How Tight Can PAC-Bayes be in the Small Data Regime?

Andrew Y. K. Foong[*]
University of Cambridge
ykf21@cam.ac.uk

Wessel P. Bruinsma[*]
University of Cambridge
Invenia Labs
wpb23@cam.ac.uk

David R. Burt
University of Cambridge
drb62@cam.ac.uk

Richard E. Turner
University of Cambridge
ret26@cam.ac.uk

Abstract

In this paper, we investigate the question: *Given a small number of datapoints, for example $N = 30$, how tight can PAC-Bayes and test set bounds be made?* For such small datasets, test set bounds adversely affect generalisation performance by discarding data. In this setting, PAC-Bayes bounds are especially attractive, due to their ability to use all the data to simultaneously learn a posterior and bound its generalisation risk. We focus on the case of i.i.d. data with a bounded loss and consider the generic PAC-Bayes theorem of Germain et al. (2009) and Bégin et al. (2016). While their theorem is known to recover many existing PAC-Bayes bounds, it is unclear what the tightest bound derivable from their framework is. Surprisingly, we show that for a fixed learning algorithm and dataset, the tightest bound of this form coincides with the tightest bound of the more restrictive family of bounds considered in Catoni (2007). In contrast, in the more natural case of distributions over datasets, we give examples (both analytic and numerical) showing that the family of bounds in Catoni (2007) can be suboptimal. Within the proof framework of Germain et al. (2009) and Bégin et al. (2016), we establish a lower bound on the best bound achievable in expectation, which recovers the Chernoff test set bound in the case when the posterior is equal to the prior. Finally, to illustrate how tight these bounds can potentially be, we study a synthetic one-dimensional classification task in which it is feasible to meta-learn both the prior and the form of the bound to obtain the tightest PAC-Bayes and test set bounds possible. We find that in this simple, controlled scenario, PAC-Bayes bounds are surprisingly competitive with comparable, commonly used Chernoff test set bounds. However, the sharpest test set bounds still lead to better guarantees on the generalisation error than the PAC-Bayes bounds we consider.

1 Introduction

Generalisation bounds are of both practical and theoretical importance. Practically, tight bounds provide certificates that algorithms will perform well on unseen data. Theoretically, the bounds and underlying proof techniques can help explain the phenomenon of learning. Among the tightest known bounds are *PAC-Bayes* (McAllester, 1999) and *test set* bounds (Langford, 2002). In this paper, we investigate their numerical tightness when applied to small datasets ($N \approx 30$–60 datapoints). The comparison between PAC-Bayes and test set bounds is particularly interesting in this setting as one cannot

[*] Equal contribution.

Preprint. Under review.

discard data to compute a test set bound without significantly harming performance. PAC-Bayes on the other hand provides valid bounds while using all of the data for learning. The small data setting can also be quite different from the big data setting, as lower-order terms in PAC-Bayes bounds have a non-negligible contribution, and the detailed structure of the bound becomes important. Fortunately, we do not have to study each PAC-Bayes bound separately: remarkably, Germain et al. (2009) and Bégin et al. (2016) showed that a wide range of bounds can be obtained as special cases of a single *generic PAC-Bayes theorem* that captures the central ideas of many PAC-Bayes proofs. This theorem has a free parameter: it holds for any convex function Δ. By choosing Δ appropriately, one can recover the well-known bounds of Langford and Seeger (2001) and Catoni (2007). We focus on two questions related to this set-up. *First*, *what is the tightest bound achievable by any convex function* Δ? An answer would characterise the limits of the generic PAC-Bayes theorem, and thereby of a wide range of bounds, by telling us how much improvement could be obtained before new ideas or assumptions are needed. *Second*, since test set bounds are the *de facto* standard for larger datasets, but PAC-Bayes has benefits when N is small, we ask: *in the small data regime, can PAC-Bayes be tighter than test set bounds?*

In Sec 3, Thm 4, we show that in the (artificial) case of a *fixed* dataset and learning algorithm, the tightest version of the generic PAC-Bayes theorem is obtained by one of the *Catoni bounds* (Catoni, 2007). In the more realistic case of a stochastic dataset, we do not fully characterise the tightest bound, but in Cor 3 we *lower bound* the tightest bound achievable (in expectation) with any Δ. We also provide numerical evidence in Fig 2 that suggests this lower bound can in some cases be attained, by parameterising a generic convex function with a neural network. Interestingly, this lower bound coincides with removing a lower-order term from the Langford and Seeger (2001) bound, and relaxes to the well-known *Chernoff test set bound* when the PAC-Bayes posterior is equal to the prior.

In Sec 4, we investigate the tightness of PAC-Bayes and test set bounds in synthetic 1D classification. The goal of this experiment is to find out how tight the bounds could be made in principle. We use meta-learning to adapt all aspects of the bounds and algorithms to the task distribution. We find that, in this setting, PAC-Bayes can be competitive with the Chernoff test set bound, but is outperformed by the *binomial tail test set bound*, of which the Chernoff bound is a relaxation. This suggests that, for standard PAC-Bayes to be quantitatively competitive with the best test set bounds on small datasets, a new proof technique leading to bounds that gracefully relax to the binomial tail bound is required.

2 Background and Related Work

We consider supervised learning. Let X and Y denote the *input space* and *output space*, and let $Z = X \times Y$. Assume there is an (unknown) probability measure[2] D over Z, with the dataset $S \sim D^N$. Denote the *hypothesis space* by $\mathcal{H} \subseteq Y^X$. A learning algorithm is then a map $Z^N \to \mathcal{H}$. In PAC-Bayes, we also consider maps $Z^N \to \mathcal{M}_1(\mathcal{H})$, where $\mathcal{M}_1$ is the set of probability measures on its argument. The performance of a hypothesis $h \in \mathcal{H}$ is measured by a *loss function* $\ell \colon Z \times \mathcal{H} \to [0, 1]$. The *(generalisation) risk* of h is $R_D(h) := \mathbb{E}_{(x,y)\sim D}[\ell((x, y), h)]$ and its *empirical risk on* S is $R_S(h) := \frac{1}{N} \sum_{(x,y)\in S} \ell((x, y), h)$. For $Q \in \mathcal{M}_1(\mathcal{H})$ its *(generalisation Gibbs) risk* is $\overline{R}_D(Q) := \mathbb{E}_{h\sim Q}[R_D(h)]$ and its *empirical (Gibbs) risk* is $\overline{R}_S(Q) := \mathbb{E}_{h\sim Q}[R_S(h)]$. In PAC-Bayes, we usually fix a *prior* $P \in \mathcal{M}_1(\mathcal{H})$, chosen without reference to S and learn a *posterior* $Q \in \mathcal{M}_1(\mathcal{H})$ which can depend on S. The *KL-divergence* between Q and P is defined as $\mathrm{KL}(Q\|P) = \int \log \frac{\mathrm{d}Q}{\mathrm{d}P}\, \mathrm{d}Q$ if $Q \ll P$ and ∞ otherwise. Let $\mathcal{C}$ denote the set of proper, convex, lower semicontinuous (l.s.c.) functions $\mathbb{R}^2 \to \mathbb{R} \cup \{+\infty\}$; if a convex function's domain is a subset of $\mathbb{R}^2$, extend it to all of $\mathbb{R}^2$ with the value $+\infty$. See App C for more details on convex analysis, which we use in Sec 3.

Test Set Bounds. *Test set bounds* rely on a subset of data which is not used to select the hypothesis, called a *test set* or *held-out set*. Let $S = S_{\text{train}} \cup S_{\text{test}}$, with $|S| = N$, $|S_{\text{train}}| = N_{\text{train}}$ and $|S_{\text{test}}| = N_{\text{test}}$. In Thms 1 and 2, we assume h is chosen independently of S_{test}. For a zero-one loss, where $\ell \in \{0, 1\}$, $N_{\text{test}} R_{S_{\text{test}}}(h)$ is a binomial random variable with parameters $(N, R_D(h))$. This leads to the following simple bound, which, for $\ell \in \{0, 1\}$, is tight among test set bounds:

Theorem 1 (Binomial tail test set bound, Langford, 2005, Thm 3.3)**.**
Let $\overline{e}(M, k, \delta) := \sup \left\{p : \delta \leq \sum_{i=1}^{k} \binom{M}{i} p^i (1-p)^{M-i}\right\}$. *For any* $h \in \mathcal{H}$, $\ell \in \{0, 1\}$ *and* $\delta \in (0, 1)$,

$$\Pr\Big(R_D(h) \leq \overline{e}(N_{\text{test}}, N_{\text{test}} R_{S_{\text{test}}}(h), \delta)\Big) \geq 1 - \delta. \tag{1}$$

[2] We will colloquially refer to measures on sets without specifying a σ-algebra. We implicitly assume functions are measurable with respect to the σ-algebras on which the relevant measures are defined.

Often, looser bounds with a simpler form are applied. These can be obtained via the Chernoff method:

Theorem 2 (Chernoff test set bound, Langford, 2005, Cor 3.7)**.**
For $q, p \in [0,1]$*, let* $\mathrm{kl}(q,p) \coloneqq q \log \frac{q}{p} + (1-q)\log\frac{1-q}{1-p}$*. For any* $h \in \mathcal{H}$*,* $\ell \in [0,1]$*, and* $\delta \in (0,1)$*,*

$$\Pr\left(\mathrm{kl}(R_{S_{\text{test}}}(h), R_D(h)) \leq \tfrac{1}{N_{\text{test}}} \log \tfrac{1}{\delta}\right) \geq 1-\delta. \tag{2}$$

PAC-Bayes Bounds. The PAC-Bayes approach bounds the generalisation Gibbs risk of *stochastic classifiers*, and does not require discarding data. Since the seminal paper of McAllester (1999), a large variety of PAC-Bayes bounds have been derived. Germain et al. (2009) and Bégin et al. (2016) prove a very general form of the PAC-Bayes theorem which encompasses many of these. Their proof technique consists of a series of inequalities shared by PAC-Bayes proofs (Jensen's, change of measure, Markov's, supremum over risk), and reveals their common structure. Thus understanding the properties of this generic theorem can give insight into many PAC-Bayes bounds at once:

Theorem 3 (Generic PAC-Bayes theorem, Bégin et al. (2016) and Germain et al. (2009))**.**[3]
Fix $P \in \mathcal{M}_1(\mathcal{H})$*,* $\ell \in [0,1]$*,* $\delta \in (0,1)$*, and* Δ *a proper, convex, l.s.c. function* $[0,1]^2 \to \mathbb{R} \cup \{+\infty\}$*. Then*

$$\Pr\left((\forall Q)\ \Delta(\overline{R}_S(Q), \overline{R}_D(Q)) \leq \tfrac{1}{N}\left[\mathrm{KL}(Q\|P) + \log\tfrac{\mathcal{I}_\Delta(N)}{\delta}\right]\right) \geq 1-\delta, \tag{3}$$

where $\mathcal{I}_\Delta(N) \coloneqq \sup_{r\in[0,1]} \sum_{k=0}^{N} \binom{N}{k} r^k (1-r)^{N-k} e^{N\Delta(k/N, r)}$.

Remark 1. *We lose no generality in assuming* $\Delta(q,\cdot)$ *is monotonically increasing for all* $q \in [0,1]$*. Otherwise, we can define a* Δ' *which is monotonic in this way and which leads to at least as good an upper bound on* $\overline{R}_D(Q)$*. See App D.*

For completeness, we provide a proof of Thm 3 in App B. Following Bégin et al. (2016), we briefly recap some of the bounds that can be recovered as special cases (or looser versions) of Thm 3. Setting $\Delta(q,p) = C_\beta(q,p) \coloneqq -\log(1 + p(e^{-\beta} - 1)) - \beta q$ for $\beta > 0$, we recover the *Catoni bounds*:

Corollary 1 (Catoni, 2007, Thm 1.2.6)**.** *For any* $\beta > 0$*,*

$$\Pr\left((\forall Q)\ \overline{R}_D(Q) \leq \tfrac{1}{1-e^{-\beta}}\left[1 - \exp\left(-\beta\overline{R}_S(Q) - \tfrac{1}{N}\left(\mathrm{KL}(Q\|P) + \log\tfrac{1}{\delta}\right)\right)\right]\right) \geq 1-\delta. \tag{4}$$

This specifies a bound for every value of $\beta > 0$. If we instead choose $\Delta(q,p) = \mathrm{kl}(q,p)$, we obtain a slightly tighter version of Maurer's refinement of the Langford-Seeger "kl-bound":

Corollary 2 (Langford and Seeger, 2001, Thm 3, Maurer, 2004, Thm 5)**.** *For* $N \geq 8$*,*

$$\Pr\left((\forall Q)\ \mathrm{kl}(\overline{R}_S(Q), \overline{R}_D(Q)) \leq \tfrac{1}{N}\left[\mathrm{KL}(Q\|P) + \log\tfrac{2\sqrt{N}}{\delta}\right]\right) \geq 1-\delta. \tag{5}$$

We refer to this as the *Maurer–Langford–Seeger* or *MLS bound*. It is very slightly looser than Thm 3 with $\Delta = \mathrm{kl}$, since Maurer (2004) upper bounds $\mathcal{I}_{\mathrm{kl}}(N)$ by $2\sqrt{N}$ using Stirling's formula. The Catoni and MLS bounds are among the tightest PAC-Bayes bounds known and have been applied in settings where numerical tightness is key, such as obtaining generalisation bounds for stochastic neural networks (Dziugaite & Roy, 2017; Zhou et al., 2019). Many other bounds can be obtained by loosening these bounds. Applying Pinsker's inequality $\mathrm{kl}(q,p) \geq 2(q-p)^2$ to Eq (5) yields the "square-root" version of the PAC-Bayes theorem (McAllester, 2003). The "PAC-Bayes-λ" (Thiemann et al., 2017) and "PAC-Bayes-quadratic" bounds (Rivasplata et al., 2019) can be derived as loosened versions of the MLS bound using the inequality $\mathrm{kl}(q,p) \geq (q-p)^2/(2p)$, valid for $q < p$. The "linear" bound in McAllester (2013) can be derived by loosening the Catoni bound using: $C_\beta(q,p) \leq A \implies p \leq \frac{1}{1-\beta/2}(q + \frac{1}{\beta}A)$, which is valid for $\beta \in (0,2)$.

How Tight Are PAC-Bayes Bounds? A fundamental question we can ask about a generalisation bound is how tight it is, and whether it can be tightened. Comparing the MLS and Chernoff test set bounds when $Q = P$ (so the PAC-Bayes bound essentially becomes a test set bound) shows they are identical except for a $\log(2\sqrt{N})/N$ on the RHS of the MLS bound. Whether this term (or similar discrepancies between PAC-Bayes and *Occam bounds* (Langford, 2002, Cor 4.6.2); see App A) can be removed has been an open question since Langford (2002, Problem 6.1.2). Maurer (2004) reduced

[3] We state a simpler version of their result WLOG, absorbing a free parameter into the function Δ.

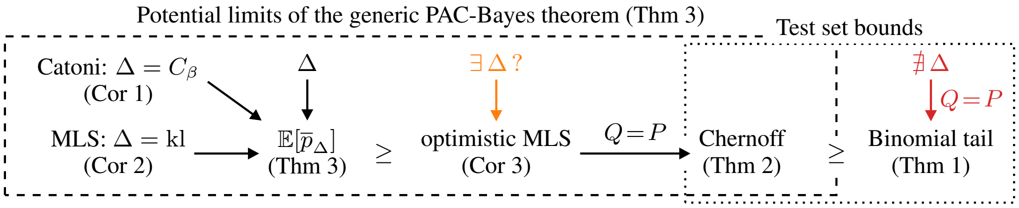

Potential limits of the generic PAC-Bayes theorem (Thm 3)
Test set bounds
Catoni: $\Delta = C_\beta$ (Cor 1)
Δ
$\exists\,\Delta\,?$
MLS: $\Delta = \mathrm{kl}$ (Cor 2)
$\mathbb{E}[\overline{p}_\Delta]$ (Thm 3)
$\geq$
optimistic MLS (Cor 3)
$Q=P$
Chernoff (Thm 2)
$\geq$
$\nexists\,\Delta$
$Q=P$
Binomial tail (Thm 1)

Figure 1: **Illustration of the relationship between various PAC-Bayes and test set bounds**; see Sec 3. It is unclear if there always exists a Δ that recovers the optimistic MLS bound (and hence the Chernoff bound when $Q = P$; Open Prob. 1), but there certainly does not exist a Δ that recovers the Binomial tail bound when $Q = P$.

this term to its current value, improving on work by Langford and Seeger (2001). Interestingly, Germain et al. (2009, Prop 2.1) shows that the expression obtained by dropping $\log(2\sqrt{N})/N$ from the MLS bound is identical to that obtained by minimising the Catoni bound with respect to β, *illegally without a union bound*. The Chernoff test set bound is itself a looser version of the binomial tail bound, raising the question of whether a PAC-Bayes bound can be found that reduces to the binomial tail bound when $Q = P$. We provide new insights into these problems in Sec 3.

