifts which often occur in practice.

2.2. Weighted Conformal Prediction for Covariate Shift

Tibshirani et al. (2019) presented an alternate proof of the conformal coverage guarantee in a way that isolated the role of exchangeability and allowed them to generalize conformal prediction to a broader class of distributions they termed *weighted exchangeable*. In particular, they thoroughly developed a specific weighted CP algorithm for the *standard covariate shift* setting (Shimodaira, 2000), where the marginal distribution of the inputs X may differ independently between training and test distributions, but the conditional distribution $Y \mid X$ is assumed to be invariant:

$$(X_i, Y_i) \overset{\text{i.i.d.}}{\sim} P_Z^{(0)} = P_X^{(0)} \times P_{Y|X},\ i = 1, \ldots n,$$
$$(X_{n+1}, Y_{n+1}) \sim P_Z^{(1)} = P_X^{(1)} \times P_{Y|X},\ \text{independently.}$$

Note that in standard covariate shift the test distribution $P_Z^{(1)}$ is *independent* from the training data; it is fixed *a priori* and cannot change depending on different draws of $Z_{1:n}$. Tibshirani et al. (2019)'s weighted CP algorithm for

[2] We focus on marginal coverage (i.e., on average over many repeated experiments) rather than conditional coverage; see Foygel Barber et al. (2021) for more details on this distinction.

[3] That is, a multiset. CP methods assume the score-fitting algorithm is symmetric with respect to permutations of its training inputs $\bar{Z}$; we maintain this assumption unless stated otherwise.

standard covariate shift generalizes Eq. (2) with likelihood (or density) ratio function weights $w(x) = p_X^{(1)}(x)/p_X^{(0)}(x)$.

Fannjiang et al. (2022) followed the overall approach of Tibshirani et al. (2019) to derive a specific weighted CP algorithm with a valid coverage guarantee (Eq. (1)) in a setting they called (one-step) *feedback covariate shift* (FCS), where $P^{(1)}_{X;Z_{1:n}}$ is explicitly *dependent* on the realized draw $Z_{1:n}$. While this work was an important extension of CP to dependent data, they did not consider a more general "multistep" FCS case that could describe how data shifts may accumulate over time, for instance in active learning. Their focus on the one-step FCS case could be because they posit as the basis of their theoretical analysis an assumption they call *pseudo-exchangeability*, which explicitly characterizes one-step FCS but does not formally describe its multistep analog (see Appendix D.3), which we introduce next.

2.3. Multistep Feedback Covariate Shift

Whereas standard and feedback covariate shift are "one-step" shifts between training and test data batches, in this work we focus our practical evaluations on a more general setting we call *multistep feedback covariate shift* (MFCS). MFCS is not only a clear example of a sequentially-dependent data distribution that goes far beyond exchangeability, but it moreover describes important instances of feedback-loop shifts inherent to the actions of AI/ML agents. For each time $t \in \{1, ..., T\}$, MFCS allows X_{n+t} to change depending on the past data $Z_{1:(n+t-1)} = \{Z_i\}_{i=1}^{n+t-1}$, while assuming that $Y|X$ remains invariant. More precisely, MFCS assumes

$$(X_i, Y_i) \overset{\text{i.i.d.}}{\sim} P_X^{(0)} \times P_{Y|X},\ i = 1, \dots n, \tag{3}$$
$$(X_{n+t}, Y_{n+t}) \sim P^{(t)}_{X;Z_{1:(n+t-1)}} \times P_{Y|X},\ t = 1, \dots, T,$$

where n is the size of an IID initialization, and t is the number of subsequent MFCS steps. Setting $T = 1$ reduces Eq. (3) to one-step FCS (Fannjiang et al., 2022) and moreover to standard covariate shift if independence is further assumed. MFCS is commonly induced by ML agents able to actively select each datapoint X_{n+t}, informed by prior training observations, so long as the predictive goal or "concept" $Y|X$ remains the same. For instance, MFCS describes active learning, where the goal is to efficiently construct a training corpus to improve model performance, and black-box optimization, where the ML model is an instrument to optimize X with respect to some unknown cost (or utility) function only indirectly accessible by each observation of Y.

3. Related Work

Conformal Prediction Under Distribution Shift Prior work can largely be categorized into variants of *weighted* CP (Tibshirani et al., 2019; Podkopaev & Ramdas, 2021; Xu & Xie, 2021; Fannjiang et al., 2022; Prinster et al., 2022; 2023; Barber et al., 2023; Farinhas et al., 2023; Nair & Janson, 2023; Yang et al., 2024) and *adaptive* CP (Gibbs & Candès, 2021; 2022; Zaffran et al., 2022; Angelopoulos et al., 2023; Feldman et al., 2023; Bhatnagar et al., 2023). The former exploits knowledge or estimates of the data shift to proactively adjust inference and thereby achieve guarantees that hold at inference time; meanwhile, the latter aims to maintain a target risk threshold by retroactively responding to threshold violations and thus obtain guarantees most meaningful "in the long run" rather than at inference time.

This work builds on the weighted CP literature, and our practical applications focus on cases of MFCS where the shift is agent-induced and thus known. Aside from Tibshirani et al. (2019) and Fannjiang et al. (2022), Prinster et al. (2022) and Prinster et al. (2023) also studied CP under (one-step) standard or feedback covariate shift by extending cross-validation-style CP methods (from Barber et al. (2021)) to those settings for flexibly balancing compute-versus-sample efficiency. Nair & Janson (2023) studied a "Y-stationarity" setting similar to MFCS with a Monte Carlo-based CP algorithm for discrete data. Otherwise, our work relates to Barber et al. (2023), which bounds the worst-case "coverage gap" (loss of coverage below Eq. (1)) if the CP weights are unknown and instead fixed *a priori* independently of the data. In contrast, we study CP for non-exchangeable distributions using data-dependent weights to achieve coverage validity guarantees as in Eq. (1) (without a coverage gap).

Conformal Prediction for Decision Making Ideas from CP are increasingly being applied to decision-making to inform utility estimates (Vovk & Bendtsen, 2018; Stanton et al., 2023; Salinas et al., 2023) or satisfy constraints (Lekeufack et al., 2023; Zhang et al., 2023; Dixit et al., 2023; Laufer-Goldshtein et al., 2023; Jin & Candès, 2023). In particular, Stanton et al. (2023) drew motivation from Fannjiang et al. (2022) to combine CP with Bayesian optimization but noted that a gap in theory prevented a formal coverage guarantee after the first round, a gap we address in this work.

4. Theory and Method Contributions

4.1. The Role of (Weighted) Exchangeability and Related Assumptions in Conformal Prediction

Informal Statement of the Key Insight One can think of CP as an inverted permutation test (with a conservative adjustment). Informally, the key insight is that—from this perspective of an inverted permutation test—the only role of exchangeability, weighted exchangeability (Tibshirani et al., 2019), or related distributional assumptions such as pseudo-exchangeability (Fannjiang et al., 2022) in CP is to enable tractability by telling us how much to "weigh" or "count" a given ordering of datapoints; in particular, by facilitating computation of the relative likelihood of each sequence of

observations, relative to some (permutation-invariant) core function, given the unordered "bag" of observed values (details in Appendix D). So, extensions of CP should be possible so long as one can quantify the relative probability of any ordering of observed datapoints (or score values).

Formal Sketch of the Key Insight We first sketch analysis from Tibshirani et al. (2019) (which corresponds to viewing CP as an inverted permutation test) followed by the key insight for our current paper's main result. Their analysis begins by conditioning on the set[4] of data values (or scores)—that is, on the event $\{Z_1, ..., Z_{n+1}\} = \{z_1, ..., z_{n+1}\}$ denoted by E_z. To be clear, E_z means the *unordered* "bag" of values is known but *not* whether $Z_i = z_i$, and so on; we know that the random variable Z_i takes on an observed value in $\{z_1, ..., z_{n+1}\}$, but we do not yet know *which* value (similarly for the scores V_i). Given E_z, the goal is to examine the probability that the test point's score V_{n+1} was actually v_i, for each $i \in \{1, ..., n+1\}$, which they write as

$$\mathbb{P}\{V_{n+1} = v_i \mid E_z\} = \mathbb{P}\{Z_{n+1} = z_i \mid E_z\} = \frac{\sum_{\sigma:\sigma(n+1)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}, \quad (4)$$

where f denotes the joint probability density function (PDF). Standard CP proceeds from this statement by substituting $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = f(z_1, ..., z_{n+1})$ via exchangeability, reducing the problem to counting permutations; weighted CP proceeds by factorizing f using weighted exchangeability, with the product of weight functions representing the "weight" given to each permutation σ; and, CP with pseudo-exchangeable data proceeds similarly to handle a specific instance of potential dependencies (details in Appendix D).

Generally, the key insight in our paper is that simplifying Eq. (4) is only a practical rather than a theoretically necessary step. For intuition on this, note that Eq. (4) makes no assumptions on f; it follows from the definition of conditional probability and the law of total probability (LOTP):

$$\mathbb{P}\{Z_{n+1} = z_i \mid E_z\} = \frac{p\{Z_{n+1} = z_i, E_z\}}{p\{E_z\}} = \frac{\sum_{\sigma:\sigma(n+1)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}.$$

That is, the last step uses LOTP in both the numerator and denominator—for instance, the denominator's sum is over every way the event E_z could occur, which amounts to all the possible permutations σ of the values. The crux of extending CP to any, possibly non-exchangeable f can thus reduce to computing $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})$ directly for all orderings σ, or factoring f into "dynamic" terms that are tractable to compute (e.g., "weight functions") and remaining permutation-invariant terms that cancel in the ratio.

4.2. A General View of Conformal Prediction

Main Formal Result With the insight from Section 4.1 in mind, we are now ready to see how—in theory—extensions of CP are possible for any, potentially non-exchangeable distribution of the calibration and test data $Z_1, ..., Z_{n+1}$ with a valid PDF[5] f (we let $t = 1$ for simplicity, but in general our argument holds for $Z_1, ..., Z_{n+t}$). This section's theorem formally states this result as the coverage guarantee for a generalized weighted CP set for arbitrary f.

Let us define nonconformity scores more explicitly as

$$\begin{cases} V_i^{(x,y)} &= \mathcal{S}(Z_i, Z_{1:n} \cup \{(x,y)\}),\ i \in \{1, ..., n\}, \\ V_{n+1}^{(x,y)} &= \mathcal{S}\big((x,y), Z_{1:n} \cup \{(x,y)\}\big), \end{cases} \quad (5)$$

and for condensed notation, let us denote $\mathbb{P}_{n+1}\{z_i|E_z\} = \mathbb{P}\{Z_{n+1} = z_i|E_z\}$, recalling Eq. (4). To enable us to write a *weighted* empirical distribution of values with probability weights $\mathbb{P}_{n+1}\{z_i|E_z\}$ applied to each value v_i, we will also now denote use δ_v to denote a point mass at the value v (e.g., in the special case of uniform weights $\frac{1}{m}$ over values $v_1, ..., v_m$, then $Q_{1-\alpha}(v_{1:m}) = Q_{1-\alpha}(\sum_{i=1}^m \frac{1}{m}\delta_{v_i})$). We now have the notation needed for our main theorem.

Theorem 4.1. *Assume that $Z_i = (X_i, Y_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, ..., n+1$ have the joint PDF f. For any measurable score function $\mathcal{S}$, and any $\alpha \in (0,1)$, define the generalized conformal prediction set (based on n calibration samples) at a point $x \in \mathbb{R}^d$ by*

$$\widehat{\mathcal{C}}_n(x) = \left\{ y \in \mathbb{R} : V_{n+1}^{(x,y)} \leq Q_{1-\alpha}\left(\sum_{i=1}^n \mathbb{P}_{n+1}\{Z_i|E_z\}\delta_{V_i^{(x,y)}} + \mathbb{P}_{n+1}\{Z_{n+1}|E_z\}\delta_\infty \right) \right\} \quad (6)$$

where $V_i^{(x,y)}, i \in \{1, ..., n+1\}$ are as in (5) and $\mathbb{P}_{n+1}\{z_i|E_z\}$ is notation for (4). Then, $\widehat{\mathcal{C}}_n$ satisfies

$$\mathbb{P}\big\{Y_{n+1} \in \widehat{\mathcal{C}}_n(X_{n+1})\big\} \geq 1 - \alpha.$$

Proof Sketch We defer the full proof for the result to Appendix A.1, but a sketch of the proof has two main steps:

1. *CP as an inverted (weighted) permutation test:* With setup as in Section 4.1, use Eq. (4) to derive a general quantile lemma, which informally states the following: For $\beta \in (0,1)$, the test point's score V_{n+1} is contained within the level β quantile of the weighted empirical distribution of score values (with weights defined as in Eq. (4)) with probability at least β.

[4] For simplicity, we assume all the values are distinct to work with sets rather than multisets; in general, the argument holds with more complex notation using uniform randomness to break ties.

[5] To be precise, we require a *non-singular* distribution so that there exists a valid PDF f, or more generally, a valid Radon-Nikodym derivative with respect to an arbitrary base measure.

2. *Conservative adjustment to connect to general CP set:* Connect the general quantile lemma from the first proof step to general CP set (Eq. (6)) by conservatively adjusting for the fact that the true test point's score value is unknown. That is, replacing the test point's score value with ∞ in the empirical distribution maintains validity, and setting $\beta = 1 - \alpha$ gives the theorem.

Remark 4.2. The score functions in Eq. (5) correspond to the "ordinary" full CP method, but the same proof argument and result follows for the "deleted" full CP method (Vovk et al., 2005), which uses leave-one-out scores to prevent overfitting. Moreover, the same result follows for split conformal prediction as a special case. (See Appendix A.2.)

Remark 4.3. As written, Theorem 4.1 maintains the convention of requiring $\mathcal{S}$ to be symmetric with respect to any training data also used in calibration. This condition is relevant for full CP methods, but for split CP, this symmetry holds trivially due to training and calibration sets being separate. (See Appendix A.3 for further details and discussion.)

Limitations and Benefits of the General View of CP There are two main obstacles to computing Eq. (6) in practice: a *computational complexity* challenge and an *epistemic* challenge regarding knowledge about f. That is, in the general case for an arbitrary or unfactorized f, Eq. (4) requires $\mathcal{O}((n+1)!)$ evaluations of f, which quickly becomes intractable. Secondly, we often do not know f; indeed, the primary motivation of distribution-free UQ is that very deficiency. Exchangeability, weighted exchangeability (Tibshirani et al., 2019), and related conditions such as pseudo-exchangeability (Fannjiang et al., 2022) introduce distributional assumptions that alleviate these challenges.

One immediate benefit of the general presentation of conformal prediction in Theorem 4.1 is a shift in perspective: because the theorem holds for any joint distribution f, for any *particular* f the question is not whether a CP set with valid coverage *exists*, it is instead whether we can *compute* the prediction set *in practice*. Consequently—from this general perspective—"quasi-exchangeability" definitions are no longer necessary conditions for CP theory; rather, the *only role* of these definitions is to enable practical tractability by simplifying and eliminating terms from the computation (of Eq. (4))—for example, at one extreme, exchangeability "assumes away" f from the computation entirely. (Appendix D provides further formal details on this point, including how previous CP validity guarantees can be viewed as corollaries to Theorem 4.1.) Specific weighted CP algorithms with corresponding coverage validity guarantees can thus be derived for any (possibly non-exchangeable) distribution. In section 4.3, we outline how to perform such derivations in general and provide the MFCS case as a worked example.

Further Discussion of Tibshirani et al. (2019) This general view of CP is implicit in a close reading of Tibshirani et al. (2019),[6] but in that paper the authors did not explicitly state or prove the general result we present in Theorem 4.1. Instead, Tibshirani et al. (2019) premised their main result on their definition of weighted exchangeability, although this condition is not strictly necessary for CP in the broad sense. While we maintain that weighted exchangeability is still a useful definition (see Appendix D.2), subsequent literature has seemingly been led to believe that new definitions of "quasi-exchangeability" are necessary for advancing and delineating the boundaries of CP theory—for example, Fannjiang et al. (2022)'s introduction of pseudo-exchangeability to study the single-step FCS setting. We hope our contribution will lead to wider recognition of the generality of Tibshirani et al. (2019)'s analysis and foster a broadened scope of possibility in the study of CP: in particular, we encourage a shift away from viewing "quasi-exchangeability" conditions as theoretical boundaries to be overcome, and instead towards primarily using such conditions to clearly specify the assumptions granted by a given setting of interest or adopted for practical tractability.

4.3. How to Find Weighted CP Validity Guarantees and Algorithms for MFCS (or Any Data Distribution)

A General Procedure for Deriving Weighted CP Algorithms with Valid Coverage Guarantees To improve the accessibility and practical utility of our paper's main theoretical contribution, we now outline a general procedure that can be used to derive weighted CP algorithms for any specific joint PDF (or mass function) f over the calibration and test data. Theorem 4.1 implies the existence of this procedure without premising it on any assumptions on f, while emphasizing that the only role of distributional assumptions like exchangeability is for tractability in the last step.

1. *List assumptions (if any):* For a given application setting, specify assumptions on f, if any, such as conditional independence or invariance assumptions. (For example, the MFCS assumptions are formally given by Eq. (3); informally, MFCS assumes $Y|X$ is invariant, while X is dynamic depending on past observations.) Alternatively, one could begin by describing an application's data-generating process by a probablistic graphical model (e.g., a Markov random field or a causal directed acyclic graph) and use the conditional inde-

[6] We note that the main insight in our paper is also invoked in the following lecture notes (Tibshirani, 2023): https://www.stat.berkeley.edu/~ryantibs/statlearn-s23/lectures/conformal_ds.pdf, which a sharp reader could use to infer Theorem 4.1. However, based on how Tibshirani et al. (2019) has been cited in the CP literature (i.e., primarily as extending CP to standard covariate shift or to weighted exchangeability), we believe this important insight has not been fully appreciated by the community and we hope that our contribution can help correct this oversight.

pendence assumptions implied by that graphical model (for an example see Appendix B.3; for details on such models Koller & Friedman (2009); Pearl (2009)).[7]

2. *Factorize f:* Factorize the joint PDF f using standard probability rules (e.g., conditional probability, chain rule, etc.) and assumptions from Step 1 to separate dynamic factors from those that are invariant (to permutations σ of the data indices). (For example, for MFCS f can be written as a product of dynamic factors $X_j \mid Z_1, \ldots, Z_{j-1}$ and invariant factors $Y \mid X$.)

