PAC-Bayesian Learning of Optimization Algorithms

M. Sucker
Department of Mathematics
University of Tübingen
michael.sucker@math.uni-tuebingen.de

P. Ochs
Department of Mathematics
University of Tübingen
ochs@math.uni-tuebingen.de

Abstract

We apply the PAC-Bayes theory to the setting of learning-to-optimize. To the best of our knowledge, we present the first framework to learn optimization algorithms with provable generalization guarantees (PAC-bounds) and explicit trade-off between a high probability of convergence and a high convergence speed. Even in the limit case, where convergence is guaranteed, our learned optimization algorithms provably outperform related algorithms based on a (deterministic) worst-case analysis. Our results rely on PAC-Bayes bounds for general, unbounded loss-functions based on exponential families. By generalizing existing ideas, we reformulate the learning procedure into a one-dimensional minimization problem and study the possibility to find a global minimum, which enables the algorithmic realization of the learning procedure. As a proof-of-concept, we learn hyperparameters of standard optimization algorithms to empirically underline our theory.

1 Introduction

Let $\ell(\cdot,\theta)$ be an instance of a class of functions $(\ell(\cdot,\theta))_{\theta\in\Theta}$. We consider the minimization problem:

$$\min_{x\in\mathbb{R}^n} \ell(x,\theta)\,. \tag{1}$$

Our goal is the construction of an algorithm $\mathcal{A}(\alpha,\theta)$, depending on some hyperparameters α, that is provably the best (on average) for the given class of problems. We contrast the majority of approaches in continuous optimization in two ways:

i) Classical optimization theory studies the worst-case behaviour, which guarantees the same convergence for all problems that arise:

$$\alpha^* \in \arg\min_{\alpha\in\mathcal{H}} \sup_{\theta\in\Theta} \ell(\mathcal{A}(\alpha,\theta),\theta)\,.$$

Thereby, this is often accompanied by rough estimates and ignores that some problems are more likely to occur than others. On the other hand, by using the additional information that θ is a realization of some random variable $\mathfrak{S}$, we seek for the average case in form of the mean function, usually called the risk:

$$\alpha^* \in \arg\min_{\alpha\in\mathcal{H}} \mathbb{E}_{\mathfrak{S}}[\ell(\mathcal{A}(\alpha,\mathfrak{S}),\mathfrak{S})]\,.$$

From an optimization perspective, this is a distinct approach leading to performance guarantees in expectation or with high probability over the draw of new problem instances. This allows us to exploit features of the considered class of problems beyond analytical accessible quantities such as the Lipschitz constant (of the gradient) or the strong convexity modulus, which are usually pessimistic and hard to compute.

ii) Instead of analytically constructing an algorithm driven by intricate worst-case estimates, we train our algorithm (by learning) to be the best one on some samples $\{\ell(\cdot,\theta_i)\}_{i=1}^N$ and prove that the performance generalizes, in a suitable sense (PAC-Bayes), to the random function $\ell(\cdot,\mathfrak{S})$. This type of problem, i.e. minimizing the expected loss, is naturally found in the whole area of machine learning and cannot be solved directly, since the mean function is generally unknown. Consequently, one typically solves an approximate problem like empirical risk minimization in the hope that the solution found there will transfer:

$$\alpha^* \in \arg\min_{\alpha\in\mathcal{H}} \frac{1}{N}\sum_{i=1}^{N} \ell(\mathcal{A}(\alpha,\theta_i),\theta_i)\,.$$

However, through this, one is left with the problem of generalization, which is one of the key problems in machine learning in general. Therefore, one of the main concerns of learning-to-optimize are generalization bounds. A

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

famous framework to provide such bounds is the PAC-Bayes framework, which allows for giving high-probability bounds on the true risk relative to the empirical risk.

In this paper, we apply the PAC-Bayesian theory to the setting of learning-to-optimize. In doing so, we provide PAC-Bayesian generalization bounds for a general optimization algorithm on a general, unbounded loss function and we show how one can trade-off convergence guarantees for convergence speed. As a proof of concept, we illustrate our approach by learning, for example, the step-size τ and the inertial parameter β, i.e., $\alpha = (\tau, \beta)$, of a fixed number of iterations of the Heavy-ball update scheme given by:

$$\begin{aligned} x^{(k+1)} &= \mathcal{HB}\Big(x^{(k)}, x^{(k-1)}, \alpha, \theta\Big) \\ &:= x^{(k)} - \tau \nabla \ell(x^{(k)}, \theta) + \beta(x^{(k)} - x^{(k-1)}), \end{aligned} \tag{2}$$

which generalizes Gradient Descent for $\beta = 0$.

1.1 Our Contributions

- We provide a general PAC-Bayes theorem for general, unbounded loss functions based on exponential families. In this framework, the role of the reference distribution (called the prior), the data dependence of the learned distribution (called the posterior) and the divergence term arise directly and naturally from the definition. Furthermore, this abstract approach allows for a unified implementation of the learning framework.
- We provide a principled way of excluding the case of the learnt algorithm's divergence from the considerations, which in turn allows us to apply our PAC theorem under a modified objective. Based on this, we give a theoretically grounded way of ensuring a given (user-specified) convergence probability during learning. Taken together, this allows us to trade-off convergence speed and the probability of convergence. To the best of our knowledge, both approaches are completely new and could also be very interesting for other learning approaches.
- We apply our PAC-Bayesian framework to the problem of learning-to-optimize and learn optimization algorithms by minimizing the PAC-Bayesian upper bound.

2 Related Work

The literature on both learning-to-optimize and the PAC-Bayes learning approach is vast. Hence, in the discussion of learning-to-optimize, we will mainly focus on approaches that provide certain theoretical guarantees. Especially, this excludes most model-free approaches, which replace the whole update step with a learnable mapping such as a neural network. Chen et al. (2021) provide a good overview of the variety of approaches in learning-to-optimize. Good introductory references for the PAC-Bayes approach are given by Guedj (2019) and Alquier (2021).

Learning-to-Optimize with Guarantees. Chen et al. (2021) point out that learned optimization methods may lack theoretical guarantees for the sake of convergence speed. That said, there are applications where convergence guarantee is of highest priority. To underline this problem, Moeller et al. (2019) provide an example where a purely learning-based approach fails to reconstruct the crucial details in medical image reconstruction. Also, they prove convergence of their method by restricting the output to descent directions, for which mathematical guarantees exist. The basic idea is to trace the learned object back to, or constrain it to, a mathematical object with convergence guarantees. Similarly, Sreehari et al. (2016) provide sufficient conditions under which the learned mapping is a proximal mapping. Related schemes under different assumptions and guarantees are given by Chan et al. (2016), Teodoro et al. (2017), Tirer and Giryes (2018), Buzzard et al. (2018), Ryu et al. (2019), Sun et al. (2019), Terris et al. (2021) and Cohen et al. (2021). A major advantage of these methods is the fact that the number of iterations is not restricted a priori. However, a major drawback is their restriction to specific algorithms and problems. Another approach, which limits the number of iterations, yet in principle can be applied to every iterative optimization algorithm, is unrolling, pioneered by Gregor and LeCun (2010). Xin et al. (2016) study the IHT algorithm and show that it is, under some assumptions, able to achieve a linear convergence rate. Likewise, Chen et al. (2018) establish a linear convergence rate for the unrolled ISTA. However, a difficulty in the theoretical analysis of unrolled algorithms is actually the notion of convergence itself, such that one rather has to consider the generalization performance. Only few works have addressed this: Either directly by means of Rademacher complexity (Chen et al., 2020), or indirectly in form of a stability analysis (Kobler et al., 2020), as algorithmic stability is linked to generalization and learnability (Bousquet and Elisseeff, 2000, 2002; Shalev-Shwartz et al., 2010). *We consider the model-based approach of unrolling a general iterative optimization algorithm and provide generalization guarantees in form of PAC-bounds.*

PAC-Bounds through Change-of-Measure. The PAC-Bayesian framework allows us to give high probability bounds on the risk, either as an oracle bound or as an empirical bound. The key ingredients are so-called change-of-measure inequalities. The choice of such an inequality strongly influences the corresponding bound. The one used most often is based on a variational representation of the Kullback–Leibler divergence due to Donsker and Varadhan (1975), employed, for example, by Catoni (2004, 2007). Yet, not all bounds are based on a variational representation, i.e., holding uniformly over all posterior distributions (Rivasplata et al., 2020). However, most bounds involve the Kullback–Leibler divergence as a measure of proximity, e.g. those by McAllester (2003b,a), Seeger (2002), Lang-

ford and Shawe-Taylor (2002), or the general PAC-Bayes bound of Germain et al. (2009). More recently, other divergences have been used: Honorio and Jaakkola (2014) prove an inequality for the χ^2-divergence, which is also used by London (2017). Bégin et al. (2016) and Alquier and Guedj (2018) use the Renyi-divergence (α-divergence). Ohnishi and Honorio (2021) propose several PAC-bounds based on the general notion of f-divergences, which includes the Kullback–Leibler-, α- and χ^2-divergences. *We develop a general PAC theorem based on exponential families. In this general approach, the role of prior, posterior, divergence and data dependence will be given naturally. Moreover, this approach allows us to implement a general learning framework that can be applied to a wide variety of algorithms.*

Boundedness of the Loss Function. A major drawback of many of the existing PAC-Bayes bounds is the assumption of a bounded loss-function. However, this assumption is mainly there to apply some exponential moment inequality like the Hoeffding- or Bernstein-inequality (Rivasplata et al., 2020; Alquier, 2021). Several ways have been developed to solve this problem: Germain et al. (2009) explicitly include the exponential moment in the bound, Alquier et al. (2016) use so-called Hoeffding- and Bernstein-assumptions, Catoni (2004) restricts to the sub-Gaussian or sub-Gamma case. Another possibility, of which we make use of here, is to ensure the generalization or exponential moment bounds by properties of the algorithm in question. London (2017) uses algorithmic stability to provide PAC-Bayes bounds for SGD. *We consider suitable properties of optimization algorithms aside from algorithmic stability to ensure the exponential moment bounds. To the best of our knowledge, this has not been done before.*

Minimization of the PAC-Bound. The PAC-bound is a relative bound and relates the risk to other terms such as the empirical risk. Yet, it does not directly say anything about the actual numbers. Thus, one aims to minimize the bound: Langford and Caruana (2001) compute non-vacuous numerical generalization bounds through a combination of PAC-bounds with a sensitivity analysis. Dziugaite and Roy (2017) extend this by minimizing the PAC-bound directly. Pérez-Ortiz et al. (2021) also consider learning by minimizing the PAC-Bayes bound and provide very tight generalization bounds. Thiemann et al. (2017) are able to solve the minimization problem resulting from their PAC-bound by alternating minimization. Further, they provide sufficient conditions under which the resulting minimization problem is quasi-convex. *We also follow this approach and consider learning as minimization of the PAC bound, however, applied to the context of learning-to-optimize.*

Choice of the Prior. A common difficulty in learning with PAC-Bayes bounds is the choice of the prior distribution, as it heavily influences the performance of the learned models and the generalization bound (Catoni, 2004; Dziugaite et al., 2021; Pérez-Ortiz et al., 2021). In part, this is due to the fact that the divergence term can dominate the bound, keeping the posterior close to the prior. This leads to the idea to choose a data- or distribution-dependent prior (Seeger, 2002; Parrado-Hernández et al., 2012; Lever et al., 2013; Dziugaite and Roy, 2018; Pérez-Ortiz et al., 2021). *As we also found the choice of the prior distribution to be crucial for the performance of our learned algorithms, we use a data-dependent prior. Further, we point out how the prior is essential in preserving necessary properties during learning. It is key to control the trade-off between convergence guarantee and convergence speed.*

More Generalization Bounds. There are many areas of research that study generalization bounds and have not been discussed here. Importantly, the vast field of stochastic optimization (SO) provides generalization bounds for specific algorithms. The main differences to our setting are the learning approach and the assumptions made:

- Learning approach: In most of the cases, the concrete algorithms studied in SO generate a single point by either minimizing the (regularized) empirical risk functional over a possibly large dataset, or by repeatedly updating the point estimate based on a newly drawn (small) batch of samples. Then, one studies the properties of this point in terms of the stationarity measure of the true risk functional (see e.g. Bottou et al. (2018); Davis and Drusvyatskiy (2022); Bianchi et al. (2022)).
- Assumptions: Since the setting in SO is more explicit, more assumptions have to be made. Typical assumptions in SO are (weak) convexity (Shalev-Shwartz et al., 2009; Davis and Drusvyatskiy, 2019), bounded gradients (Défossez et al., 2022), bounded noise (Davis and Drusvyatskiy, 2022), or at least smoothness (Kavis et al., 2022), just to name a few.