Researchers have also compared PAC-Bayes bounds numerically on actual learning problems. Langford (2005) and Germain et al. (2009) were able to obtain reasonable guarantees on small datasets. However, Langford (2005) found that on datasets with $N \approx 145$, PAC-Bayes was outperformed by test set bounds. Dziugaite and Roy (2017) and Langford and Caruana (2002) and Pérez-Ortiz et al. (2020) provide non-vacuous bounds for neural networks using PAC-Bayes. Even so, Dziugaite et al. (2021) states that tighter bounds would be obtained using a test set instead. In Sec 4 we find that if the bounds and learning algorithms are optimised for a task distribution, PAC-Bayes can be tight enough to compete with the Chernoff test set bound, but not the binomial tail test set bound.

3 Characterising the Limits of the Generic PAC-Bayes Proof Technique

This section establishes our main theoretical contributions, which characterise the limits of the generic PAC-Bayes theorem (Thm 3). For a convex $\Delta \in \mathcal{C}$, Thm 3 gives a high-probability upper bound on $\Delta(\overline{R}_S(Q), \overline{R}_D(Q))$. Define $B[f, y] := \sup\{p \in [0,1] : f(p) \leq y\}$ for $f\colon [0,1] \to \mathbb{R}$ and $y \in \mathbb{R}$, where the supremum of the empty set is taken to be 1. This upper bound can be "inverted" to obtain a high-probability upper bound on $\overline{R}_D(Q)$: with probability at least $1 - \delta$, for all $Q \in \mathcal{M}_1(\mathcal{H})$,

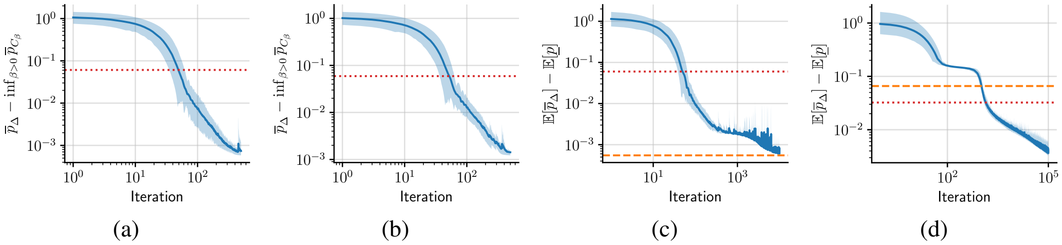
$$\overline{R}_D(Q) \leq \overline{p}_\Delta \quad \text{where} \quad \overline{p}_\Delta := B\big[\Delta(\overline{R}_S(Q), \cdot), \tfrac{1}{N}\big(\mathrm{KL}(Q\|P) + \log\tfrac{\mathcal{I}_\Delta(N)}{\delta}\big)\big]. \tag{6}$$

Since (6) holds for all $\Delta \in \mathcal{C}$, a natural question is: *Which Δ minimises $\overline{p}_\Delta$?* This would characterise how tight, numerically, PAC-Bayes theorems can be made without introducing ideas beyond those needed to prove the bounds stated in Sec 2. Before considering the general case of a stochastic dataset, we first characterise the optimal Δ in the simplified scenario of a fixed dataset S and posterior Q (Thm 4). This is artificial because it is not valid to choose Δ based on S. Next, from Thm 4, we derive a lower bound on the best possible generic PAC-Bayes bound (in expectation) in the more realistic stochastic case (Cor 3). We then connect this lower bound to various existing PAC-Bayes and test set bounds. An overview is shown in Fig 1. We now state our first result.

Theorem 4. *For a fixed dataset S and any $Q, P \in \mathcal{M}_1(\mathcal{H})$, the tightest Catoni bound is as tight as the tightest bound possible within the generic PAC-Bayes theorem (Thm 3). Formally, let $\Delta \in \mathcal{C}$ and $\delta \in (0,1)$. Choose some fixed $\overline{R}_S(Q) =: q \in [0,1]$ and $\mathrm{KL}(Q\|P) =: \mathrm{KL} \in [0,\infty)$. If $q > 0$, then there exists a $\beta \in (0,\infty)$ such that $\overline{p}_\Delta \geq \overline{p}_{C_\beta}$. Moreover, if $q = 0$, then $\overline{p}_\Delta \geq \lim_{\beta\to\infty} \overline{p}_{C_\beta}$.*

Remark 2. *By Thm 4, for all $\Delta \in \mathcal{C}$, we have $\overline{p}_\Delta \geq \inf_{\beta>0} \overline{p}_{C_\beta}$, and, by Prop 2.1 of Germain et al. (2009), $\inf_{\beta>0} \overline{p}_{C_\beta} = B[\mathrm{kl}(q,\,\cdot\,), \frac{1}{N}(\mathrm{KL} + \log\frac{1}{\delta})]$. Hence, for all $\Delta \in \mathcal{C}$, it holds that $\overline{p}_\Delta \geq B[\mathrm{kl}(q,\,\cdot\,), \frac{1}{N}(\mathrm{KL} + \log\frac{1}{\delta})]$, which is also shown directly in the proof of Thm 4 (Eq (17)). Note that optimising β in this way is illegal, and would typically require a union bound to be valid.*

We defer the proof of Thm 4 to the end of this section. We numerically verify Thm 4 by optimising $\overline{p}_\Delta$ with respect to an arbitrary convex Δ for various settings of fixed q and KL. To parametrise a general convex Δ, we use a one-hidden-layer neural network with positive weights at the output



(a) (b) (c) (d)

Figure 2: **The tightest Catoni bound is the optimal generic PAC-Bayes bound for a fixed dataset, but not the optimal expected bound for a random dataset.** We optimise a convex function with H hidden units to minimise $\overline{p}_\Delta$ with $\delta = 0.1, N = 30$. **(a) and (b) consider fixed** q **and** KL (precise values below) and show the difference with the best Catoni bound (Thm 4); **(c) and (d) consider random** q **and** KL and show the *expected* difference with the optimistic MLS bound (Cor 3). Shaded regions show the minimum and maximum over ten initialisations. All plots show the MLS bound (dotted red) and (c) and (d) show the optimal Catoni bound with parameter β^* (dashed orange). All runs quickly converged to non-vacuous values. (a): $(q, \mathrm{KL}) = (2\%, 1)$, $\beta^* \approx 2.24$, $H = 256$. (b): $(q, \mathrm{KL}) = (5\%, 2)$, $\beta^* \approx 1.84$, $H = 256$. (c): $(q, \mathrm{KL}) \in \{(2\%, 1), (5\%, 2)\}$ uniformly, $\beta^* \approx 1.99$, $H = 512$. (d): $(q, \mathrm{KL}) \in \{(30\%, 1), (40\%, 50)\}$ uniformly, $\beta^* \approx 2.32$, $H = 1024$.

layer and softplus nonlinearities. The inversion performed by B is approximated numerically by discretising the inputs to Δ and detecting an upcrossing. Gradients are then approximated using the inverse function theorem: $\frac{\mathrm{d}}{\mathrm{d}\theta} B[f_\theta, c(\theta)] = (\partial_\theta c(\theta) - \partial_\theta f_\theta(x))/\partial_x f_\theta(x)$. See App F for details. Figs 2a and 2b show the difference between the numerically optimised Δ and the best Catoni bound for two settings of fixed q and KL. In both cases, $\overline{p}_\Delta - \inf_{\beta>0} \overline{p}_{C_\beta}$ appears to converge to zero from above, as expected from Thm 4. Interestingly, App F shows that the learned Δ can deviate substantially from C_β, suggesting there are choices for Δ besides Catoni's which achieve $\inf_{\Delta \in \mathcal{C}} \overline{p}_\Delta$.

For a fixed dataset, Thm 4 states that the tightest bound is one of the Catoni bounds; precisely: $\inf_{\Delta \in \mathcal{C}} \overline{p}_\Delta = \inf_{\beta>0} \overline{p}_{C_\beta}$. The more interesting question is whether, when S is sampled randomly, one of the Catoni bounds can still achieve the tightest bound (in expectation). The answer is no: Fig 2d gives a numerical counterexample where $\inf_{\Delta \in \mathcal{C}} \mathbb{E}[\overline{p}_\Delta] < \mathbb{E}[\overline{p}_{\mathrm{kl}}] < \inf_{\beta>0} \mathbb{E}[\overline{p}_{C_\beta}]$. Since the Catoni family of bounds cannot generally achieve the tightest bound in expectation, which Δ do? And how tight is $\inf_{\Delta \in \mathcal{C}} \mathbb{E}[\overline{p}_\Delta]$? Whilst we do not have a full answer, we establish a simple lower bound on $\inf_{\Delta \in \mathcal{C}} \mathbb{E}[\overline{p}_\Delta]$. Define the *optimistic MLS bound* $\underline{p}$ as the quantity from Rem 2, which equals the MLS bound without the $\frac{1}{N} \log \mathcal{I}_{\mathrm{kl}}(N)$ term:

$$\underline{p} \coloneqq B[\mathrm{kl}(q, \cdot\,), \tfrac{1}{N}(\mathrm{KL}(Q\|P) + \log \tfrac{1}{\delta})]. \tag{7}$$

The optimistic MLS bound is *not* proven to be a valid generalisation bound. Rem 2 then yields:

Corollary 3. *Consider the setting from Thm 3. Then the expected optimistic MLS bound* $\mathbb{E}[\underline{p}]$ *gives a lower bound on all expected generalisation bounds obtained through the generic PAC-Bayes theorem (Thm 3). That is, for any distribution over datasets, any prior, and any learning algorithm,*

$$\inf_{\Delta \in \mathcal{C}} \mathbb{E}[\overline{p}_\Delta] \geq \mathbb{E}[\underline{p}] \tag{8}$$

Moreover, there exists a distribution over datasets, a prior, and a posterior such that equality holds. For example, let (x, y) *be constant almost surely, which reduces to the setting of Thm 4.*

Fig 1 shows how Cor 3 fits into the picture so far. The optimistic MLS bound is at least as tight as the bound achieved by any Δ, but Cor 3 does not establish the existence of a Δ which achieves it. Cor 3 has practical utility: for any choice of Δ and corresponding generalisation bound, the optimistic MLS bound can quantify potential slack in the bound due to a suboptimal choice of Δ. App H considers an example of this application of Cor 3 in the simplified scenario where $\overline{R}_S(Q) = \frac{1}{2}$ almost surely. The optimistic MLS bound recovers the Chernoff test set bound (Thm 2) when setting $Q = P$. Since the binomial tail bound (Thm 1) is strictly tighter than the Chernoff bound, this shows there does not exist a Δ such that the generic PAC-Bayes bound (Thm 3) recovers the Binomial tail bound when $Q = P$; this is illustrated in Fig 1. What is unclear, however, is whether there always exists a Δ such that Thm 3 recovers the Chernoff test set bound; or, alternatively, such that the optimistic MLS bound is attained. A positive answer to the latter would establish that the optimistic MLS bound is a valid generalisation bound.[4] As a first piece of evidence, the traces from Figs 2c and 2d suggest that a

[4] By Rem 2, $\overline{p}_\Delta \geq \underline{p}$, so $\mathbb{E}[\overline{p}_\Delta] = \mathbb{E}[\underline{p}]$ implies that $\overline{p}_\Delta = \underline{p}$ a.s., meaning that $\underline{p}$ is a valid gen. bound.

convex function could actually achieve $\mathbb{E}[\underline{p}]$; see App G for more traces. We leave a full resolution of this question as an open problem; see Sec 5. Interestingly, Fig 2c shows that a Catoni bound is sometimes *nearly* optimal even in the stochastic case; we will see another example of this in Fig 3.

We end this section with the proof of Thm 4. Recall that the Catoni family of bounds follows from Thm 3 by considering $\Delta(q,p) = C_\beta(q,p) \coloneqq \mathcal{F}_\beta(p) - \beta q$ with $\mathcal{F}_\beta(p) \coloneqq -\log(p(e^{-\beta}-1)+1)$ and $\beta > 0$. To simplify the notation, we denote $\alpha = \frac{1}{N}(\mathrm{KL} + \log\frac{1}{\delta}) \in (0,\infty)$.

Proof of Thm 4. The proof proceeds in three steps. In the first two steps, we lower bound $\frac{1}{N}\log\mathcal{I}_\Delta(N)$ and upper bound Δ. In the third step, we use these bounds to lower bound $B[\Delta(q,\,\cdot\,),\alpha + \frac{1}{N}\log\mathcal{I}_\Delta(N)]$ and identify the result with a particular Catoni bound.

Lower bound on $\frac{1}{N}\log\mathcal{I}_\Delta(N)$: Since $\Delta \in \mathcal{C}$, it is equal to its own double convex conjugate: $\Delta(q,p) = \Delta^{**}(q,p) = \sup_{c_q,c_p\in\mathbb{R}}(c_q q + c_p p - \Delta^*(c_q,c_p))$, where * denotes convex conjugation. Let $X \sim \mathrm{Bin}(r,N)$. Then

$$\mathcal{I}_\Delta(N) = \sup_{r\in[0,1]}\mathbb{E}[e^{N\Delta(X/N,r)}] = \sup_{r\in[0,1]}\mathbb{E}[e^{\sup_{c_q,c_p\in\mathbb{R}}(c_q X + N c_p r - N\Delta^*(c_q,c_p))}] \tag{9}$$

$$\geq \sup_{r\in[0,1]}\sup_{c_q,c_p\in\mathbb{R}} e^{Nc_p r - N\Delta^*(c_q,c_p)}\mathbb{E}[e^{c_q X}] \tag{10}$$

where $\mathbb{E}[e^{c_q X}] = (r(e^{c_q}-1)+1)^N$ is the moment-generating function of X. Consequently, taking log, dividing by N, and noting that $\frac{1}{N}\log\mathbb{E}[e^{c_q X}] = -\mathcal{F}_{-c_q}(r)$,

$$\tfrac{1}{N}\log\mathcal{I}_\Delta(N) \geq A \quad \text{where} \quad A \coloneqq \sup_{c_q,c_p\in\mathbb{R}}[-\Delta^*(c_q,c_p) + \sup_{r\in[0,1]}(c_p r - \mathcal{F}_{-c_q}(r))]. \tag{11}$$

Upper bound on Δ: We upper bound Δ by making Δ^* as small as possible without exceeding the supremum from (11). Note that A is finite, because Δ^* is proper. Define $\tilde{\Delta}^*$ as follows: $\tilde{\Delta}^*(c_q,c_p) = -A + \sup_{r\in[0,1]}(c_p r - \mathcal{F}_{-c_q}(r))$. Note that $\tilde{\Delta}^*$ is proper, convex as a pointwise supremum of convex functions, and l.s.c. as a supremum of l.s.c. functions. In fact, $\tilde{\Delta}^*$ is finite for all inputs. As the notation suggests, define $\tilde{\Delta} \coloneqq (\tilde{\Delta}^*)^*$. Then $\tilde{\Delta}^*$ is indeed the convex conjugate of $\tilde{\Delta}$, because $\tilde{\Delta}^* \in \mathcal{C}$, so it is equal to its own double convex conjugate. Moreover,

$$\tilde{\Delta}(q,p) = A + \sup_{c_q,c_p\in\mathbb{R}}[c_q q + c_p p - \sup_{r\in[0,1]}(c_p r - \mathcal{F}_{-c_q}(r))] \tag{12}$$

$$= A + \sup_{c_q\in\mathbb{R}}[c_q q + \sup_{c_p\in\mathbb{R}}[c_p p - \mathcal{F}^*_{-c_q}(c_p)]] \tag{13}$$

$$= A + \sup_{c_q\in\mathbb{R}}[c_q q + \mathcal{F}_{-c_q}(p)], \tag{14}$$

by observing that $p \mapsto \mathcal{F}_{-c_q}(p) \in \mathcal{C}$, so it is equal to its own double convex conjugate. Therefore,

$$\tilde{\Delta}(q,p) = A + \sup_{c_q\in\mathbb{R}} C_{-c_q}(q,p) \overset{\text{(i)}}{=} A + \mathrm{kl}(q,p) \tag{15}$$

where (i) follows from a direct computation; see Lem E.1 (App E). *Claim:* For all $q,p \in [0,1]$, $\tilde{\Delta}(q,p) \geq \Delta(q,p)$. This follows from the definitions and finiteness of $\tilde{\Delta}^*$ and A: for all $c_q, c_p \in \mathbb{R}$,

$$-\tilde{\Delta}^*(c_q,c_p) + \sup_{r\in[0,1]}(c_p r - \mathcal{F}_{-c_q}(r)) = A \geq -\Delta^*(c_q,c_p) + \sup_{r\in[0,1]}(c_p r - \mathcal{F}_{-c_q}(r)), \tag{16}$$

which means that $\tilde{\Delta}^* \leq \Delta^*$, so $\tilde{\Delta} \geq \Delta$ by the order-reversing property of the convex conjugate.