3. *Compute or estimate weights:* Plug the factorized form of f from Step 2 into Eq. (4) to obtain the normalized calibration and test point weights, and simplify by canceling out permutation-invariant and constant factors. Compute or estimate the simplified weights, and plug them into (6) to obtain the CP set. If computation is performed exactly, then the resulting CP set has a valid coverage guarantee premised on the assumptions from Step 1 (as a corollary to Theorem 4.1).

Deriving CP for Agent-Induced MFCS For a concrete demonstration of how this general procedure can be used to derive weighted CP methods for a given practical setting, we now return to the MFCS setting introduced in Section 2.3, which characterizes common instances of ML-agent-induced feedback-loop data shifts. We provide only a sketch of key ideas here; for full details see Appendix B. Step 1 of this procedure for MFCS begins by recalling Eq. (3), the conditional independence and invariance assumptions that we used to define MFCS. Using standard probability rules and leveraging these MFCS assumptions, for Step 2 we are able to factorize the joint PDF f to separate the time-dependent factors corresponding to $X_j \mid Z_1, \ldots, Z_{j-1}$ from the time-invariant factors corresponding to $Y \mid X$:

$$f(z_1, ..., z_{n+t}) = \prod_{j=1}^{n+t} \Big[\underbrace{p(x_j \mid z_1, ..., z_{j-1})}_{\text{Time-dependent factors}} \Big] \cdot \underbrace{\prod_{j=1}^{n+t} \Big[p(y_j|x_j) \Big]}_{\text{Time-invariant factor}}. \quad (7)$$

We focus on *time* dependence and invariance here only because we assume that the data indices indicate timesteps; in general, Step 2 aims to separate terms that depend on permutations of the data indices from those that are invariant to such permutations (if any). Lastly, for Step 3, we use our factorization of f to simplify the CP calibration and test point weights. To do so, we slightly modify our prior notation by letting $E_z^{(t)}$ denote the event $\{Z_1, ..., Z_{n+t}\} = \{z_1, ..., z_{n+t}\}$; then, plugging Eq. (7) into the analog of Eq. (4) for test point $n+t$ yields

$$\mathbb{P}\{Z_{n+t} = z_i \mid E_z^{(t)}\} = \frac{\sum_{\sigma:\sigma(n+t)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+t)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+t)})}$$
$$= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=1}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=1}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}, \quad (8)$$

after canceling out $\prod_{j=1}^{n+t} p(y_{\sigma(j)}|x_{\sigma(j)})$ from the numerator and denominator, as it is invariant to permutations σ.

As a corollary to Theorem 4.1, a CP method with weights given by Eq. (8) has a valid coverage guarantee premised on the MFCS assumptions in Eq. (3). We can thus turn to practical estimation. Firstly, in the agent-induced MFCS settings that we focus on in our experimental evaluations (Section 5), the epistemic challenge is overcome because the dynamic components of f are known entirely. That is, the invariant factor corresponding to $Y|X$ cancels in the ratio, and in our practical evaluations the remaining factors $p(x|Z_1, ..., Z_{j-1})$ represent ML-agent-controlled query probability functions at each timestep j. Next, we examine computational complexity. Due to the MFCS IID initialization implying $p(x|Z_1, ..., Z_{j-1}) = p(x)$ for $j \leq n$, and when $p(x|Z_1, ..., Z_{j-1})$ is computed from an ML model that treats $Z_1, ..., Z_{j-1}$ symmetrically (as in our experiments), the exact computation of Eq. (8) has complexity $\mathcal{O}(\prod_{j=1}^{t}(n+j))$. Though reduced from $\mathcal{O}((n+t)!)$, the complexity for arbitrary f, this still quickly becomes intractable for large t. To alleviate this remaining complexity bottleneck, we propose estimating Eq. (8) by using only the "highest-order" terms from the d most recent timesteps. That is, for a user-specified estimation depth $d \in \{1, ..., t\}$, we define our "d-step" estimate of Eq. (8) as

$$\widehat{p}_{n+t}^{(d)}\{z_i \mid E_z^{(t)}\} = \widehat{p}^{\,(d)}\{Z_{n+t} = z_i \mid E_z^{(t)}\} \quad (9)$$
$$= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=n+t+1-d}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=n+t+1-d}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})},$$

which reduces the complexity to $\mathcal{O}(\prod_{j=t+1-d}^{t}(n+j))$, a tractable polynomial when d is small. Appendix B.2 provides a derivation for our specific MFCS CP algorithms, which use a recursive implementation for further efficiency.

Full versus Split CP Algorithms The derivation procedure and MFCS methods presented in this section apply to variants of both Full and Split CP. Full CP efficiently uses the same dataset for both training and calibration (often resulting in more accurate and sharper prediction intervals). Full CP's data efficiency comes at a steep computational cost, with its complexity dominated by the number of predictors that must be *trained* for each candidate label: For an arbitrary predictor, computing the (leave-one-out) Full CP MFCS weights in Eq. (8) requires training $|\mathcal{Y}| \cdot \prod_{j=1}^{t}(n+j)$

[7] Probabilistic graphical models can intuitively represent a data-generating process as a graph with nodes corresponding to random variables (or random vectors). Missing edges in such a graph imply conditional independences in the joint PDF f; the converse depends on further assumptions (Pearl, 2009).

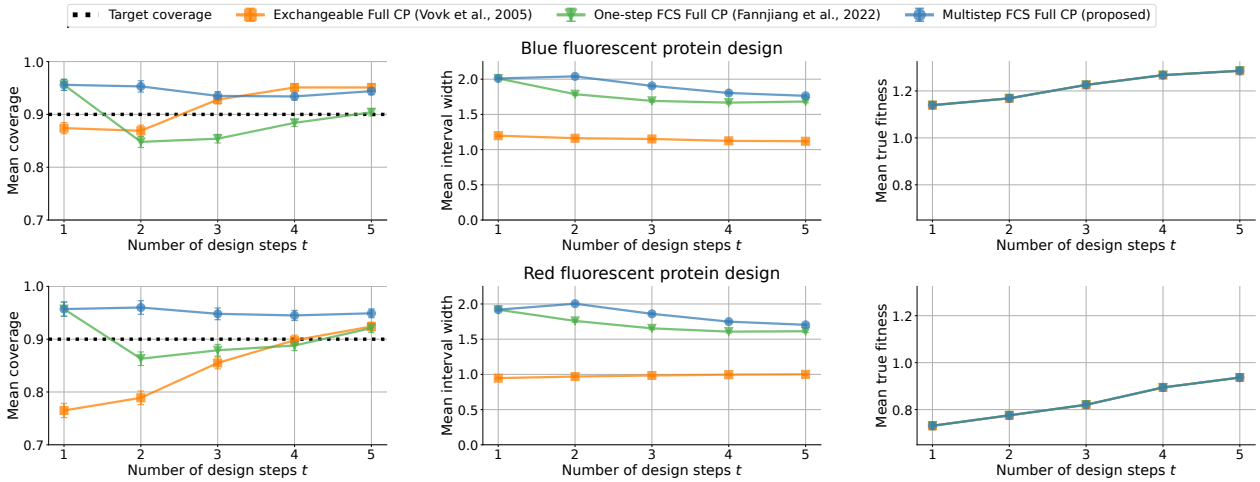

Figure 1. Results for protein design ($n = 32$, $\lambda = 8$) with linear ridge regression, comparing the proposed MFCS Full CP method ($d = 2$) to standard Full CP and One-Step FCS Full CP. Values are computed over 1,000 random seeds; error bars for mean coverage and mean predicted fitness are standard errors, while error bars for median interval width are interquartile ranges. MFCS Full CP maintains coverage where the baselines do not. Its intervals are wider than those of the one-step FCS baseline where the latter loses coverage ($t = 2, 3, 4$), but similar where the baseline maintains coverage ($t = 1, 5$). (Further experimental details in Appendix E.1 Table 1.)

distinct predictors, where $|\mathcal{Y}|$ is the cardinality of the label space.[8] Certain predictor classes (e.g., linear ridge regression) have closed-form solutions as a function of y (Vovk et al., 2005), which reduces the training cost to $\prod_{j=1}^{t}(n+j)$.

Variants of Split CP (Papadopoulos, 2008) avoid the training cost of Full CP by splitting the available data into a "proper" training set used only to fit a single ML model (for each updated training dataset), and a separate holdout "calibration" set on which the fitted model computes CP scores to construct the CP set. Split CP's computational complexity at inference time thus corresponds to the *evaluation* cost of the pre-fitted model on the calibration and test data.

5. Experimental Results

We evaluate the performance of our proposed CP methods for ML-agent-induced MFCS on synthetic black-box optimization (specifically protein design) and active learning tasks. All experiments are a regression setting with all CP methods using the absolute residual score $\widehat{\mathcal{S}}(x, y) = |y - \widehat{\mu}_t(x)|$, where $\widehat{\mu}_t(x)$ is the prediction of the model fit to $Z^{(t)}_{\text{train}}$, the training data available just prior to time t (for Split CP these are separate from calibration data, for Full CP $Z^{(t)}_{\text{train}} = Z_{1:(n+t-1)}$). Our proposed methods are defined by estimating Eq. (6) with weights as in Eq. (9). The primary desired criterion for reliability is whether its prediction intervals maintain empirical coverage at or above the target coverage level $1 - \alpha$. Secondarily—if target coverage is achieved—then smaller predictive sets are more informative (i.e., predicting the full label space $\widehat{\mathcal{C}}_n(x) = \mathcal{Y}$ could trigger deferral to an expert but does not itself convey new insights from the predictor). In all experiments we compare to Exchangeable CP (Vovk et al., 2005) and One-Step FCS CP (Fannjiang et al., 2022), and in the active learning experiments we also compare with Adaptive Conformal Inference (ACI) (Gibbs & Candès, 2021). Our proposal distributions follow the form $p(x \mid Z^{(t)}_{\text{train}}) \propto \exp(\lambda \cdot u_t(x))$, where u_t is some utility function and $\lambda \geq 0$ is a hyperparameter that controls the magnitude of agent-induced covariate shift. In all experiments we corrupt labels with Gaussian noise to simulate noisy observations. Each query point is added to training for full CP experiments, and for split CP, to training or calibration by flipping a fair coin. (See Appendix E for details, Appendices C and F for additional experiments.)

[8] In practice for regression, a grid approximates the label space.

5.1. Multistep Protein Design Experiments

Optimization-Induced MFCS Experimental Details Our protein design procedure is similar to a common approach in the biomolecular engineering literature (Biswas et al., 2021; Zhu et al., 2024) and related works in CP (Fannjiang et al., 2022; Prinster et al., 2023), but extended to a more realistic online setting where protein sequence proposals and model updates are interleaved and dependent (Stanton et al., 2022; Gruver et al., 2023; Angermueller et al., 2023). All methods start with an IID initialization by sampling n points uniformly at random from a combinatorially complete library (Poelwijk et al., 2019), splitting the data equally into training and calibration sets if using split CP. (See Appendix E.1 for details.) We evaluate the mean coverage and median intervals width as a function of t (the number of optimization

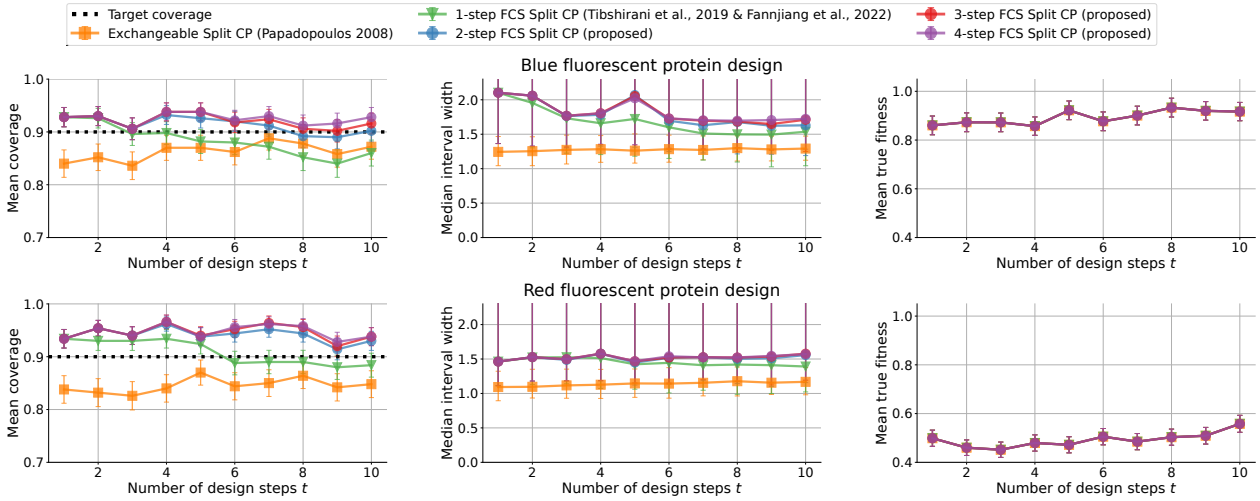

Figure 2. Results for fluorescent protein design with a multi-layer perceptron (MLP) regressor (initial training and calibration data sizes: $n_{\text{train}} = n_{\text{cal}} = 32$; $\lambda = 5$), comparing the proposed MFCS Split CP methods to Standard Split CP and the One-Step FCS Split CP baselines. Values are computed over 500 random seeds. Error bars for mean coverage and mean predicted fitness are standard errors; for median interval width they are the interquartile ranges. Error bars extending beyond the top of the figure indicate infinite upper quartiles. MFCS Split CP maintains coverage where the baselines do not, but its intervals are occasionally very wide, suggesting the proposal distribution is too aggressive for dependably informative uncertainty estimation. (More experimental details in Appendix E.1 Table 2.)

steps past initialization), fitting the regressor to training data observed prior to time t, and sampling one query point with replacement (with added measurement noise) from the library, each with probability $p(x \mid Z_{\text{train}}^{(t)}) \propto \exp(\lambda \cdot \widehat{\mu}_t(x))$.

Full CP Results Figure 1 reports the results for our proposed MFCS Full CP method (blue circles) with hyperparameters reported in the caption. On both the blue and red fluorescent protein design datasets, and on all five evaluated design steps, our proposed MFCS Full CP method maintains coverage above the target level even when both baselines lose coverage below this level. The MFCS method's coverage is maintained by increasing the conservativeness (size) of its prediction intervals relative to the One-Step baseline after the first design step. The superimposed fitness curves on the right verify that differences in coverage can be attributed to the different CP methods, not the data collected.

Split CP Results with Neural Network Regressor The results in Figure 2 use an MLP regressor on the protein design datasets to evaluate our MFCS Split CP method. We moreover compare the effect of several estimation depths d for estimating the MFCS weights as in Eq. (9). For weight estimation depths $d \in \{3, 4\}$, our proposed MFCS CP methods maintain target coverage across all 10 evaluated design steps, for both the blue and red fluorescent protein design datasets, even when the baselines break down. Increasing the estimation depth d tends to improve coverage at later design steps, for instance relative to the One-Step FCS method, apparently with diminishing returns (the red 3-step and purple 4-step methods have nearly overlapping performance). Coverage is maintained by the MFCS Split CP methods by slightly increasing the size of the intervals relative to the one-step baseline at later design steps. However, both the MFCS Split CP method and the One-Step FCS Split CP method occasionally have noninformative (infinite width) intervals, which could be unacceptable in some application settings. In Appendix C, we show that adaptively bounding the ML agent's query probability function can achieve dependably informative (finite) interval widths without losing coverage, by restricting the ML agent's feedback shift.

5.2. Active Learning-Induced MFCS

Now we turn to an active learning setting where MFCS is induced by a max-entropy data acquisition strategy, where the ML agent selects new queries for annotation by maximizing its (Bayesian) model uncertainty. In this experiment we use a Gaussian process (GP) regressor as the predictor and take $u_t(x) = \mathbb{V}(\mu(x) \mid Z_{\text{train}}^{(t)})$ (i.e. the GP posterior variance). See Appendix E.2 for model and dataset details.

Active Learning Experimental Results Figure 3 demonstrates the performance of our proposed MFCS Split Conformal methods for several depths of weight approximation. Whereas the baseline methods lose coverage, the proposed MFCS Split CP methods maintain target coverage for all datasets and across the 70 evaluated active learning steps. As the active learning process reaches later iterations where more labeled data is available for both training and calibration, the proposed MFCS Split CP methods are still able to maintain target coverage with sharper intervals.