We provide a principled way to learn a distribution over hyperparameters of an abstract algorithm under weak assumptions. Further, the methodology is independent of a concrete implementation and independent of the concrete choice of hyperparameters. Furthermore, we go explicitly beyond analytically tractable quantities.

3 Preliminaries and Notation

If not further specified, we will endow every topological space X with the corresponding Borel-σ-algebra $\mathcal{B}(X)$. If we consider a product space $X \times Y$ of two measurable spaces $(X, \mathcal{A})$ and $(Y, \mathcal{B})$, we endow it with the product-σ-algebra $\mathcal{A} \otimes \mathcal{B}$. We use the Fraktur-font to denote random variables. Let $\big(\Omega, \mathcal{F}, \mathbb{P}\big)$ be a probability space, Θ be a

Polish space and

$$\mathfrak{S} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \Theta$$

be a random variable. Its distribution is denoted by $\mathbb{P}_{\mathfrak{S}}$, following the general notation $\mathbb{P}_{\mathfrak{X}}$ to denote the distribution of a random variable $\mathfrak{X}$. Integration w.r.t. $\mathbb{P}_{\mathfrak{X}}$ is denoted by $\mathbb{E}_{\mathfrak{X}}[g] := \mathbb{E}_{\mathfrak{X}}[g(\mathfrak{X})] := \int g(x)\ \mathbb{P}_{\mathfrak{X}}(dx)$. Finally, $\mathbb{1}_A$ denotes the indicator function of a set A, which is one for $x \in A$ and zero else, and $\log$ denotes the natural logarithm.

Definition 3.1. *Let $N \in \mathbb{N}$. Further, let $\big(\Omega, \mathcal{F}, \mathbb{P}\big)$ be a probability space, $\mathfrak{S}_i : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \Theta$, $i = 1, ..., N$, be random variables. A measurable function*

$$\mathfrak{D}_N : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \Big(\prod_{i=1}^N \Theta, \bigotimes_{i=1}^N \mathcal{B}(\Theta)\Big),\ \omega \mapsto \prod_{i=1}^N \mathfrak{S}_i(\omega)$$

is called a dataset*. If the induced distribution $\mathbb{P}_{\mathfrak{D}_N}$ factorizes into the product of the marginals, i.e., if it satisfies $\mathbb{P}_{\mathfrak{D}_N} = \bigotimes_{i=1}^N \mathbb{P}_{\mathfrak{S}_i}$, it is called* independent *and if, additionally, it satisfies $\mathbb{P}_{\mathfrak{D}_N} = \bigotimes_{i=1}^N \mathbb{P}_{\mathfrak{S}}$, it is called* i.i.d.

Notation 3.2. *The space $(\prod_{i=1}^N \Theta,\ \bigotimes_{i=1}^N \mathcal{B}(\Theta))$ will be denoted by $(\mathcal{D}_N, \mathcal{B}(\mathcal{D}_N))$. Since $\bigotimes_{i=1}^N \mathcal{B}(\Theta)$ is indeed the Borel-σ-algebra of $\mathcal{D}_N$, it will not be mentioned anymore.*

In the PAC-Bayesian framework, generalization bounds typically involve a so-called posterior distribution, which in turn is referred to as a data-dependent distribution. Often, this term is left unspecified. However, as also pointed out by Rivasplata et al. (2020), this is an instance of a Markov kernel. Another commonly used name are regular conditional probabilities, following the definition of a regular conditional distribution (Catoni, 2004; Alquier, 2008).

Definition 3.3. *Let $\mathfrak{D}_N : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \mathcal{D}_N$ be a dataset and $\mathcal{H}$ a Polish space. A Markov kernel from $\mathcal{D}_N$ to $\mathcal{H}$ is called a* data-dependent distribution*.*

Remark 3.4. *The assumption of a Polish space is not very restrictive (for practical considerations) and sufficient to ensure the existence of such Markov kernels. Both definitions can be found in the supplementary material C.1.*

The following theory will be based on exponential families, which are a special class of probability distributions with a specific, mathematically convenient form.

Definition 3.5. *Let $\Lambda \subset \mathbb{R}^k$. A family of probability measures $(\mathbb{Q}_\lambda)_{\lambda\in\Lambda}$ on a measurable space $(\mathcal{H}, \mathcal{B}(\mathcal{H}))$ is called an* exponential family*, if there is a dominating probability measure $\mathbb{P}_{\mathfrak{H}}$, measurable functions $\eta_1, ..., \eta_k : \Lambda \longrightarrow \mathbb{R}$, a measurable function $A : \Lambda \longrightarrow \mathbb{R}_{>0}$, measurable functions $T_1, ..., T_k : \mathcal{H} \longrightarrow \mathbb{R}$ and $h : \mathcal{H} \longrightarrow \mathbb{R}_{>0}$, such that every $\mathbb{Q}_\lambda$ has a $\mathbb{P}_{\mathfrak{H}}$-density of the form:*

$$\frac{d\mathbb{Q}_\lambda}{d\mathbb{P}_{\mathfrak{H}}}(\alpha) = h(\alpha)A(\lambda)\exp\big(\langle\eta(\lambda), T(\alpha)\rangle\big), \quad \mathbb{P}_{\mathfrak{H}} - a.s.$$

where $\eta := (\eta_1, ..., \eta_k)$ and $T := (T_1, ..., T_k)$.

In the PAC-Bayesian setting, the dominating measure $\mathbb{P}_{\mathfrak{H}}$ is usually referred to as the prior and every distribution $\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}$ is referred to as a posterior. Note that this deviates from the standard definitions of prior and posterior in Bayesian statistics, which are linked through the likelihood. We use a similar notation as in Barndorff–Nielsen (2014) and denote

$$\begin{aligned} c(\lambda) &:= \int_{\mathcal{H}} h(\alpha)\exp(\langle\eta(\lambda), T(\alpha)\rangle)\ \mathbb{P}_{\mathfrak{H}}(d\alpha) \\ \kappa(\lambda) &:= \log(c(\lambda)), \end{aligned} \tag{3}$$

or short, $\kappa = \log(c)$. It holds that $A(\lambda) = c(\lambda)^{-1}$.

Remark 3.6. *In the case $h = 1$ and $\eta(\lambda) = \lambda$, c is the Laplace transform (moment generating function) of the push-forward measure $\mathbb{P}_{\mathfrak{H}} \circ T^{-1}$ and κ the corresponding log-Laplace transform (cumulant-generating function). Further, if $\eta(\lambda)$ actually describes a lower-dimensional manifold or curve in $\mathbb{R}^k$, $(\mathbb{Q}_\lambda)_{\lambda\in\Lambda}$ is sometimes also called a curved exponential family (Efron, 1975).*

Remark 3.7. *In the following we will consider data-dependent exponential families, i.e., the sufficient statistic T additionally depends on a dataset $\mathfrak{D}_N$. Hence, also c and κ do depend on $\mathfrak{D}_N$. Thus, we will assume that $T : \mathcal{H} \times \mathcal{D}_N \longrightarrow \mathbb{R}$ is measurable. In this case, $\mathbb{Q}_\lambda$ is indeed a data-dependent distribution.*

Notation 3.8. *For notational simplicity, we will omit the dependence of $\mathbb{Q}_\lambda$, T, c and κ on the dataset $\mathfrak{D}_N$.*

For the rest of the paper, we assume that we are given an exponential family $(\mathbb{Q}_\lambda)_{\lambda\in\Lambda}$ w.r.t. $\mathbb{P}_{\mathfrak{H}}$ of the form:

$$\frac{d\mathbb{Q}_\lambda}{d\mathbb{P}_{\mathfrak{H}}}(\alpha) = \frac{h(\alpha)}{c(\lambda)}\exp\Big(\langle\eta(\lambda), T(\alpha)\rangle\Big). \tag{4}$$

Finally, since the loss-function is neither assumed to be bounded nor to satisfy any self-bounding or bounded-difference property, the following result will be needed. It states that non-negative random variables with finite second moment satisfy a one-sided sub-Gaussian inequality. It can be found as Exercise 2.9 on page 47 in the book by Boucheron et al. (2013).

Lemma 3.9. *Let $\mathfrak{X}$ be a non-negative random variable with finite second moment. Then, for every $\lambda > 0$ it holds:*

$$\mathbb{E}\Big[\exp\Big(-\lambda(\mathfrak{X} - \mathbb{E}[\mathfrak{X}])\Big)\Big] \leq \exp\Big(\frac{\lambda^2}{2}\mathbb{E}[\mathfrak{X}^2]\Big).$$

4 Problem Setup

As described in the introduction, we aim to solve the following minimization problem with a random objective function ℓ under Assumption 1:

$$\min_{x\in\mathbb{R}^n} \ell(x, \mathfrak{S}).$$

Assumption 1. *Θ is a Polish space, $\mathfrak{S} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \Theta$ is a random variable, and $\ell : \mathbb{R}^n \times \Theta \longrightarrow \mathbb{R}$ is measurable and non-negative.*

Remark 4.1. *The non-negativity assumption is not restrictive, as any lower-bounded function f can be rescaled according to $\ell(x, \theta) := f(x, \theta) - \inf_{x \in \mathbb{R}^n} f(x, \theta)$.*

To actually solve this problem for a concrete realization θ, we apply an optimization algorithm $\mathcal{A}$ to ℓ. For this, we will consider a similar setting as in London (2017), i.e., randomized algorithms are considered as deterministic algorithms with randomized hyperparameters.