Conclusion: Assume that $\overline{p}_\Delta < 1$; otherwise, any $\beta > 0$ works. To begin with, use the previous steps:

$$\overline{p}_\Delta = B[\Delta(q,\,\cdot\,),\alpha + \tfrac{1}{N}\log\mathcal{I}_\Delta(N)] \overset{\text{(11), claim}}{\geq} B[\tilde{\Delta}(q,\,\cdot\,),\alpha + A] \overset{(15)}{=} B[\mathrm{kl}(q,\,\cdot\,),\alpha] = \underline{p}. \tag{17}$$

Since $\alpha > 0$, clearly $\underline{p} > q$, so $0 \leq q < \underline{p} < 1$. Hence, if $q > 0$, then there exists a $\beta > 0$ such that $\mathrm{kl}(q,\underline{p}) = C_\beta(q,\underline{p})$ (Lem E.2; App E). Using that $p \mapsto C_\beta(q,p)$ is continuous and strictly increasing for all $\beta > 0$, we have that $\underline{p} = B[C_\beta(q,\,\cdot\,),\alpha]$, so

$$\overline{p}_\Delta \geq B[C_\beta(q,\,\cdot\,),\alpha] \overset{\text{(i)}}{=} B[C_\beta(q,\,\cdot\,),\alpha + \tfrac{1}{N}\log\mathcal{I}_{C_\beta}(N)] = \overline{p}_{C_\beta}, \tag{18}$$

where (i) uses that $\frac{1}{N}\log\mathcal{I}_{C_\beta}(N) = 0$ (Lem E.3; App E). If $q = 0$, then $\mathrm{kl}(0,\underline{p}) = \lim_{\beta\to\infty} C_\beta(0,\underline{p})$ (Lem E.2; App E), so $\overline{p}_\Delta \geq B[\lim_{\beta\to\infty} C_\beta(0,\cdot),\alpha]$, and conclude like in (18) using Lem E.4 (App E). □

4 Meta-Learning the Tightest Bounds for Synthetic Classification

We now consider, for a particular distribution over tasks, how tight each bound can be made in expectation. Two questions naturally arise: *Which PAC-Bayes bounds are tightest?* and *Can PAC-Bayes bounds be tighter than test set bounds?* Our goal is *not* to compare these bounds when using standard practice, but to see how tight they can be *in principle* if we use every tool in our toolbox to minimise the expected bounds. While these optimisations will be impractical for large models and datasets, they can provide some statistical insight.

Learning Algorithm. Certain learning algorithms may work better with test set bounds, and others with PAC-Bayes bounds. Instead of choosing a fixed algorithm, we *meta-learn* (Schmidhuber, 1987; Thrun & Pratt, 2012) separate algorithms to optimise each bound in expectation: we parametrise a hypothesis space $\mathcal{H}_\theta$ and a *posterior map* $Q_\theta \colon Z^N \to \mathcal{M}_1(\mathcal{H}_\theta)$ by a finite dimensional vector θ, which is trained to optimise the expected bound. This is explained in more detail below. This way, we obtain algorithms that are optimised for each bound. After meta-learning, we can further refine each PAC-Bayes posterior by minimising the PAC-Bayes bound, see App I.4.

Task Distribution. In meta-learning, we refer to a data-generating distribution D and dataset $S \sim D^N$ as a *task*. We consider a *distribution* over tasks, $D \sim \mathcal{T}$, and aim to find the best expected bounds for this distribution achievable by an optimised algorithm.[5] We choose especially simple learning tasks — synthetic 1-dimensional binary classification problems, generated by thresholding Gaussian process (GP) samples — which allows us to fully control the task distribution and easily inspect predictive distributions visually to diagnose learning. App I.1 contains full details.

Priors. The choice of prior is crucial in PAC-Bayes, and the role of *data-dependent priors* (DDPs) (Ambroladze et al., 2007; Parrado-Hernández et al., 2012) has been gaining increased attention. This involves splitting the dataset into $N = N_{\text{prior}} + N_{\text{risk}}$ datapoints. The DDP is allowed to depend on the *prior set* of size N_{prior} (standard priors use $N_{\text{prior}} = 0$), and the empirical risk is computed on the *risk set* of size N_{risk}. Crucially, *the bound is valid when the posterior depends on all N datapoints*. Recently, Dziugaite et al. (2021) showed that DDPs can lead to tighter expected bounds than the optimal non-data-dependent prior, and are sometimes even *required* to obtain non-vacuous bounds. In our experiments we meta-learn a DDP as a map from the prior set to the prior, $P_\theta \colon Z^{N_{\text{prior}}} \to \mathcal{M}_1(\mathcal{H})$ (P_θ is not the same map as Q_θ, but we amalgamate all meta-learned parameters into the single vector θ to ease notation). To compare PAC-Bayes DDPs against test set bounds, we sweep the prior/train set proportion from 0 to 0.8 and see what the tightest value obtained is. Strictly this would require a union bound over the proportions, but here we are primarily interested in comparing the various bounds against each other on an even footing, and vary the proportion for illustrative purposes.

The Meta-Learning Objective. We now discuss meta-learning in more detail. During meta-training, θ is trained to optimise the expected PAC-Bayes generalisation bound over the task distribution:

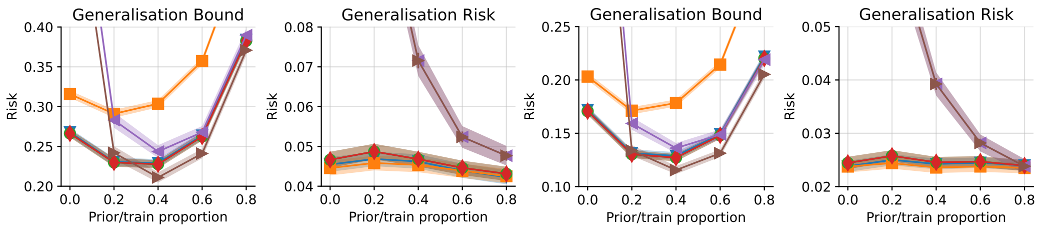
$$\mathbb{E}_{D\sim\mathcal{T}}\mathbb{E}_{S\sim D^N}\, B\big[\Delta_\theta(\overline{R}_{S_{\text{risk}}}(Q_\theta(S)),\,\cdot\,),\, \tfrac{1}{N_{\text{risk}}}\big(\mathrm{KL}(Q_\theta(S)\|P_\theta(S_{\text{prior}})) + \log\tfrac{\mathcal{I}_{\Delta_\theta}(N_{\text{risk}})}{\delta}\big)\big], \quad (19)$$

where the θ in Δ_θ denotes that some bounds (Catoni and learned convex) have meta-learnable parameters. Alternatively, for a meta-learner that minimises a test set bound, the objective is simply $\mathbb{E}_{D\sim\mathcal{T}}\mathbb{E}_{S\sim D^N}\, \overline{R}_{S_{\text{test}}}(Q_\theta(S_{\text{train}}))$, since all test set bounds are monotonic in the test set risk. We use the $0/1$ loss. As the classifiers are stochastic, the empirical risk is still differentiable with respect to θ. In contrast to PAC-Bayes, the predictor that minimises the test set bound can be made deterministic after θ is learned, since it tends to eventually learn essentially deterministic classifiers; see App I.2. We sample $T = 80\,000$ tasks $D_t \sim \mathcal{T}$, with associated datasets $S_t \sim D_t^N$. These form the *meta-trainset*. Additionally, we sample 1024 tasks that form a *meta-testset* used to estimate the average bounds over $\mathcal{T}$ after meta-training. For the PAC-Bayes bounds, we then Monte Carlo estimate (19). Hence, the final objective for a PAC-Bayes meta-learner is (a minibatched version of):

$$\tfrac{1}{T}\textstyle\sum_{t=1}^{T}\, B\big[\Delta_\theta(\overline{R}_{S_{t,\text{risk}}}(Q_\theta(S_t)),\,\cdot\,),\, \tfrac{1}{N_{\text{risk}}}\big(\mathrm{KL}(Q_\theta(S_t)\|P_\theta(S_{t,\text{prior}})) + \log\tfrac{\mathcal{I}_{\Delta_\theta}(N_{\text{risk}})}{\delta}\big)\big]. \quad (20)$$

Similarly, the objective for the test set bound meta-learner is $\frac{1}{T}\sum_{t=1}^{T} \overline{R}_{S_{t,\text{test}}}(Q_\theta(S_{t,\text{train}}))$. The bounds we compute on datasets in the meta-testset, after meta-training is complete and θ is frozen, are valid even though θ was optimised on the meta-trainset. This highlights a contrast between

[5] We could also consider drawing all datasets from a *single* task D, which would more directly match Sec 3. We regard this case as less interesting, since we would often want a bound to perform well on a variety of tasks.



(a) $N = 30$ datapoints.

(b) $N = 60$ datapoints.

Figure 3: **Average generalisation bound and actual generalisation risk** (± two standard errors) for CNN-NP meta-learners trained to optimise Catoni (▼), MLS (■), optimistic MLS (●), learned convex (♦), Chernoff test set (◀), and binomial tail test set (▶) bounds. Catoni, optimistic MLS, and learned convex overlap. The generalisation risks for Chernoff and binomial tail test set bounds are identical as they share the same meta-learner; only the bound computation differs. The bounds are valid with failure probability $\delta = 0.1$ except for optimistic MLS, which should be a lower bound on the best bound achievable with Thm 3. Corresponding plots for the MLP-NP are in App J.2.

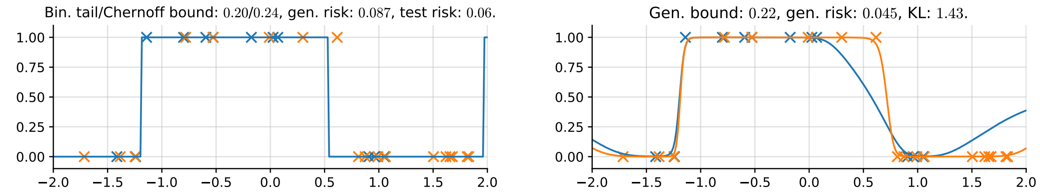
our procedure and the PAC-Bayes meta-learning in Amit and Meir (2018), Liu et al. (2021), and Rothfuss et al. (2020) and Farid and Majumdar (2021). While those works use PAC-Bayes to analyse generalisation of a meta-learner on new tasks, we use PAC-Bayes to analyse generalisation *within* individual tasks.

Parametrising the Meta-Learner and Hypothesis Space. We now describe how to parametrise the hypothesis space $\mathcal{H}_\theta$ and the maps Q_θ, P_θ. We meta-learn a feature map $\phi_\theta \colon \mathbb{R} \to \mathbb{R}^K$ and choose $\mathcal{H}_\theta = \{h_w : h_w(x) = \operatorname{sign}\langle w, \phi_\theta(x)\rangle,\ w \in \mathbb{R}^K\}$. For Q_θ and P_θ Gaussian, this hypothesis space allows us to compute the empirical Gibbs risk without Monte Carlo integration; see App I.3 for details. For the form of Q_θ, we take inspiration from Neural Processes (NPs) (Garnelo, Rosenbaum, et al., 2018; Garnelo, Schwarz, et al., 2018; Kim et al., 2019). NPs use neural networks to flexibly parametrise a map from datasets to predictive distributions that respects the permutation invariance of datasets (Zaheer et al., 2017). They are regularly benchmarked on 1D meta-learning tasks, making them ideally suited. We make a straightforward modification to NPs to make them output Gaussian measures over weight vectors $w \in \mathbb{R}^K$. Hence, they act as parametrisable maps from Z^N to the set of Gaussian measures on $\mathbb{R}^K$.

We considered two kinds of NP, one based on multilayer perceptrons (MLP-NP) and another based on convolutional neural networks (CNN-NP) (detailed in Apps I.5 and I.6) Although the MLP-NP is very flexible, the state-of-the-art in NPs on 1D tasks is given by CNN-based NPs (Bruinsma et al., 2021a; Foong et al., 2020; Gordon et al., 2020). We use an architecture closely based on the *Gaussian Neural Process* (Bruinsma et al., 2021a), which outputs full-covariance Gaussians. As expected, we found the CNN-NP to produce tighter (or comparable) average bounds to the MLP-NP, while using far fewer parameters, and training much more reliably and quickly. Hence, we focus on the CNN-NP, but report some results for the MLP-NP in App J.2. Hyperparameter details are given in App I.7.

Results. We show example classification tasks and average bounds on the meta-test set in Figs 3 and 4. Note that the test set classifier became deterministic and makes hard predictions whereas the PAC-Bayes classifier shows uncertainty; see App I.2 for a discussion. The MLS bound is loosest, which is unsurprising as it has no optimisable parameters to adapt to $\mathcal{T}$.[6] Surprisingly, the results for Catoni, optimistic MLS, and learned convex are nearly identical. As long as optimisation has succeeded reasonably, Cor 3 then implies that one of the Catoni bounds is very nearly optimal among all convex functions for this task distribution — there is not much "slack" from choosing suboptimal Δ here. We also see that the Catoni and learned convex bounds with prior proportion 0.4 are tighter than any Chernoff test set bound. Hence, *PAC-Bayes can provide slightly tighter (or comparable) generalisation bounds to a Chernoff test set bound*. However, we see that *the binomial tail test set bound with train set proportion* 0.4 *leads to the tightest generalisation bounds overall*. Cor 3 sheds light on this behaviour: the optimal generic PAC-Bayes bound reduces, at best, to the Chernoff test set bound when the posterior equals the prior. However, the Chernoff bound is itself looser than the binomial tail bound. Of course, the posterior does not equal the prior here, but Cor 3 indicates there

[6] This is in contrast with usual applications of PAC-Bayes, where one does not have a meta-dataset with which to optimise parameters of the bound. In that setting, it can be an advantage to not have tunable parameters.



(a) **Binomial tail/Chernoff test set** bounds, showing the learned hypothesis (—), the train set (×) of size 12 and the test set (×) of size 18.

(b) **Learned convex** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

Figure 4: **Predictions on one of the 1D datasets in the meta-test set** with $N = 30$ and prior/train proportion 0.4. For each method, we report the generalisation bound and actual generalisation risk. For the test set model, we also show the risk on the test set, and for the PAC-Bayes model we show the KL-divergence. The learned convex bound meta-learner has learned a DDP that provides a "first guess" given the prior set, which is then refined by the posterior. Figures for other PAC-Bayes bounds and datasets are provided in App J.1.

is an extra source of looseness that PAC-Bayes has to overcome relative to the binomial tail bound. Finally, although the test set meta-learner leads to the tightest generalisation bounds, its generalisation risk is roughly double that of the PAC-Bayes meta-learner when the prior/train set proportion is 0.4.