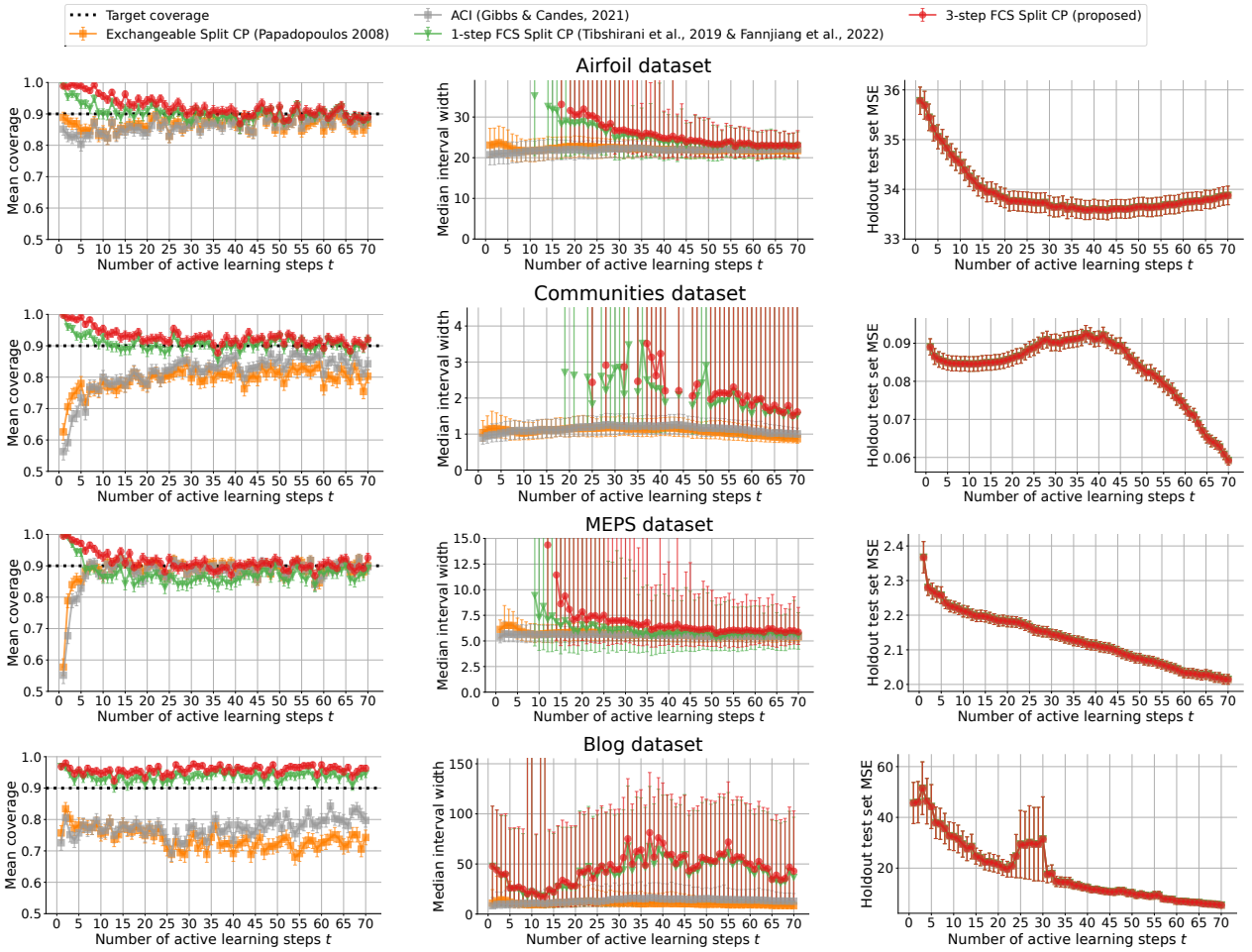

Figure 3. Active learning experiments of proposed MFCS Split CP method for $d = 3$ (red) compared to baselines of unweighted Split CP (orange), One-Step Split CP (green), and ACI (gray). Y-axes represent mean coverage, median interval width, and mean squared error on a holdout test set to track the accuracy of the base predictor. X-axes correspond to the number of active learning query steps, with each query based on posterior variance of a GP regressor. All values are computed over 350 distinct randomm seeds (full experimental details in Appendix E.2). The proposed MFCS CP method maintains target coverage over a long time horizon even where baselines do not.

6. Discussion

In this paper we have demonstrated how weighted conformal prediction can theoretically generalize to any joint data distribution. Exchangeability conditions are only necessary to rigorously specify when *a specific practical algorithm* (e.g., standard CP) is equivalent to the more general formulation, which is always valid but may not be computable. Promising future directions include further addressing practical bottlenecks, extending our general marginal coverage guarantees to other loss functions (Angelopoulos et al., 2022), and identifying minimal assumptions for strong conditional coverage in practice (Barber et al., 2021; Gibbs et al., 2023).

We have also provided practical CP algorithms for multiple rounds of agent-induced, data-dependent covariate shifts, which maintains coverage on data collected by active learning and black-box optimization agents. Finally, we have shown that these agents can attain both empirically valid coverage and informative intervals by restricting the degree of shift introduced (Appendix C). These findings bring us to the control-consistency dilemma at the heart of statistical decision-making. To control is to change, but statistics requires consistency. Like the explore-exploit dilemma, the right tradeoff depends on the situation at hand. As machine learning agents assume increasing levels of responsibility in gravely consequential settings, it is more important than ever that we face the control-consistency dilemma head-on.

Code Availability

Repository with code for all experiments: `https://github.com/drewprinster/conformal-mfcs`

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work which we do not think must be specifically highlighted here. However, we note that our contributions are specifically aimed at improving the reliability of methods for uncertainty quantification of machine learning predictions, which is often critical for managing risks and improving the trustworthiness of such systems.

Acknowledgements

D.P. was partially funded by a fellowship from the Johns Hopkins University (JHU) Mathematical Institute for Data Science (MINDS); D.P. and S. Saria were partially funded by National Science Foundation (NSF) grant #1840088; D.P., A.L., and S. Saria were partially funded by the Gordon and Betty Moore Foundation grant #12128. A.L. was also partially funded by a JHU Discovery Award and a JHU Institute for Assured Autonomy (IAA) seed grant. The authors would like to thank Anastasios Angelopoulos for early discussions that later helped inform this work and Clara Fannjiang for helpful feedback on a manuscript draft.

References

Angelopoulos, A. N., Bates, S., Fisch, A., Lei, L., and Schuster, T. Conformal risk control. *arXiv preprint arXiv:2208.02814*, 2022.

Angelopoulos, A. N., Candes, E. J., and Tibshirani, R. J. Conformal pid control for time series prediction. *Advances in neural information processing systems*, 36, 2023.

Angermueller, C., Mariet, Z., Jester, B., Engelhart, E., Emerson, R., Alipanahi, B., Lin, C., Shikany, C., Guion, D., Nelson, J., et al. High-throughput ml-guided design of diverse single-domain antibodies against sars-cov-2. *bioRxiv*, pp. 2023–12, 2023.

Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. Predictive inference with the jackknife+. 2021.

Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 51(2):816–845, 2023.

Bhatnagar, A., Wang, H., Xiong, C., and Bai, Y. Improved online conformal prediction via strongly adaptive online learning. *arXiv preprint arXiv:2302.07869*, 2023.

Biswas, S., Khimulya, G., Alley, E. C., Esvelt, K. M., and Church, G. M. Low-n protein engineering with data-efficient deep learning. *Nature methods*, 18(4):389–396, 2021.

Brooks, T., Pope, D., and Marcolini, M. Airfoil Self-Noise. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C5VW2C.

Buza, K. BlogFeedback. UCI Machine Learning Repository, 2014. DOI: https://doi.org/10.24432/C58S3F.

Dixit, A., Lindemann, L., Wei, S. X., Cleaveland, M., Pappas, G. J., and Burdick, J. W. Adaptive conformal prediction for motion planning among dynamic agents. In *Learning for Dynamics and Control Conference*, pp. 300–314. PMLR, 2023.

Ezzati-Rice, T. M., Rohde, F., and Greenblatt, J. *Sample design of the Medical Expenditure Panel Survey household component, 1998-2007*. US Department of Health & Human Services, Agency for Healthcare Research and . . . , 2008.

Fannjiang, C., Bates, S., Angelopoulos, A. N., Listgarten, J., and Jordan, M. I. Conformal prediction under feedback covariate shift for biomolecular design. *Proceedings of the National Academy of Sciences*, 119(43):e2204569119, 2022.

Farinhas, A., Zerva, C., Ulmer, D., and Martins, A. F. Non-exchangeable conformal risk control. *arXiv preprint arXiv:2310.01262*, 2023.

Feldman, S., Ringel, L., Bates, S., and Romano, Y. Achieving risk control in online learning settings. *Transactions on Machine Learning Research*, 2023.

Foygel Barber, R., Candes, E. J., Ramdas, A., and Tibshirani, R. J. The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2):455–482, 2021.

Frank, A. Uci machine learning repository. *http://archive. ics. uci. edu/ml*, 2010.

Gibbs, I. and Candès, E. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Gibbs, I. and Candès, E. Conformal inference for online prediction with arbitrary distribution shifts. *arXiv preprint arXiv:2208.08401*, 2022.

Gibbs, I., Cherian, J. J., and Candès, E. J. Conformal prediction with conditional guarantees. *arXiv preprint arXiv:2305.12616*, 2023.

Gruver, N., Stanton, S., Frey, N. C., Rudner, T. G., Hotzel, I., Lafrance-Vanasse, J., Rajpal, A., Cho, K., and Wilson, A. G. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36, 2023.

Jin, Y. and Candès, E. J. Selection by prediction with conformal p-values. *Journal of Machine Learning Research*, 24(244):1–41, 2023.

Koller, D. and Friedman, N. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Kolmogorov, A. N. *Foundations of the theory of probability: Second English Edition*. Chelsea Publishing Co., 1956.

Laufer-Goldshtein, B., Fisch, A., Barzilay, R., and Jaakkola, T. Risk-controlling model selection via guided bayesian optimization. *arXiv preprint arXiv:2312.01692*, 2023.

Lekeufack, J., Angelopoulos, A. A., Bajcsy, A., Jordan, M. I., and Malik, J. Conformal decision theory: Safe autonomous decisions from imperfect predictions. *arXiv preprint arXiv:2310.05921*, 2023.

Nair, Y. and Janson, L. Randomization tests for adaptively collected data. *arXiv preprint arXiv:2301.05365*, 2023.

Papadopoulos, H. Inductive conformal prediction: Theory and application to neural networks. In *Tools in artificial intelligence*. Citeseer, 2008.

Pearl, J. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.

Pearl, J. *Causality: Models, Reasoning and Inference*. Cambridge University Press, USA, 2nd edition, 2009. ISBN 052189560X.

Podkopaev, A. and Ramdas, A. Distribution-free uncertainty quantification for classification under label shift. In *Uncertainty in Artificial Intelligence*, pp. 844–853. PMLR, 2021.

Poelwijk, F. J., Socolich, M., and Ranganathan, R. Learning the pattern of epistasis linking genotype and phenotype in a protein. *Nature communications*, 10(1):4213, 2019.

Prinster, D., Liu, A., and Saria, S. Jaws: Auditing predictive uncertainty under covariate shift. *Advances in Neural Information Processing Systems*, 35:35907–35920, 2022.

Prinster, D., Saria, S., and Liu, A. Jaws-x: addressing efficiency bottlenecks of conformal prediction under standard and feedback covariate shift. In *International Conference on Machine Learning*, pp. 28167–28190. PMLR, 2023.

Redmond, M. Communities and Crime. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C53W3X.

Salinas, D., Golebiowski, J., Klein, A., Seeger, M., and Archambeau, C. Optimizing hyperparameters with conformal quantile regression. *arXiv preprint arXiv:2305.03623*, 2023.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Stanton, S., Maddox, W., Gruver, N., Maffettone, P., Delaney, E., Greenside, P., and Wilson, A. G. Accelerating bayesian optimization for biological sequence design with denoising autoencoders. In *International Conference on Machine Learning*, pp. 20459–20478. PMLR, 2022.

Stanton, S., Maddox, W., and Wilson, A. G. Bayesian optimization with conformal prediction sets. In *International Conference on Artificial Intelligence and Statistics*, pp. 959–986. PMLR, 2023.

Tibshirani, R. Conformal prediction under distribution shift. https://www.stat.berkeley.edu/~ryantibs/statlearn-s23/lectures/conformal_ds.pdf, 2023. [Lecture notes for "Advanced Topics in Statistical Learning"].

Tibshirani, R. J., Foygel Barber, R., Candes, E., and Ramdas, A. Conformal prediction under covariate shift. *Advances in neural information processing systems*, 32, 2019.

Vovk, V. and Bendtsen, C. Conformal predictive decision making. In *Conformal and Probabilistic Prediction and Applications*, pp. 52–62. PMLR, 2018.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Xu, C. and Xie, Y. Conformal prediction interval for dynamic time-series. In *International Conference on Machine Learning*, pp. 11559–11569. PMLR, 2021.

Yang, Y., Kuchibhotla, A. K., and Tchetgen Tchetgen, E. Doubly robust calibration of prediction sets under covariate shift. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, pp. qkae009, 2024.

Zaffran, M., Féron, O., Goude, Y., Josse, J., and Dieuleveut, A. Adaptive conformal predictions for time series. In *International Conference on Machine Learning*, pp. 25834–25866. PMLR, 2022.

Zhang, Y., Park, S., and Simeone, O. Bayesian optimization with formal safety guarantees via online conformal prediction. *arXiv preprint arXiv:2306.17815*, 2023.

Zhu, D., Brookes, D. H., Busia, A., Carneiro, A., Fannjiang, C., Popova, G., Shin, D., Donohue, K. C., Lin, L. F., Miller, Z. M., et al. Optimal trade-off control in machine learning–based library design, with application to adeno-associated virus (aav) for gene therapy. *Science Advances*, 10(4):eadj3786, 2024.

A. Main Result Proof and Details

A.1. Proof of Theorem 4.1

In this section we present the proof for our main result given in Theorem (4.1). Our proof builds on ideas in Tibshirani et al. (2019)'s alternate proof of their Lemma 1 (for conformal prediction assuming exchangeable data) and their proof of their Lemma 3 (for conformal prediction assuming weighted exchangeable data). However, those proofs make key substitutions using the definitions of exchangeability and weighted exchangeability respectively, which thereby yields guarantees that are premised on those definitions. In contrast, the proof we present here makes no such substitution, and thus holds for an arbitrary joint probability density function (PDF)[9] f. We will call attention to the key step where our proof critically differs from those in Tibshirani et al. (2019) in substance, and even for steps that we have in common with Tibshirani et al. (2019) we aim to provide more explicit derivation explanations for increased accessibility (e.g., where explicit step justifications may be omitted in Tibshirani et al. (2019)).

As we sketch in Section 4.1 of our main text, our proof corresponds to a view of CP as an inverted permutation test. Let E_z denote the event of observing the unordered set[10] of our data values. In particular, E_z denotes the event that $\{Z_1, ..., Z_{n+1}\} = \{z_1, ..., z_{n+1}\}$, but importantly, the lack of ordering means that this does *not* imply that $Z_i = z_i$ for all $i \in \{1, ..., n+1\}$; we do not yet know whether z_1 is the value obtained by Z_1, or by Z_2, or by Z_{n+1}, and so on. Similarly, let E_v denote the event $\{V_1, ..., V_{n+1}\} = \{v_1, ..., v_{n+1}\}$ for the scores. Also, recall the simplifying notation $v_i = \widehat{\mathcal{S}}(z_i)$ for the value computed by our fitted score function $\widehat{\mathcal{S}}$ with input z_i (where $\widehat{\mathcal{S}}(z_i) = \mathcal{S}(z_i, z_{1:(n+1)})$, that is $\widehat{\mathcal{S}}$ is fit on the set $z_{1:(n+1)} = \{z_1, ..., z_{n+1}\}$, the second input to $\mathcal{S}$).

With this notation, we consider the probability that, conditioned on E_v, the value corresponding to the random variable V_{n+1} was actually v_i. That is, we consider

$$\mathbb{P}\{V_{n+1} = v_i \mid E_v\} = \mathbb{P}\{Z_{n+1} = z_i \mid E_z\},\ i \in \{1, ..., n+1\},$$

which follows since, conditioned on E_z, there are an equal number of scores $v_1, ..., v_{n+1}$ as data points $z_1, ..., z_{n+1}$, thus implying that the fitted (symmetric) score function $\widehat{\mathcal{S}}$ induces a bijection between the discrete set of distinct data observations and the discrete set of distinct scores.[11] Then, using the definition of conditional probability and the law of total probability (LOTP), it follows that

$$\begin{aligned}\mathbb{P}\{Z_{n+1} = z_i \mid E_z\} &= \frac{p\{Z_{n+1} = z_i, E_z\}}{p\{E_z\}} \\ &= \frac{\sum_{\sigma:\sigma(n+1)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}\end{aligned} \tag{10}$$

where the last step follows by applying the LOTP in both the numerator and the denominator. For example, in the denominator the summation is over all possible permutations σ of the value, i.e., all the different orderings in which the event E_z could have occurred. Even more explicit notation could be to write out $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = f_{Z_1,...,Z_{n+1}}(z_{\sigma(1)}, ..., z_{\sigma(n+1)})$ if f is a PDF, or $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = f(Z_1 = z_{\sigma(1)}, ..., Z_{n+1} = z_{\sigma(n+1)})$ if f is a probability mass function (PMF), to emphasize that the permutation is on the *observed values* and not on the random variables in the joint distribution.

Here is where our proof differs critically from the analogous proofs in Tibshirani et al. (2019). At this point, their proof corresponding to exchangeable conformal prediction uses exchangeability to substitute in $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = f(z_1, ..., z_{n+1})$, since under exchangeability f is invariant to permutations σ (see Appendix D.1 for details); meanwhile, their proof corresponding to weighted conformal prediction substitutes in $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = \prod_{j=1}^{n+1} w_j(z_{\sigma(j)}) g(z_1, ..., z_{n+1})$, since under their definition of weighted exchangeability f factorizes into weight functions w_j and a core function g that is invariant to σ (see Appendix D.2 for details on weighted exchangeability). Here, unlike in Tibshirani et al. (2019), we instead maintain the generality of Eq. (10) by avoiding any substitutions, and this will allow us to prove the more general conformal prediction result, which holds for any f.

[9] More generally, f can be a valid Radon-Nikodym derivative with respect to an arbitrary base measure: in the discrete case f is a probability mass function (PMF); it could also be a mixture of PDFs and PMFs.

[10] As in Tibshirani et al. (2019) we assume for simplicity that the score values $V_1, ..., V_{n+1}$ are distinct almost surely, though the result also holds in more general cases (but with more complicated notation involving randomness from uniform random variables to break ties).