Definition 4.2. *Let $\mathcal{H}$ be a Polish space. A measurable function*

$$\mathcal{A} : \mathcal{H} \times \mathbb{R}^n \times \Theta \longrightarrow \mathbb{R}^n$$

is called a parametric algorithm *and $\mathcal{H}$ is called the* hyperparameter space *of $\mathcal{A}$. A random variable*

$$\mathfrak{H} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \mathcal{H}$$

is called a hyperparameter *of $\mathcal{A}$.*

Remark 4.3. *$\mathcal{H}$ corresponds to the hyperparameters of the algorithm, $\mathbb{R}^n$ to the initialization and Θ to the parameters of the problem instance.*

Learning now refers to learning a distribution $\mathbb{Q}$ on $\mathcal{H}$. For this, one needs a reference distribution:

Assumption 2. *$\mathcal{A}$ is a parametric optimization algorithm with hyperparameter space $\mathcal{H}$. The prior $\mathbb{P}_{\mathfrak{H}}$ is induced by hyperparameters $\mathfrak{H} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \longrightarrow \mathcal{H}$ that are independent of the dataset $\mathfrak{D}_N$ and $\mathfrak{S}$. The initialization $x^{(0)} \in \mathbb{R}^n$ is given and fixed.*

The initialization and the probability space $\big(\Omega, \mathcal{F}, \mathbb{P}\big)$ will not be mentioned anymore. We define the risk of a randomized parametric algorithm in the usual way:

Definition 4.4. *Let $N \in \mathbb{N}$ and let $\mathfrak{D}_N = (\mathfrak{S}_1, ..., \mathfrak{S}_N)$ be a data set. Further, let $\mathcal{A}$ be a parametric algorithm with hyperparameter space $\mathcal{H}$. Furthermore, let $\mathfrak{S} \sim \mathbb{P}_{\mathfrak{S}}$ be independent of $\mathfrak{D}_N$. Finally, let $\ell : \mathbb{R}^n \times \Theta \longrightarrow \mathbb{R}_{\geq 0}$ satisfy Assumption 1. The risk of $\mathcal{A}$ is defined as the measurable function:*

$$\mathcal{R} : \mathcal{H} \longrightarrow \mathbb{R}_{\geq 0},\ \alpha \mapsto \mathbb{E}_{\mathfrak{S}}[\ell\big(\mathcal{A}(\alpha, \mathfrak{S}), \mathfrak{S}\big)]\,.$$

Similarly, the empirical risk of $\mathcal{A}$ on $\mathfrak{D}_N$ is defined as the measurable map $\hat{\mathcal{R}} : \mathcal{H} \times \mathcal{D}_N \longrightarrow \mathbb{R}_{\geq 0}$ with:

$$\hat{\mathcal{R}}(\alpha, \mathfrak{D}_N) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(\mathcal{A}(\alpha, \mathfrak{S}_i), \mathfrak{S}_i\big)\,.$$

Notation 4.5. *We also use $\ell(\alpha, \theta) := \ell(\mathcal{A}(\alpha, \theta), \theta)$.*

5 General PAC-Bayesian Theorem

In this section we derive a general PAC-Bayes bound, which will be used to bound the generalization risk of the learned parametric optimization algorithm $\mathcal{A}$. As stated above, PAC-Bayesian theorems are usually based on a change-of-measure (in-)equality. The following lemma is a form of the Donsker–Varadhan variational formulation. Though it is not new, we state it nevertheless for the sake of completeness. The proof is especially easy in this case and is given in the supplementary material A.1.

Lemma 5.1. *Let $(\mathbb{Q}_\lambda)_{\lambda \in \Lambda}$ be an exponential family on $\mathcal{H}$ w.r.t. $\mathbb{P}_{\mathfrak{H}}$ of the form (4) and κ as in (3). Then it holds:*

$$\kappa(\lambda) = \sup_{\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}} \mathbb{E}_{\mathbb{Q}}\big[\langle \eta(\lambda), T\rangle + \log(h)\big] - D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}})\,.$$

Furthermore, the supremum is attained at $\mathbb{Q}_\lambda$.

This change-of-measure equality allows to directly give the PAC-Bayesian theorem in its general form. Basically, one uses Markov's inequality to give a probabilistic bound on $\kappa(\lambda)$. The restriction to a finite set is made such that the bound also holds uniformly in $\lambda \in \Lambda$ by a union-bound. This idea appeared previously (Langford and Caruana, 2001; Catoni, 2007; Alquier, 2021).

Theorem 5.2. *Let $(\mathbb{Q}_\lambda)_{\lambda \in \Lambda}$ be an exponential family on $\mathcal{H}$ of the form (4). Further, let Λ be a finite set with cardinality $|\Lambda|$ and let $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$ for all $\lambda \in \Lambda$. Then, for $\epsilon > 0$, it holds that:*

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\forall \lambda \in \Lambda,\ \forall \mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}} : \mathbb{E}_{\mathbb{Q}}\big[\langle \eta(\lambda), T\rangle + \log(h)\big] \leq D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}}) + \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big\} \geq 1 - \epsilon\,.$$

The proof of Theorem 5.2 is given in the supplementary material A.2.

Remark 5.3. *The restriction to a finite set gets problematic, if the term $\log(|\Lambda|)$ influences the bound strongly. In our application the loss is usually much larger than $\log(|\Lambda|)$, such that this is not the case even for large $|\Lambda|$.*

Remark 5.4. *By a chaining argument, the finiteness assumption on Λ can be relaxed to assuming that Λ is totally bounded (e.g. compact) and that the growth of κ can be controlled on balls of radius δ. For more details, we refer to the supplementary material C.6, as this was only found after the rebuttal phase. Also, note that the experiments in Section 7 were carried out with the setting in Theorem 5.2.*

For the rest of the paper, we will have $h \equiv 1$. Corollary 5.5 shows how to transform this general result into a high-probability bound on the risk. It follows directly by using the properties of the Euclidean scalar product.

Corollary 5.5. *Let the assumptions of Theorem 5.2 hold. Furthermore, assume that there are $T' : \mathcal{H} \times \mathcal{D}_N \longrightarrow$*

$\mathbb{R}^{k-1}$, $\eta' : \Lambda \longrightarrow \mathbb{R}^{k-1}$ *and* $\eta_1 : \Lambda \longrightarrow \mathbb{R}_{>0}$*, such that* η *and* T *are given by:*

$$\eta(\lambda) = \big(\eta_1(\lambda),\ \eta'(\lambda)\big)$$
$$T(\alpha, \mathfrak{D}_N) = \big(\mathcal{R}(\alpha) - \hat{\mathcal{R}}(\alpha, \mathfrak{D}_N),\ T'(\alpha, \mathfrak{D}_N)\big)\,.$$

Then it holds for $\epsilon > 0$*:*

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\forall \lambda \in \Lambda, \forall \mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}} \ :\ \mathbb{E}_{\mathbb{Q}}[\mathcal{R}] \leq \mathbb{E}_{\mathbb{Q}}[\hat{\mathcal{R}}] + \frac{1}{\eta_1(\lambda)}\big(D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}}) + \log\Big(\frac{|\Lambda|}{\epsilon}\Big) - \mathbb{E}_{\mathbb{Q}}\big[\langle \eta'(\lambda), T'\rangle\big]\big)\Big\} \geq 1 - \epsilon\,. \tag{5}$$

In Section 6 we provide sufficient conditions, such that $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$ holds for all $\lambda > 0$.

5.1 Minimization of the PAC-Bound

In this whole subsection we use η and T from Corollary 5.5. We seek for $\lambda \in \Lambda$ and $\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}$ that minimizes the right-hand side of the generalization bound in (5). By factoring out $-\frac{1}{\eta_1(\lambda)}$ again, this is actually the same as:

$$\inf_{\lambda\in\Lambda} -\frac{1}{\eta_1(\lambda)}\Big(\sup_{\mathbb{Q}\ll\mathbb{P}_{\mathfrak{H}}} \mathbb{E}_{\mathbb{Q}}[\langle \eta(\lambda), \tilde{T}\rangle] - D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}}) - \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big),$$

where $\tilde{T}(\alpha, \mathfrak{D}_N) := \big(-\hat{\mathcal{R}}(\alpha, \mathfrak{D}_N),\ T'(\alpha, \mathfrak{D}_N)\big)$. Since $\log(|\Lambda|/\epsilon)$ is a constant, Lemma 5.1 shows that the term inside the brackets is actually given by $\tilde{\kappa}(\lambda) - \log(|\Lambda|/\epsilon)$, where $\tilde{\kappa}$ corresponds to the exponential family $\mathbb{Q}_\lambda$ built upon $\tilde{T}, \eta$ and $h \equiv 1$. Furthermore, it shows that the optimal posterior distribution is given by the corresponding member of the exponential family (usually called the Gibbs posterior (Alquier, 2021)):

$$\frac{d\mathbb{Q}_\lambda}{d\mathbb{P}_{\mathfrak{H}}}(\alpha) = \frac{\exp(\langle \eta(\lambda), \tilde{T}(\alpha)\rangle)}{\mathbb{E}_{\mathfrak{H}}[\exp(\langle \eta(\lambda), \tilde{T}\rangle)]}\,.$$

By denoting $F(\lambda) := -\frac{1}{\eta_1(\lambda)}(\tilde{\kappa}(\lambda) - \log(|\Lambda|/\epsilon))$, one is left with solving the following minimization problem:

$$\min_{\lambda\in\Lambda} F(\lambda), \tag{6}$$

which for $\Lambda \subset \mathbb{R}$ is one-dimensional. Under mild assumptions, one can show that $\arg\min_{\lambda>0} F(\lambda)$ lies in a bounded interval. Thus, one can control the accuracy of the solution of the minimization problem (6) by the choice of Λ. The computational cost for evaluating this one-dimensional function several times is low compared to solving several minimization problems during training.

6 Learning Optimization Algorithms with Theoretical Convergence Guarantees

In this section, we consider properties of optimization algorithms, that assert the necessary condition $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$ for all $\lambda \in \Lambda$ to employ the PAC-Bayes bound from Section 5. Typically, this yields the functions η' and T'.

6.1 Guaranteed Convergence

The following convergence property is sufficient to ensure the assumptions of Theorem 5.2. Essentially, it requires the loss of the algorithm's output to be bounded. Nevertheless, it is shown in 6.2 that it is too restrictive to learn hyperparameters that allow for a significant acceleration compared to the standard choices from a worst-case analysis.

Assumption 3. *There is a constant* $C \geq 0$ *and a measurable function* $\rho : \mathcal{H} \longrightarrow \mathbb{R}_{\geq 0}$*, such that it holds:*

$$\ell(\mathcal{A}(\alpha, \mathfrak{S}), \mathfrak{S}) \leq C\rho(\alpha)\ell(x^{(0)}, \mathfrak{S}) \qquad \forall \alpha \in \mathcal{H}\,.$$

Remark 6.1. *The basic motivation for Assumption 3 is to take the (possibly known) convergence behaviour of an optimization algorithm into account.*

Theorem 6.2. *Let* $N \in \mathbb{N}$ *and* $\mathfrak{D}_N$ *be an i.i.d. dataset. Assume* $\mathcal{A}$ *satisfies Assumption 3. Further, assume that* $\mathbb{E}_{\mathfrak{S}}\big[\ell(x^{(0)}, \mathfrak{S})^2\big] < \infty$*. Define* $\eta : \mathbb{R}_{>0} \longrightarrow \mathbb{R}^2$ *and* $T : \mathcal{H} \times \mathcal{D}_N \longrightarrow \mathbb{R}^2$ *through:*

$$\eta(\lambda) := \Big(\lambda,\ -\frac{\lambda^2}{2}\frac{C^2}{N}\mathbb{E}_{\mathfrak{S}}\big[\ell(x^{(0)}, \mathfrak{S})^2\big]\Big),$$
$$T(\alpha, \mathfrak{D}_N) := (\mathcal{R}(\alpha) - \hat{\mathcal{R}}(\alpha, \mathfrak{D}_N),\ \rho^2(\alpha))\,.$$

Then, it holds that $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$*, for all* $\lambda > 0$*.*

The proof of Theorem 6.2 is given in the supplementary material A.3.