5 Conclusions, Open Problems, and Limitations

PAC-Bayes presents a potentially attractive framework for obtaining tight generalisation bounds in the small-data regime. We have investigated the tightness of PAC-Bayes and test set bounds in this regime both theoretically and experimentally. Theoretically, we showed that the generic PAC-Bayes theorem of Germain et al. (2009) and Bégin et al. (2016) which encompasses a wide range of PAC-Bayes bounds, cannot produce tighter bounds in expectation than the expression obtained by discarding the $\log(2\sqrt{N})/N$ term in the Langford and Seeger (2001) bound (*i.e.*, the *optimistic MLS bound*; Cor 3). Although we did not prove that the optimistic MLS bound is a valid generalisation bound, numerical evidence suggests (Figs 2c and 2d) that there may exist a convex function Δ which achieves it, at least for the distributions over empirical risk and KL-divergence we considered. This suggests the following open problem:

Open Problem 1. *For an arbitrary distribution over datasets, does there exist a choice of Δ such that the expected optimistic MLS bound is attained (Cor 3)? If not, how close can one get to the expected optimistic MLS bound?*

If such a Δ exists, then that would imply the optimistic MLS bound is a valid generalisation bound (see Sec 3) and resolve Problem 6.1.2 of Langford (2002) in the affirmative.

We then considered, in a controlled experimental setting where meta-learning all parameters of the bounds and learning algorithms was feasible, whether PAC-Bayes bounds could be tighter than test set bounds. Although we found PAC-Bayes competitive with Chernoff bounds, both were outperformed by the binomial tail test set bound. This motivates a second open problem:

Open Problem 2. *Can a PAC-Bayes bound be found that relaxes gracefully to the binomial tail test set bound (Thm 1) when the posterior is equal to the prior?*

Resolving these problems could have a significant impact on the tightness of PAC-Bayes applied to small-data, and clarify our understanding of the relationship between PAC-Bayes and test set bounds.

Limitations. In this paper, we concern ourselves with understanding the tightness of bounds in what might be called the *standard PAC-Bayes setting* of supervised learning: bounded losses, i.i.d. data, and Gibbs risk. We also focus on bounds that are first order in the sense that they rely only on the empirical Gibbs risk, though extending the analysis to consider other PAC-Bayes theorems (*e.g.* Tolstikhin and Seldin (2013)) would be of interest, especially with regards to Open Problems 1 and 2. For many practical applications in which performance guarantees are needed (*e.g.* health care), the i.i.d. assumption should be considered carefully, as it is likely an unrealistic simplification. Furthermore, Gibbs classifiers are not often used in practice. To address these and other concerns, PAC-Bayes has been generalised in many directions beyond the scope of the standard setting we consider. Examples include bounds for non-i.i.d. data (Alquier & Guedj, 2018; Rivasplata et al., 2020; Seldin et al., 2012), unbounded losses (Germain et al., 2016), derandomised classifiers (Blanchard & Fleuret, 2007; Viallard et al., 2021), and Bayes risk (Germain et al., 2015; Masegosa et al., 2020). Bounds based on other divergences besides the KL have also been proposed (Alquier & Guedj, 2018; Bégin et al., 2016). As our proof relies primarily on tools from convex analysis, and Jensen's inequality is ubiquitous in PAC-Bayes bounds, it would be interesting to see if our arguments can be extended beyond the limited setting we focus on. Finally, our meta-learning experiments only considered 1D classification, and the results might not necessarily be representative of more realistic datasets. Scaling these up is an important, but potentially challenging, avenue for future work.

Acknowledgements

We would like to thank Pierre Alquier and Yann Dubois for insightful discussions, John Langford for clarifying a remark on test set bounds, and David Janz, Will Tebbutt, and Austin Tripp for providing helpful comments on a draft version of this paper. Andrew Y. K. Foong gratefully acknowledges funding from a Trinity Hall Research Studentship and the George and Lilian Schiff Foundation. David R. Burt acknowledges funding from the Qualcomm Innovation Fellowship and the Williams College Herchel Smith Fellowship. Richard E. Turner is supported by Google, Amazon, ARM, Improbable, Microsoft, EPSRC grant EP/T005637/1, and the UKRI Centre for Doctoral Training in the Application of Artificial Intelligence to the study of Environmental Risks (AI4ER).

References

Alquier, P., & Guedj, B. (2018). Simpler PAC-Bayesian bounds for hostile data. *Machine Learning*, *107*(5), 887–902 (cit. on p. 10).

Ambroladze, A., Parrado-Hernández, E., & Shawe-Taylor, J. (2007). Tighter PAC-Bayes bounds. *Advances in neural information processing systems* (cit. on p. 7).

Amit, R., & Meir, R. (2018). Meta-learning by adjusting priors based on extended PAC-Bayes theory. *International Conference on Machine Learning* (cit. on p. 8).

Bégin, L., Germain, P., Laviolette, F., & Roy, J.-F. (2016). PAC-Bayesian bounds based on the Rényi divergence. *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS* (cit. on pp. 1–3, 9, 10, 13).

Blanchard, G., & Fleuret, F. (2007). Occam's hammer. *International Conference on Computational Learning Theory*, 112–126 (cit. on p. 10).

Boucheron, S., Lugosi, G., & Massart, P. (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford University Press. (Cit. on p. 13).

Bruinsma, W. P., Requeima, J., Foong, A. Y., Gordon, J., & Turner, R. E. (2021a). The Gaussian neural process. *arXiv preprint arXiv:2101.03606* (cit. on pp. 8, 24).

Bruinsma, W. P., Requeima, J., Foong, A. Y., Gordon, J., & Turner, R. E. (2021b). *The Gaussian neural process — contributed talk*. `https://wesselb.github.io/assets/talks/Bruinsma,%20The%20Gaussian%20Neural%20Process%20(Handout).pdf`. (Cit. on p. 24)

Catoni, O. (2007). PAC-Bayesian supervised classification: The thermodynamics of statistical learning. Institute of Mathematical Statistics, Beachwood Ohio, USA. (Cit. on pp. 1–3, 16).

Cremer, C., Li, X., & Duvenaud, D. (2018). Inference suboptimality in variational autoencoders. *International Conference on Machine Learning*, 1078–1086 (cit. on p. 23).

Dziugaite, G. K., Hsu, K., Gharbieh, W., Arpino, G., & Roy, D. (2021). On the role of data in PAC-Bayes. *International Conference on Artificial Intelligence and Statistics*, 604–612 (cit. on pp. 4, 7).

Dziugaite, G. K., & Roy, D. M. (2017). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *Uncertainty in Artificial Intelligence, UAI* (cit. on pp. 3, 4, 18).

Farid, A., & Majumdar, A. (2021). PAC-bus: Meta-learning bounds via PAC-Bayes and uniform stability. *arXiv preprint arXiv:2102.06589* (cit. on p. 8).

Foong, A. Y. K., Bruinsma, W. P., Gordon, J., Dubois, Y., Requeima, J., & Turner, R. E. (2020). Meta-learning stationary stochastic process prediction with convolutional neural processes. *arXiv preprint arXiv:2007.01332* (cit. on p. 8).

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., & Eslami, S. A. (2018). Conditional neural processes. *International Conference on Machine Learning*, 1704–1713 (cit. on pp. 8, 23).

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., & Teh, Y. W. (2018). Neural processes. *arXiv preprint arXiv:1807.01622* (cit. on p. 8).

Germain, P., Bach, F., Lacoste, A., & Lacoste-Julien, S. (2016). PAC-Bayesian theory meets Bayesian inference. *Advances in Neural Information Processing Systems* (cit. on p. 10).

Germain, P., Lacasse, A., Laviolette, F., March, M., & Roy, J.-F. (2015). Risk bounds for the majority vote: From a PAC-Bayesian analysis to a learning algorithm. *Journal of Machine Learning Research*, *16*(26), 787–860 (cit. on p. 10).

Germain, P., Lacasse, A., Laviolette, F., & Marchand, M. (2009). PAC-Bayesian learning of linear classifiers. *Proceedings of the 26th Annual International Conference on Machine Learning*, 353–360 (cit. on pp. 1–4, 9, 16).

Gordon, J., Bruinsma, W. P., Foong, A. Y., Requeima, J., Dubois, Y., & Turner, R. E. (2020). Convolutional conditional neural processes. *International Conference on Learning Representations (ICLR), 8th* (cit. on pp. 8, 24, 25).

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., & Teh, Y. W. (2019). Attentive neural processes. *arXiv preprint arXiv:1901.05761* (cit. on p. 8).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations, ICLR* (cit. on pp. 23, 25).

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. *International Conference on Learning Representations, ICLR* (cit. on p. 23).

Langford, J. (2002). *Quantitatively tight sample complexity bounds* (Doctoral dissertation). Carnegie Mellon University. (Cit. on pp. 1, 3, 9, 13).

Langford, J. (2005). Tutorial on practical prediction theory for classification. *Journal of machine learning research*, *6*(3) (cit. on pp. 2–4, 13).

Langford, J., & Caruana, R. (2002). (Not) bounding the true error. *Advances in neural information processing systems*. (Cit. on p. 4).

Langford, J., & Seeger, M. (2001). *Bounds for averaging classifiers* (tech. rep.). Carnegie Mellon University. (Cit. on pp. 2–4, 9).

Liu, T., Lu, J., Yan, Z., & Zhang, G. (2021). PAC-Bayes bounds for meta-learning with data-dependent prior. *arXiv preprint arXiv:2102.03748* (cit. on p. 8).

Masegosa, A., Lorenzen, S., Igel, C., & Seldin, Y. (2020). Second order PAC-Bayesian bounds for the weighted majority vote. *Advances in Neural Information Processing Systems* (cit. on p. 10).

Maurer, A. (2004). A note on the PAC Bayesian theorem. *arXiv preprint arXiv:cs/0411099* (cit. on pp. 3, 13).

McAllester, D. (2013). A PAC-Bayesian tutorial with a dropout bound. *arXiv preprint arXiv:1307.2118* (cit. on p. 3).

McAllester, D. A. (1999). PAC-Bayesian model averaging. *Proceedings of the twelfth annual conference on Computational learning theory*, 164–170 (cit. on pp. 1, 3).

McAllester, D. A. (2003). PAC-Bayesian stochastic model selection. *Machine Learning*, *51*(1), 5–21 (cit. on p. 3).

Parrado-Hernández, E., Ambroladze, A., Shawe-Taylor, J., & Sun, S. (2012). PAC-Bayes bounds with data dependent priors. *The Journal of Machine Learning Research*, *13*(1), 3507–3531 (cit. on p. 7).

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in PyTorch (cit. on p. 22).

Pérez-Ortiz, M., Rivasplata, O., Shawe-Taylor, J., & Szepesvári, C. (2020). Tighter risk certificates for neural networks. *arXiv preprint arXiv:2007.12911* (cit. on p. 4).

Rivasplata, O., Kuzborskij, I., Szepesvári, C., & Shawe-Taylor, J. (2020). PAC-Bayes analysis beyond the usual bounds. *arXiv preprint arXiv:2006.13057* (cit. on p. 10).

Rivasplata, O., Tankasali, V. M., & Szepesvari, C. (2019). PAC-Bayes with backprop. *arXiv preprint arXiv:1908.07380* (cit. on p. 3).

Rockafellar, R. T., & Wets, R. J. (2010). *Variational analysis*. Springer. (Cit. on pp. 14, 15).

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, 234–241 (cit. on p. 25).

Rothfuss, J., Fortuin, V., Josifoski, M., & Krause, A. (2020). PACOH: Bayes-optimal meta-learning with PAC-guarantees. *arXiv preprint arXiv:2002.05551* (cit. on p. 8).

Schmidhuber, J. (1987). Evolutionary principles in self-referential learning. On learning how to learn: The meta-meta-meta...-hook [diploma thesis]. (Cit. on p. 7).

Seldin, Y., Laviolette, F., Cesa-Bianchi, N., Shawe-Taylor, J., & Auer, P. (2012). PAC-Bayesian inequalities for martingales. *IEEE Transactions on Information Theory*, *58*(12), 7086–7093 (cit. on p. 10).

Thiemann, N., Igel, C., Wintenberger, O., & Seldin, Y. (2017). A strongly quasiconvex PAC-Bayesian bound. *International Conference on Algorithmic Learning Theory*, 466–492 (cit. on p. 3).

Thrun, S., & Pratt, L. (2012). *Learning to learn*. Springer Science & Business Media. (Cit. on p. 7).

Tolstikhin, I. O., & Seldin, Y. (2013). PAC-Bayes-empirical-Bernstein inequality. *Advances in Neural Information Processing Systems* (cit. on p. 9).

Viallard, P., Germain, P., Habrard, A., & Morvant, E. (2021). A general framework for the derandomization of PAC-Bayesian bounds. *arXiv preprint arXiv:2102.08649* (cit. on p. 10).

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., & Smola, A. (2017). Deep sets (cit. on p. 8).

Zhou, W., Veitch, V., Austern, M., Adams, R. P., & Orbanz, P. (2019). Non-vacuous generalization bounds at the imagenet scale: A PAC-Bayesian compression approach (cit. on p. 3).

A Relationship Between PAC-Bayes and Occam Bound

The well-known *Occam bounds* can be derived by a simple application of the union bound to a countable hypothesis class $\mathcal{H}$. In particular, we can consider a "prior" distribution P, which functions similarly to the PAC-Bayes prior. Then by applying the union bound and weighting each hypothesis h with a failure probability of $P(\{h\})\delta$, we can convert any test set bound into a corresponding train set bound. Applying this to Thm 1, we obtain:

Theorem A.1 (Binomial tail Occam bound, Langford (2002), Thm 4.6.1)**.** *Let $\mathcal{H}$ be countable, and fix $P \in \mathcal{M}_1(\mathcal{H})$, $\ell \in \{0,1\}$ and $\delta \in (0,1)$. Then*

$$\Pr\Big((\forall h)\ R_D(h) \le \overline{e}\big(N, NR_S(h), P(\{h\})\delta\big)\Big) \ge 1-\delta, \tag{21}$$

where $\overline{e}$ is defined in Thm 1.

Alternatively, applying this procedure to Thm 2 yields a looser bound:

Theorem A.2 (Chernoff Occam bound, Langford (2002), Cor 4.6.2)**.** *Let $\mathcal{H}$ be countable, and fix $P \in \mathcal{M}_1(\mathcal{H})$, $\ell \in [0,1]$ and $\delta \in (0,1)$. Then*

$$\Pr\left((\forall h)\ \mathrm{kl}(R_S(h), R_D(h)) \le \frac{1}{N}\left[\log\frac{1}{P(\{h\})} + \log\frac{1}{\delta}\right]\right) \ge 1-\delta. \tag{22}$$

Following Langford (2005, Sec 5.1), it is instructive to compare the Chernoff Occam bound with the MLS bound (Cor 2) when $\mathcal{H}$ is countable and Q is constrained to be a point mass, *i.e.* $Q = Q_h := \delta_h$, where δ_h denotes the Dirac measure at h. In that case, $\mathrm{KL}(Q_h \| P)$ reduces to $\log(1/P(\{h\}))$, and the Gibbs risks $\overline{R}_S(Q_h), \overline{R}_D(Q_h)$ simply reduce to the risks $R_S(h), R_D(h)$. Then the MLS bound states that:

$$\Pr\left((\forall h)\ \mathrm{kl}(R_S(h), R_D(h)) \le \frac{1}{N}\left[\log\frac{1}{P(\{h\})} + \log\frac{2\sqrt{N}}{\delta}\right]\right) \ge 1-\delta. \tag{23}$$

Comparing Eq (23) with Thm A.2, we see that the MLS bound leads to a bound on $\mathrm{kl}(R_S(h), R_D(h))$ which is looser by an additive constant of $\log(2\sqrt{N})/N$ compared to the Chernoff Occam bound. Hence the PAC-Bayes bound does not relax gracefully to the Occam bound in this case, which motivates Open Problem 6.1.2 in Langford (2002). In fact, by Rem 2 and Cor 3, we know that *if* we could find a convex Δ that allowed us to remove this $\log(2\sqrt{N})/N$ term (*i.e.*, the optimistic MLS bound), this would be the tightest possible bound obtainable from the generic PAC-Bayes theorem (Thm 3). This motivates Open Problem 1. Finally, noting that Thm A.2 is itself a looser version of Thm A.1, we see that a PAC-Bayes bound that relaxes gracefully to Thm A.1 is not obtainable from Thm 3, motivating Open Problem 2.