[11] To be thorough and precise, the equation $\mathbb{P}\{V_{n+1} = v_i \mid E_v\} = \mathbb{P}\{V_{n+1} = v_i \mid E_z\} = \mathbb{P}\{Z_{n+1} = z_i \mid E_z\}$ relies on the convention in conformal prediction that $\mathcal{S}$ is a *symmetric* score function (see Remark 4.3). We elaborate on this point in Appendix A.3.

For simpler notation, let $\mathbb{P}\{Z_{n+1} = z_i \mid E_z\} = \mathbb{P}_{n+1}\{z_i \mid E_z\}$. Then, statement (10) implies that, conditioned on E_z, the random variable V_{n+1} is distributed according to the weighted empirical distribution of the values $\{v_1, ..., v_{n+1}\}$, with weight $\mathbb{P}\{Z_{n+1} = z_i \mid E_z\}$ at each δ_{v_i}:

$$V_{n+1}|E_v \sim \sum_{i=1}^{n+1} \mathbb{P}_{n+1}\{z_i \mid E_z\} \cdot \delta_{v_i}.$$

Letting Q_β denote a level-β quantile function, this implies

$$\mathbb{P}\left\{V_{n+1} \leq Q_\beta\left(\sum_{i=1}^{n+1} \mathbb{P}_{n+1}\{z_i \mid E_z\} \cdot \delta_{v_i}\right) \;\middle|\; E_z\right\} \geq \beta.$$

Due to the conditioning on E_z, this is equivalent to

$$\mathbb{P}\left\{V_{n+1} \leq Q_\beta\left(\sum_{i=1}^{n+1} \mathbb{P}_{n+1}\{Z_i \mid E_z\} \cdot \delta_{V_i}\right) \;\middle|\; E_z\right\} \geq \beta,$$

and marginalizing over draws of the event E_z yields

$$\mathbb{P}\left\{V_{n+1} \leq Q_\beta\left(\sum_{i=1}^{n+1} \mathbb{P}_{n+1}\{Z_i \mid E_z\} \cdot \delta_{V_i}\right)\right\} \geq \beta \tag{11}$$

We note that statement (11) can be viewed as a general quantile lemma, which generalizes both Lemma 1 and Lemma 3 in Tibshirani et al. (2019). We now connect statement (11) to the general conformal prediction set defined in Theorem 4.1 to obtain the result. To do so, we observe that

$$Q_\beta\left(\sum_{i=1}^{n+1} \mathbb{P}_{n+1}\{Z_i \mid E_z\} \cdot \delta_{V_i}\right) \leq Q_\beta\left(\sum_{i=1}^{n} \left[\mathbb{P}_{n+1}\{Z_i \mid E_z\} \cdot \delta_{V_i}\right] + \mathbb{P}_{n+1}\{Z_{n+1} \mid E_z\} \cdot \delta_\infty\right), \tag{12}$$

where the right-hand side is obtained from the left-hand side by replacing $\delta_{V_{n+1}}$ with δ_∞ to conservatively increase the value of the right side relative to the left. Now, with the notation $V_i = V_i^{(X_{n+1}, Y_{n+1})}$ for $i \in \{1, ..., n+1\}$, note that the right hand side of (12) is the same quantile as is used in the construction of the general conformal prediction interval in 4.1. That is, $Y_{n+1} \in \widehat{\mathcal{C}}_n(X_{n+1})$ if and only if

$$V_{n+1} \leq Q_\beta\left(\sum_{i=1}^{n} \left[\mathbb{P}_{n+1}\{Z_i \mid E_z\} \cdot \delta_{V_i}\right] + \mathbb{P}_{n+1}\{Z_{n+1} \mid E_z\} \cdot \delta_\infty\right). \tag{13}$$

Due to statement (12), the probability of the event (13) is thus lower bounded by statement (11). Setting $\beta = 1 - \alpha$ in the quantile function then gives the result.

A.2. Additional Details for Remark 4.2

In Theorem 4.1 and the proof in Appendix A.1 we use score functions that correspond to the "ordinary" full conformal method Vovk et al. (2005). However, the same result and proof technique would also apply to generalizing the "deleted" or "leave-one-out" full conformal method (which is computationally more expensive than ordinary full CP but can obtain more informative prediction sets by avoiding overfitting in its construction) (Vovk et al., 2005) or the split conformal method (Papadopoulos, 2008). The score functions for the deleted full CP method would be $V_i^{(x,y)} = \mathcal{S}(Z_i, Z_{-i} \cup \{(x, y)\})$ for $i \in \{1, ..., n\}$, where $Z_{-i} = Z_{1:n} \backslash Z_i$, and $V_{n+1}^{(x,y)} = \mathcal{S}\big((x, y), Z_{1:n}\big)$. For split CP, the score functions could be defined using independent training and calibration sets. Letting $Z_{1:m}^0 = \{Z_i^0\}_{i=1}^m$ denote the training data and $Z_{1:n} = \{Z_i\}_{i=1}^n$ denote the calibration data, the split CP score functions would be $V_i^{(x,y)} = \mathcal{S}(Z_i, Z_{1:m}^0)$ for $i \in \{1, ..., n\}$ and $V_{n+1}^{(x,y)} = \mathcal{S}\big((x, y), Z_{1:m}^0\big)$.

A.3. Discussion on Symmetric Versus Nonsymmetric Score Functions

In the main paper, we maintained the convention in conformal prediction of requiring that $\mathcal{S}$ be a symmetric score function (e.g., see Remark 4.3). Here, we briefly consider how our main result would need to be modified for nonsymmetric score

functions, while leaving more rigorous analysis for future work. First for an example, if $\widehat{\mu}_{\bar{Z}} \leftarrow \mathcal{A}(\bar{Z})$ is an ML predictor fit on training examples $\bar{Z}$ by an algorithm $\mathcal{A}$, then the absolute value residual score $\mathcal{S}\big((x,y),\bar{Z}\big) = |y - \widehat{\mu}_{\bar{Z}}(x)|$ is symmetric if the algorithm $\mathcal{A}$ treats its data symmetrically (i.e., if it is invariant to shuffling the ordering of its inputs); meanwhile, this score function is nonsymmetric if the fitted ML preditor output by the algorithm $\widehat{\mu}_{\bar{Z}} \leftarrow \mathcal{A}(\bar{Z})$ depends on the ordering of the elements in $\bar{Z}$. As we note in Remark 4.3, this distinction is relevant for full conformal methods, but it holds trivially for split conformal methods due to a separate dataset being used for training and calibration (the symmetry needed is for how the score function treats calibration and test data).

For a more precise distinction, let $\mathcal{S}$ be a real-valued function that takes as its input a point (x,y) and a *sequence* of other examples $(z_1,...,z_n)$, so that $\mathcal{S}\big((x,y),(z_1,...,z_n)\big)$ can be interpreted as how "nonconforming" or "strange" (x,y) is relative to the examples. Then, we can define symmetric and nonsymmetric score functions as follows:[12]

- $\mathcal{S}$ is a **symmetric score function** if $\mathcal{S}\big((x,y),(z_1,...,z_n)\big) = \mathcal{S}\big((x,y),(z_{\sigma(1)},...,z_{\sigma(n)})\big)$ for all $n \geq 1$, all permutations σ of the indices $\{1,...,n\}$, all $(x,y) \in \mathcal{X} \times \mathcal{Y}$, and all $(z_1,...,z_n) = ((x_1,y_1),...,(x_n,y_n))$.
- Otherwise, $\mathcal{S}$ is a **nonsymmetric score function**; for example, if $\mathcal{S}$ is sensitive to the ordering of its examples such that $\mathcal{S}\big((x,y),(z_1,...,z_n)\big) \neq \mathcal{S}\big((x,y),(z_{\sigma(1)},...,z_{\sigma(n)})\big)$ for some permutation σ of the indices $\{1,...,n\}$.

In other words, symmetric score functions treat their training examples $(z_1,...,z_n)$ symmetrically, so they can equivalently be defined (as in Section 2) by requiring the examples to be a bag or multiset (rather than a sequence), wherein there is no ordering to the elements. In contrast, nonsymmetric score functions may be sensitive to the ordering of their training inputs.

The proof of Theorem 4.1 provided in Appendix A.1 relies on $\mathcal{S}$ being a symmetric score function to equate score probabilies to data probabilities; that is, the equation $\mathbb{P}\{V_{n+1} = v_i \mid E_v\} = \mathbb{P}\{Z_{n+1} = z_i \mid E_z\}$ relies on score-function symmetry. To see this, first recall that we justified this equality by stating that (conditioned on E_z), the *fitted* and *symmetric* score function $\widehat{\mathcal{S}}$ induces a bijection between the data points $z_1,...,z_{n+1}$ and the scores $v_1,...,v_{n+1}$. As we defined $\widehat{\mathcal{S}}(x,y) = \mathcal{S}\big((x,y),z_{1:(n+1)}\big)$, where $z_{1:(n+1)} = \{z_1,...,z_{n+1}\}$ (an unordered set), this is equivalent to stating that the *unfitted* (and symmetric) score function $\mathcal{S}$ induces a bijection between the $n+1$ "data objects" $(z_1, z_{1:(n+1)}),...,(z_{n+1}, z_{1:(n+1)})$ and the $n+1$ scores $v_1,...,v_{n+1}$.

On the other hand, if we allow $\mathcal{S}$ to be a nonsymmetric score function, then in general it can be the case that $\mathbb{P}\{V_{n+1} = v_i \mid E_v\} \neq \mathbb{P}\{Z_{n+1} = z_i \mid E_z\}$. This can be seen for instance by comparing the events E_v and E_z: observing an unordered set of data values (i.e., the event E_z) could result in many different possible sets of scores from a nonsymmetric score function. For nonsymmetric $\mathcal{S}$, we thus are not able to make this substitution to equate score probabilities $\mathbb{P}\{V_{n+1} = v_i \mid E_v\}$ to data probabilities $\mathbb{P}\{Z_{n+1} = z_i \mid E_z\}$.

However, a similar argument to the proof in Appendix A.1 could proceed without making this substitution. The main difference is that, rather than the corresponding general result requiring an arbitrary joint PDF f of the *data* in Eq. (10), instead a the result would require an arbitrary joint PDF of the *score values*. By making this observation, we aim to point out that extensions to allow for nonsymmetric score functions could be a interesting direction, but one we leave for future work. Such future work could potentially leverage ideas from the "swap step" in Barber et al. (2023), which the authors introduced to accomodate nonsymmetric algorithms.

[12] Note that these definitions implicitly assume that any randomness is fixed, e.g., so that no equality or inequality is trivial due to differing random seeds.

B. Deriving Weighted Conformal Validity Guarantees and Algorithms for Data under MFCS

This section describes in detail how one can use the general procedure presented in Section 4.3 to derive conformal prediction validity guarantees and practical algorithms for multistep feedback covariate shift (MFCS).

Regarding notation: We will use f to denote the joint probability density function (PDF) for the sequence of random variables $Z_1, ..., Z_{n+t}$ and p to denote more specific (conditional) PDFs, for example $p_{Y|X} = dP_{Y|X}$. To lighten notation we will sometimes omit implicit random variables; for example, we will write $p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid Z_1 = z_1, ..., Z_{j-1} = z_{j-1})$ equivalently as $p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1})$ or even as $p(x_j \mid z_1, ..., z_{j-1})$, if $X_j|Z_1, ..., Z_{j-1}$ are clear from context and if the shorthand may help declutter the notation.[13]

B.1. Deriving MFCS Conformal Validity Guarantees (Sketch Given in Section 4.3)

Here we use the general procedure outlined in Section 4.3 to derive conformal validity guarantees for MFCS.

Step 1: List assumptions

We begin by restating the formal assumptions we used to characterize MFCS in Eq. (3):

$$Z_i = (X_i, Y_i) \overset{\text{i.i.d.}}{\sim} P_X^{(0)} \times P_{Y|X},\ i = 1, \dots n,$$
$$Z_{n+t} = (X_{n+t}, Y_{n+t}) \sim P^{(t)}_{X;Z_{1:(n+t-1)}} \times P_{Y|X},\ t = 1, \dots, T.$$

Recall that informally, these assumptions are stating that (after an IID initialization of n datapoints) the distribution of the covariates X at time t can change arbitrarily depending on past observations $Z_{1:(n+t-1)} = \{Z_1, ..., Z_{n+t-1}\}$, while the conditional label distribution $Y \mid X$ is assumed to remain invariant. Note that $Y \mid X$ being invariant implies $Z_1, ..., Z_{n+t-1} \perp\!\!\!\perp Y_{n+t} \mid X_{n+t}$, where the latter can be read as "Y_{n+t} is independent of past observations $Z_1, ..., Z_{n+t-1}$, given X_{n+t}," and moreover that $Y_i|X_i \overset{d}{=} Y|X$ for all $i \in \{1, ..., n+T\}$.

Step 2: Factorize f

Next, we factorize the joint probability density function (PDF) f into "dynamic" and "invariant" factors—specifically, into factors that are dependent on versus are invariant to *permutations* of the data indices. Because in MFCS it is useful to interpret the data indices as timesteps (where the n IID initialized points as distinct observations all occuring at timestep $t = 0$), we can think of this distinction as one of time-dependence versus time-invariance.

First, for simplicity, let us focus on factorizing out terms for the most recent timestep $n + t$:

$$\begin{aligned}
&f(z_1, ..., z_{n+t}) \\
&\quad = f(z_1, ..., z_{n+t-1}, x_{n+t}, y_{n+t}) && z_{n+t} = (x_{n+t}, y_{n+t}) \\
&\quad = p_{Z_1,...,Z_{n+t-1}}(z_1, ..., z_{n+t-1}) \cdot p_{X_{n+t}|Z_1,...,Z_{n+t-1}}(x_{n+t} \mid z_1, ..., z_{n+t-1}) \\
&\qquad \cdot p_{Y_{n+t}|Z_1,...,Z_{n+t-1},X_{n+t}}(y_{n+t} \mid z_1, ..., z_{n+t-1}, x_{n+t}) && \text{Chain rule of probability or conditional probability density def.} \\
&\quad = p_{Z_1,...,Z_{n+t-1}}(z_1, ..., z_{n+t-1}) \cdot p_{X_{n+t}|Z_1,...,Z_{n+t-1}}(x_{n+t} \mid z_1, ..., z_{n+t-1}) \\
&\qquad \cdot p_{Y|X}(y_{n+t} \mid x_{n+t}) && \begin{array}{c}\text{MFCS invariance assumption:} \\ Y \mid X \text{ invariant} \implies \\ Z_1, ..., Z_{n+t-1} \perp\!\!\!\perp Y_{n+t} \mid X_{n+t} \\ \text{and } Y_{n+t}|X_{n+t} \overset{d}{=} Y|X\end{array}
\end{aligned}$$

These steps are not specific to $n + t$, however; we can repeat the same procedure for $p_{Z_1,...,Z_{n+t-1}}(z_1, ..., z_{n+t-1})$ and the

[13] We also use $p(x \mid Z_1, ..., Z_{j-1})$ to denote the generic the conditional density function that can change depending on different realized values of $Z_1, ..., Z_{j-1}$ (e.g., it could be that $p(x \mid Z_1 = z_1, ..., Z_{j-1} = z_{j-1}) \neq p(x \mid Z_1 = z'_1, ..., Z_{j-1} = z'_{j-1})$ for some $(z_1, ..., z_{j-1}) \neq (z'_1, ..., z'_{j-1})$), whereas $p(x \mid z_1, ..., z_{j-1})$ denotes the specific conditional density, conditional on the joint event $Z_1 = z_1, ..., Z_{j-1} = z_{j-1}$.

index $n+t-1$, and so on. When these steps are performed for all indices $i \in \{1, ..., n+t\}$, we obtain

$$f(z_1, ..., z_{n+t}) = \prod_{j=1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1}) \cdot p_{Y|X}(y_j|x_j) \Big]$$
$$= \prod_{j=1}^{n+t} \Big[\underbrace{p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1})}_{\text{Time-dependent factors}} \Big] \cdot \underbrace{\prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \Big]}_{\text{Time-invariant factor}},$$

that is, we obtain the factorization into time-dependent and time-invariant factors provided in the main paper Eq. (7) (where in the main paper we omitted the random variables from the subscripts for ligher notation).

Step 3: Compute or estimate weights

Lastly, we plug the result of our factorization from Step 2 into Eq. (4) to compute or estimate the conformal weights (for the calibration and test points). Recall that with slightly modified notation from our proof in Appendix A.1 to accommodate $n+t$ data indices, we let $E_z^{(t)}$ denote the event $\{Z_1, ..., Z_{n+t}\} = \{z_1, ..., z_{n+t}\}$.

$$\mathbb{P}\{Z_{n+t} = z_i \mid E_z^{(t)}\} = \frac{\sum_{\sigma:\sigma(n+t)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+t)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+t)})}$$
$$= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)}) \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_{\sigma(j)} \mid x_{\sigma(j)}) \Big]}{\sum_{\sigma} \prod_{j=1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)}) \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_{\sigma(j)} \mid x_{\sigma(j)}) \Big]}$$
$$= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=1}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=1}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})},$$

where this result is equivalent to Eq. (8) in the main paper; it follows because $\prod_{j=1}^{n+t} p_{Y|X}(y_{\sigma(j)} \mid x_{\sigma(j)})$ is invariant to permutations σ, and thus cancels out in the ratio. For more concise notation to refer to these weights derived using MFCS assumptions, let us define

$$\mathbb{P}_{n+t}^{(\text{MFCS})}\{z_i \mid E_z^{(t)}\} := \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=1}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=1}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}. \tag{14}$$

A weighted conformal prediction algorithm with these weights $\mathbb{P}_{n+t}^{(\text{MFCS})}\{z_i \mid E_z^{(t)}\}$ for $i \in \{1, ..., n+t\}$ has a valid coverage guarantee as a corollary[14] to Theorem 4.1, premised on the MFCS assumptions. This is because our derivation only relied on standard probability rules that hold for any joint PDF[15] f along with the MFCS assumptions listed in Step 1. We now state this corollary for completeness.