6.2 Conditioning on Convergence

Most of the time, the previous approach is only able to learn hyperparameters that ensure convergence. When the considered class of functions $(\ell(\cdot, \theta))_{\theta\in\Theta}$ is that of general quadratic functions, the convergence behaviour is accurately represented by analytic quantities from a worst-case analysis. Thereby, Assumption 3 prevents "aggressive" step-size parameters that lie outside the worst-case convergence regime. This is also encoded in Assumption 3, as C and ρ are independent of $\mathfrak{S}$. Moreover, it can be difficult to compute them. Hence, in this section, a different approach is taken: We allow for divergence, if it only occurs in rare cases with a controllable quantity. Essentially, one only considers the loss for all the hyperparameters, where convergence occurs, as well as the probability for that. In Section 6.3, we develop a technique that allows

the user to control this probability. Clearly, a higher convergence guarantee trades for convergence speed. To the best of our knowledge, the following way of dealing with the rare, unwanted case is completely new.

Definition 6.3. *The* convergence set *is defined as the set-valued mapping $C : \mathcal{H} \rightrightarrows \Theta$ with*

$$C(\alpha) := \{\theta \in \Theta \mid \ell(\mathcal{A}(\alpha, \theta), \theta) \leq \ell(x^{(0)}, \theta)\}\,.$$

Remark 6.4. *Other definitions of the convergence set are possible and the concrete choice will influence the resulting PAC-bound. For proving the result, the essential property is that the loss after application of $\mathcal{A}$ can be bounded, such that the (conditional) expectation is finite.*

For every $\alpha \in \mathcal{H}$, the set $C(\alpha)$ is measurable, as the map $\theta \mapsto \ell(\mathcal{A}(\alpha, \theta), \theta) - \ell(x^{(0)}, \theta)$ is measurable. Nevertheless, we have to make the following assumption:

Assumption 4. *The map $p : \mathcal{H} \longrightarrow [0, 1], \quad \alpha \mapsto p(\alpha) := \mathbb{P}_{\mathfrak{S}}[C(\alpha)]$ is measurable.*

Remark 6.5. *Although at first Assumption 4 seems very restrictive, it is actually very mild in the sense that in our use case one can always find a measurable version of p. For more details see Lemma C.3, whose assumptions are always met in our setting.*

Now we define the convergence risk as the expect loss conditioned on the convergence of the algorithm:

Definition 6.6. *The* convergence risk *is defined as the conditional expectation of the loss given $C(\alpha)$:*

$$\begin{aligned}\mathcal{R}_c(\alpha) &:= \mathbb{E}_{\mathfrak{S}}[\ell(\mathcal{A}(\alpha, \mathfrak{S}), \mathfrak{S}) \mid C(\alpha)] \\ &= \begin{cases} \frac{1}{p(\alpha)} \mathbb{E}_{\mathfrak{S}}[\mathbb{1}_{C(\alpha)}(\mathfrak{S}) \ell(\alpha, \mathfrak{S})], & \text{if } p(\alpha) > 0\,; \\ 0, & \text{else}\,. \end{cases}\end{aligned}$$

Given a dataset $\mathfrak{D}_N = (\mathfrak{S}_1, ..., \mathfrak{S}_N)$, the empirical convergence risk *is defined as:*

$$\hat{\mathcal{R}}_c(\alpha, \mathfrak{D}_N) := \frac{1}{p(\alpha)} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{C(\alpha)}(\mathfrak{S}_i) \ell(\alpha, \mathfrak{S}_i)\,.$$

The following theorem is a generalization of Theorem 6.2. The proof is given in the supplementary material A.4.

Theorem 6.7. *Assume that $\mathbb{P}_{\mathfrak{H}}\{p > 0\} = 1$ and $\mathbb{E}_{\mathfrak{S}}[\ell(x^{(0)}, \mathfrak{S})^2] < \infty$. Define $\eta : \mathbb{R}_{>0} \longrightarrow \mathbb{R}^2$ and $T : \mathcal{H} \times D_N \longrightarrow \mathbb{R}^2$ through*

$$\begin{aligned}\eta(\lambda) &:= \left(\lambda, -\frac{\lambda^2}{2} \frac{1}{N} \mathbb{E}_{\mathfrak{S}}\left[\ell(x^{(0)}, \mathfrak{S})^2\right]\right), \\ T(\alpha, \mathfrak{D}_N) &:= \left(\mathcal{R}_c(\alpha) - \hat{\mathcal{R}}_c(\alpha, \mathfrak{D}_N), \frac{1}{p(\alpha)^2}\right).\end{aligned}$$

Then, it holds that $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$, for all $\lambda > 0$.

Remark 6.8. *$\mathbb{P}_{\mathfrak{H}}\{p > 0\} = 1$ says that, under the prior, the algorithm should not diverge exclusively.*

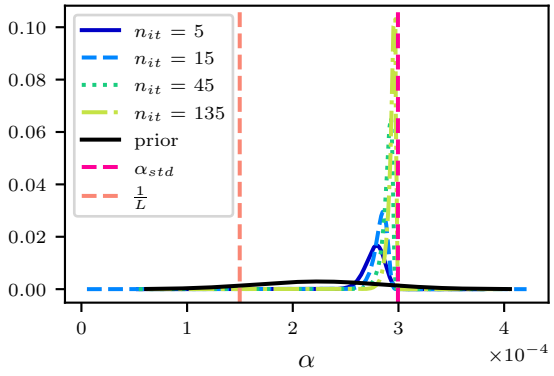

Figure 1: Posterior for an increasing number of iterations: The initial prior is chosen as a Gaussian centered at $\frac{1}{2}(\frac{1}{L} + \frac{2}{L})$. The posterior distributions for $N_{it} \in \{5, 15, 45, 135\}$ are shown. For an increasing number of iterations the posterior puts increasingly more mass close to $\alpha_{std} = \frac{2}{L+\mu}$.

6.3 Guarantee of Convergence with High Probability

In the previous approach, care has to be taken in the choice of the prior $\mathbb{P}_{\mathfrak{H}}$: Constructing the prior in a way that minimizes the upper bound as much as possible can lead to the case where a high convergence probability is neglected, i.e., the algorithm converges only on a small subset of the parameters and for them especially fast, because the term $\frac{1}{p(\alpha)}$ might not compensate for the smaller convergence risk. Thus, if a certain convergence probability ϵ_{conv} has to be satisfied, one has to ensure this in another way. We propose to use a direct consequence of absolute continuity:

Lemma 6.9. *Let $\epsilon_{conv} \in [0, 1]$ and $\mathbb{P}_{\mathfrak{H}}$ be such that $\mathbb{P}_{\mathfrak{H}}\{p < \epsilon_{conv}\} = 0$. Then it holds for every $\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}$:*

$$\mathbb{Q}\{p < \epsilon_{conv}\} = 0\,.$$

Though the proof is trivial, this lemma has a very important consequence, which we want to emphasize here: If one can guarantee that a required property is satisfied for the prior, it will be preserved during the PAC-Bayes learning process, i.e., if the prior only puts mass on hyperparameters that ensure a certain convergence probability, also the posterior will allow only hyperparameters that ensure the same convergence probability. Thus, ensuring a convergence probability will be part of the construction of the prior.

7 Experiments

In all experiments, we use $n = 50$ and a quadratic loss function for which we can choose the smallest and largest eigenvalue, i.e., a loss of the form $\frac{1}{2}\|Ax - b\|^2$. As optimization algorithms we unroll either the Heavy-ball (2) or Gradient Descent update step for a fixed number of itera-

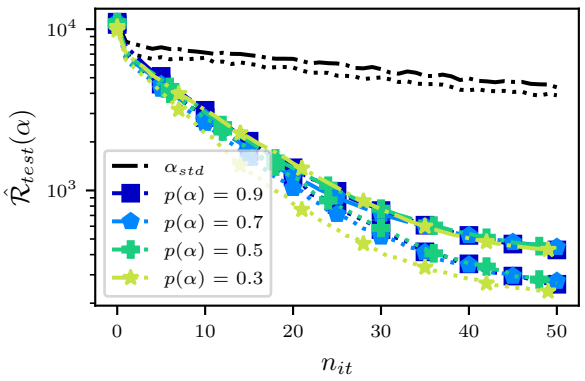

Figure 2: Test loss over the iterations: The black lines are for the standard choices of the hyperparameters. The empirical mean is given by the dashed line, and the median by the dotted one. The other lines show the test loss for $p(\alpha) \in \{0.9, 0.7, 0.5, 0.3\}$. By excluding the worst-case, one can accelerate the optimization procedure significantly.

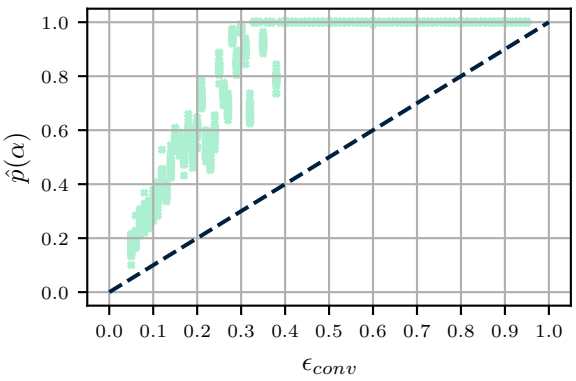

Figure 3: Empirical convergence probability: The dashed diagonal indicates the user-specified convergence probability. Each cross represents the empirical convergence probability on a separate test set. All empirical convergence probabilities lie well above the diagonal, i.e., the algorithm indeed ensures the user-specified convergence probability.

tions. In the case of Gradient Descent we learn the (constant) step-size, and in the case of heavy-ball we learn the step-size and the extrapolation parameter (both constant). Note that all results are created with a single sample from the posterior and do not show the expected value under the posterior. The experiments are a proof-of-concept for our theory in an easily controllable setting. Actually, our theory does not require a quadratic, in fact not even convex, loss function. More details about the learning procedure are given in the supplementary material.

7.1 Convergence of the Posterior

The first experiment considers the posterior distribution over the step-size parameter of Gradient Descent. The parameter $\mathfrak{S}$ is given by the right-hand side b of the quadratic problem, i.e., all problems have the same strong convexity parameter μ and the same smoothness parameter L (smallest and largest eigenvalue of $A^T A$). We use $N_{train} = 200$ and build the exponential family with η and T from Section 6.1, i.e., convergence is guaranteed during learning. We are interested in how the posterior distribution evolves for an increasing number of iterations of the algorithm. Since it is known that $\alpha_{std} = \frac{2}{L+\mu}$ yields the optimal rate in the worst-case (Nesterov, 2018), one would expect that the posterior puts increasingly more mass onto step-sizes close to α_{std}. Figure 1 confirms this intuition. Also, it shows that Assumption 3 prohibits step-sizes larger than $\frac{2}{L}$, which could lead to divergence easily.

7.2 Conditioning on Convergence

Here, the parameters $\mathfrak{S}$ of ℓ are given by the quadratic matrix and the right-hand side, i.e., the problems have a differing strong convexity parameter μ and smoothness parameter L. We sample these from a uniform distribution over $[\mu_-, \mu_+]$ and $[L_-, L_+]$. This simulates a situation where these parameters can only be estimated roughly. We use the Heavy-ball method for 50 iterations. The standard choice for the hyperparameters are given by $\tau_{std} = \left(\frac{2}{\sqrt{L_+}+\sqrt{\mu_-}}\right)^2$ and $\beta_{std} = \left(\frac{\sqrt{L_+}-\sqrt{\mu_-}}{\sqrt{L_+}+\sqrt{\mu_-}}\right)^2$ (Nesterov, 2018). We use $N_{prior} = 100$, $N_{train} = 100$ and $N_{test} = 200$. Figure 2 shows the convergence behaviour for different convergence guarantees. As one can see, excluding the worst-case ($\epsilon_{conv} \geq 0.9$) leads to a significantly better convergence result. However, a further decrease of the convergence guarantee does not lead to a further acceleration. This does not match the expected behaviour, yet is explained by the next experiment.