B Proof of Generic PAC-Bayes Theorem (Thm 3)

We provide a proof of Thm 3 here for convenience, which closely follows the proof given in Bégin et al. (2016). We first require a well-known lemma:

Lemma B.1 (Kullback-Leibler change of measure inequality, Boucheron et al., 2013, Cor 4.15)**.** *For any set $\mathcal{H}$, probability measures $P, Q \in \mathcal{M}_1(\mathcal{H})$, and measurable function $\phi : \mathcal{H} \to \mathbb{R}$,*

$$\mathbb{E}_{h\sim Q}\phi(h) \le \mathrm{KL}(Q\|P) + \log\left(\mathbb{E}_{h\sim P} e^{\phi(h)}\right). \tag{24}$$

In order to deal with general bounded losses $\ell \in [0,1]$, we also use a lemma proven in Maurer (2004):

Lemma B.2 (Maurer (2004), Lem 3)**.** *For any $[0,1]$-valued random variable z, let z' denote the unique Bernoulli random variable with $\mathbb{E}[z'] = \mathbb{E}[z]$. Let $S = (z_1, \ldots, z_N)$ and $S' = (z'_1, \ldots, z'_N)$ denote tuples of N such random variables. Then for any convex function $f : [0,1]^N \to \mathbb{R}$,*

$$\mathbb{E}[f(S)] \le \mathbb{E}[f(S')]. \tag{25}$$

We can now prove Thm 3:

Proof of Thm 3. Applying Jensen's inequality followed by Lem B.1, we have, for all $Q \in \mathcal{M}_1(\mathcal{H})$:

$$\begin{aligned} N\Delta(\overline{R}_S(Q), \overline{R}_D(Q)) &= N\Delta(\mathbb{E}_{h\sim Q} R_S(h), \mathbb{E}_{h\sim Q} R_D(h)) \qquad (26) \\ &\leq \mathbb{E}_{h\sim Q} N\Delta(R_S(h), R_D(h)) \qquad (27) \\ &\leq \mathrm{KL}(Q\|P) + \log\left(\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))}\right). \qquad (28) \end{aligned}$$

Applying Markov's inequality to the random variable $\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))}$ (which is random through $S \sim D^N$), we obtain, for any $\delta \in (0,1)$:

$$\Pr\left(\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))} \leq \frac{\mathbb{E}_{S\sim D^N}\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))}}{\delta}\right) \geq 1-\delta. \qquad (29)$$

Combining this with Eq (28) yields, with probability at least $1-\delta$, for all Q simultaneously:

$$\Delta(\overline{R}_S(Q), \overline{R}_D(Q)) \leq \frac{1}{N}\left[\mathrm{KL}(Q\|P) + \log\frac{\mathbb{E}_{S\sim D^N}\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))}}{\delta}\right]. \qquad (30)$$

Finally, we upper bound $\mathbb{E}_{S\sim D^N}\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))}$ by a quantity than can be computed without knowing the true distribution D. By Tonelli's theorem:

$$\mathbb{E}_{S\sim D^N}\mathbb{E}_{h\sim P} e^{N\Delta(R_S(h),R_D(h))} = \mathbb{E}_{h\sim P}\mathbb{E}_{S\sim D^N} e^{N\Delta(R_S(h),R_D(h))}. \qquad (31)$$

We will now upper bound the inner expectation by a quantity that is independent of h. Denote the datapoints in S as $S = ((x_1,y_1),\dots,(x_N,y_N))$. Recall that $R_S(h) := \frac{1}{N}\sum_{n=1}^N \ell((x_n,y_n),h)$, and note that $\ell((x_n,y_n),h)$ is a $[0,1]$-valued random variable. Let L denote the N-tuple of these random variables for each datapoint in S, *i.e.* $L = (\ell((x_1,y_1),h),\dots,\ell((x_N,y_N),h))$, and let $M(L) := \frac{1}{N}\sum_{n=1}^N L_n$ be the arithmetic mean of L. Then the function $f : L \mapsto e^{N\Delta(M(L),R_D(h))}$ is convex since $M(L)$ is linear in L, Δ is convex and the exponential function is convex and nondecreasing. Hence defining L' as the N-tuple of Bernoulli random variables such that $\mathbb{E}[L'_n] = \mathbb{E}[L_n] = R_D(h)$ for $1 \leq n \leq N$ and applying Lem B.2,

$$\begin{aligned} \mathbb{E}_{S\sim D^N} e^{N\Delta(R_S(h),R_D(h))} &= \mathbb{E}\, e^{N\Delta(M(L),R_D(h))} \qquad (32) \\ &= \mathbb{E}\, f(L) \qquad (33) \\ &\leq \mathbb{E}\, f(L') \qquad (34) \\ &= \textstyle\sum_{k=0}^N \Pr\big(M(L') = k/N\big) e^{N\Delta(k/N,R_D(h))} \qquad (35) \\ &= \textstyle\sum_{k=0}^N \binom{N}{k} R_D(h)^k (1-R_D(h))^{N-k} e^{N\Delta(k/N,R_D(h))} \qquad (36) \\ &\leq \textstyle\sup_{r\in[0,1]} \sum_{k=0}^N \binom{N}{k} r^k (1-r)^{N-k} e^{N\Delta(k/N,r)} \qquad (37) \\ &= \mathcal{I}_\Delta(N). \qquad (38) \end{aligned}$$

Substituting this into Eq (31) and then Eq (30) completes the proof. □

C Basic Facts from Convex Analysis

A function $f\colon \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ is called *proper* if it not everywhere ∞ and nowhere $-\infty$. The convex conjugate $f^*\colon \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ of f is defined as $f^*(c) = \sup_{x\in\mathbb{R}^n}(\langle c,x\rangle - f(x))$. If f is proper, convex, and l.s.c., then f^* is also proper, convex, and l.s.c. Moreover, if f is proper, convex, and l.s.c., then f is equal to its double convex conjugate: $f(x) = \sup_{c\in\mathbb{R}^n}(\langle c,x\rangle - f^*(c))$. A pointwise supremum of convex functions is again convex; and a pointwise supremum of l.s.c. functions remains l.s.c. Convex functions $f\colon A \to \mathbb{R}$ defined on only a convex subset $A \subseteq \mathbb{R}^n$ are extended to the whole of $\mathbb{R}^n$ by setting $f|_{\mathbb{R}^n\setminus A} = +\infty$. See, for example, Rockafellar and Wets (2010) for proofs of these results and an introduction to the topic.

D Monotonicity of Δ

Proposition D.1. *For* $\Delta\colon [0,1]^2 \to \mathbb{R}\cup\{+\infty\}$ *a proper, convex, and lower semi-continuous function,* $q \in [0,1]$, $\delta \in (0,1]$, *and* $\mathrm{KL} \geq 0$, *define*

$$\overline{p}_\Delta = \sup\left\{p \in [0,1] : \Delta(q,p) \leq \frac{1}{N}\left(\mathrm{KL} + \log\frac{\mathcal{I}_\Delta(N)}{\delta}\right)\right\} \qquad (39)$$

Then there exists a proper, convex, lower semi-continuous $\Delta' \colon [0,1]^2 \to \mathbb{R} \cup \{+\infty\}$ *such that* $\overline{p}_{\Delta'} \le \overline{p}_{\Delta}$*, and for every* $q \in [0,1]$*,* $\Delta'(q,\cdot)$ *is monotonically increasing.*

Proof. Define $\Delta'(q,p) = \inf_{p' \ge p} \Delta(q,p')$. We will prove that Δ' has the desired properties. First, Δ' is not infinity everywhere, as $\Delta'(q,p) \le \Delta(q,p)$ and Δ is proper. Second, since Δ is l.s.c. and proper, it obtains a minimum on the compact set $[0,1]^2$, hence Δ' does not take the value $-\infty$. Therefore, Δ' is proper.

Since Δ is l.s.c., the strict sublevel sets of Δ are open; that is, for all $y \in \mathbb{R}$, $\{(q,p) : \Delta(q,p) < y\}$ is open. Then,

$$\{(q,p) : \Delta'(q,p) < y\} = \{(q,p) : \inf_{p' \ge p} \Delta(q,p') < y\} \tag{40}$$

$$= \bigcup_{p' \in [p,1]} \{(q,p') : \Delta(q,p') < y\}. \tag{41}$$

The equality follows from noting that the infimum on the closed set $[p,1]$ must be achieved as Δ is l.s.c.[7] As we have written $\{(q,p) : \Delta'(q,p) < y\}$ as a union of open sets, it is open. Hence, the sublevel sets of Δ' are open, implying Δ' is l.s.c.

We next show that Δ' is convex. Define the function $D : \mathbb{R}^3 \to \mathbb{R} \cup \{+\infty\}$ by,

$$D(q,p',p) = \begin{cases} \Delta(q,p) & p' \ge p \\ +\infty & \text{otherwise.} \end{cases}$$

$D(q,p,p')$ is convex since Δ is convex and $p' \ge p$ is a convex set. Also, $\inf_{p \in \mathbb{R}} D(q,p',p) = \inf_{p' \ge p} \Delta(q,p') = \Delta'(q,p)$. As the infimum projection of a convex function is convex (Rockafellar & Wets, 2010, Proposition 2.22), Δ' is convex. Also, $\Delta'(q,p)$ is monotonically increasing in p as the infimum is taken over a smaller set for larger p.

It remains to show that $\overline{p}_{\Delta'} \le \overline{p}_{\Delta}$. For all pairs (q,p), $\Delta'(q,p) = \inf_{p' \ge p} \Delta(q,p') \le \Delta(q,p)$. From this it follows that

$$\frac{1}{N}\left(\mathrm{KL} + \log \frac{\mathcal{I}_{\Delta'}(N)}{\delta}\right) \le \frac{1}{N}\left(\mathrm{KL} + \log \frac{\mathcal{I}_{\Delta}(N)}{\delta}\right). \tag{42}$$

Finally, for any $\beta \in \mathbb{R} \cup \{+\infty\}$,

$$p'_\beta := \sup\ \{p \in [0,1] : \Delta'(q,p) \le \beta\} = \sup\ \{p \in [0,1] : \Delta(q,p) \le \beta\} =: p_\beta. \tag{43}$$

One inequality follows from $\Delta' \le \Delta$. For the other, for any $p' \ge p \ge p_\beta$, we have $\Delta(q,p') > \beta$. Taking an infimum over such p', noting that Δ is lower semi-continuous and therefore obtains a minimum on the closed interval $[p,1]$, $\Delta'(q,p) > \beta$. Hence $p'_\beta \le p_\beta$. The result follows from combining Eq (42) and Eq (43). □

E Lemmas for Theorem 4

Let $C_\beta(p,q) := -\log(p(e^{-\beta} - 1) + 1) - \beta q$ for $\beta > 0$ and

$$\mathcal{I}_\Delta(N) = \sup_{r \in [0,1]} \mathbb{E}_{X \sim \mathrm{Bin}(r,N)}[e^{N\Delta(X/N,r)}] \tag{44}$$

Lemma E.1. *Consider* $q,p \in [0,1]$. *Then*

$$\sup_{\beta \in \mathbb{R}} C_\beta(q,p) = \mathrm{kl}(q,p). \tag{45}$$

Proof. If $q = p = 0$, then $C_\beta(q,p) = 0 = \mathrm{kl}(q,p)$; and, if $q = p = 0$, then also $C_\beta(q,p) = 0 = \mathrm{kl}(q,p)$. If, on the other hand, $q = 0$ but $p = 1$, then clearly $\sup_{\beta \in \mathbb{R}} C_\beta(q,p) = \infty = \mathrm{kl}(q,p)$; and if $q = 1$ but $p = 0$, then also clearly $\sup_{\beta \in \mathbb{R}} C_\beta(q,p) = \infty = \mathrm{kl}(q,p)$. It remains to deal with the case that $q,p \in (0,1)$. In that case, to compute the supremum, set the derivative to zero:

$$q = \frac{pe^{-\beta}}{p(e^{-\beta} - 1) + 1} \tag{46}$$

[7] The equality holds if Δ is not l.s.c by the definition of the infimum as well

and verify that we indeed have a maximum. This gives

$$-\beta = \log \frac{1-p}{1-q} + \log \frac{q}{p}, \tag{47}$$

so

$$-\log(p(e^{-\beta}-1)+1) = \beta + \log \frac{q}{p} = \log \frac{1-q}{1-p}. \tag{48}$$

Therefore,

$$\sup_{\beta \in \mathbb{R}} C_\beta(q,p) = -\log \frac{1-p}{1-q} + q \log \frac{1-p}{1-q} + q \log \frac{q}{p} = \mathrm{kl}(q,p). \tag{49}$$

□

The following lemma is essentially Prop 2.1 from Germain et al. (2009), but stated in a slightly more careful form:

Lemma E.2 (Germain et al. (2009))**.** *Consider* $0 \le q < p < 1$*. If* $q > 0$*, then there exists a unique* $\beta > 0$ *such that*

$$C_\beta(q,p) = \mathrm{kl}(q,p). \tag{50}$$

On the other hand, if $q = 0$*, then*

$$\lim_{\beta \to \infty} C_\beta(0,p) = \mathrm{kl}(0,p). \tag{51}$$

Proof. If $0 < q < p < 1$, then the unique β follows from the proof of Lem E.1:

$$\beta = \log \frac{1-q}{1-p} + \log \frac{p}{q} \in (0,\infty). \tag{52}$$

On the other hand, if $q = 0 < p < 1$, then

$$\mathrm{kl}(0,p) = -\log(1-p) = \lim_{\beta \to \infty} C_\beta(0,p). \tag{53}$$

□

Lemma E.3 (Catoni (2007) and Germain et al. (2009))**.** *For every* $\beta > 0$*, it holds that* $\mathcal{I}_{C_\beta}(N) = 1$*.*

Proof. Let $r \in [0,1]$ and $X \sim \mathrm{Bin}(r,N)$. Note that

$$\mathbb{E}[e^{-\beta X}] = (r(e^{-\beta}-1)+1)^N. \tag{54}$$

Therefore,

$$\mathbb{E}[e^{NC_\beta(X/N,r)}] = \mathbb{E}[e^{-N\log(r(e^{-\beta}-1)+1)-\beta X}] = \frac{\mathbb{E}[e^{-\beta X}]}{(r(e^{-\beta}-1)+1)^N} = 1. \tag{55}$$

□

Lemma E.4. *Let* $y \ge 0$*. Then* $B[\lim_{\beta\to\infty} C_\beta(0,\,\cdot\,), y] = \lim_{\beta\to\infty} B[C_\beta(0,\,\cdot\,), y]$*.*

Proof. Note that

$$\lim_{\beta\to\infty} C_\beta(0,p) = -\log(1-p), \qquad C_\beta(0,p) = -\log(p(e^{-\beta}-1)+1). \tag{56}$$

Therefore,

$$B[\lim_{\beta\to\infty} C_\beta(0,\,\cdot\,), y] = 1 - e^{-y}, \qquad B[C_\beta(0,\,\cdot\,), y] = \min\left(\frac{1-e^{-y}}{1-e^{-\beta}}, 1\right) \tag{57}$$

The result then follows from the observation that

$$\lim_{\beta\to\infty} \frac{1-e^{-y}}{1-e^{-\beta}} = 1 - e^{-y}. \tag{58}$$

□

F Learning a Convex Function

In Sec 3, we optimised an objective with respect to a function $\Delta\colon [0,1]^2 \to \mathbb{R} \cup \{+\infty\}$ that was proper, l.s.c., and convex. In this appendix, we describe how a function $\Delta\colon [0,1]^2 \to \mathbb{R}$ that is differentiable and convex can be generally parametrised. We also discuss two challenges encountered during the optimisation: (1) computing and differentiating through a supremum and (2) computing and optimising a partial inverse.

F.1 Parametrising a Convex Function

To generally parametrise a differentiable and convex $\Delta\colon [0,1]^2 \to \mathbb{R}$, we use the sum of an affine function and a one-hidden-layer neural network with softplus activation functions and positive weights at the output layer. The combination of positive weights and softplus activation functions ensures that the neural network is a convex function. The number of hidden units used is varied between 128 and 1024; the precise numbers are specified in the descriptions of the experiments.