Corollary B.1. *Assume that $Z_i = (X_i, Y_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, ..., n+t$ have the joint PDF f and are generated under multistep feedback covariate shift as in Eq. (3). For any measurable score function $\mathcal{S}$, and any $\alpha \in (0, 1)$, define the generalized conformal prediction set (based on n calibration samples) at a point $x \in \mathbb{R}^d$ by*

$$\widehat{\mathcal{C}}_n(x) = \left\{ y \in \mathbb{R} : V_{n+t}^{(x,y)} \leq Q_{1-\alpha}\left(\sum_{i=1}^{n} \mathbb{P}_{n+t}^{(MFCS)}\{Z_i \mid E_z\} \cdot \delta_{V_i^{(x,y)}} + \mathbb{P}_{n+t}^{(MFCS)}\{Z_{n+t} \mid E_z\} \cdot \delta_{\infty} \right) \right\} \tag{15}$$

where $V_i^{(x,y)}, i \in \{1, ..., n+t\}$ are as in (5) and $\mathbb{P}_{n+t}^{(MFCS)}\{Z_i \mid E_z\}$ is defined in (14). Then, $\widehat{\mathcal{C}}_n$ satisfies

$$\mathbb{P}\big\{Y_{n+1} \in \widehat{\mathcal{C}}_n(X_{n+1})\big\} \geq 1 - \alpha.$$

[14] We call this result a "corollary" rather than a "theorem" primarily to emphasize that it follows from our main result, Theorem 4.1, when assuming data are generated under MFCS; however, we encourage future authors to use the term they find most fitting for this or other conformal validity guarantees derived from Theorem 4.1.

[15] Or more generally, for any valid Radon-Nikodym derivative with respect to an arbitrary base measure; this also includes discrete probability mass functions (PMFs) and mixtures of PDFs and PMFs.

B.2. Deriving Practical Conformal Algorithms under Agent-Induced MFCS

With the conformal coverage validity guarantee for MFCS in Corollary B.1 at hand, we now turn to deriving practical algorithms for computing or estimating the conformal prediction set defined in Eq. (15) under ML-agent-induced MFCS. As we discuss in Section 4.3 of our main paper, the primary practical bottleneck in this case is computational complexity: While the epistemic challenge is overcome when MFCS is agent-induced (due to $p(x_j \mid Z_1, ..., Z_{j-1})$ representing an ML-agent-controlled query function at time j), the complexity for computing the weights in Eq. (14) (equivalent to our main paper's Eq. (8)) is still $\mathcal{O}(\prod_{j=1}^{t}(n+j))$, which quickly becomes intractable for large t. To alleviate this bottleneck, we thus proposed an estimation that only uses the "highest-order" terms from the d most recent timesteps,

$$\widehat{p}^{(d)}_{n+t}\{z_i \mid E^{(t)}_z\} = \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=n+t+1-d}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=n+t+1-d}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})},$$

where we call d our "estimation depth" and $\widehat{p}^{(d)}_{n+t}\{z_i \mid E^{(t)}_z\}$ is our "d-step" approximation of the MFCS weights.[16] This d-step approximation of the MFCS thus has reduced complexity $\mathcal{O}(\prod_{j=t+1-d}^{t}(n+j))$, which is a tractable polynomial when d is small. This estimation depth d can be specified by a user based on their computational budget, with larger values of d better approximating the coverage validity guarantee at the expense of increasing computational demand.

When the ML model controlling the query function $p(x_j \mid Z_1, ..., Z_{j-1})$ treats its training data $Z_1, ..., Z_{j-1}$ symmetrically at each timestep (as in our experiments), we can factorize this even further and derive a recursive algorithm for computing the weights, wherein d corresponds to the recursion depth. To provide intuition for this derivation and recursive implementation, let us start by letting $d = 3$ and return to the simpler notation used in the main text, which drops the random variables (i.e., leaves them implicit to be more concise):

$$\begin{aligned}\widehat{p}^{(3)}_{n+t}\{z_i \mid E^{(t)}_z\} &= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=n+t-2}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})}{\sum_{\sigma} \prod_{j=n+t-2}^{n+t} p(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})} \\ &= \frac{\sum_{\sigma:\sigma(n+t)=i} p(x_{\sigma(n+t)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-1)}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_{\sigma} p(x_{\sigma(n+t)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-1)}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})},\end{aligned}$$

where the second line simply writes out the product. Next, it will soon help us to write the denominator's summation $\sum_{\sigma}$ equivalently as $\sum_i \sum_{\sigma:\sigma(n+t)=i}$, for $i \in \{1, ..., n+t\} = [n+t]$

$$\widehat{p}^{(3)}_{n+t}\{z_i \mid E^{(t)}_z\} = \frac{\sum_{\sigma:\sigma(n+t)=i} p(x_{\sigma(n+t)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-1)}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i \sum_{\sigma:\sigma(n+t)=i} p(x_{\sigma(n+t)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-1)}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})},$$

as we will now make use of knowing that $\sigma(n+t) = i$ within each of the summations $\sum_{\sigma:\sigma(n+t)=i}$ in the numerator and denominator. To do so, we need slightly more notation: for any subset of the data indices $K \subseteq \{1, ..., n+t\} = [n+t]$, we write $z_{-K} = \{z_i : i \in [n+t] \backslash K\}$ to denote the set of data whose indices are *not* in K; for example, $z_{-\{i,j\}} = \{1, ..., n+t\} \backslash \{i, j\}$. And, since we are assuming ML models that treat their data symmetrically at each timestep, we allow our conditional probabilities to ignore the ordering of data observations being conditioned on (e.g., $p(x \mid z_{-K})$ does not depend on the ordering of the indices in z_{-K}). So, with this assumption and notation, the fact that $\sigma(n+t) = i$ within the summation $\sum_{\sigma:\sigma(n+t)=i}$ implies $p(x_{\sigma(n+t)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-1)}) = p(x_i \mid z_{-\{i\}})$. So, we can make this substitution and

[16] We note that, due to assuming the first n points are IID in MFCS it follows that $p_{X_j|Z_1,...,Z_{j-1}}(x \mid Z_1, ..., Z_{j-1}) = p_X(x)$ for $j \in \{1, ..., n\}$, which can allow for including the terms corresponding to $j \in \{1, ..., n\}$ in the product with only marginally more computation, regardless of d (whereas as we have defined $\widehat{p}^{(d)}_{n+t}\{z_i \mid E^{(t)}_z\}$, selecting $d \in \{1, ..., t\}$ excludes these terms from the product). However, in certain cases such as uniform random IID initialization where all points initial density on each point is a constant, these terms cancel out in the ratio and thus can be ignored without cost. Accordingly, for this reason and for ease of exposition, we focus on the estimated weights $\widehat{p}^{(d)}_{n+t}\{z_i \mid E^{(t)}_z\}$ as we have introduced above.

factor out $p(x_i \mid z_{-\{i\}})$ from $\sum_{\sigma:\sigma(n+t)=i}$, since it no longer depends on σ:

$$\begin{aligned}
&\widehat{p}^{(3)}_{n+t}\{z_i \mid E^{(t)}_z\} \\
&= \frac{\sum_{\sigma:\sigma(n+t)=i} p(x_i \mid z_{-\{i\}}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i \sum_{\sigma:\sigma(n+t)=i} p(x_i \mid z_{-\{i\}}) \cdot p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{\sigma:\sigma(n+t)=i} p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{\sigma:\sigma(n+t)=i} p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}.
\end{aligned}$$

Next, observe that the index $\sigma(n+t)$ no longer appears within the summation $\sum_{\sigma:\sigma(n+t)=i}$ in either the numerator or denominator (it has been factored out). So, similarly as we did before in the denominator, we can equivalently write the summation $\sum_{\sigma:\sigma(n+t)=i}$ equivalently as $\sum_{j\neq i} \sum_{\sigma:\sigma(n+t-1)=j}$, for

$$\begin{aligned}
&\widehat{p}^{(3)}_{n+t}\{z_i \mid E^{(t)}_z\} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} \sum_{\sigma:\sigma(n+t-1)=j} p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} \sum_{\sigma:\sigma(n+t-1)=j} p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})},
\end{aligned}$$

and similarly as before, within the summation $\sigma(n+t-1) = j$, so $p(x_{\sigma(n+t-1)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-2)}) = p(x_j \mid z_{-\{i,j\}})$. Making this substitution and factoring, and then repeating the pattern again for the third term ultimately gives

$$\begin{aligned}
&\widehat{p}^{(3)}_{n+t}\{z_i \mid E^{(t)}_z\} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} \sum_{\sigma:\sigma(n+t-1)=j} p(x_j \mid z_{-\{i,j\}}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} \sum_{\sigma:\sigma(n+t-1)=j} p(x_j \mid z_{-\{i,j\}}) \cdot p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{\sigma:\sigma(n+t-1)=j} p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{\sigma:\sigma(n+t-1)=j} p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{k\neq i,j} \sum_{\sigma:\sigma(n+t-2)=k} p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{k\neq i,j} \sum_{\sigma:\sigma(n+t-2)=k} p(x_{\sigma(n+t-2)} \mid z_{\sigma(1)}, ..., z_{\sigma(n+t-3)})} \\
&= \frac{p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{k\neq i,j} p(x_k \mid z_{-\{i,j,k\}})}{\sum_i p(x_i \mid z_{-\{i\}}) \sum_{j\neq i} p(x_j \mid z_{-\{i,j\}}) \cdot \sum_{k\neq i,j} p(x_k \mid z_{-\{i,j,k\}})}.
\end{aligned}$$

More generally, for an arbitrary estimation depth $d \in [n+t]$ we have the following, with annotations added to indicate how increasing our the depth of recursion d will influence the computation,

$$\widehat{p}^{(d)}_{n+t}\{z_{i_1} \mid E^{(t)}_z\} = \frac{\overbrace{p(x_{i_1} \mid z_{-\{i_1\}})}^{\text{recursion depth 1}} \overbrace{\sum_{i_2\neq i_1} p(x_{i_2} \mid z_{-\{i_1,i_2\}})}^{\text{recursion depth 2}} \cdots \overbrace{\sum_{i_d\neq i_1,...,i_{d-1}} p(x_{i_d} \mid z_{-\{i_1,i_2,...,i_d\}})}^{\text{recursion depth } d}}{\sum_{i_1} p(x_{i_1} \mid z_{-\{i_1\}}) \sum_{i_2\neq i_1} p(x_{i_2} \mid z_{-\{i_1,i_2\}}) \cdots \sum_{i_d\neq i_1,...,i_{d-1}} p(x_{i_d} \mid z_{-\{i_1,i_2,...,i_d\}})}, \tag{16}$$

and where the denominator can be obtained by summing over all the numerator values for $i_1 \in [n+t]$.

Code for our specific implementations of MFCS Full CP and Split CP methods are available at the following GitHub repository: https://github.com/drewprinster/conformal-mfcs.

B.3. Alternative Approaches to Deriving MFCS Conformal Guarantees and Algorithms

Here we describe certain alternative approaches to the derivation steps in Appendix B.1, and we briefly discuss how these alternate steps could have implications for practical algorithm design for MFCS settings and more broadly.

Alternative Step 1: Assumptions via a Probabilistic Graphical Model

Rather than beginning with the MFCS conditional independence assumptions (Eq. (3)) for Step 1, we could have instead begun by describing our data-generating process using a probabilistic graphical model (Koller & Friedman, 2009; Pearl, 2009) and used the conditional independence conditions implied by the missing edges in that model.[17] One advantage of graphical models is that they can be more visually intuitive than conditional independence or invariance assumptions. For example, consider the directed acyclic graph (DAG) model in Figure 4, on which we take a causal interpretation to view each directed edge $A \to B$ as representing that A has a causal effect on B.[18] As in MFCS, in this causal DAG model we first assume $Z_{1:n} = \{Z_1, ..., Z_n\}$ are independent and identically distributed (IID), and in the DAG we represent these IID variables with a single node to be concise; for the remaining nodes, we allow each X_{n+t} to depend causally on all prior observations (i.e., each X_{n+t} has incoming edges from all prior nodes), while we only allow each Y_{n+t} to be affected by X_{n+t} (i.e., Y_{n+t} only has an incoming edge from X_{n+t}). We use blue arrows to represent relationships that we further assume are equivalent (in distribution).

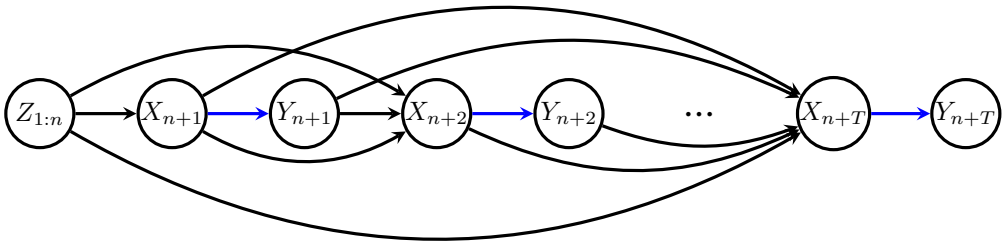

Figure 4. A causal directed acyclic graph (DAG) model that implies $Z_1, ..., Z_{n+t-1} \perp\!\!\!\perp Y_{n+t} \mid X_{n+t}$ (as does MFCS). Moreover, the blue edges represent relationships that are further assumed to be equivalent, implying $Y_{n+t}|X_{n+t} \stackrel{d}{=} Y|X$ for all $t \in \{1, ..., T\}$.

By the rules of d-separation for DAGs (Pearl, 1988), for any $t \in \{1, ..., T\}$, conditioning on X_{n+t} blocks all paths from prior nodes to Y_{n+t}; so, this DAG implies the same conditional independence relation that we obtained from our original MFCS invariance assumptions: $Z_1, ..., Z_{n+t-1} \perp\!\!\!\perp Y_{n+t} \mid X_{n+t}$. Moreover, the blue edges represent relationships that are further assumed to be equivalent in distribution, implying $Y_{n+t}|X_{n+t} \stackrel{d}{=} Y|X$ for all $t \in \{1, ..., T\}$.

Alternative Steps 2 and 3: Factoring f with likelihood-ratio factors

The following is an alternative approach to Steps 2 and 3 in Appendix B.1, which factorizes f using likelihood-*ratio* factors, which more closely resemble the specific CP algorithms for standard covariate shift in Tibshirani et al. (2019) and (one-step) feedback covariate shift in Fannjiang et al. (2022). We will refer to the factorization given in Step 2 of Appendix B.1 as the *direct* likelihood factorization, and the alternative derivation we present here as the likelihood *ratio*-based factorization; we will soon discuss tradeoffs of these different viewpoints regarding interpretability and implementation.

[17] Missing edges in probabilistic graphical models imply conditional independences in the joint PDF f; the converse depends on further assumptions (Pearl, 2009).

[18] We take posit a *causal* DAG here so that the edges better map onto our intuition about cause and effect; however, in other settings different graphical models could instead be used so long as the appropriate Markov property is used to obtain the implied conditional independence relations.

First, recall the result of Step 2 in Appendix B.1:

$$f(z_1, ..., z_{n+t}) = \prod_{j=1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1}) \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \Big].$$

Whereas the direct factorization in Appendix B.1 proceeds immediately from here to Step 3 by plugging this statement into Eq. (4), now we consider further steps to obtain explicit likelihood-ratio factors. To do so, we simplify using our assumption that the first n points are IID, which implies $p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1}) = p_X(x_j)$ for $j \in \{1, ..., n\}$, and then use a "multiply by one" trick to multiply by $1 = \frac{\prod_{j=n+1}^{n+t} p_X(x_j)}{\prod_{j=n+1}^{n+t} p_X(x_j)}$, where $p_X := dP_X^{(0)}$ denotes the density for the features in the initial IID distribution $P_Z^{(0)} = P_X^{(0)} \times P_{Y|X}$:

$$\begin{aligned}
f(z_1, ..., z_{n+t}) &= \prod_{j=1}^{n} \Big[p_X(x_j) \Big] \cdot \prod_{j=n+1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1}) \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \Big] \\
&= \prod_{j=1}^{n} \Big[p_X(x_j) \Big] \cdot \prod_{j=n+1}^{n+t} \Big[p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1}) \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \Big] \cdot \frac{\prod_{j=n+1}^{n+t} p_X(x_j)}{\prod_{j=n+1}^{n+t} p_X(x_j)} \\
&= \prod_{j=1}^{n+t} \Big[p_X(x_j) \Big] \cdot \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1})}{p_X(x_j)} \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \Big] \\
&= \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1})}{p_X(x_j)} \Big] \cdot \prod_{j=1}^{n+t} \Big[p_{Y|X}(y_j|x_j) \cdot p_X(x_j) \Big] \\
&= \underbrace{\prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_j \mid z_1, ..., z_{j-1})}{p_X(x_j)} \Big]}_{\substack{\text{Time-dependent} \\ \text{likelihood-ratio factors}}} \cdot \underbrace{\prod_{j=1}^{n+t} \Big[p_Z(z_j) \Big]}_{\substack{\text{Time-invariant} \\ \text{"core" function}}}.
\end{aligned}$$

There are two main tradeoffs between the direct likelihood factorization (Step 2 of Appendix B.1) and the alternative likelihood-ratio factorization presented here—the first tradeoff is with respect to interpretability, and the second tradeoff is with respect to implementation. First, regarding interpretability: a benefit of the factorization presented in Appendix B.1 is that its permutation-invariant factor $\prod_{j=1}^{n+t} p_{Y|X}(y_j \mid x_j)$ clearly and intuitively corresponds to the assumption that $Y \mid X$ is invariant in MFCS, which is one reason we chose to use that factorization in our main paper's presentation; on the other hand, a benefit of the current likelihood-ratio factorization is that it can be interpreted more clearly as "the likelihood obtained by *adjusting relative to if all the data were IID*," since the permutation-invariant factor $\prod_{j=1}^{n+t} p_Z(z_j)$ represents the probability of the sequence of observations *if* they were all IID, and the likelihood ratios represent the necessary adjustment factors needed to compute the true probability. The current likelihood ratio-based factorization also relates more closely to the analysis and algorithms in Tibshirani et al. (2019) and Fannjiang et al. (2022).