7.3 Ensuring a Certain Convergence Probability

We use the same setup as in Section 7.2 and investigate the empirical convergence probability on several test sets. We use $N_{prior} = 100$, $N_{train} = 100$ and 25 test sets of size $N_{test} = 250$ per user-specified convergence probability to estimate the true convergence probability of the algorithm. Note that we use the same datasets for all different convergence probabilities, i.e., we create them beforehand. We use the standard estimator for binomial distributions as empirical estimate for the convergence probability, i.e., $\hat{p}(\alpha) = \frac{N_{conv}}{N_{test}}$. Figure 3 shows the result of this experiment: All empirical convergence probabilities lie well

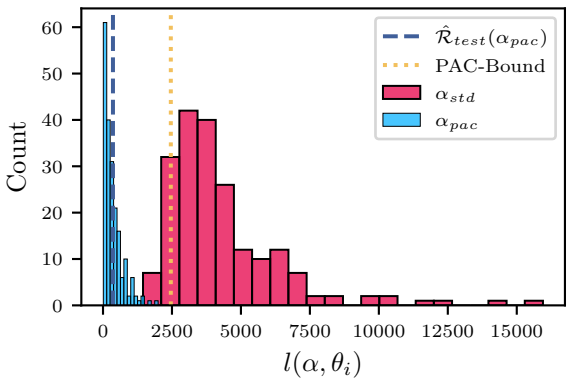

Figure 4: Test loss as histogram: The blue thin bars represent the learned hyperparameters and the red thick bars the ones from a worst-case analysis. The vertical lines represent the empirical mean for the learned hyperparameters (blue dashed) and the corresponding PAC-bound (orange dotted). The learned hyperparameters clearly outperform the standard ones, yet the PAC-bound is not perfectly tight.

above the diagonal, i.e., the algorithm indeed ensures the user-specified convergence probability. However, one can also see that it clearly favors a higher convergence probability than necessary, which can hinder the performance and explains the somewhat unexpected behaviour in the previous experiment. As indicated by the theory, this behaviour is probably due to our construction of the prior distribution.

7.4 Evaluation of the PAC-Bound

This experiment looks at the tightness of the PAC-bound. We adopt the setting from Section 7.2. Based on the previous experiment, we choose $\epsilon_{conv} = 0.9$ as convergence guarantee. Further, we use $N_{prior} = 200$, $N_{train} = 1000$ and $N_{test} = 200$. The training dataset is chosen larger than before, since the PAC-bound is not yet very tight for small datasets. Figure 4 shows the resulting losses on the test set as histogram plot, as well as the empirical mean and the PAC-Bayes bound. One can clearly see the improved performance of Heavy-ball with the learned hyperparameters. Further, one can see that the PAC-bound is not perfectly tight, however provides a good estimate of the true mean.

8 Conclusion

We presented a general PAC-Bayes theorem based on exponential families, which allows for a unified implementation of the learning framework. We applied this framework to the setting of learning-to-optimize and derived properties, under which the theorem is applicable to a given algorithm. Further, we provided a principled way to exclude unwanted cases by using conditional expectations and showed how to preserve necessary properties during learning. We believe that both approaches can be of great interest even aside the setting of learning-to-optimize. Finally, we provided a proof-of-concept of our theory on several experiments.

Limitations. We mainly see four limitations of our work: First, a theoretical guarantee to find the global minimum in (6) is still missing. Second, the construction of the prior is difficult and time-consuming. Third, we expect similar scaling problems for high-dimensional hyperparameters as with other probabilistic methods. And fourth, the trade-off between convergence speed and convergence probability is partly rather conservative. These problems are very related and will be addressed in future work.

Acknowledgements

We acknowledge funding by the German Research Foundation under Germany's Excellence Strategy – EXC number 2064/1 – 390727645 and the project DFG OC 150/5-1. Furthermore, we thank J. Fadili and one of the anonymous reviewers for an important hint for the extension of the union bound argument to compact sets.

References

Alquier, P. (2008). PAC-Bayesian bounds for randomized empirical risk minimizers. *Mathematical Methods of Statistics*, 17(4):279–304.

Alquier, P. (2021). User-friendly introduction to PAC-Bayes bounds. *arXiv preprint arXiv:2110.11216*.

Alquier, P. and Guedj, B. (2018). Simpler PAC-Bayesian bounds for hostile data. *Machine Learning*, 107(5):887–902.

Alquier, P., Ridgway, J., and Chopin, N. (2016). On the properties of variational approximations of Gibbs posteriors. *Journal of Machine Learning Research*, 17(1):8374–8414.

Barndorff–Nielsen, O. (2014). *Information and exponential families: in statistical theory*. John Wiley & Sons.

Bégin, L., Germain, P., Laviolette, F., and Roy, J.-F. (2016). PAC-Bayesian bounds based on the Rényi divergence. In *Artificial Intelligence and Statistics*, pages 435–444. PMLR.

Bianchi, P., Hachem, W., and Schechtman, S. (2022). Convergence of constant step stochastic gradient descent for non-smooth non-convex functions. *Set-Valued and Variational Analysis*, pages 1–31.

Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311.

Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press.

Bousquet, O. and Elisseeff, A. (2000). Algorithmic stability and generalization performance. *Advances in Neural Information Processing Systems*, 13.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, 2:499–526.

Buzzard, G. T., Chan, S. H., Sreehari, S., and Bouman, C. A. (2018). Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium. *SIAM Journal on Imaging Sciences*, 11(3):2001–2020.

Catoni, O. (2004). *Statistical learning theory and stochastic optimization: Ecole d'Eté de Probabilités de Saint-Flour, XXXI-2001*, volume 1851. Springer Science & Business Media.

Catoni, O. (2007). PAC-Bayesian supervised classification: The thermodynamics of statistical learning. *Lecture Notes-Monograph Series*, 56:i–163.

Chan, S. H., Wang, X., and Elgendy, O. A. (2016). Plug-and-play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98.

Chen, T., Chen, X., Chen, W., Heaton, H., Liu, J., Wang, Z., and Yin, W. (2021). Learning to optimize: A primer and a benchmark. *arXiv preprint arXiv:2103.12828*.

Chen, X., Liu, J., Wang, Z., and Yin, W. (2018). Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds. *Advances in Neural Information Processing Systems*, 31.

Chen, X., Zhang, Y., Reisinger, C., and Song, L. (2020). Understanding deep architecture with reasoning layer. *Advances in Neural Information Processing Systems*, 33:1240–1252.

Cohen, R., Elad, M., and Milanfar, P. (2021). Regularization by denoising via fixed-point projection. *SIAM Journal on Imaging Sciences*, 14(3):1374–1406.

Davis, D. and Drusvyatskiy, D. (2019). Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239.

Davis, D. and Drusvyatskiy, D. (2022). Graphical convergence of subgradients in nonconvex optimization and learning. *Mathematics of Operations Research*, 47(1):209–231.

Défossez, A., Bottou, L., Bach, F., and Usunier, N. (2022). A simple convergence proof of adam and adagrad. *Transactions on Machine Learning Research*.

Donsker, M. D. and Varadhan, S. S. (1975). Asymptotic evaluation of certain Markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47.

Dziugaite, G. K., Hsu, K., Gharbieh, W., Arpino, G., and Roy, D. (2021). On the role of data in PAC-Bayes bounds. In *International Conference on Artificial Intelligence and Statistics*, pages 604–612. PMLR.

Dziugaite, G. K. and Roy, D. M. (2017). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press.

Dziugaite, G. K. and Roy, D. M. (2018). Data-dependent PAC-Bayes priors via differential privacy. *Advances in neural information processing systems*, 31.

Efron, B. (1975). Defining the curvature of a statistical problem (with applications to second order efficiency). *The Annals of Statistics*, pages 1189–1242.

Germain, P., Lacasse, A., Laviolette, F., and Marchand, M. (2009). PAC-Bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 353–360.

Gregor, K. and LeCun, Y. (2010). Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on international conference on machine learning*, pages 399–406.

Guedj, B. (2019). A primer on PAC-Bayesian learning. In *Proceedings of the second congress of the French Mathematical Society*, volume 33.

Honorio, J. and Jaakkola, T. (2014). Tight bounds for the expected risk of linear classifiers and PAC-Bayes finite-sample guarantees. In *Artificial Intelligence and Statistics*, pages 384–392. PMLR.

Kavis, A., Levy, K. Y., and Cevher, V. (2022). High probability bounds for a class of nonconvex algorithms with adagrad stepsize. In *International Conference on Learning Representations*.

Kobler, E., Effland, A., Kunisch, K., and Pock, T. (2020). Total deep variation: A stable regularizer for inverse problems. *arXiv preprint arXiv:2006.08789*.

Langford, J. and Caruana, R. (2001). (Not) bounding the true error. In Dietterich, T., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Langford, J. and Shawe-Taylor, J. (2002). PAC-Bayes and margins. *Advances in neural information processing systems*, 15.

Lever, G., Laviolette, F., and Shawe-Taylor, J. (2013). Tighter PAC-Bayes bounds through distribution-dependent priors. *Theoretical Computer Science*, 473:4–28.

London, B. (2017). A PAC-Bayesian analysis of randomized learning with application to stochastic gradient descent. *Advances in Neural Information Processing Systems*, 30.

McAllester, D. (2003a). PAC-Bayesian stochastic model selection. *Machine Learning*, 51(1):5–21.

McAllester, D. (2003b). Simplified PAC-Bayesian margin bounds. In *Learning theory and Kernel machines*, pages 203–215. Springer.

Moeller, M., Mollenhoff, T., and Cremers, D. (2019). Controlling neural networks via energy dissipation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3256–3265.

Nesterov, Y. (2018). *Lectures on convex optimization*, volume 137. Springer.

Ohnishi, Y. and Honorio, J. (2021). Novel change of measure inequalities with applications to PAC-Bayesian bounds and Monte Carlo estimation. In *International Conference on Artificial Intelligence and Statistics*, pages 1711–1719. PMLR.

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

Pérez-Ortiz, M., Rivasplata, O., Shawe-Taylor, J., and Szepesvári, C. (2021). Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22(227):1–40.

Rivasplata, O., Kuzborskij, I., Szepesvári, C., and Shawe-Taylor, J. (2020). PAC-Bayes analysis beyond the usual bounds. *Advances in Neural Information Processing Systems*, 33:16833–16845.

Ryu, E., Liu, J., Wang, S., Chen, X., Wang, Z., and Yin, W. (2019). Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. PMLR.

Seeger, M. (2002). PAC-Bayesian generalisation error bounds for Gaussian process classification. *Journal of Machine Learning Research*, 3:233–269.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. (2009). Stochastic convex optimization. In *COLT*, volume 2, page 5.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. (2010). Learnability, stability and uniform convergence. *Journal of Machine Learning Research*, 11:2635–2670.

Sreehari, S., Venkatakrishnan, S. V., Wohlberg, B., Buzzard, G. T., Drummy, L. F., Simmons, J. P., and Bouman, C. A. (2016). Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423.

Sun, Y., Wohlberg, B., and Kamilov, U. S. (2019). An online plug-and-play algorithm for regularized image reconstruction. *IEEE Transactions on Computational Imaging*, 5(3):395–408.

Teodoro, A. M., Bioucas-Dias, J. M., and Figueiredo, M. A. (2017). Scene-adapted plug-and-play algorithm with convergence guarantees. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE.