F.2 Computing and Differentiating Through a Supremum

The generic PAC-Bayes theorem (Thm 3) involves $\mathcal{I}_\Delta(N)$, which in turn involves a supremum of a function over $r \in [0,1]$. When optimising with respect to Δ, we therefore need to compute and differentiate through a supremum. To compute the supremum, we finely discretise $[0,1]$ and compute the maximum over this grid. Technically, by approximating the supremum in this way, the bound is approximate, which means that it might not be a valid generalisation bound. However, by making the discretisation very fine, using an inter-point spacing of 10^{-5}, the error on the generalisation bound is negligible. To differentiate the supremum, we simply run automatic differentiation on the approximation. In the remainder of this subsection, we give a plausible explanation for why this procedure also approximates the gradients correctly. The following discussion is based on `https://math.stackexchange.com/questions/3753495/derivative-of-argmin-in-a-constrained-problem`.

Consider $f\colon [0,1] \times \mathbb{R} \to \mathbb{R}$ continuously differentiable in its interior. We aim to compute

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \sup_{x \in [0,1]} f(x,\theta) = \frac{\mathrm{d}}{\mathrm{d}\theta} \max_{x \in [0,1]} f(x,\theta) \tag{59}$$

where the supremum turns into a maximum by compactness of $[0,1]$ and continuity of f. We assume that the maximum is uniquely obtained and write

$$z(\theta) = \arg\max_{x \in [0,1]} f(x,\theta). \tag{60}$$

Then

$$\sup_{x \in [0,1]} f(x,\theta) = f(z(\theta),\theta), \tag{61}$$

so we can compute the derivative with respect to θ with the chain rule if we can compute $z'(\theta)$.

Case 1: The constraint $x \in [0,1]$ is not binding. In that case, the stationarity condition is satisfied in a neighbourhood of θ:

$$\partial_x f(z(\theta),\theta) = 0. \tag{62}$$

Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \sup_{x \in [0,1]} f(x,\theta) = \partial f(z(\theta),\theta) z'(\theta) + \partial_\theta f(z(\theta),\theta) = \partial_\theta f(z(\theta),\theta). \tag{63}$$

Case 2: The constraint $x \in [0,1]$ is binding. In that case, $\partial_x f(z(\theta),\theta) \neq 0$, so $\partial_x f(z(\theta),\theta) < 0$ and you can argue that the optimum will remain to be attained at the constraint in a neighbourhood of θ. Therefore, $z'(\theta) = 0$, which means that again

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \sup_{x \in [0,1]} f(x,\theta) = \partial f(z(\theta),\theta) z'(\theta) + \partial_\theta f(z(\theta),\theta) = \partial_\theta f(z(\theta),\theta). \tag{64}$$

In either case,

$$\frac{\mathrm{d}}{\mathrm{d}\theta} \sup_{x \in [0,1]} f(x, \theta) = \partial_\theta f(z(\theta), \theta), \tag{65}$$

which shows that the derivative with respect to the maximiser can be ignored. Assuming that $z(\theta)$ can be well approximated by computing the maximiser over the fine discretisation, and that in turn leads to a good approximation of $\partial_\theta f(z(\theta), \theta)$, this provides justification for our approach of simply running automatic differentiation on the approximation to the supremum.

Finally, we note that, although computing the derivative accurately is useful for the optimisation to succeed, the bounds we compute are valid regardless of how accurate the derivative is (subject to the computation of the supremum itself being sufficiently accurate). In practice, we observe that the learned convex bound decreases steadily during optimisation (Fig 2), and that it approaches, but never goes below, the optimistic MLS bound, as per Cor 3, which provides evidence that the implementation is sufficiently accurate for our purposes.

F.3 Computing and Optimising a Partial Inverse

The objective that we optimise with respect to Δ is $\overline{p}_\Delta$. Recall from Eq (6) that

$$\overline{p}_\Delta \coloneqq B\big[\Delta(\overline{R}_S(Q), \,\cdot\,), \tfrac{1}{N}\big(\mathrm{KL}(Q\|P) + \log \tfrac{\mathcal{I}_\Delta(N)}{\delta}\big)\big]. \tag{66}$$

We now abbreviate $f \coloneqq \Delta(\overline{R}_S(Q), \,\cdot\,)$ and $c \coloneqq \frac{1}{N}\big(\mathrm{KL}(Q\|P) + \log \frac{\mathcal{I}_\Delta(N)}{\delta}\big)$, so that the objective is $B[f, c] = \sup\{p \in [0,1] : f(p) \le c\}$ for $f\colon [0,1] \to \mathbb{R}$ convex and $c \in \mathbb{R}$. Assuming that $f = f_\theta$ and $c = c(\theta)$ depend on some parameters θ (*i.e.*, the parameters of the neural network defining Δ), our goal is to compute $B[f_\theta, c(\theta)]$ and optimise it with respect to θ.

A possible issue that can be run into during optimisation is that, if $f(p) \le c$ for all $p \in [0,1]$, then $B[f, c] = 1$ and the gradient with respect to θ may be zero, which means that the optimisation may fail to make progress. A similar issue is discussed by Dziugaite and Roy (2017) when trying to optimise the MLS bound: the derivative of the inverse Bernoulli KL can be zero if c is large enough. In Dziugaite and Roy (2017, Sec 2.2) this is handled by upper bounding the inverse Bernoulli KL using Pinsker's inequality. This can lead to upper bounds which are greater than 1 (whereas the exact computation of $B[f, c]$ never allows this to happen), but has the advantage of always providing a useful gradient signal.

Similarly, we can define $\overline{B}[f, c] \coloneqq \sup\{p \in \mathbb{R}_{\ge 0} : f(p) \le c\}$ for $f\colon \mathbb{R}_{\ge 0} \to \mathbb{R}$ and $c \in \mathbb{R}$, which ignores the constraint that $p \le 1$. Note that in our case, since f_θ is defined by a neural network, it is trivial to extend its domain from $[0,1]$ to $\mathbb{R}_{\ge 0}$. This will allow us to obtain a useful derivative even when the bound is vacuous. Moreover, in our case f_θ will be convex, which means that $\overline{B}[f_\theta, c(\theta)] = B[f_\theta, c(\theta)]$ if $B[f_\theta, c(\theta)] < 1$: $B[f_\theta, c(\theta)]$ is characterised by an upcrossing[8] of $c(\theta)$ by f_θ, and, by convexity, f_θ can have at most one such upcrossing.

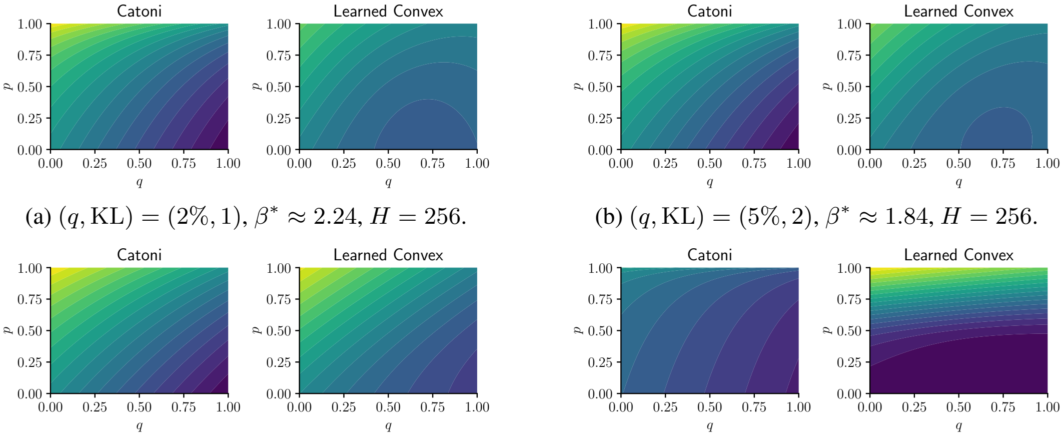
We now describe how $\overline{B}[f_\theta, c(\theta)]$ can be (approximately) computed and differentiated with respect to θ. To compute $\overline{B}[f_\theta, c(\theta)]$, we evaluate f_θ on a discretisation of the interval $[0, u]$ using an inter-point spacing of 10^{-4} and attempt to detect an upcrossing of $c(\theta)$. Assume that this procedure finds an upcrossing; otherwise, either increase u (*e.g.*, by doubling) and try again or return u and set the derivative to zero (failure). Denote $x = \overline{B}[f_\theta, c(\theta)]$ and note that f_θ is continuously differentiable in its interior, because it is a neural network with softplus activations. Assume that $x > 0$, which in practice turns out to nearly always be the case. Using continuity of f_θ, it holds that $f_\theta(x) = c(\theta)$. It also holds that $\partial_x f_\theta(x) > 0$ (f_θ upcrosses $c(\theta)$ at x). Restricting f_θ to an appropriate neighbourhood, the derivative of $\overline{B}[f_\theta, c(\theta)]$ with respect to θ comes down to computing the derivative of $f_\theta^{-1}(c(\theta))$ with respect to θ. The latter derivative can be computed as follows:

$$\partial_\theta c(\theta) = \frac{\mathrm{d}}{\mathrm{d}\theta} f_\theta(f_\theta^{-1}(c(\theta))) = \partial_x f_\theta(x) \frac{\mathrm{d}}{\mathrm{d}\theta} f_\theta^{-1}(c(\theta)) + \partial_\theta f_\theta(x), \tag{67}$$

which implies that

$$\frac{\mathrm{d}}{\mathrm{d}\theta} f_\theta^{-1}(c(\theta)) = \frac{\partial_\theta c(\theta) - \partial_\theta f_\theta(x)}{\partial_x f_\theta(x)}, \tag{68}$$

[8] We say that a function $f\colon \mathbb{R} \to \mathbb{R}$ upcrosses $y \in \mathbb{R}$ at $x \in \mathbb{R}$ if there exists some $\varepsilon > 0$ such that $f(x') < y$ for all $x' \in (x - \varepsilon, x)$ and $f(x') > y$ for all $x' \in (x, x + \varepsilon)$.



(a) $(q, \mathrm{KL}) = (2\%, 1)$, $\beta^* \approx 2.24$, $H = 256$.

(b) $(q, \mathrm{KL}) = (5\%, 2)$, $\beta^* \approx 1.84$, $H = 256$.

(c) $(q, \mathrm{KL}) = (30\%, 1)$, $\beta^* \approx 0.976$, $H = 256$.

(d) $(q, \mathrm{KL}) = (40\%, 50)$, $\beta^* \approx 4.40$, $H = 256$.

Figure 5: **Although the tightest Catoni bound is the optimal generic PAC-Bayes bound for a fixed dataset, evidence suggests that there are other choices for Δ which also achieve the tightest bound.** For various settings of fixed q and KL, we optimise a convex function with H hidden units to minimise $\overline{p}_\Delta$ with $\delta = 0.1$ and $N = 30$. We compare the optimal Catoni C_{β^*} (left sides) with the learned convex function (right sides). For all runs, $\overline{p}_\Delta$ converged to $\inf_{\beta>0} \overline{p}_{C_\beta}$ within a small tolerance.

recalling that $\partial_x f_\theta(x) > 0$.

Similarly to App F.2, although computing the gradient through the partial inverse accurately is useful for optimising the convex function, the bound itself will be valid as long as the value of the partial inverse itself is computed sufficiently accurately.

G Additional Results for Numerical Verification of Theory

Fig 5 complements Fig 2 by comparing the optimal Catoni C_β to examples of the learned convex functions, which demonstrates that there are other choices than C_β which achieve bounds that are close to $\inf_{\Delta\in\mathcal{C}} \overline{p}_\Delta$ within a small tolerance. Fig 6 complements Fig 2 by considering three slightly more complicated cases of a random dataset. Note that, in Figs 6b and 6c, during iterations 10^4–10^6, the optimiser struggles: the trace jumps around. The are various reasons for why this might have happened: the neural network parametrising Δ has too few hidden units, the learning rate of the optimiser is too large, the various approximations that involve Δ (Apps F.2 and F.3) introduce too much error, or Open Problem 1 might be false.

H Worked Example for Corollary 3

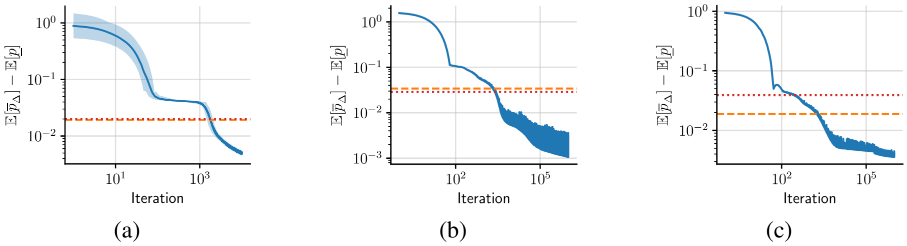
In this section, we illustrate an application of Cor 3 to determine when, in the simplified scenario where $R_S(Q) = \frac{1}{2}$ almost surely, the expected MLS bound is tighter than the tightest expected Catoni bound. This verifies with an analytic example, that, as we claim in Sec 3, although the Catoni bound is optimal for a fixed dataset and learning algorithm, it is not optimal in expectation in the general case of a random dataset. To make the example more concrete, we also compute the bounds for $\mathrm{KL} = 1$ with probability $\frac{1}{2}$ and $\mathrm{KL} = 100$ otherwise. This choice for KL is motivated with the conclusion at the end the section.

Denote $\alpha = \frac{1}{N}(\mathrm{KL}(Q\|P) + \log\frac{1}{\delta})$. We can solve for the optimal expected Catoni bound:

$$\inf_{\beta>0} \frac{1}{1-e^{-\beta}} \left(1 - e^{-\frac{1}{2}\beta}\mathbb{E}[e^{-\alpha}]\right). \tag{69}$$

Denote $u = \mathbb{E}[e^{-\alpha}]$ and set the derivative with respect to β to zero:

$$(1-e^{-\beta})\tfrac{1}{2}ue^{-\frac{1}{2}\beta} - e^{-\beta}(1 - ue^{-\frac{1}{2}\beta}) = 0. \tag{70}$$



(a) (b) (c)

Figure 6: **Extra results indicating that the tightest expected Catoni bound is not the optimal generic PAC-Bayes bound for a random dataset.** We optimise a convex function with H hidden units to minimise $\overline{p}_\Delta$ with $\delta = 0.1$ and $N = 30$. All plots consider random q and KL and show the *expected* difference with the optimistic MLS bound (Cor 3). In (a), the shaded regions show the minimum and maximum over ten initialisations. Due to computational considerations, (b) and (c) only show one repetition, since they are run for much longer than (a) (roughly 25 hours for each run). All plots show the MLS bound (dotted red) and (c) and (d) show the optimal Catoni bound with parameter β^* (dashed orange). All runs quickly converged to non-vacuous values. (a): $(q, \mathrm{KL}) \in \{(35\%, 5), (45\%, 30), (40\%, 7), (43\%, 25)\}$ uniformly, $\beta^* \approx 2.23$, $H = 1024$. (b): $(q, \mathrm{KL}) \in \{(3.0\%, 46.0), (9.6\%, 0.52), (14.2\%, 48.9)\}$ (rounded values) uniformly, $\beta^* \approx 3.21$, $H = 1024$. (c): $(q, \mathrm{KL}) \in \{(18.1\%, 2.56), (8.0\%, 5.83), (16.8\%, 30.0)\}$ (approximate values) uniformly, $\beta^* \approx 2.20$, $H = 1024$.