Next, to discuss implementation tradeoffs, it may be helpful to first complete Step 3 by plugging our factorization into the equation (Eq. (4)) for the CP weights:

$$\begin{aligned}
\frac{\sum_{\sigma:\sigma(n+t)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})} &= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)},...,z_{\sigma(j-1)})}{p_X(x_{\sigma(j)})} \Big] \prod_{j=1}^{n+1} \Big[p_Z(z_{\sigma(j)}) \Big]}{\sum_{\sigma} \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)},...,z_{\sigma(j-1)})}{p_X(x_{\sigma(j)})} \Big] \prod_{j=1}^{n+1} \Big[p_Z(z_{\sigma(j)}) \Big]} \\
&= \frac{\sum_{\sigma:\sigma(n+t)=i} \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)},...,z_{\sigma(j-1)})}{p_X(x_{\sigma(j)})} \Big]}{\sum_{\sigma} \prod_{j=n+1}^{n+t} \Big[\frac{p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)},...,z_{\sigma(j-1)})}{p_X(x_{\sigma(j)})} \Big]}.
\end{aligned}$$

Completing this Step 3 emphasizes that it is the "dynamic" factors (that are not permutation invariant) which are of practical concern for implementation, since the permutation-invariant factor $\prod_{j=1}^{n+t} p_Z(z_{\sigma(j)})$ has cancelled out in the ratio.

One implementation tradeoff between the direct (Appendix B.1) and the likelihood ratio-based factorization here is an epistemic one: Implementations using the direct factorization rely on computing or estimating the product of likelihoods $\prod_{j=1}^{n+t} p_{X_j|Z_1,...,Z_{j-1}}(x_{\sigma(j)} \mid z_{\sigma(1)}, ..., z_{\sigma(j-1)})$; we leverage this direct approach in our main paper largely because our practical experiments focus on ML-agent-induced MFCS where $p_{X_j|Z_1,...,Z_{j-1}}(x \mid Z_1, ..., Z_{j-1})$ is known (it is the ML agent's query function). On the other hand, for a hypothetical MFCS instance where $p_{X_j|Z_1,...,Z_{j-1}}(x \mid Z_1, ..., Z_{j-1})$ is *not* known, an implementation based on the likelihood-ratio factorization may be preferred, because likelihood-ratio (density-ratio) estimation can often be easier than direct likelihood (density) estimation.

Another implementation consideration to note is the "failure mode" of specific permutations σ that have zero probability density, for example due to $p_X(x_{\sigma(j)}) = 0$ for some $j \in \{n+1, ..., n+t\}$, which could occur if $X_{n+t} = x_{n+t}$ is observed out of the support of $P_X^{(0)}$, the initial distribution that the first n points are assumed to be drawn IID from in MFCS. Implementations based on the likelihood ratio-based factorization would be undefined in such cases due to requiring dividing by zero, and thus they may need to exclude these cases by assuming that the density in the numerator is absolutely continuous with respect to that of the denominator (e.g., Tibshirani et al. (2019) and Fannjiang et al. (2022) made this assumption for their specific CP algorithms for one-step standard and feedback covariate shift). On the other hand, implementations using the direct factorization (Appendix B.1) are not necessarily undefined in these cases—that is, if one commits to a definition of conditional probability densities where $p(A|B) = 0$ if $p(B) = 0$, rather allowing the conditional density to be undefined in such cases (see e.g., Kolmogorov (1956)). For a specific example, let us assume $T = 1$ and that $X_{n+1} = x_{n+1}$ is observed out of the support of $P_X^{(0)}$, so $p_X(x_{n+1}) = 0$. Then (assuming a conditional probability density definition where $p(A|B) = 0$ if $p(B) = 0$), the MFCS weights obtained via direct factorization in Eq. (14) would reduce to $\mathbb{P}_{n+1}^{(\text{MFCS})}\{z_i \mid E_z^{(t)}\} = 0$ for all $i \neq n+1$, and $\mathbb{P}_{n+1}^{(\text{MFCS})}\{z_{n+1} \mid E_z^{(t)}\} = 1$. Accordingly, the resulting CP set would maintain coverage trivially by concentrating all the weight on δ_∞ (the point mass at ∞) in the (conservative) empirical score distribution, thus forcing a non-informative prediction set $\widehat{C}_n(x) = \mathcal{Y}$.

C. Bounded Query Function Experiments for Adaptive Exploration

C.1. Bounding the Query Function to Enforce Informative (Finite) Prediction Intervals

Because noninformative prediction sets ($\widehat{\mathcal{C}}_n(x) = \mathcal{Y}$) occur in our weighted CP MFCS experiments when a test point's normalized weight as in Eq. (9) (implemented as in Eq. (16)) exceeds α, we now consider redesigning the ML query probability functions to prevent this from happening. For simplicity, we impose this constraint on the one-step weights. In particular, we consider bounding the query probability function $p(x \mid Z_1, ..., Z_{j-1})$ so that instead of being proportional to $\exp(\lambda \cdot u_t(x))$, the probabilities are instead proportional to a bounded function:

$$p(x \mid Z_1, ..., Z_{j-1}) \propto \min\big(\exp(\lambda \cdot u_t(x)), B\big), \quad \text{where } B = \sup\left\{b > 0 : \frac{b}{\sum_{i=1}^n \min\big(\exp(\lambda \cdot u_t(X_i)), b\big) + b} < \alpha\right\}.$$

That is, we select B as the largest bound that we could use to avoid infinite-width prediction intervals (in the depth $d = 1$ weights). The selection of B above avoids noninformative prediction sets because

$$\min\big(\exp(\lambda \cdot u_t(x)), B\big) \leq B$$
$$\implies \frac{\min\big(\exp(\lambda \cdot u_t(x)), B\big)}{\sum_{i=1}^n \min\big(\exp(\lambda \cdot u_t(X_i)), B\big) + \min\big(\exp(\lambda \cdot u_t(x)), B\big)} \leq \frac{B}{\sum_{i=1}^n \min\big(\exp(\lambda \cdot u_t(X_i)), B\big) + B} < \alpha.$$

In our implementations, we solve for B via a binary search algorithm over all values of the unbounded function $\exp(\lambda \cdot u_t(x))$ for data in the input space $x \in \mathcal{X}$, which is tractable in our experiments where $\mathcal{X}$ is the finite pool or library of possible query points. That is, we specifically solve for $B = \max\left\{b = \exp(\lambda \cdot u_t(x)) \ : \ x \in \mathcal{X}, \ \frac{b}{\sum_{i=1}^n \min(\exp(\lambda \cdot u_t(X_i)), b) + b} < \alpha\right\}$. Code for our implementations is available here: https://github.com/drewprinster/conformal-mfcs.

Across the four real datasets and the same experimental conditions we used for our active learning experiments in Figure 3 of our main paper, the following results empirically validate that imposing the proposed adaptive bound on our MFCS CP method's query function prevents any noninformative (infinite width) intervals, all without sacrificing coverage even over a long time horizon (70 active learning steps). For easy comparison, for each dataset we provide the corresponding results for the original unbounded query function from Figure 3. We moreover plot the "relative magnitude" of the bound B—i.e., we plot $B/B_{\max}$, where $B_{\max} = \max_{x \in \mathcal{X}}(\exp(\lambda \cdot \hat{\sigma}(x)))$—across the same trajectory of active learning steps.

Airfoil dataset

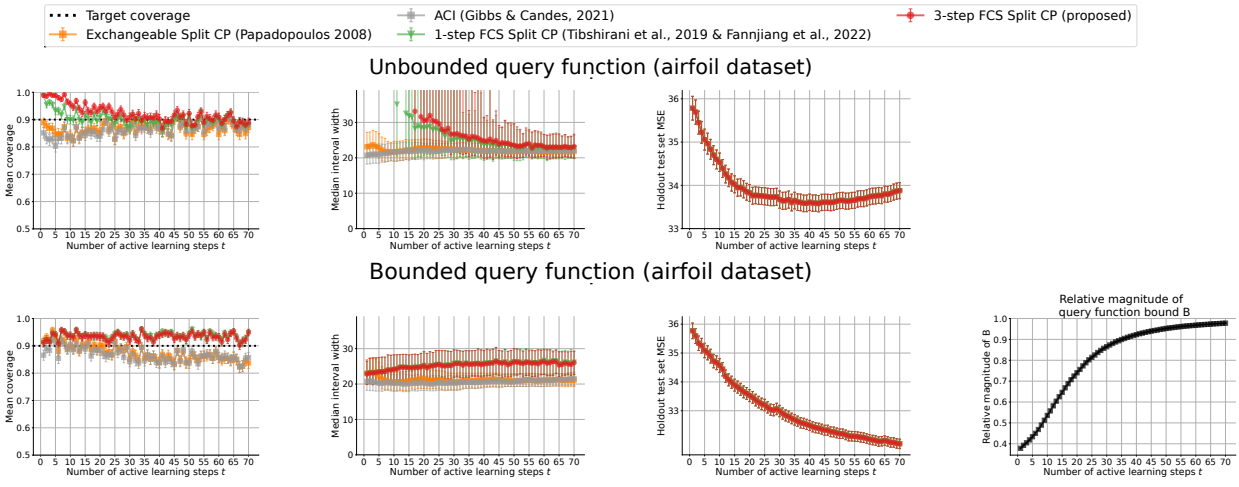

Figure 5. Unbounded (top row) versus bounded (bottom row) active learning experiments of proposed multistep split CP method for $d = 3$ (red circles) compared to baselines of unweighted split CP (orange squares), one-step split CP (green triangles), and ACI (gray squares) on the airfoil dataset. The Y-axes represent mean coverage, median interval width, and mean squared error on a holdout test set; the X-axes correspond to the number of active learning query steps, with each query based on posterior variance of a GP regressor. All values are computed over 350 distinct random seeds. Hyperparameters for the experiments are given in Appendix E.2 Table 3.

Communities dataset

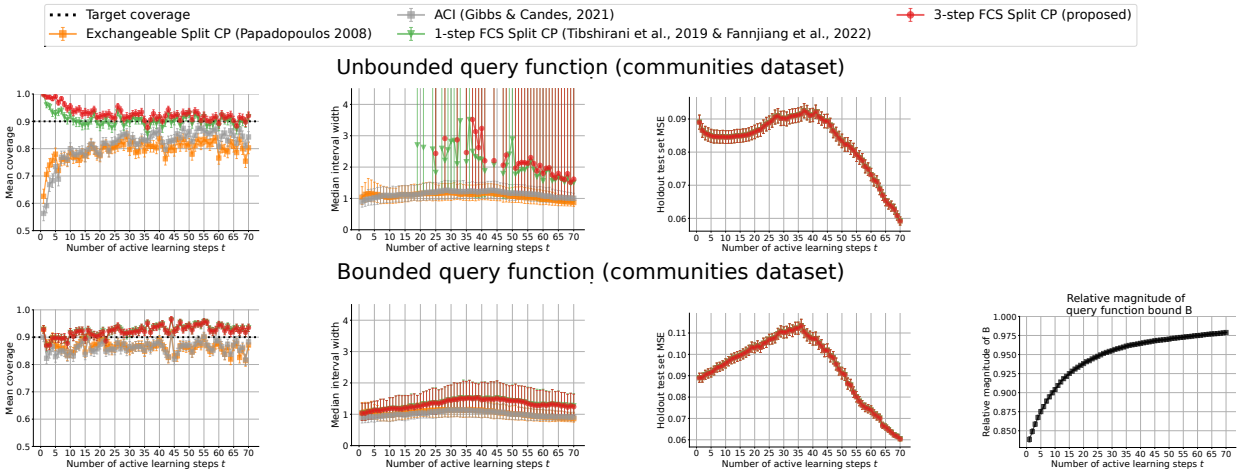

Figure 6. Unbounded (top row) versus bounded (bottom row) active learning experiments of proposed multistep split CP method for $d = 3$ (red circles) compared to baselines of unweighted split CP (orange squares), one-step split CP (green triangles), and ACI (gray squares) on the communities dataset. The Y-axes represent mean coverage, median interval width, and mean squared error on a holdout test set; the X-axes correspond to the number of active learning query steps, with each query based on posterior variance of a GP regressor. All values are computed over 350 distinct random seeds. Hyperparameters for the experiments are given in Appendix E.2 Table 3.

Medical Expenditure Panel Survey (MEPS) dataset

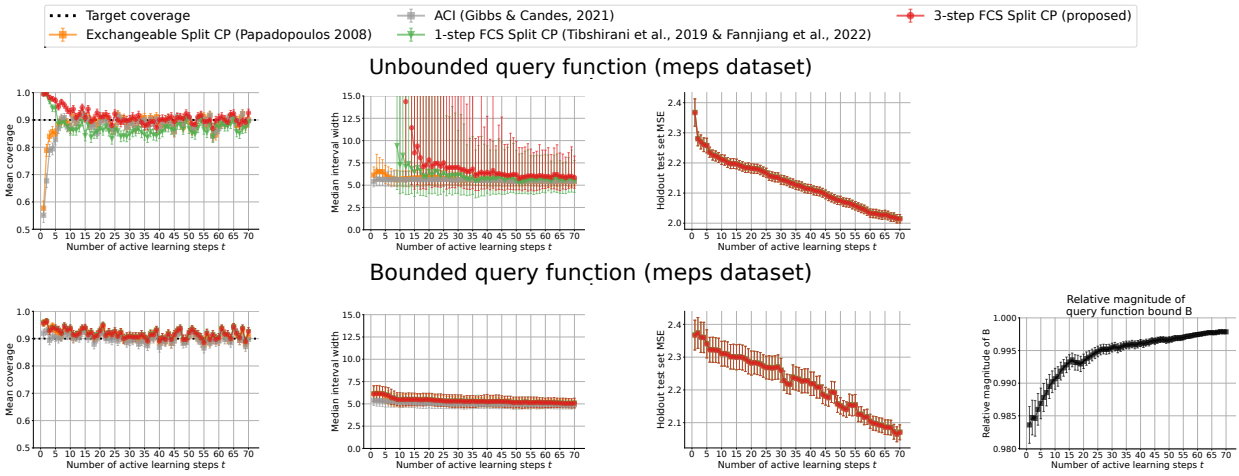

Figure 7. Unbounded (top row) versus bounded (bottom row) active learning experiments with proposed multistep split CP method for estimation depth $d = 3$ (red circles) compared to baselines of unweighted split CP (orange squares), one-step split CP (green triangles), and ACI (gray squares) on the Medical Expenditure Panel Survey (MEPS) dataset. The Y-axes represent mean coverage, median interval width, mean squared error on a holdout test set, and the relative bound magnitude $B/B_{\max}$; the X-axes correspond to the number of active learning query steps, with each query based on posterior variance of a GP regressor. All values are computed over 350 distinct random seeds. Hyperparameters for the experiments are given in Appendix E.2 Table 3.

Blog dataset

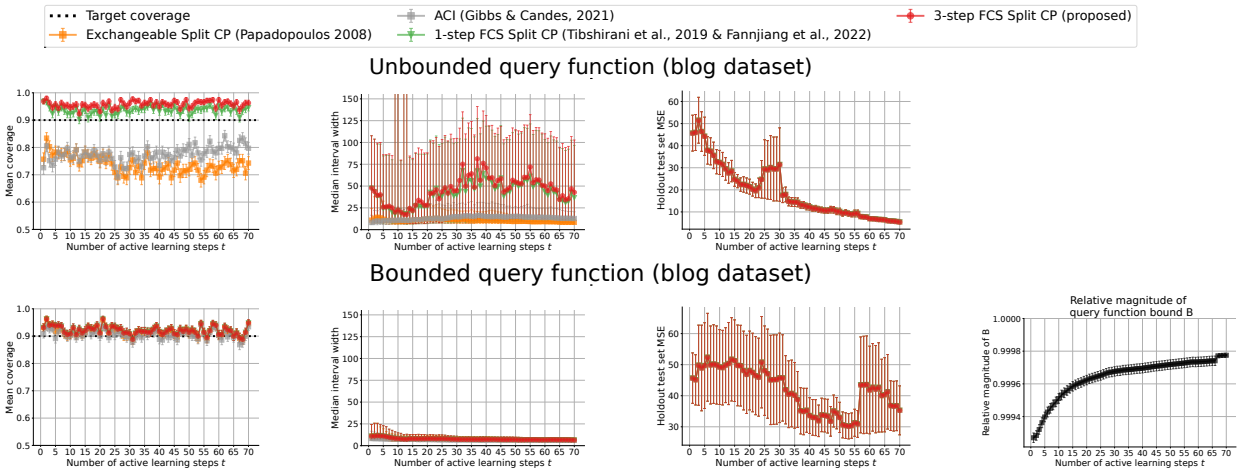

Figure 8. Unbounded (top row) versus bounded (bottom row) active learning experiments of proposed multistep split CP method for $d = 3$ (red circles) compared to baselines of unweighted split CP (orange squares), one-step split CP (green triangles), and ACI (gray squares) on the blog dataset. The Y-axes represent mean coverage, median interval width, and mean squared error on a holdout test set; the X-axes correspond to the number of active learning query steps, with each query based on posterior variance of a GP regressor. All values are computed over 350 distinct random seeds. Hyperparameters for the experiments are given in Appendix E.2 Table 3.