Terris, M., Repetti, A., Pesquet, J.-C., and Wiaux, Y. (2021). Enhanced convergent pnp algorithms for image restoration. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 1684–1688. IEEE.

Thiemann, N., Igel, C., Wintenberger, O., and Seldin, Y. (2017). A strongly quasiconvex PAC-Bayesian bound. In *International Conference on Algorithmic Learning Theory*, pages 466–492. PMLR.

Tirer, T. and Giryes, R. (2018). Image restoration by iterative denoising and backward projections. *IEEE Transactions on Image Processing*, 28(3):1220–1234.

Witting, H. (2013). *Mathematische Statistik I: Parametrische Verfahren bei festem Stichprobenumfang*. Springer-Verlag.

Xin, B., Wang, Y., Gao, W., Wipf, D., and Wang, B. (2016). Maximal sparsity with deep networks? *Advances in Neural Information Processing Systems*, 29.

A MISSING PROOFS

In this section, we collect all the missing proofs from the main text.

A.1 Proof of Lemma 5.1

Proof. Recall that $\kappa(\lambda) = \log\Big(\int h \exp\Big(\langle \eta(\lambda), T\rangle\Big) d\mathbb{P}_{\mathfrak{H}}\Big)$ and $A(\lambda) = c(\lambda)^{-1}$. We have to show:

1) $\kappa(\lambda) = \sup_{\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}} \mathbb{E}_{\mathbb{Q}}\big[\langle \eta(\lambda), T\rangle + \log(h)\big] - D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}})$.

2) The supremum is attained at $\mathbb{Q}_\lambda$.

For this, we first show $\kappa(\lambda) \geq \mathbb{E}_{\mathbb{Q}}\big[\langle \eta(\lambda), T\rangle + \log(h)\big] - D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}})$ for an arbitrary $\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}$ and then equality for $\mathbb{Q}_\lambda$. Thus, let $\mathbb{Q} \ll \mathbb{P}_{\mathfrak{H}}$ and denote by $\frac{d\mathbb{Q}}{d\mathbb{P}_{\mathfrak{H}}}$ its Radon-Nikodym derivative w.r.t. $\mathbb{P}_{\mathfrak{H}}$. Then it holds:

$$\begin{aligned}\mathbb{E}_{\mathbb{Q}}\Big[\langle \eta(\lambda), T\rangle + \log(h)\Big] - D_{KL}(\mathbb{Q} \,\|\, \mathbb{P}_{\mathfrak{H}}) &= \int \langle \eta(\lambda), T\rangle + \log(h) - \log\Big(\frac{d\mathbb{Q}}{d\mathbb{P}_{\mathfrak{H}}}\Big)\, d\mathbb{Q} \\ &= \int \log\Big(\frac{h}{\frac{d\mathbb{Q}}{d\mathbb{P}_{\mathfrak{H}}}} \exp\Big(\langle \eta(\lambda), T\rangle\Big)\Big)\, d\mathbb{Q}\,.\end{aligned}$$

Since the logarithm is concave, by Jensen's inequality this can be bounded by:

$$\begin{aligned}&\leq \log\Big(\int \frac{h}{\frac{d\mathbb{Q}}{d\mathbb{P}_{\mathfrak{H}}}} \exp\Big(\langle \eta(\lambda), T\rangle\Big)\, d\mathbb{Q}\Big) \\ &= \log\Big(\int h \exp\Big(\langle \eta(\lambda), T\rangle\Big)\, d\mathbb{P}_{\mathfrak{H}}\Big) \\ &= \kappa(\lambda)\,.\end{aligned}$$

It remains to show the equality for $\mathbb{Q}_\lambda$:

$$\begin{aligned}D_{KL}(\mathbb{Q}_\lambda \,\|\, \mathbb{P}_{\mathfrak{H}}) &= \int \log\Big(\frac{d\mathbb{Q}_\lambda}{d\mathbb{P}_{\mathfrak{H}}}\Big)\, d\mathbb{Q}_\lambda \\ &= \int \log\Big(hA(\lambda)\exp(\langle \eta(\lambda), T\rangle)\Big)\, d\mathbb{Q}_\lambda \\ &= \int \log(h) + \langle \eta(\lambda), T\rangle\, d\mathbb{Q}_\lambda + \log(A(\lambda)) \\ &= \mathbb{E}_{\mathbb{Q}_\lambda}\Big[\log(h) + \langle \eta(\lambda), T\rangle\Big] - \log(c(\lambda)) \\ &= \mathbb{E}_{\mathbb{Q}_\lambda}\Big[\log(h) + \langle \eta(\lambda), T\rangle\Big] - \kappa(\lambda),\end{aligned}$$

which yields:

$$\kappa(\lambda) = \mathbb{E}_{\mathbb{Q}_\lambda}\Big[\log(h) + \langle \eta(\lambda), T\rangle\Big] - D_{KL}(\mathbb{Q}_\lambda \,\|\, \mathbb{P}_{\mathfrak{H}})\,.$$

Thus, the supremum is attained at $\mathbb{Q}_\lambda$. This concludes the proof. □

A.2 Proof of Theorem 5.2

Proof. We will use $c(\lambda)$ and $\kappa(\lambda)$ as a short-hand for $c(\lambda, \mathfrak{D}_N)$ and $\kappa(\lambda, \mathfrak{D}_N)$ respectively. $c(\lambda)$ is a non-negative random variable and $\exp$ is a monotonically increasing function. Thus, since $\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)] \leq 1$ for all $\lambda \in \Lambda$, one gets for every $\lambda \in \Lambda$ from Markov's inequality for every $s \in \mathbb{R}$:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{c(\lambda) > \exp(s)\Big\} \leq \frac{\mathbb{E}_{\mathfrak{D}_N}[c(\lambda)]}{\exp(s)} \leq \exp(-s)\,.$$

Since $c(\lambda) > \exp(s) \;\Leftrightarrow\; \kappa(\lambda) = \log(c(\lambda)) > s$, this is the same as:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\kappa(\lambda) > s\Big\} \leq \exp(-s)\,.$$

This implies by the union-bound argument, that:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\,\kappa(\lambda) > s\Big\} = \mathbb{P}_{\mathfrak{D}_N}\Big\{\bigcup_{\lambda\in\Lambda}\{\kappa(\lambda) > s\}\Big\} \leq \sum_{\lambda\in\Lambda}\mathbb{P}_{\mathfrak{D}_N}\Big\{\kappa(\lambda) > s\Big\} \leq |\Lambda|\exp(-s)\,.$$

Inserting $s = \log\Big(\frac{|\Lambda|}{\epsilon}\Big)$ gives:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\,\kappa(\lambda) > \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big\} \leq \epsilon\,.$$

Hence, the complementary event yields:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\kappa(\lambda) \leq \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big\} \geq 1-\epsilon\,.$$

Using $\kappa(\lambda) = \sup_{\mathbb{Q}\ll\mathbb{P}_{\mathfrak{H}}}\mathbb{E}_{\mathbb{Q}}\big[\langle\eta(\lambda),T\rangle + \log(h)\big] - D_{KL}(\mathbb{Q}\,\|\,\mathbb{P}_{\mathfrak{H}})$ then gives:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\,\sup_{\mathbb{Q}\ll\mathbb{P}_{\mathfrak{H}}}\mathbb{E}_{\mathbb{Q}}[\log(h)+\langle\eta(\lambda),T\rangle] - D_{KL}(\mathbb{Q}\,\|\,\mathbb{P}_{\mathfrak{H}}) \leq \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big\} \geq 1-\epsilon\,.$$

Rearranging and reformulating then yields the result:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\forall\lambda\in\Lambda,\;\forall\mathbb{Q}\ll\mathbb{P}_{\mathfrak{H}}\;:\;\mathbb{E}_{\mathbb{Q}}[\log(h)+\langle\eta(\lambda),T\rangle] \leq D_{KL}(\mathbb{Q}\,\|\,\mathbb{P}_{\mathfrak{H}}) + \log\Big(\frac{|\Lambda|}{\epsilon}\Big)\Big\} \geq 1-\epsilon\,.$$

□

A.3 Proof of Theorem 6.2

Proof. We use the following short-hand notation:

$$\mathfrak{L}(\alpha) := \ell(\mathcal{A}(\alpha,\mathfrak{S}),\mathfrak{S}),\;\;\mathfrak{L}_i(\alpha) := \ell(\mathcal{A}(\alpha,\mathfrak{S}_i),\mathfrak{S}_i),\;\;\mathfrak{L}_0 := \ell(x^{(0)},\mathfrak{S})\,.$$

By the i.i.d. assumption, one can write for every fixed $\alpha\in\mathcal{H}$:

$$\begin{aligned}\mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(\lambda(\mathcal{R}(\alpha)-\hat{\mathcal{R}}(\alpha,\mathfrak{D}_N))\Big)\Big] &= \mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(-\frac{\lambda}{N}\sum_{i=1}^{N}\Big(\mathfrak{L}_i - \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}]\Big)\Big)\Big]\\ &= \mathbb{E}_{\mathfrak{D}_N}\Big[\prod_{i=1}^{N}\exp\Big(-\frac{\lambda}{N}(\mathfrak{L}_i - \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}])\Big)\Big] \overset{iid}{=} \prod_{i=1}^{N}\mathbb{E}_{\mathfrak{S}}\Big[\exp\Big(-\frac{\lambda}{N}\Big(\mathfrak{L} - \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}]\Big)\Big)\Big]\,.\end{aligned}$$

Since the loss-function is non-negative and $\mathcal{A}$ satisfies the convergence property, one gets that $\mathfrak{L}$ is a non-negative random variable with finite second-moment:

$$\begin{aligned}\mathbb{E}_{\mathfrak{S}}[\mathfrak{L}^2] &= \mathbb{E}_{\mathfrak{S}}\Big[\ell\big(\mathcal{A}(\alpha,\mathfrak{S}),\mathfrak{S}\big)^2\Big]\\ &\leq C^2\rho(\alpha)^2\mathbb{E}_{\mathfrak{S}}\Big[\ell\big(x^{(0)},\mathfrak{S}\big)^2\Big] = C^2\rho(\alpha)^2\mathbb{E}_{\mathfrak{S}}[\mathfrak{L}_0^2]\,.\end{aligned}$$

Thus, by lemma 3.9, one gets the following bound:

$$\leq \prod_{i=1}^{N}\exp\Big(\frac{\lambda^2}{2N^2}\mathbb{E}_{\mathfrak{S}}[\mathfrak{L}^2]\Big) = \exp\Big(\frac{\lambda^2}{2N}\mathbb{E}_{\mathfrak{S}}[\mathfrak{L}^2]\Big)\,.$$

Since the exponential function is monotonically increasing, by the convergence property this can again be bounded by:

$$\leq \exp\Big(\frac{\lambda^2}{2N} C^2 \rho(\alpha)^2 \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}_0^2]\Big) .$$

Thus, for any $\alpha \in \mathcal{H}$ one arrives at the following inequality:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(\lambda(\mathcal{R}(\alpha) - \hat{\mathcal{R}}(\alpha, \mathfrak{D}_N))\Big)\Big] \leq \exp\Big(\frac{\lambda^2}{2N} C^2 \rho(\alpha)^2 \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}_0^2]\Big) .$$

Since the right-hand side is a constant w.r.t. $\mathbb{P}_{\mathfrak{D}_N}$, rearranging terms gives:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(\lambda(\mathcal{R}(\alpha) - \hat{\mathcal{R}}(\alpha, \mathfrak{D}_N)) - \frac{\lambda^2}{2}\frac{C^2}{N} \rho(\alpha)^2 \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}_0^2]\Big)\Big] \leq 1 .$$