Letting $x = e^{-\frac{1}{2}\beta}$, this equation becomes a quadratic equation:

$$x^2 - \frac{2}{u}x + 1 = 0 \implies x = \frac{1}{u} \pm \sqrt{\frac{1}{u^2} - 1}. \tag{71}$$

Therefore,

$$\beta = 2\log\frac{u}{1 \pm \sqrt{1-u^2}} = 2\log\frac{\mathbb{E}[e^{-\alpha}]}{1 \pm \sqrt{1-\mathbb{E}[e^{-\alpha}]^2}}, \tag{72}$$

so the positive solution for β is given by

$$\beta = 2\log\frac{\mathbb{E}[e^{-\alpha}]}{1 - \sqrt{1-\mathbb{E}[e^{-\alpha}]^2}} \approx 2.803. \tag{73}$$

Plugging this back into the Catoni bound gives that

$$\inf_{\beta>0} \mathbb{E}[\overline{p}_{C_\beta}] = \frac{\sqrt{1-u^2}}{1 - \left(\frac{1-\sqrt{1-u^2}}{u}\right)^2} = \tfrac{1}{2}\left(1 + \sqrt{1-\mathbb{E}[e^{-\alpha}]^2}\right) \approx 0.943. \tag{74}$$

We compare this with the choice $\Delta = \mathrm{kl}$, which corresponds to the MLS bound. In that case, the expected bound is given by

$$\mathbb{E}[\overline{p}_{\mathrm{kl}}] = \mathbb{E}\, B[\mathrm{kl}(\tfrac{1}{2}, \,\cdot\,), \alpha + \gamma] \tag{75}$$

where

$$\gamma = \frac{1}{N}\log\mathcal{I}_{\mathrm{kl}}(N) = \frac{1}{N}\log\sum_{n=0}^{N}\binom{N}{n}\left(\frac{N}{n}\right)^n\left(1-\frac{N}{n}\right)^{N-n}. \tag{76}$$

To compute $B[\mathrm{kl}(\frac{1}{2}, \,\cdot\,), \alpha + \gamma]$, note that

$$\tfrac{1}{2}\log\frac{\frac{1}{2}}{p} + \tfrac{1}{2}\log\frac{\frac{1}{2}}{1-p} = y \implies y_\pm = \tfrac{1}{2}\left(1 \pm \sqrt{1-e^{-2y}}\right). \tag{77}$$

Therefore,

$$\mathbb{E}[\overline{p}_{\mathrm{kl}}] = \tfrac{1}{2}\left(1 + \mathbb{E}\sqrt{1-e^{-2(\alpha+\gamma)}}\right) \approx 0.836. \tag{78}$$

This is better than the Catoni bound by more than 10%. Finally, by omitting γ, we find the optimistic MLS bound:

$$\mathbb{E}[\underline{p}] = \tfrac{1}{2}\left(1 + \mathbb{E}\sqrt{1-e^{-2\alpha}}\right) \tag{79}$$

Note how similar the computed bounds are:

$$\inf_{\beta>0} \mathbb{E}[\overline{p}_{C_\beta}] = \tfrac{1}{2}\left(1 + \sqrt{1 - \mathbb{E}[e^{-\alpha}]^2}\right), \qquad \text{(optimal Catoni)} \tag{80}$$

$$\mathbb{E}[\overline{p}_{\mathrm{kl}}] = \tfrac{1}{2}\left(1 + \mathbb{E}\sqrt{1 - e^{-2(\alpha+\gamma)}}\right), \qquad \text{(MLS)} \tag{81}$$

$$\mathbb{E}[\underline{p}] = \tfrac{1}{2}\left(1 + \mathbb{E}\sqrt{1 - e^{-2\alpha}}\right). \qquad \text{(optimistic MLS)} \tag{82}$$

Define

$$\phi(x) = 1 - \tfrac{1}{2}\sqrt{1 - x^2}, \tag{83}$$

which is convex. Define the ϕ-entropy of a random variable X by

$$\mathbb{H}_\phi(X) = \mathbb{E}[\phi(X)] - \phi(\mathbb{E}[X]). \tag{84}$$

Observe that $\mathbb{H}_\phi(X)$ quantifies the slack in Jensen's inequality, which, in particular, means that $\mathbb{H}_\phi(X) \geq 0$. We then find that

$$\inf_{\beta>0} \mathbb{E}[\overline{p}_{C_\beta}] - \mathbb{E}[\underline{p}] = \mathbb{H}_\phi(e^{-\alpha}), \tag{85}$$

$$\mathbb{E}[\overline{p}_{\mathrm{kl}}] - \mathbb{E}[\underline{p}] = \mathbb{E}[\phi(e^{-\alpha}) - \phi(e^{-\gamma}e^{-\alpha})]. \tag{86}$$

Therefore, the MLS bound is tighter if and only if

$$\mathbb{E}[\phi(e^{-\alpha})] - \mathbb{E}[\phi(e^{-\gamma}e^{-\alpha})] \leq \mathbb{H}_\phi(e^{-\alpha}). \tag{87}$$

In words, the expected MLS bound is tighter than the tighest expected Catoni bound if the slack in Jensen's inequality is more than the slack introduced by scaling by $e^{-\gamma}$, which, for example, will be the case if $\mathrm{KL}(Q\|P)$ attains both small and large values.

I Additional Details for Synthetic Classification

I.1 Data Generation Details

We now provide details of the task generation for the 1D classification experiments. For each task, we sample a 1D function f from a Gaussian process (GP) with an exponentiated quadratic kernel with lengthscale 0.7 and variance 1. This is then turned into a classification problem by thresholding: $S = \big((x_n, \operatorname{sign}(f(x_n)))\big)_{n=1}^{N}$, where $x_n \sim \mathcal{U}[-2, 2]$. Finally, we only select tasks that are approximately balanced, so that the risk of a trivial predictor is ≈ 0.5. This is done in a way that preserves the i.i.d. assumptions. In more detail, when sampling from the GP, in addition to sampling the N points that make up the dataset, we also sample an additional 300 points that make up an extra held-out set which is unseen by any of the meta-learners, and whose sole purpose is for us to be able to estimate the actual generalisation risk of each posterior, which is what we report under "Generalisation Risk" in, *e.g.*, Fig 3. Furthermore, jointly with the $N + 300$ datapoints already sampled, we sample an *additional* 300 datapoints which form a "balance set". The sole purpose of the balance set is for us to check if the dataset is roughly balanced between positive and negative examples. If the prevalence of each class in the balance set is not ≈ 0.5, then we discard the entire GP sample. Since the balance set is disjoint from the original dataset N (and also the 300 datapoints forming the extra held-out set), doing this does not jeopardise the i.i.d. property within each dataset.[9] Approximately balancing the datasets in this way is convenient because it allows us to interpret results more easily, since the risk of the trivial classifier that always predicts the majority class in the observed dataset is ≈ 0.5. We generate two disjoint meta-train sets (along with their corresponding meta-test sets) this way: one with $N = 30$ and another with $N = 60$. The meta-learners are either meta-trained and meta-tested exclusively with $N = 30$ or exclusively with $N = 60$.

I.2 Deterministic Classification for Test Set Model

PAC-Bayes bounds naturally lead to stochastic classifiers (also known as Gibbs classifiers), whereby a fresh sample $h \sim Q$ is drawn whenever the classifier is presented with an input. However, this does

[9] The tasks themselves are also still i.i.d. from the same task distribution, although this does not affect the validitiy of our bounds, which only requires the i.i.d. assumption to hold *within* each dataset.

not need to be the case for *test set* bounds. In fact, it may be easier to bound the risk of a deterministic classifier with a test set bound than a stochastic one, since for a deterministic classifier, each term in the sum defining the empirical risk is a Bernoulli random variable, and hence it is trivial to apply Thm 1, which leads to significantly tighter bounds than Thm 2 in the small data regime. Additionally, we observed that when optimising $\frac{1}{T}\sum_{t=1}^{T}\overline{R}_{S_{t,\mathrm{test}}}(Q_\theta(S_{t,\mathrm{train}}))$ as in Sec 4, the learned posterior map Q_θ eventually became essentially deterministic once meta-training was complete.

Another way to use the binomial tail test set bound in the case when $\ell \in [0,1]$ (as it effectively is for Gibbs classifiers, once the zero-one loss is integrated over Q to form $\mathbb{E}_{h\sim Q}[\ell_{0/1}((x,y),h)] \in [0,1]$), is to randomise the computation of the empirical loss. In particular, for each $z \in S_{\mathrm{test}}$, one could sample a Bernoulli random variable with parameter $\ell(z,h)$ and set the empirical risk in Thm 1 to be the average of these Bernoulli random variables. For test set bounds, these are i.i.d. hence the sum is binomially distributed and Thm 1 can be applied directly. We do not pursue this here, as it does not make a significant difference when the classifier is nearly deterministic, as we found.

For these reasons, at meta-*test* time we convert the test set bound meta-learners into *deterministic* classifiers by using a Bayes classifier instead of a Gibbs classifier. That is, the final predictor for a test set bound meta-learner with posterior Q is given by

$$\hat{y}(x) \coloneqq \operatorname{sign}\left(\mathbb{E}_{w\sim Q}[w^\mathsf{T}\phi_\theta(x)]\right) \tag{88}$$

The risk of this predictor on a dataset S, which is the quantity we report and upper bound for the test set bound meta-learners in Sec 4, is then simply the usual (non-Gibbs) risk: $R_S(\hat{y}) = \frac{1}{N}\sum_{(x,y)\in S}\ell_{0/1}((x,y),\hat{y})$. We emphasise that this change primarily serves to simplify the test set bound computation (and allow the use of the tighter Thm 1 instead of just Thm 2), and essentially does not affect the performance of the test set classifiers — the Gibbs and Bayes risks are nearly identical because the Gibbs classifier learned by the test set meta-learners was already nearly deterministic.

I.3 Computing the Empirical Risk

In this section we provide additional details on how to compute the empirical risk term for the meta-learners. This applies for the PAC-Bayes meta-learners at both meta-train time and meta-test time, but only applies to the test set bound meta-learners during meta-train time — at meta-test time we use a Bayes classifier for the test set bound meta-learners instead of a Gibbs one; see App I.2 for a discussion. Recall that we consider hypotheses of the form $h_w(x) \coloneqq \operatorname{sign}(w^\mathsf{T}\phi_\theta(x))$. Then the loss function is:

$$\ell_{0/1}((x,y),h_w) = \mathbb{1}[y \neq \operatorname{sign}(w^\mathsf{T}\phi_\theta(x))] \tag{89}$$

We can then compute the empirical Gibbs risk as

$$\begin{aligned}\overline{R}_S(Q) &= \frac{1}{|S|}\sum_{(x,y)\in S}\mathbb{E}_{w\sim Q}[\mathbb{1}[y \neq \operatorname{sign}(w^\mathsf{T}\phi_\theta(x))]] &(90)\\ &= \frac{1}{|S|}\sum_{(x,y)\in S}\Pr\left(yw^\mathsf{T}\phi_\theta(x) < 0\right) &(91)\end{aligned}$$

We now specialise to the case of Gaussian $Q \coloneqq \mathcal{N}(\mu,\Sigma)$. In this case, we can compute the empirical Gibbs risk in Eq (91) in closed form (up to the error function, which has a standard implementation in PyTorch (Paszke et al., 2017)):

$$yw^\mathsf{T}\phi_\theta(x) \sim \mathcal{N}(y\mu^\mathsf{T}\phi_\theta(x),\, \phi_\theta(x)^\mathsf{T}\Sigma\phi_\theta(x)), \tag{92}$$

$$\Pr\left(yw^\mathsf{T}\phi_\theta(x) < 0\right) = \Phi\left(\frac{-y\mu^\mathsf{T}\phi_\theta(x)}{\sqrt{\phi_\theta(x)^\mathsf{T}\Sigma\phi_\theta(x)}}\right) \tag{93}$$

where Φ is the standard normal cumulative distribution function and where we have used the fact that $y \in \{-1,+1\}$ so $y^2 = 1$. Now recall that Φ is related to the error function $\operatorname{erf}(x)$ (as defined in PyTorch) by $\Phi(x) = \frac{1}{2}[1+\operatorname{erf}(\frac{x}{\sqrt{2}})]$, which gives:

$$\overline{R}_S(Q) = \frac{1}{|S|}\sum_{(x,y)\in S}\frac{1}{2}\left[1+\operatorname{erf}\left(\frac{-y\mu^\mathsf{T}\phi_\theta(x)}{\sqrt{2\phi_\theta(x)^\mathsf{T}\Sigma\phi_\theta(x)}}\right)\right]. \tag{94}$$

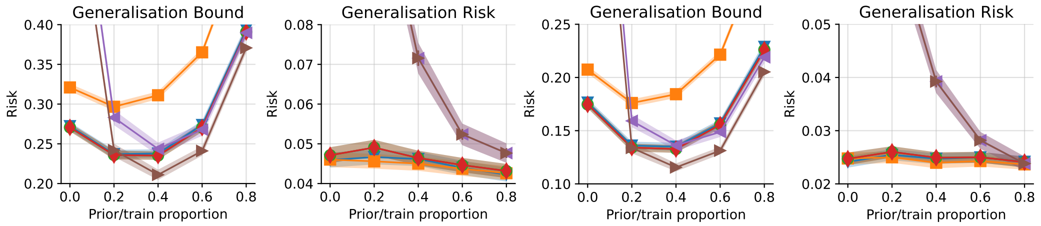
Hence we can backpropagate through the empirical Gibbs risk without the need for Monte Carlo integration over Q.

I.4 Post-Hoc Optimisation of Posteriors

It is well-known that when performing amortised variational inference (VI) (Kingma & Welling, 2014), there is an *amortisation gap* (Cremer et al., 2018), which is the gap between the performance of the amortised inference network, and the performance obtained when optimising each variational problem separately. The meta-learners we consider in Sec 4 have similarities with amortised VI, except that PAC-Bayes bound minimisation is being amortised, rather than VI. Similarly, there is an amortisation gap for our meta-learners, which is the gap between the bound obtained by the meta-learner when the posterior that was outputted by the posterior map is directly used, versus the bound we obtain when optimising, for each dataset, the posterior using gradient-based methods (in our case, ADAM (Kingma & Ba, 2015)). Optimising the posterior for each dataset individually is costly, but since we are concerned with obtaining the tightest bounds possible, we perform this *post-hoc optimisation* for all of our meta-learners (including the results reported in Sec 4). Fortunately, each optimisation does not take too long, since we can initialise the posterior at the distribution output by the meta-learner.

So far, we have discussed post-hoc optimisation of the PAC-Bayes bound. However, we can also run post-hoc optimisation for the test set bound, *as long as the optimised posterior does not depend on the test set*. We consider post-hoc optimising the *train risk* for each dataset. In principle, this could possibly lead to overfitting the train set. In practice, we observe that this *improves* performance slightly for the MLP-NP (indicating that the MLP-NP test set meta-learner was underfitting the train set somewhat), and leaves performance essentially completely unaffected for the CNN-NP, because the train set risk is *already* essentially zero for the CNN-NP test set meta-learner before post-optimisation. Note that post-hoc optimisation is completely legal as a means of obtaining bounds — it does not affect the validity of the bounds we consider, but merely closes the amortisation gap.

As an ablation study, we can compare the performance of the meta-learners with and without post-hoc optimisation. Figs 7 and 14 show the performance of the CNN-NP and MLP-NP meta-learners *without* post-hoc optimisation, which should be compared to Figs 3 and 13, which show their performance *with post-optimisation*. Comparing Fig 7 with Fig 3 we see that post-hoc optimisation improves the performance of the PAC-Bayes meta-learners slightly but leaves the test set meta-learners essentially unaffected for the CNN-NP. Comparing Fig 14 with Fig 13, we see that post-hoc optimisation tightens the generalisation bounds for all meta-learners slightly. In conclusion, post-hoc optimisation sometimes leads to a small benefit, so we perform it for all meta-learners.



(a) $N = 30$ datapoints.

(b) $N = 60$ datapoints.

Figure 7: Average generalisation bound and actual generalisation risk **for CNN-NP without post-hoc optimisation** (± two standard errors) for Catoni (▼), MLS (■), optimisitc MLS (●), learned convex (♦), Chernoff test set bound (◀), and binomial tail test set bound (▶). All bounds hold with failure probability $\delta = 0.1$ except for optimistic MLS which is not proven to be a valid generalisation bound.

I.5 The Multilayer Perceptron Neural Process

We now describe the multilayer perceptron (MLP)-NP model, which is closely related to (but not identical with) the *conditional neural process* model first described in Garnelo, Rosenbaum, et al. (2018).[10] When using the MLP-NP, we use an MLP to implement the feature map ϕ_θ. Additionally, each of the maps Q_θ/P_θ consists of two MLPs: the *encoder* and *decoder*. The encoder e_θ maps

[10] The original conditional neural process outputs a Gaussian distribution directly in function space. This leads to complications when considering the KL term in the PAC-Bayes bounds, hence we modify it to output a Gaussian distribution over the parameters of a linear model.