The bounded query function's improvement in prediction-interval informativeness (smaller prediction intervals relative to the unbounded query function) sometimes comes at the cost of an initially delayed decrease in MSE, though the long-run MSE for the bounded and unbounded query functions are comparable for most datasets. The bounded query function can thus be thought of as an "informativeness constraint" that forces the AI/ML agent to initially be more "cautious" or "slow" in its exploration until it has accumulated enough training and calibration data to explore more aggressively (as evidenced by the increasing bound magnitude over time).

D. Details on the Roles of Exchangeability, Weighted Exchangeability, and Pseudo-Exchangeability in Conformal Prediction

D.1. The Role of Exchangeability in Conformal Prediction

Summary: Standard conformal prediction can be viewed as an inverted permutation test, where exchangeability allows us to "weigh" or count each permutation equally.

Appendix A.6 of Tibshirani et al. (2019) provides an alternate proof to a key Lemma underlying (standard) conformal prediction's validity, which corresponds to viewing conformal prediction as an inverted permutation test (though they do not explicitly state this). We reproduce most of that argument here, with some additional explanatory commentary to highlight the role played by the exchangeability assumption in standard conformal prediction.

The general strategy in Tibshirani et al. (2019) is "to condition on the unlabeled multiset of values $\{v_1, ..., v_{n+1}\}$ obtained by our random variables $V_1, ..., V_{n+1}$, and then inspect the probabilities that the last random variable V_{n+1} attains each one of these values." That is, Tibshirani et al. (2019) condition on the event $\{V_1, ..., V_{n+1}\} = \{v_1, ..., v_{n+1}\}$, which they denote by E_v—recall that this is *not* saying that $V_i = v_i$ for all $i \in \{1, ..., n+1\}$. Rather, it is only saying that the set or "bag"[19] of values $\{v_1, ..., v_{n+1}\}$ are the values obtained by the random variables $V_1, ..., V_{n+1}$; we do not yet know whether v_1 is the value obtained by V_1, or by V_2, or by V_{n+1}, etc.

With this notation, Tibshirani et al. (2019) consider the following probability:

$$\mathbb{P}\{V_{n+1} = v_i \mid E_v\}, i = 1, ..., n+1,$$

that is, the probability that the value obtained by V_{n+1} is v_i, conditioned on E_v (conditioned on the bag of values).

Next is where Tibshirani et al. (2019) implicitly use an inverted permutation test. With f denoting the joint density function, they state the following, which is analogous to Eq. (4) in our main paper:

$$\mathbb{P}\{V_{n+1} = v_i \mid E_v\} = \frac{\sum_{\sigma:\sigma(n+1)=i} f(v_{\sigma(1)}, ..., v_{\sigma(n+1)})}{\sum_{\sigma} f(v_{\sigma(1)}, ..., v_{\sigma(n+1)})}.$$

This statement can be thought of as an inverted permutation test as it equates the probability $\mathbb{P}\{V_{n+1} = v_i \mid E_v\}$ to the total probability mass of all permutations σ that map $n+1$ to i, normalized across all possible permutations of the $n+1$ values. For intuition, more explicit notation with random variables can be helpful: That is, if f is a PDF, then writing $f(v_{\sigma(1)}, ..., v_{\sigma(n+1)}) = f_{V_1,...,V_{n+1}}(v_{\sigma(1)}, ..., v_{\sigma(n+1)})$; if f is a PMF, then writing $f(v_{\sigma(1)}, ..., v_{\sigma(n+1)}) = f(V_1 = v_{\sigma(1)}, ..., V_{n+1} = v_{\sigma(n+1)})$. In either case, writing the random variables explicitly can help to emphasize that each argument in f still corresponds to the index of the *random variable*; meanwhile, the *observed values* are being permuted to different argument locations.

Next is where exchangeability comes in. That is, exchangeability means $f(v_{\sigma(1)}, ..., v_{\sigma(n+1)}) = f(v_1, ..., v_{n+1})$ for permutations σ of the indices $\{1, ..., n+1\}$, so by plugging in this substitution and factoring out $f(v_1, ..., v_{n+1})$ from the summation, we can simplify to reduce our statement to counting permutations:

$$\begin{aligned}
\mathbb{P}\{V_{n+1} = v_i \mid E_v\} &= \frac{\sum_{\sigma:\sigma(n+1)=i} f(v_1, ..., v_{n+1})}{\sum_{\sigma} f(v_1, ..., v_{n+1})} \\
&= \frac{f(v_1, ..., v_{n+1}) \sum_{\sigma:\sigma(n+1)=i} 1}{f(v_1, ..., v_{n+1}) \sum_{\sigma} 1} \\
&= \frac{\sum_{\sigma:\sigma(n+1)=i} 1}{\sum_{\sigma} 1} \\
&= \frac{n!}{(n+1)!} = \frac{1}{n+1}.
\end{aligned}$$

That is, the last line follows because (in the denominator's summation) there are $(n+1)!$ possible permutations of $n+1$ values and (in the numerator's summation) there are $n!$ possible permutations of n values (since in the numerator's summation

[19] For simplicity, Tibshirani et al. (2019) assume there are almost surely no ties among those scores (to work with sets rather than multisets/bags); the argument applies in the general case but with more cumbersome notation.

we only consider σ where $\sigma(n+1) = 1$, there are only n values unknown). In other words, under the assumption of exchangeability, every permutation of the data is equally likely, so the probability of the event $\{V_{n+1} = v_i \mid E_v\}$ can be computed as the total number of permutations (of the $n+1$ values) where $\sigma(n+1) = i$, divided by the total number of all possible permutations (of the $n+1$ values).

The remainder of the argument in Tibshirani et al. (2019) then follows similarly as in the proof given in Appendix A.1, but premised on the assumption of exchangeability. That is, Tibshirani et al. (2019) explain that the above result shows that the distribution of $V_{n+1}|E_z$ is uniform on the set $\{v_1, ..., v_{n+1}\}$:

$$V_{n+1}|E_v \sim \frac{1}{n+1}\sum_{i=1}^{n+1}\delta_{v_i},$$

where δ_v denotes a point mass at the value v. The remaining steps of the proof then follow by the definition of a quantile, by the meaning of conditioning on E_z, and by marginalizing (similarly as in Appendix A.1). The resulting standard CP coverage guarantee can be viewed as a corollary to Theorem 4.1 because its proof (as presented by Tibshirani et al. (2019)) follows the same steps, except also relying on the simplifying assumption of exchangeability to allow us to count every permutation equally.

Lastly, we note that while standard conformal prediction is often defined by taking the level $(1-\alpha)\cdot\frac{n+1}{n}$ quantile over the values $\{v_1, ..., v_n\}$, this is equivalent to taking the level $1-\alpha$ quantile of values $\{v_1, ..., v_n\} \cup \{\infty\}$. Tibshirani et al. (2019) introduced the latter with the point mass δ notation we have been using to enable writing a *weighted* empirical distribution of score values.

D.2. The Role of Weighted Exchangeability in Conformal Prediction

Summary: Weighted CP can be viewed as an inversion of *weighted* permutation tests, where the weight functions tell us how much to "weigh" or count each permutation.

First, we restate the definition of weighted exchangeability from Tibshirani et al. (2019):

Definition 1: *Random variables $V_1, ..., V_n$ are said to be* weighted exchangeable, *with weight functions $w_1, ..., w_n$, if the density[20] f, of their joint distribution can be factorized as*

$$f(v_1, ..., v_n) = \prod_{i=1}^{n} w_i(v_i)\cdot g(v_1, ..., v_n) \tag{17}$$

where g is any function that does not depend on the ordering of its inputs, i.e., $g(v_{\sigma(1)}, ..., v_{\sigma(n)}) = g(v_1, ..., v_n)$ for any permutation σ of $1, ..., n$.

We now sketch the argument in appendix A.7 of Tibshirani et al. (2019), which proves a key lemma underlying the validity of weighted CP, with the goal of understanding the role of the weight functions in an inversion of a weighted permutation test.

Similarly as before and as in equation (4),[21]

$$\begin{aligned}\mathbb{P}\{V_{n+1} = v_i \mid E_z\} =& \mathbb{P}\{Z_{n+1} = z_i \mid E_z\}\\ =& \frac{\sum_{\sigma:\sigma(n+1)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}.\end{aligned}$$

Assuming that the data are weighted exchangeability, this becomes

$$\mathbb{P}\{V_{n+1} = v_i \mid E_z\} = \frac{\sum_{\sigma:\sigma(n+1)=i}\prod_{j=1}^{n+1} w_j(z_{\sigma(j)})g(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma}\prod_{j=1}^{n+1} w_j(z_{\sigma(j)})g(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}.$$

[20] Tibshirani et al. (2019) give the following footnote here: "As before, f may be the Radon-Nikodym derivative with respect to an arbitrary base measure."

[21] Again, for simplicity, Tibshirani et al. (2019) assume that there are almost surely no ties in the values $\{v_1, ..., v_{n+1}\}$ to work with sets rather than multisets/bags, but the results holds more generally (with more cumbersome notation).

Recalling the definition of weighted exchangeability wherein g is a permutation-invariant function, we are able to substitute $g(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = g(z_1, ..., z_{n+1})$, analogously to how we substituted $f(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = f(z_1, ..., z_{n+1})$ using exchangeability in Appendix D.1:

$$\begin{aligned}\mathbb{P}\{V_{n+1} = v_i \mid E_z\} =& \frac{\sum_{\sigma:\sigma(n+1)=i} \prod_{j=1}^{n+1} w_j(z_{\sigma(j)}) g(z_1, ..., z_{n+1})}{\sum_{\sigma} \prod_{j=1}^{n+1} w_j(z_{\sigma(j)}) g(z_1, ..., z_{n+1})} \\ =& \frac{\sum_{\sigma:\sigma(n+1)=i} \prod_{j=1}^{n+1} w_j(z_{\sigma(j)})}{\sum_{\sigma} \prod_{j=1}^{n+1} w_j(z_{\sigma(j)})}.\end{aligned}$$

From this simplified statement, it is relatively easier to see that the product $\prod_{j=1}^{n+1} w_j(z_{\sigma(j)})$ can be thought of as how much more likely it would be to observe $Z_1 = z_{\sigma(1)}, ..., Z_{n+1} = z_{\sigma(n+1)}$ (given E_z), relative to the measure that our core function g assigns to this event—i.e., relative to $g(z_{\sigma(1)}, ..., z_{\sigma(n+1)}) = g(z_1, ..., z_{n+1})$. Put more simply, $\prod_{j=1}^{n+1} w_j(z_{\sigma(j)})$ tells us how much we should "weigh" or "count" a given permutation σ.

D.3. The Role of Pseudo-Exchangeability in Conformal Prediction

Summary: Pseudo-exchangeability, as defined in Fannjiang et al. (2022), has a similar role as weighted exchangeability but with "factor functions" in the place of weight functions, and where the factor functions are written to explicitly depend on other datapoints. While pseudo-exchangeability describes *one-step* feedback covariate shift as a special case (as it was seemingly introduced mainly for this purpose), it does not formally describe MFCS, as an artifact of the definition.

First, we recall the definition of pseudo-exchangeability from (the appendix of) Fannjiang et al. (2022):

Definition 1: *Random variables $V_1, ..., V_{n+1}$ are* pseudo-exchangeable *with factor functions $g_1, ..., g_{n+1}$ and core function h if the density, f, of their joint distribution can be factorized as*

$$f(v_1, ..., v_{n+1}) = \prod_{i=1}^{n+1} g_i(v_i; v_{-i}) \cdot h(v_1, ..., v_{n+1}) \tag{18}$$

where $v_{-i} = v_{1:(n+1)} \backslash v_i$, each $g_i(\cdot; v_{-i})$ is a function that depends on the multiset v_{-i} (that is, on the values in v_{-i} but not on their ordering), and h is a function that does not depend on the ordering of its $n+1$ inputs.

Note that pseudo-exchangeability is similar to weighted exchangeability, except the "factor functions" $g_i(v_i; v_{-i})$ take the place of the weight functions $w_i(v_i)$ to explicitly describe how v_i can depend (but only symmetrically, as per the definition) on the other observations v_{-i}. The analogous argument in Fannjiang et al. (2022) is very similar to that of Tibshirani et al. (2019) except using pseudo-exchangeability.[22] As before, we begin with the statement of equation (4):

$$\begin{aligned}\mathbb{P}\{V_{n+1} = v_i \mid E_z\} =& \mathbb{P}\{Z_{n+1} = z_i \mid E_z\} \\ =& \frac{\sum_{\sigma:\sigma(n+1)=i} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} f(z_{\sigma(1)}, ..., z_{\sigma(n+1)})},\end{aligned}$$

Using pseudo-exchangeability, this becomes

$$\mathbb{P}\{Z_{n+1} = z_i \mid E_z\} = \frac{\sum_{\sigma:\sigma(n+1)=i} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)}) h(z_{\sigma(1)}, ..., z_{\sigma(n+1)})}{\sum_{\sigma} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)}) h(z_{\sigma(1)}, ..., z_{\sigma(n+1)})},$$

where the core function h does not depend on the ordering of its inputs, so we have

$$\begin{aligned}\mathbb{P}\{Z_{n+1} = z_i \mid E_z\} =& \frac{\sum_{\sigma:\sigma(n+1)=i} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)}) h(z_1, ..., z_{n+1})}{\sum_{\sigma} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)}) h(z_1, ..., z_{n+1})} \\ =& \frac{\sum_{\sigma:\sigma(n+1)=i} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)})}{\sum_{\sigma} \prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)})}.\end{aligned} \tag{19}$$

[22] Again, for simplicity, Fannjiang et al. (2022) assume that there are almost surely no ties to work with sets rather than multisets/bags, but the results holds more generally (with more cumbersome notation).

Here, in the same way that for weighted exchangeable data the product $\prod_{j=1}^{n+1} w_j(z_{\sigma(j)})$ represents the (relative) likelihood of observing the joint sequence of events $Z_1 = z_{\sigma(1)}, ..., Z_{n+1} = z_{\sigma(n+1)}$ (given E_Z), for pseudo-exchangeable data the analogous role is played by the product $\prod_{j=1}^{n+1} g_j(z_{\sigma(j)}; z_{-\sigma(j)})$: the role of this product is to tell us how much to "weigh" or "count" a given permutation σ. However, the specification in pseudo-exchangeability's definition that "$g_i(\cdot; v_{-i})$ is a function that depends on the *multiset* v_{-i} (that is, on the values in v_{-i} but not on their ordering)" (emphasis added) prevents pseudo-exchangeability from formally describing multistep feedback covariate shift. We state this more formally in the following remark.

Remark D.1. Pseudo-exchangeability does not formally characterize *multistep* FCS. Consider a two-step instance of FCS for a counterexample, that is with data $Z_1, ..., Z_{n+2}$ generated under MFCS. The factor function for the first step $g_{n+1}(z_{n+1}\ ;\ Z_1 = z_1, ..., Z_n = z_n)$ depends on $Z_1, ..., Z_n$ differently than it does on Z_{n+2} (i.e., it does not depend on Z_{n+2} at all). So, the factor function $g_{n+1}(\cdot\ ;\ Z_1 = z_1, ..., Z_n = z_n)$ cannot be said to depend on the multiset of values $z_{-(n+1)} = \{z_1, ..., z_n, z_{n+2}\}$ independently of the ordering of the values (i.e., it requires knowing which of the values corresponds to $Z_{1:n}$ versus Z_{n+2}), as is required by the defintion of pseudo-exchangeability from Fannjiang et al. (2022).

E. Additonal Details for Main Paper Experiments

E.1. Black-box Optimization Experimental Details

Protein Design Datasets Our multistep biomolecular design experiments leverage fluorescent protein data from Poelwijk et al. (2019), previously used for evaluation of related single-step FCS CP methods in Fannjiang et al. (2022) and Prinster et al. (2023). This dataset has the benefit of a combinatorially complete set of labels, which simulates measuring a design in an experiment. In particular, both a "blue" and "red" wavelength fluorescence strength were experimentally measured for each of $2^{13} = 8,192$ possible combinations of binary variations to a wild-type (natural) fluorescent protein at 13 positions on the primary sequence.

Measurement Noise for Protein Design Experiments We followed the same procedure for adding measurement noise as described in Fannjiang et al. (2022)'s Supplementary Materials S3. That is, every time the i-th sequence was sampled from the combinatorially complete library (including for initial training and calibration data as well as for each query point), zero-mean Guassian noise was added (to the label value) with standard deviation set to the absolute residual between that i-th sequence's true label and the prediction on that i-th sequence from a separately fit linear model. The separately-fit linear model had as its input covariates the up to seventh-order terms identified as statistically significant by Poelwijk et al. (2019), as described by Fannjiang et al. (2022).

Table 1. Hyperparameters for Full CP protein design experiments with ridge regression predictor (Figure 1).

Hyperparameter name	Value(s)
α (Target miscoverage rate)	0.1
n (Initial number of training/calibration samples)	32
λ (Shift magnitude or "inverse temperature")	8.0
d (MFCS estimation depth)	$\{1, 2\}$
T (Number of MFCS steps past IID initialization)	5
Ridge regression regularization strength	0.01
Random seeds	$\{0, ..., 999\}$

Table 2. Hyperparameters for Split CP protein design neural network experiments (Figure 2) with the `MLPRegressor` from `scikit-learn` (L-BFGS solver and logistic activation function, default parameters otherwise).