By integrating both sides with respect to $\mathbb{P}_{\mathfrak{H}}$ and using Fubini's theorem (note that this is possible, since $\mathbb{P}_{\mathfrak{H}}$ is independent of $\mathfrak{D}_N$), one gets:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\mathbb{E}_{\mathfrak{H}}\Big[\exp\Big(\lambda(\mathcal{R}(\mathfrak{H}) - \hat{\mathcal{R}}(\mathfrak{H}, \mathfrak{D}_N))\Big) - \frac{\lambda^2}{2}\frac{C^2}{N} \rho(\mathfrak{H})^2 \mathbb{E}_{\mathfrak{S}}[\mathfrak{L}_0^2]\Big]\Big] \leq 1 .$$

Inserting the definition of η and T now gives:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\mathbb{E}_{\mathfrak{H}}\Big[\exp\Big(\langle \eta(\lambda), T(\mathfrak{H}, \mathfrak{D}_N)\rangle\Big)\Big]\Big] \leq 1 .$$

By definition of the Laplace transform, this is the same as:

$$\mathbb{E}_{\mathfrak{D}_N}[c(\lambda, \mathfrak{D}_N)] \leq 1 .$$

□

A.4 Proof of Theorem 6.7

Proof. The proof is very similar to the proof of lemma 5.1 and basically follows the same line of argumentation. We use $\ell_c(\alpha, \theta) := \mathbb{1}_{C(\alpha)}(\theta)\ell(\alpha, \theta)$ as short-hand and call this the convergence loss. First, consider $\alpha \in \mathcal{H}$ fixed with $p(\alpha) > 0$. Then it holds:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\exp(\lambda(\mathcal{R}_c(\alpha) - \hat{\mathcal{R}}_c(\alpha, \mathfrak{D}_N)))\Big] = \mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(-\frac{\lambda}{Np(\alpha)} \sum_{i=1}^{N}\Big(\ell_c(\alpha, \mathfrak{S}_i) - \mathbb{E}_{\mathfrak{S}}[\ell_c(\alpha, \mathfrak{S})]\Big)\Big)\Big]$$

$$= \mathbb{E}_{\mathfrak{D}_N}\Big[\prod_{i=1}^{N} \exp\Big(-\frac{\lambda}{Np(\alpha)}\Big(\ell_c(\alpha, \mathfrak{S}_i) - \mathbb{E}_{\mathfrak{S}}[\ell_c(\alpha, \mathfrak{S})]\Big)\Big)\Big] .$$

Since $\mathfrak{D}_N$ is assumed to be i.i.d., this is the same as:

$$= \prod_{i=1}^{N} \mathbb{E}_{\mathfrak{S}}\Big[\exp\Big(-\frac{\lambda}{Np(\alpha)}\Big(\ell_c(\alpha, \mathfrak{S}) - \mathbb{E}_{\mathfrak{S}}[\ell_c(\alpha, \mathfrak{S})]\Big)\Big)\Big] .$$

Since the convergence loss is non-negative and has a finite second-moment (since $\mathbb{E}_{\mathfrak{S}}[\ell_c(\alpha, \mathfrak{S})^2] \leq \mathbb{E}_{\mathfrak{S}}[\ell(x^{(0)}, \mathfrak{S})^2] < \infty$), by lemma 3.14 this can be bounded by:

$$\leq \prod_{i=1}^{N} \exp\Big(\frac{\lambda^2}{2N^2 p(\alpha)^2} \mathbb{E}_{\mathfrak{S}}\Big[\ell_c(\alpha, \mathfrak{S})^2\Big]\Big) = \exp\Big(\frac{\lambda^2}{2Np(\alpha)^2} \mathbb{E}_{\mathfrak{S}}\Big[\ell_c(\alpha, \mathfrak{S})^2\Big]\Big) .$$

By definition of the convergence set, this can in turn be bounded by:

$$\leq \exp\Big(\frac{\lambda^2}{2Np(\alpha)^2} \mathbb{E}_{\mathfrak{S}}\Big[\mathbb{1}_{C(\mathfrak{S})}\ell(x^{(0)}, \mathfrak{S})^2\Big]\Big) \leq \exp\Big(\frac{\lambda^2}{2Np(\alpha)^2} \mathbb{E}_{\mathfrak{S}}\Big[\ell(x^{(0)}, \mathfrak{S})^2\Big]\Big) .$$

Thus, one gets $\mathbb{P}_{\mathfrak{H}}$-a.s.:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\exp(\lambda(\mathcal{R}_c(\alpha) - \hat{\mathcal{R}}_c(\alpha, \mathfrak{D}_N)))\Big] \leq \exp\Big(\frac{\lambda^2}{2Np(\alpha)^2}\mathbb{E}_{\mathfrak{S}}\Big[\ell(x^{(0)}, \mathfrak{S})^2\Big]\Big)\,.$$

Since the right-hand side is independent of $\mathfrak{D}_N$, this is equivalent to:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\exp\Big(\lambda(\mathcal{R}_c(\alpha) - \hat{\mathcal{R}}_c(\alpha, \mathfrak{D}_N)) - \frac{\lambda^2}{2Np(\alpha)^2}\mathbb{E}_{\mathfrak{S}}\Big[\ell(x^{(0)}, \mathfrak{S})^2\Big]\Big)\Big] \leq 1\,.$$

Since $\mathbb{P}_{\mathfrak{H}}\Big\{p(\mathfrak{H}) > 0\Big\} = 1$, one can integrate both sides w.r.t. $\mathbb{P}_{\mathfrak{H}}$. Furthermore, since $\mathbb{P}_{\mathfrak{H}}$ is independent of $\mathfrak{D}_N$, one can use Fubini's theorem to get:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\mathbb{E}_{\mathfrak{H}}\Big[\exp\Big(\lambda(\mathcal{R}_c(\mathfrak{H}) - \hat{\mathcal{R}}_c(\mathfrak{H}, \mathfrak{D}_N)) - \frac{\lambda^2}{2Np(\alpha)^2}\mathbb{E}_{\mathfrak{S}}\Big[\ell(x^{(0)}, \mathfrak{S})^2\Big]\Big)\Big]\Big] \leq 1\,.$$

Using the definition of η and T, this is the same as:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[\mathbb{E}_{\mathfrak{H}}\Big[\exp(\langle \eta(\lambda), T(\mathfrak{H}, \mathfrak{D}_N)\rangle)\Big]\Big] \leq 1,$$

which is equivalent to:

$$\mathbb{E}_{\mathfrak{D}_N}\Big[c(\lambda, \mathfrak{D}_N)\Big] \leq 1\,.$$

□

B Description of the Learning Procedure

In this section we provide further details about the implementation.

B.1 General Setup

We use $n = 50$ as dimension of the optimization problem, 50 iterations of the algorithm, $x^{(0)} = 0 \in \mathbb{R}^n$ as initialization and $\epsilon = 0.01$ as threshold in the PAC-bound. For every implementation of the parametric optimization algorithm $\mathcal{A}$, we specify all (learnable) hyperparameters in a named dictionary, such that we can match all involved quantities like corresponding priors during learning by their unique names. Furthermore, since we perform first-order gradient-based optimization, we implement every algorithm in the form $\mathcal{A}(x^{(0)}, \theta, \nabla_x \ell, \alpha, n_{it})$, where $\nabla_x \ell$ denotes the gradient of ℓ w.r.t. x as function of θ. Through this, the following learning procedure can be applied to all tested algorithms in the same way.

B.2 Creation of the Parametric Problems

Fixed strong convexity and smoothness parameters. We create the matrix $A \in \mathbb{R}^{n \times n}$ randomly, where every entry is created by sampling an integer in $\{-10, ..., 10\}$ uniformly at random and then adding noise from a standard normal distribution. This matrix is fixed across the different instances of the problem, such that all problems have the same strong convexity and smoothness parameter. For the right-hand side $b \in \mathbb{R}^n$, which in this case is the only parameter of the parametric optimization problem, we first create a mean m and a covariance matrix Σ by sampling every entry uniformly at random in $\{-5, ..., 5\}$ (and updating $\Sigma \leftarrow \Sigma^T \Sigma$ to make it positive definite), and then we sample $N_{prior} + N_{train} + N_{test}$ right-hand sides from the multivariate normal distribution $\mathcal{N}(m, \Sigma)$.

Varying strong convexity and smoothness parameters. The creation of the right-hand sides is the same as in the previous paragraph. Thus, we will only describe the creation of the matrices A, which define the strong convexity parameter μ and smoothness parameter L. First of all, we restrict to a diagonal matrix. Further, since we found the strong convexity parameter μ to have only a negligible influence in previous experiments (if the problem is not generally well-conditioned, in which case one would not have to learn anything), we fix it (typically $\mu = 0.05$) and only vary the smoothness parameter L. First, we sample $N_{prior} + N_{train} + N_{test}$ smoothness parameters uniformly at random in $[1, 5000]$. Then, for each smoothness parameter we create the matrix A by linearly interpolating between $\sqrt{\mu}$ and $\sqrt{L}$ and inserting these elements (randomly permutated) into the diagonal of A.

B.3 Learning Procedure

At first, we setup the sufficient statistics T and the natural parameters η as functions in α and λ, which can be called during training. We hand these, together with the specified priors, over to the general implementation of the learning procedure, which performs the following steps:

i) First, we create samples from the initial prior (depending on the experiment between 50 and 500).

ii) Then we evaluate the sufficient statistics T on these samples and find $\arg\min_{\lambda \in \Lambda} F(\lambda)$ by a simple grid search. *For this we use a linear grid* Λ *over* $(0, 1]$ *with 25000 entries (note that this corresponds to* $\log(|\Lambda|) \approx 10$ *and has, compared to solving the minimization problems during learning, a negligible computational cost).* Note that this also directly yields the PAC-bound.

iii) Then, we calculate the posterior density on these samples through the formula for the Gibbs posterior, i.e., if f denotes the density of the prior (w.r.t. the Lebesgue measure), we calculate $f(\alpha_i) \frac{\exp(\langle \eta(\lambda), T(\alpha_i) \rangle)}{\mathbb{E}_{\mathfrak{H}}[\exp(\langle \eta(\lambda), T \rangle)]}$ for every sample α_i. Here we use the empirical mean as approximation for the integral.

iv) Finally, we normalize the resulting values, such that we have a distribution over these samples.

Through this, we effectively build a discrete distribution. For visualization purposes, we take a single instance (the argmax from the discrete posterior) as learned hyperparameter.

B.4 Construction of the Prior

If we actively construct the prior for a given hyperparameter (instead of using a fixed one as in the first experiment), we do this in an iterative fashion (typically two iterations) on a separate dataset:

i) First, since we assume that we have access to the standard choice of the hyperparameters α_{std}, we put a uniform prior around α_{std}, i.e., $\mathcal{U}[C_1, C_2]$, where $C_1 < \alpha_{std}$ and $C_2 > \alpha_{std}$ depend on the user-specified convergence probability, i.e., they are chosen more "aggressive", if a smaller convergence guarantee has to be satisfied. *In our experiments, we actually used* $C_1 = \frac{0.5}{\epsilon_{conv}} \frac{2}{L_{max}}$ *and* $C_2 = \frac{3}{\epsilon_{conv}} \frac{2}{L_{max}}$ *for the step-size parameter and* $C_1 = \frac{1}{2}\beta_{std}$ *and* $C_2 = 2\beta_{std}$ *for the extrapolation parameter. Here* ϵ_{conv} *denotes the user-specified convergence probability. Initially, we started also with more "aggressive" values for the extrapolation parameter depending on the convergence probability. However, we found that the learned values almost exclusively ended up in that range, such that we directly restricted it.*

ii) Then we run the learning procedure with this prior dataset. As described above, this yields a discrete distribution over some samples from the initial prior. From these samples, we retain only those that satisfy the user-specified convergence probability (see Section B.5) and, if these are many, only those with highest posterior density.

iii) Then we build a new uniform distribution $\mathcal{U}[a, b]$ as initial distribution for the next iteration (i.e., start from ii) again). For this, we use the standard estimators for a and b, i.e., $\min$ and $\max$ over the remaining samples.