$\mathbb{R} \times \{-1, +1\} \to \mathbb{R}^M$, where $\mathbb{R}^M$, $M \in \mathbb{N}$ is the *representation space*. The decoder d_θ maps $\mathbb{R}^M \to \mathcal{G}(\mathbb{R}^K)$, where $\mathcal{G}(\mathbb{R}^K)$ is the set of all Gaussian distributions over $\mathbb{R}^K$ (in practice, the decoder outputs a vector in $\mathbb{R}^{K+K(K+1)/2}$, which is converted into the mean of the Gaussian, and also the lower-triangular part of the Cholesky decomposition of the covariance matrix). When given a dataset S, the encoder computes a permutation-invariant representation of the dataset as $r_\theta(S) \coloneqq \frac{1}{N} \sum_{(x,y) \in S} e_\theta((x, y))$. The decoder d_θ then computes a Gaussian posterior distribution over the hypothesis space as $d_\theta(r_\theta(S))$.

I.6 The CNN-Based Gaussian Neural Process

In contrast to the MLP-NP, which uses MLPs to implement the feature map ϕ_θ, the CNN-Based Gaussian Neural Process (Bruinsma et al., 2021a) (CNN-NP) lets the k^{th} component of the feature map be $\phi_{\theta,k}(x) = \exp(-\frac{1}{2\ell^2}(x - x_k)^2)$, a Gaussian basis function centred at some fixed input x_k, with a learnable lengthscale ℓ. The centres of the Gaussian basis functions $(x_k)_{k=1}^K$ are evenly spread out through the interval $[-2, 2]$. The CNN-NP lets Q_θ and P_θ be parametrisations of maps from datasets to full-covariance Gaussian posteriors over the weights of these basis functions where the maps incorporate a symmetry called *translation equivariance*: if all inputs of the observed data are shifted by some amount, then the posterior over the weights for the basis functions should be shifted accordingly. Translation equivariance enables the CNN-NP to use CNNs for Q_θ and P_θ instead of MLPs.

We now give a brief high-level description how the CNN architecture for the posterior mean of the Gaussian works. This follows the way that the mean of the *Convolutional Conditional Neural Process* (ConvCNP) is computed,[11] and we refer the reader to Sec 4 and especially Fig 1 of Gordon et al. (2020) for a full description. First, the dataset is embedded into a 1D function with two channels, known as the *data channel* and the *density channel*. This 1D function is then evaluated on a discretised grid, and then fed into a CNN. The CNN output then defines mean of the Gaussian predictive distribution over functions. However, unlike in Gordon et al. (2020) and Bruinsma et al. (2021a), we modify this setup slightly, so that, instead of interpreting the CNN output as the mean of the Gaussian predictive over functions, it is interpreted as the mean of the Gaussian posterior over *weights* of the basis functions in ϕ_θ. Defining the posteriors in weight space instead of function space makes it much easier to compute the KL-divergence.

We also give a brief description of how the CNN architecture for computing the posterior *covariance* works. As this computation is more involved than the computation for the mean, we refer the reader to Sec 3 and App E.2 of Bruinsma et al. (2021a) for a detailed description, `https://github.com/wesselb/NeuralProcesses.jl` for a full implementation, and Bruinsma et al. (2021b) for a useful visual description of the Gaussian neural process architecture, on which we base our CNN-NP architecture used in Sec 4. To compute the covariance matrix of the weights of the basis functions, the dataset S is first embedded into three images on $[-2, 2] \times [-2, 2]$. The embedding is performed by placing a Gaussian basis function[12] corresponding to each datapoint along the *diagonal* of the $[-2, 2] \times [-2, 2]$ square. These three images are known as the *data channel*, *density channel* and *source channel* respectively. As explained in Bruinsma et al. (2021a), the data channel incorporates information about the y-values of the observations in S, the density channel records information about how many points in S are observed at any particular x-location, and the source channel is simply in the shape of an identity matrix which, intuitively speaking, allows CNN-NP to begin with a "white noise" covariance matrix that afterwards is modulated to include correlations. These continuous images, after being appropriately discretised on a regular 2D grid[13] are passed through a 2D CNN, which outputs an image which is interpreted as a covariance matrix over the interval $[-2, 2]$. In order to ensure that the covariance matrix output is positive semi-definite, we multiply the output by itself: $\Sigma \coloneqq MM^\mathsf{T}$, where Σ is the covariance matrix and M is the $K \times K$ matrix output by the CNN. This covariance matrix is finally interpolated onto the grid defined by the locations of the basis functions in ϕ_θ, which then defines the covariance of the weights of the basis functions.

[11] The predictive *mean* of the ConvCNP (Gordon et al., 2020) and that of the later, full-covariance Gaussian Neural Process (Bruinsma et al., 2021a) are computed in the same way.

[12] These basis functions are distinct from the basis functions used to define the feature map ϕ_θ.

[13] This discretisation need not be the same as the spacing used for the Gaussian basis functions which make up the feature map ϕ_θ.

I.7 Hyperparameters

General meta-learner training details. We fix the failure probability at $\delta = 0.1$ for all of the meta-learning experiments. During meta-training, for the PAC-Bayes models we found it was more numerically stable to optimise the logarithm of the objective described in Eq (20). In particular, for the Catoni bound model, we used the numerically stable implementation of $\log(1 - e^{-x})$ referenced in `https://github.com/pytorch/pytorch/issues/39242`. For all meta-learners, we use a mini-batch estimate of the objective in Eq (20), with a batch size of 16 datasets. We use a weight decay of 1×10^{-5} for all meta-learners.

MLP-NP hyperparameters. We use a relatively large architecture for the MLP-NP, as we found during preliminary experiments that larger architectures performed better. The feature dimension of the linear model (see App I.5) was set at $K = 256$. The MLPs implementing the feature map ϕ_θ, encoder e_θ and decoder d_θ all had two hidden layers, each with a width of 512. The MLP-NP was trained for 100 epochs on the meta-train set, with a learning rate of 2×10^{-5} (we found that higher learning rates could lead to instabilities during training) with the ADAM optimiser (Kingma & Ba, 2015). We did some manual hyperparameter tuning to choose these hyperparameters, but they were not selected exhaustively. To avoid overfitting to the meta-train set when doing manual hyperparameter tuning, we also sampled a meta-validation set of datasets, which we used when tuning hyperparameters.

CNN-NP hyperparameters. For the CNN in the architecture, we use a U-Net (Ronneberger et al., 2015). The U-Net, we use has 12 layers, with the number of channels in each layer being $8, 16, 16, 32, 32, 64, 64, 64, 64, 32, 32, 16$ respectively. This architecture matches that used by Gordon et al. (2020). The number of Gaussian basis functions per unit of input space (which determines the number of features in ϕ_θ) was set at 16. The discretisation of the Gaussian Neural Process (*i.e.*, the spacing at which the continuous representation is evaluated before passing it through the CNN) is set at 32 points per unit. Then CNN-NP was trained for 10 epochs on the meta-train set, with a learning rate of 1×10^{-3} with the ADAM optimiser. We did very little manual hyperparameter tuning for the CNN-NP, because we found that it was fairly robust to the choice of learning rate and basis function spacing. In all cases, the CNN-NP optimised much more quickly than the MLP-NP.

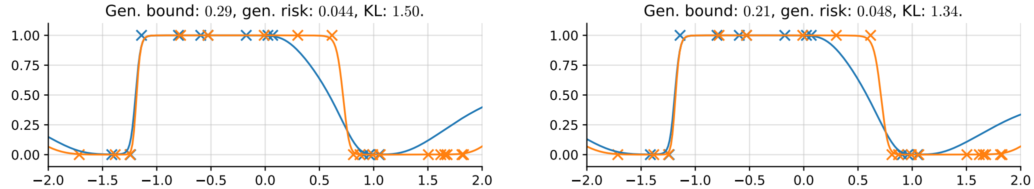
Post-hoc optimisation. We perform post-hoc optimisation at meta-test time, as discussed in App I.4. Given a dataset, we initialise the posterior at the Gaussian distribution which is output by the NP. We then use the ADAM optimiser (Kingma & Ba, 2015) with a learning rate of 3×10^{-4} for a maximum of $3\,000$ optimisation steps to target either the PAC-Bayes bound (for PAC-Bayes meta-learners), or the train risk (for test set meta-learners). If, after 100 optimisation steps, the generalisation bound has not decreased by at least 0.0001, then the optimisation is ended early.

Compute. We used roughly 500–1000 GPU-hours divided NVIDIA Tesla V100 and GeForce RTX 2080 graphics cards using both an internal cluster and Amazon Web Services. Most of the computational budget was spent on the meta-learning experiments. Of these, the MLP-NP was more costly to run than then CNN-NP, since it took longer to train.

J Additional Plots for Synthetic Classification

J.1 Predictive Distributions

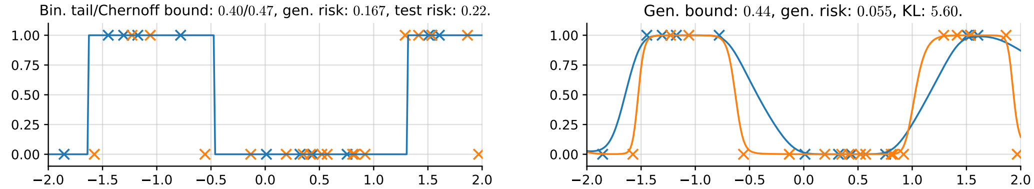
In this appendix we include extra plots of 1D classification tasks from the meta-test set, similar to Fig 4:



(a) **MLS** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

(b) **Catoni** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

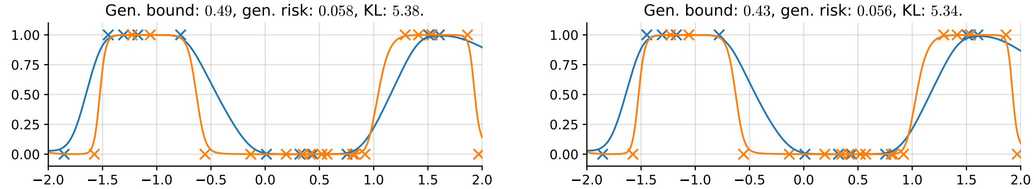
Figure 8: Predictions and bounds on one of the 1D datasets with **30 datapoints**. The bounds hold with failure probability $\delta = 0.1$.



(a) **Test set** bounds, showing the learned hypothesis, (—), the train set (×) of size 12 and the test set (×) of size 18.

(b) **Catoni** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

Figure 9: Predictions and bounds on one of the 1D datasets with **30 datapoints**. The bounds hold with failure probability $\delta = 0.1$.



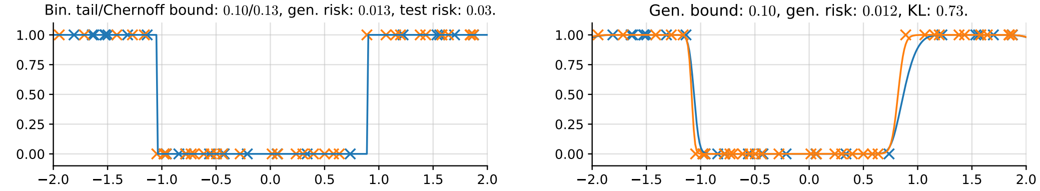
(a) **MLS** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

(b) **Learned convex** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 12 and risk set (×) of size 18.

Figure 10: Predictions and bounds on one of the 1D datasets with **30 datapoints**. The bounds hold with failure probability $\delta = 0.1$.

J.2 Performance of MLP-NP

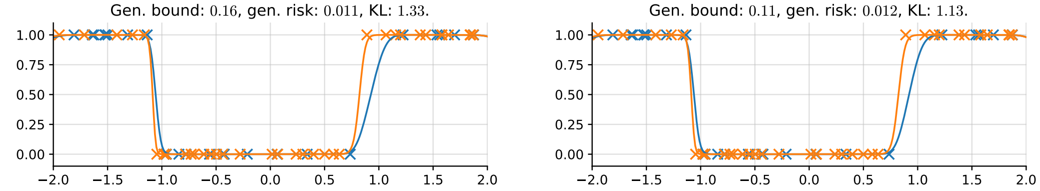
In the main body, we considered the CNN-NP model, since it performed better while training much faster and requiring fewer parameters then the MLP-NP. In Figs 13 and 14 we also show the performance of the MLP-NP for the test set meta-learners and also the Catoni bound meta-learner, both with and without post-hoc optimisation (see App I.4). We see that the MLP-NP test set meta-learner performs very similarly to the CNN-NP one when $N = 30$, but performs slightly worse when $N = 60$. The MLP-NP Catoni meta-learner is either as tight as the CNN-NP Catoni meta-learner, or slightly looser, except when $N = 30$ and the prior proportion is 0 or 0.8, in which case the MLP-NP seems to have encountered learning difficulties. Also note that generalisation risk is generally higher for the MLP-NP than the CNN-NP.



(a) **Test set bounds**, showing the learned hypothesis, (—), the train set (×) of size 24 and the test set (×) of size 36.

(b) **Catoni** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 24 and risk set (×) of size 36.

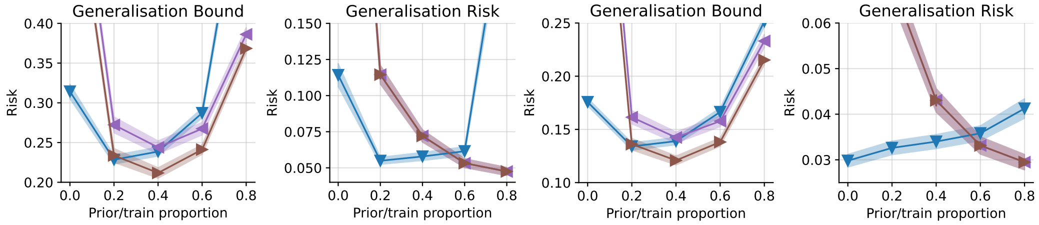
Figure 11: Predictions and bounds on one of the 1D datasets with **60 datapoints**. The bounds hold with failure probability $\delta = 0.1$.



(a) **MLS** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 24 and risk set (×) of size 36.

(b) **Learned convex** bound with data-dependent prior, showing the prior (—) and posterior (—) predictive, prior set (×) of size 24 and risk set (×) of size 36.

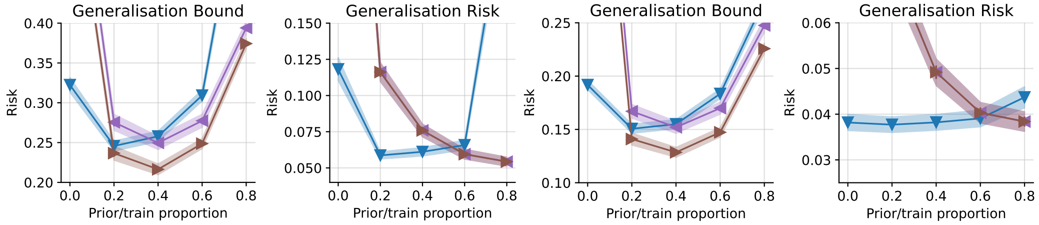
Figure 12: Predictions and bounds on one of the 1D datasets with **60 datapoints**. The bounds hold with failure probability $\delta = 0.1$.



(a) $N = 30$ datapoints.

(b) $N = 60$ datapoints.

Figure 13: Average generalisation bound and actual generalisation risk **for MLP-NP with post-hoc optimisation** (± two standard errors) for Catoni (▼), Chernoff test set bound (◀), and binomial tail test set bound (▶). All bounds hold with failure probability $\delta = 0.1$.



(a) $N = 30$ datapoints.

(b) $N = 60$ datapoints.

Figure 14: Average generalisation bound and actual generalisation risk **for MLP-NP without post-hoc optimisation** (± two standard errors) for Catoni (▼), Chernoff test set bound (◀), and binomial tail test set bound (▶). All bounds hold with failure probability $\delta = 0.1$.