Hyperparameter name	Value(s)
α (Target miscoverage rate)	0.1
n_{train} (Initial number of training samples)	32
n_{cal} (Initial number of calibration samples)	32
λ (Shift magnitude or "inverse temperature")	5.0
d (MFCS estimation depth)	$\{1, 2, 3, 4\}$
T (Number of MFCS steps past IID initialization)	10
Probability each queried point is added to training (versus calibration) data	0.5
Random seeds	$\{0, ..., 499\}$

E.2. Active Learning Experimental Details

Datasets for Active Learning Experiments We evaluated on four datasets from the UCI Machine Learning Repository (Frank, 2010) that are commonly used for evaluation in the CP literature (Tibshirani et al., 2019; Barber et al., 2021; Prinster et al., 2022; 2023) and represent a range of sample sizes and dimensionalities: the NASA airfoil self-noise dataset (1503 samples, $p = 5$) (Brooks et al., 2014), the communities and crime dataset (1994 samples, $p = 99$) (Redmond, 2009), the Medical Expenditure Panel Survey 2016 data set[23] (33005 samples, $p = 107$) (preprocessed as in Barber et al. (2021);

[23] https://meps.ahrq.gov/mepsweb/data_stats/download_data_files_detail.jsp?cboPufNumber=HC-192

details for an older version of the data are in Ezzati-Rice et al. (2008)), and the blog feedback dataset (60021 samples, $p = 281$) (Buza, 2014).

Sampling Holdout Test Set and Initial Training Set For all experiments, a holdout test set of 250 samples was first sampled uniformly at random from the full dataset to track the accuracy of the base predictor as the active learning progressed. To simulate a practical active learning setting where there often tends to be sample-selection bias in how the initial training data is obtained relative to the desired test distribution, we sampled the initial training and calibration datasets from the remaining non-test-set data with probabilities proportional to $\exp(\gamma \cdot X_{\text{PCA1}}^{\text{normed}})$, where $X_{\text{PCA1}}^{\text{normed}}$ is the min-max-normed first principal component representation of the data (that is, the dataset projected onto the first principal component extracted from the same dataset).

Measurement Noise for Active Learning Experiments Every time the i-th datapoint was sampled (including for the holdout test set, initial training/calibration data, and each queried point), zero-mean Gaussian noise was added to the label value. The standard deviation for the Gaussian noise for the i-th datapoint was set as 0.05 times the absolute residual between that i-th point's true label value and the prediction for that i-th point by a separately fit kernel ridge regression model that was fit to all of the data samples. In particular, the separately-fit regressor was the `KernelRidge` method from `scikit-learn`, with regularization strength 1.0 and default parameters otherwise.

Table 3. Hyperparameters for Split CP active learning experiments (Figure 3) with the `GaussianProcessRegressor` from `scikit-learn` as the active learning agent (kernel=DotProduct(sigma_0=0.05)+WhiteKernel(noise_level=0.05), default parameters otherwise).

Hyperparameter name	Value(s)
α (Target miscoverage rate)	0.1
n_{train} (Initial number of training samples)	64
n_{cal} (Initial number of calibration samples)	16
λ (Shift magnitude)	10.0
d (MFCS estimation depth)	$\{1, 3\}$
T (Number of MFCS steps past IID initialization)	70
γ (Bias magnitude in initial training data IID sampling)	3.0
ACI baseline "step size" (default parameter in Gibbs & Candès (2021))	0.005
Probability each queried point is added to training (versus calibration) data	0.5
Random seeds	$\{0, ..., 349\}$

F. Additional Experiments

F.1. Shift Magnitude Ablation Study

These ablation studies examine the effect of changing the shift magnitude λ, or the "aggressiveness" of the ML agent's query strategy, focusing on the Split CP blue fluorescent protein design case. Other than the different values of λ, the experimental setting and other hyperparameters are the same as in the blue protein results reported in the main paper Figure 2, where $\lambda = 5.0$ (full hyperparameters reported in Appendix E.1, Table 2). Increasing λ tends to exacerbate the discrepancy in coverage between the $\{1, 2, 3, 4\}$-step MFCS CP methods (with higher-order methods maintaining stronger coverage); increase interval widths of the MFCS CP methods; and increase the fitness values of the designed protein sequence.

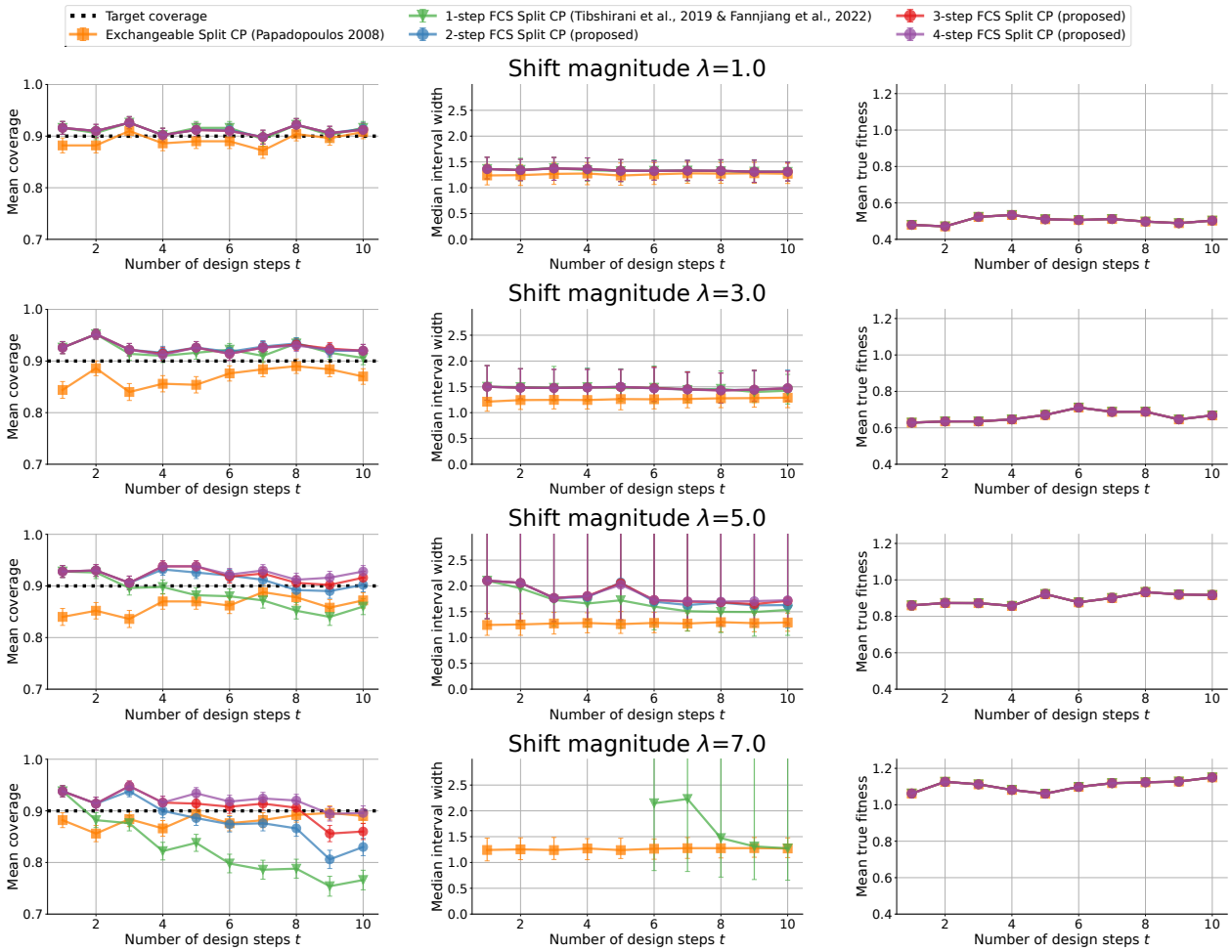

Figure 9. Ablation studies for shift magnitude λ, focused on Split CP experiments on the blue protein design dataset, where each row corresponds to a different shift magnitude $\lambda \in \{1.0, 3.0, 5.0, 7.0\}$. All experiments are conducted with the `MLPRegressor` from `scikit-learn` (with LBFGS solver, logistic activation, and default parameters otherwise) as the ML predictor, initially trained on 32 datapoints and with 32 points in the calibration set. All values are computed over 500 random seeds. Other than λ, full hyperparameters are as in Appendix E.1, Table 2.

F.2. Ablation Study for Initial Amount of Training and Calibration Data

These ablation studies examine the effect of changing the initial number of training points (n_{train}) and the initial number of calibration points (n_{cal}), focusing on the Spit CP red fluorescent protein design case. Other than the different initial sample sizes (where $n_{\text{train}} = n_{\text{cal}}$), the experimental setting and other hyperparameters are the same as in the blue protein results reported in the main paper Figure 2, where $n_{\text{train}} = n_{\text{cal}} = 32$ (full hyperparameters reported in Appendix E.1, Table 2). All values are computed over 500 random seeds.

Increasing the initial training and calibration sample size tends to hurt the coverage of baselines and increase the discrepancy in coverage between the $\{1, 2, 3\}$-step CP methods in these experiments. This may be because more training data improves the performance of the ML model, which can enable it to propose more extreme design sequences. This explanation is plausible because the increase in mean protein fitness values with larger initial training sets is an indication of those design procedures being more aggressive. Interval widths of the MFCS Split CP methods tend to increase with larger initial sample sizes, potentially for the same reason, to maintain coverage when the shift is more extreme.

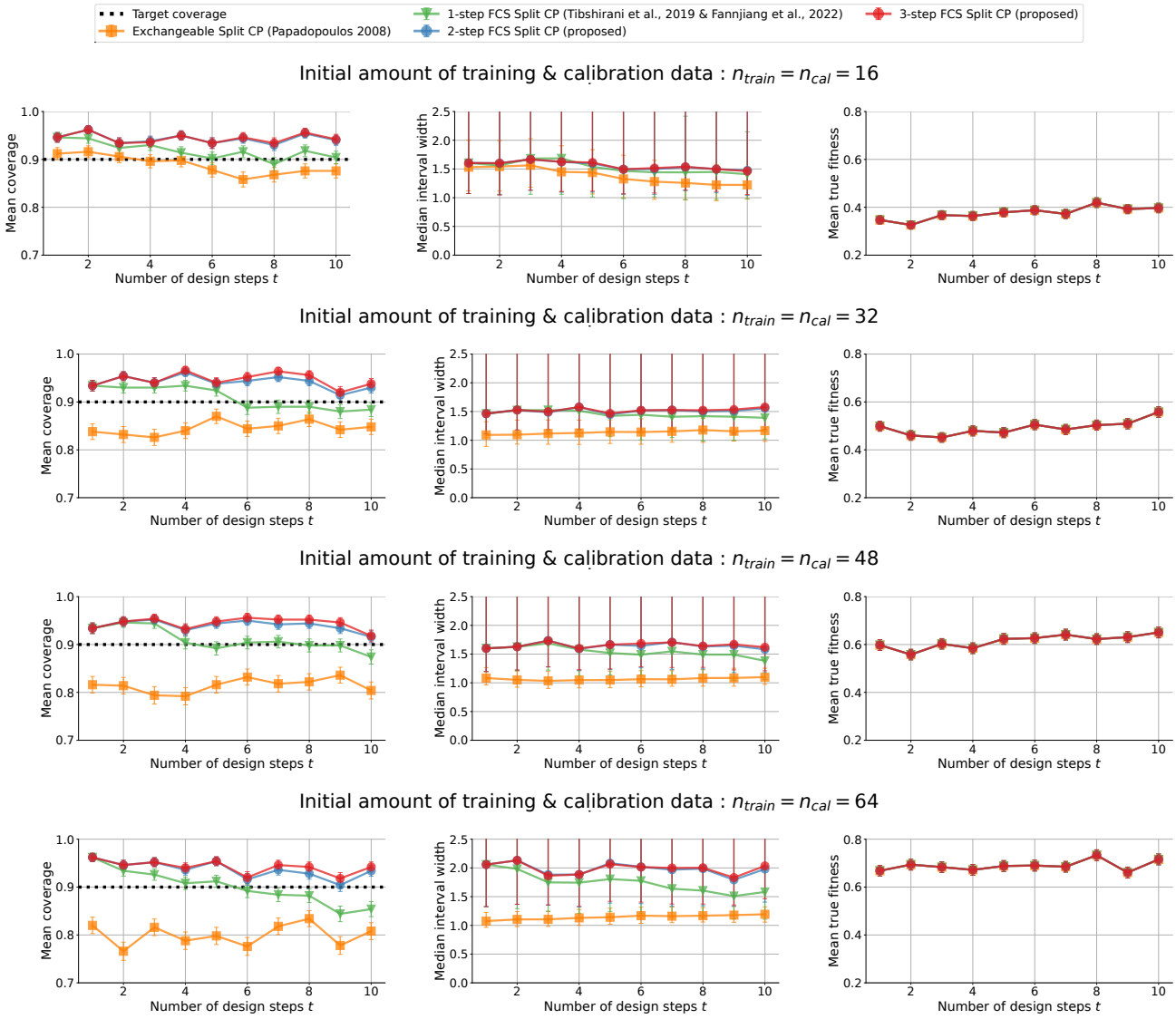

Figure 10. Ablation studies for the initial amount of training and calibration data ($n_{\text{train}} = n_{\text{cal}}$,) focused on Split CP experiments on the red protein design dataset, where each row corresponds to a different initial sample size $n_{\text{train}} = n_{\text{cal}} \in \{16, 32, 48, 64\}$. The ML predictor is the `MLPRegressor` from `scikit-learn` (with LBFGS solver, logistic activation, and default parameters otherwise). All values are computed over 500 random seeds. Other than n_{train} and n_{cal}, full hyperparameters are as in Appendix E.1, Table 2.

F.3. ACI Step Size Ablation Study

Here we report sensitivity analyses on an ACI baseline's "step size" hyperparameter that controls how strongly the method responds to recent over- or under-coverage (Gibbs & Candès, 2021) (by adjusting its empirical quantile level with the aim of having its long-run empirical coverage be close to the target coverage rate). Larger values of the ACI step size correspond to making larger adjustments to recent mistakes, and thus make the method more adaptive. These results are for the airfoil dataset, with hyperparameters as in Appendix E.2, Table 3, aside from the ACI adaptiveness and the reported number of active learning iterations (here $T = 25$). In these experiments, increasing the adaptiveness (step size) of ACI tends to *hurt* its coverage performance, possibly due to changes to the CP calibration set causing its retroactive updates to be "out-of-date."

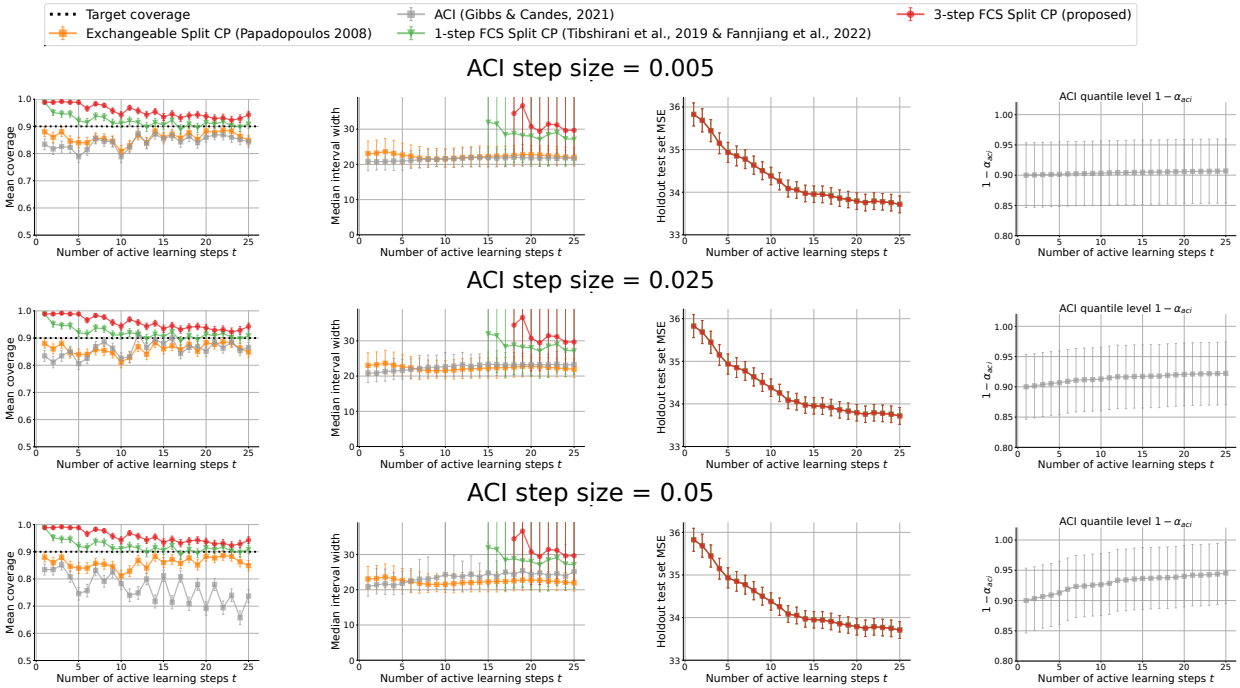

Figure 11. Ablation studies for ACI baseline's step size over values in $\{0.005, 0.25, 0.05\}$. All these experiments are conducted on the airfoil dataset with the `GaussianProcessRegressor` from `scikit-learn`, initially trained on 64 datapoints and with 16 points in the calibration set, and at each active learning step the queried point is added to either training or calibration with equal probability. All values are computed over 350 random seeds.

F.4. Recursion Depth Wall-Clock Runtimes

Table 4. Wall-clock runtimes for computing estimated MFCS weights (Eq. 9) with different estimation or recursion depths d. In these experiments, initially $n_{\text{cal}} = 16$ points are in the calibration set and every queried point is added to the calibration set deterministically. Mean and standard error values are computed over 100 repeated experiments. Increasing the depth by one increases the wall-clock runtime by over an order of magnitude (for each evaluated recursion depth $d \in \{1, ..., 6\}$), which could quickly become intractable for larger d.

MFCS recursion depth d	Wall-clock runtime in seconds ($\pm$ std. err.)
1	1.297 ($\pm$0.063) e$-$6
2	9.595 ($\pm$0.042) e$-$5
3	1.607 ($\pm$0.008) e$-$3
4	3.194 ($\pm$0.036) e$-$2
5	6.628 ($\pm$0.030) e$-$1
6	1.444 ($\pm$0.001) e+1