Note that this procedure is contractive, i.e., it does not yield a distribution that puts mass outside the very first initial distribution.

B.5 Ensuring a Certain Convergence Probability

As described above and in the main text, ensuring the convergence probability is part of the construction of the prior. For this, we simply split the prior data set into two parts of size $N_{prior,1}$ and $N_{prior,2}$ (typically $N_{prior,1} \approx N_{prior,2}$). The first one is used in the learning procedure in the construction of the prior as described above in Section B.4, and the second one is used as a separate test set to check for the convergence probability. Here we use the standard estimator for the binomial distribution $\hat{p}_{conv}(\alpha) = \frac{N_{conv}}{N_{prior,2}}$. Based on this estimate, we only keep those samples in Section B.4 that satisfy the user-specified convergence probability during the construction of the prior. Hence, since the construction of the prior is contractive (as described in Section B.4), this constrains the prior to only put mass on regions that satisfy the convergence guarantee. However, as seen in the experiments, it is also partly rather conservative.

C Further Remarks & Definitions

This section provides further details on statements made throughout the paper for which no proof has yet been provided. Furthermore, we provide a few (standard) definitions that were used in the main text, but might not be known by every reader.

C.1 Further Definitions

The following two definitions are needed in Definition 3.3 of a data-dependent distribution.

Definition C.1 (Polish Space)**.** *A topological space $\mathcal{X}$ that is separable, i.e. it has a countable dense subset, and completely metrizable, i.e. there is a complete metric that generates the topology, is called a Polish space.*

Definition C.2 (Markov Kernel)**.** *Let $(\Omega_1, \mathcal{A}_1)$, $(\Omega_2, \mathcal{A}_2)$ be measurable spaces. A map*

$$\kappa : \Omega_1 \times \mathcal{A}_2 \to [0, 1]$$

is called a Markov kernel, if:

i) For every $\omega_1 \in \Omega_1$, the map $A_2 \mapsto \kappa(\omega_1, A_2)$ is a probability measure on $\mathcal{A}_2$.

ii) For every $A_2 \in \mathcal{A}_2$, the map $\omega_1 \to \kappa(\omega_1, A_2)$ is measurable.

C.2 On the Measurability Assumption of $p(\alpha)$

Lemma C.3. *Let $\big(\Omega, \mathcal{F}, \mathbb{P}\big)$ be a probability space and $\mathfrak{H} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \to \mathcal{H}$, $\mathfrak{S} : \big(\Omega, \mathcal{F}, \mathbb{P}\big) \to \Theta$ be random variables. Assume that Θ is a Polish space and that $\mathfrak{S}$ and $\mathfrak{H}$ are independent. Then there is a measurable function $p' : \mathcal{H} \to [0, 1]$, such that it holds:*

$$p' \circ \mathfrak{H} = p \circ \mathfrak{H} \qquad \mathbb{P}_{\mathfrak{H}} - a.s.$$

Proof. Denote by:

$$g(\alpha, \theta) := \ell(\mathcal{A}(\alpha, \theta), \theta) - \ell(x^{(0)}, \theta)\,.$$

Then, by definition, it holds for $\alpha \in \mathcal{H}$:

$$p(\alpha) = \mathbb{P}_{\mathfrak{S}}\{g_\alpha(\mathfrak{S}) \leq 0\} = \int_\Theta \mathbb{1}_{(-\infty,0]}(g_\alpha(\theta))\, \mathbb{P}_{\mathfrak{S}}(d\alpha)\,,$$

where $g_\alpha(\theta) : \Theta \to \mathbb{R},\ \theta \mapsto g(\alpha, \theta)$ denotes g with fixed argument $\alpha \in \mathcal{H}$. Since Θ is a Polish space, there exists a regular version of the conditional probability

$$(\alpha, B) \mapsto \mathbb{P}_{\mathfrak{S}|\mathfrak{H}=\alpha}(B)$$

of $\mathfrak{S}$ given $\mathfrak{H} = \alpha$. With this, for every measurable function $f : \mathcal{H} \times \Theta \to \mathbb{R}$, such that $\mathbb{E}[f(\mathfrak{H}, \mathfrak{S})]$ exists, it holds (see e.g. (Witting, 2013, p.124, Thm. 1.122)):

$$\mathbb{E}[f(\mathfrak{H}, \mathfrak{S}) \mid \mathfrak{H} = \alpha] = \int_\Theta f_\alpha(\theta)\, \mathbb{P}_{\mathfrak{S}|\mathfrak{H}=\alpha}(d\theta) \qquad \mathbb{P}_{\mathfrak{H}} - a.s.$$

By independence, one further gets that (see e.g. (Witting, 2013, p.126, Thm. 1.123)):

$$\mathbb{P}_{\mathfrak{S}|\mathfrak{H}=\alpha} = \mathbb{P}_{\mathfrak{S}} \qquad \mathbb{P}_{\mathfrak{H}} - a.s.$$

Hence, in total one gets $\mathbb{P}_{\mathfrak{H}}$-a.s.:

$$\begin{aligned}
\mathbb{P}\{g(\mathfrak{H}, \mathfrak{S}) \mid \mathfrak{H} = \alpha\} &= \int_\Theta \mathbb{1}_{(-\infty,0]}(g_\alpha(\theta))\, \mathbb{P}_{\mathfrak{S}|\mathfrak{H}=\alpha}(d\theta) \\
&= \int_\Theta \mathbb{1}_{(-\infty,0]}(g_\alpha(\theta))\, \mathbb{P}_{\mathfrak{S}}(d\theta) \\
&= p(\alpha)\,.
\end{aligned}$$

Since $\alpha \mapsto \mathbb{P}\{g(\mathfrak{H}, \mathfrak{S}) \mid \mathfrak{H} = \alpha\}$ is measurable by definition of regular conditional probabilities, the claim follows. □

C.3 On the Finiteness Assumption of Λ

We denote the open ball of radius r around a point x by $\mathcal{B}(x;r)$ and the corresponding closed ball by $\mathcal{B}[x;r]$. For a set S, the notation $|S|$ denotes the cardinality of S.

Definition C.4 (Totally Bounded Space)**.** *A metric space $(\mathcal{X},d)$ is called* totally bounded*, if for every $\epsilon>0$ there exists $n\in\mathbb{N}$, $x_1,...,x_n\in X$, such that*

$$\mathcal{X}\subseteq\bigcup_{i=1}^{n}\mathcal{B}(x_i;\epsilon)\,.$$

The typical example of a totally bounded space is a compact space. The important property of this space, which is used in the following, is that they have a finite covering number. In the end, this allows again to apply the union bound argument.

Definition C.5 (δ-Covering Number)**.** *Let $(\mathcal{X},d)$ be a totally bounded metric space and let $\delta>0$. A* proper δ-covering *of $\mathcal{X}$ is a finite set $X_\delta\subset\mathcal{X}$, such that*

$$\mathcal{X}\subseteq\bigcup_{x\in X_\delta}\mathcal{B}[x;\delta]\,.$$

The minimal cardinality of any δ-covering is denoted $\mathcal{N}_c(\delta,\mathcal{X})$ and is called the δ-covering number *of $\mathcal{X}$:*

$$\mathcal{N}_c(\delta,\mathcal{X}):=\min\{|X_\delta|\ :\ X_\delta \text{ is a proper } \delta\text{-covering of } \mathcal{X}\}\,.$$

Taken together, and using the proof of Theorem 5.2 as entry point, one gets the following Lemma. This is a direct generalization of the result in Theorem 5.2, as in the case where Λ is finite, Λ can be covered by itself, such that it holds $\mathcal{N}_c(\delta,\Lambda)=|\Lambda|$ and $C=0$.

Lemma C.6. *Let (Λ,d) be a totally bounded metric space and let $\delta>0$. Assume that there is a constant $C:=C(\delta)$, such that for all $\tilde{\lambda}\in\Lambda$ it holds:*

$$\sup_{\lambda\in\mathcal{B}[\tilde{\lambda},\delta]}\kappa(\lambda)-\kappa(\tilde{\lambda})\leq C\,.$$

Finally, assume that $\mathbb{P}_{\mathfrak{D}_N}\{\kappa(\lambda)>s\}\leq\exp(-s)$ for all $s\in\mathbb{R}$, $\lambda\in\Lambda$. Then it holds that:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\kappa(\lambda)\leq\log\Big(\frac{\mathcal{N}_c(\delta,\Lambda)}{\epsilon}\Big)+C\Big\}\geq 1-\epsilon\,.$$

Proof. Since (Λ,d) is a totally bounded metric space, its covering number $\mathcal{N}_c(\delta,\Lambda)$ is well-defined and finite. For notational simplicity, set $N:=\mathcal{N}_c(\delta,\Lambda)$. Hence, there are $\lambda_1,...,\lambda_N\in\Lambda$, such that:

$$\Lambda\subseteq\bigcup_{i=1}^{N}\mathcal{B}[\lambda_i,\delta]\,.$$

Therefore, one directly gets:

$$\sup_{\lambda\in\Lambda}\kappa(\lambda)\leq\max_{i=1,...,N}\sup_{\lambda\in\mathcal{B}[\lambda_i,\delta]}\kappa(\lambda)\,.$$

Further, by assumption it holds:

$$\sup_{\lambda\in\mathcal{B}[\lambda_i,\delta]}\kappa(\lambda)=\kappa(\lambda_i)+\sup_{\lambda\in\mathcal{B}[\lambda_i,\delta]}\Big(\kappa(\lambda)-\kappa(\lambda_i)\Big)\leq\kappa(\lambda_i)+C\,.$$

Hence, in total one gets for $s \in \mathbb{R}$:

$$\begin{aligned}
\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\kappa(\lambda) > s\Big\} &\leq \mathbb{P}_{\mathfrak{D}_N}\Big\{\max_{i=1,\dots,N}\kappa(\lambda_i) + C > s\Big\} \\
&= \mathbb{P}_{\mathfrak{D}_N}\Big\{\bigcup_{i=1}^{N}\{\kappa(\lambda_i) + C > s\}\Big\} \\
&\leq \sum_{i=1}^{N}\mathbb{P}_{\mathfrak{D}_N}\Big\{\kappa(\lambda_i) + C > s\Big\} \\
&\leq \sum_{i=1}^{N}\exp(C-s) \\
&= N\exp(C-s)\,.
\end{aligned}$$

Since $\epsilon = N\exp(C-s) \iff s = \log\Big(\frac{N}{\epsilon}\Big) + C$, one gets:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\kappa(\lambda) > \log\Big(\frac{N}{\epsilon}\Big) + C\Big\} \leq \epsilon\,.$$

Taking the complementary event yields the result:

$$\mathbb{P}_{\mathfrak{D}_N}\Big\{\sup_{\lambda\in\Lambda}\kappa(\lambda) \leq \log\Big(\frac{N}{\epsilon}\Big) + C\Big\} \geq 1-\epsilon\,.$$

□