When MAML Learns Quickly, Does It Generalize Well?

First Author[1][0000−1111−2222−3333], Second Author[2,3][1111−2222−3333−4444], and Third Author[3][2222−−3333−4444−5555]

[1] Princeton University, Princeton NJ 08544, USA
[2] Springer Heidelberg, Tiergartenstr. 17, 69121 Heidelberg, Germany
lncs@springer.com
http://www.springer.com/gp/computer-science/lncs
[3] ABC Institute, Rupert-Karls-University Heidelberg, Heidelberg, Germany
{abc,lncs}@uni-heidelberg.de

Abstract. Model-Agnostic Meta-Learning is a meta-learning method that achieved state-of-the-art performance on few-shot image classification benchmarks at the time of its introduction. MAML's strength is its ability to quickly adapt to a new task in time-critical settings, while still being general enough for any gradient-based model. Although there is no need for quick adaptation in most of the few-shot learning benchmarks, MAML is still being utilized as a benchmark in this context. We investigate the benefit of limiting the adaptation steps of MAML in settings where quick adaptation is not required by the problem. In this initial study, the expected performance of MAML is compared to some conventional base learners for synthetic linear and nonlinear regression problems. Our experimental results show that considering few gradient steps in the presence of small task variance the limitation of the gradient steps improves generalization performance when tuned properly.

Keywords: meta-learning · expected performance · model-agnostic meta-learning

1 Introduction

Learning to learn, also referred to as meta-learning, treats the training of a machine learning model as a learning problem in itself. In this setting, there exist multiple learning problems and they are treated together. A machine learning model is "learning to learn" if the performance on each task improves with training experience obtained from the other tasks [18]. A major use of meta-learning is to tackle few-shot learning problems, where there is little data available from the learning task that is of prime interest, whereas there is an abundance of data from other similar tasks.

Early works of the learning-to-learn paradigm relied upon two different models working together, where one model tries to improve performance on the specific task and the other tries to improve performance over the observed tasks together [18]. MAML (Model-Agnostic Meta-Learning) [6] provides an algorithm that

circumvents the need for multiple models. This method tackles meta-learning by providing a model initialization (which is always required for models that are optimized with Stochastic Gradient Descent) that facilitates quick adaptation and maximum generalization.

Since it is model and problem independent MAML finds a wide application area in the context of few-shot meta-learning. Specifically, for supervised and reinforcement learning problems under that paradigm, where the losses differ from each other. Moreover, MAML also aims to improve a specific task performance quickly (with a few gradient steps). This is an additional aspect of the definition of meta-learning.

The quick adaptation feature can prove useful in certain settings, for instance, in robotics research, where the reaction/adaptation time of the agents in dynamic environments imposes time limitations. However, this limitation is not present in supervised learning problems, where MAML or its variants are utilized as a baseline. (*e.g.* [8,15,17,3,10] etc.) Most supervised problems are benchmarked with an image classification problem, where N-way K-shot classification problem (N different classes with K labeled training data) is tackled, where memory or time limitations do not constitute a major issue.

The main aim of this paper is to investigate MAML in settings where quick adaptation is not needed, and where most of the applications and variants of this method are benchmarked. This will be achieved by looking at the expected performance of MAML under two synthetic regression scenarios, and comparing its performance to conventional base learners (*e.g.* Linear Regression, Ridge Regression, Kernel Ridge Regression, etc.). By doing this we aim to investigate the effect of the limited adaptation step, and whether or not there is a benefit to this limitation.

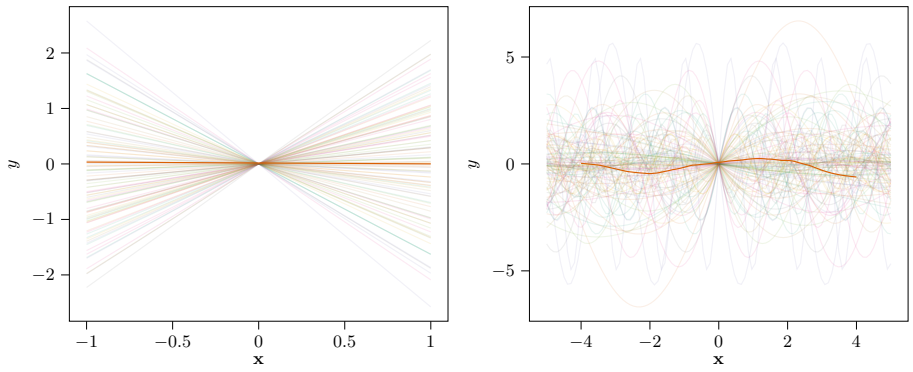
2 Model-Agnostic Meta-Learning (MAML)

MAML aims to obtain an intermediate model $\mathcal{M}(\bar{\mathbf{w}})_{\text{meta}}$ that can generalize well after adaptation with gradient descent to a dataset $\mathcal{Z}$ observed from a new and unseen task $\mathcal{T}$ drawn from $p_{\mathcal{T}}$ where the number of training points N and the number of iterations n_{iter} is limited.

In order to obtain $\bar{\mathbf{w}}_{\text{meta}}$ first, a batch of tasks $\{\mathcal{T}_i\}_{i=1}^{M}$ from $p_{\mathcal{T}}$ is observed with each having a corresponding dataset $\{\mathcal{Z}_i\}_{i=1}^{M}$. Then, the future gradients concerning each task are observed and gradient descent is utilized to get possible parameters $\bar{\mathbf{w}}'$ for each task and a gradient descent iteration is made by collecting all the possible parameters from an observed batch of tasks for the real parameter update. The general procedure for supervised learning problems is given in Algorithm 1. Authors of [6] indicate that by using this procedure the model $\mathcal{M}(\bar{\mathbf{w}})_{\text{meta}}$ requires few gradient updates from a specific task, achieving good generalization performance. Examples of the intermediate model $\mathcal{M}(\bar{\mathbf{w}})_{\text{meta}}$ prediction with the observed tasks on the background can be seen in Figures **??** and **??**.

Data: $p_{\mathcal{T}}$, α, β
Result: Intermediate Model $\mathcal{M}(\bar{\mathbf{w}}_{meta})$
initialize $\bar{\mathbf{w}}$ randomly;
while *not done* **do**
  sample a batch of tasks $\mathcal{T}_i$ from $p_{\mathcal{T}}$
  forall $\mathcal{T}_i$ **do**
    Obtain future gradients: $\nabla_{\bar{\mathbf{w}}}\mathcal{L}_{\mathcal{T}_i}(\mathcal{M}(\bar{\mathbf{w}}))$ wrt. $\mathcal{Z}_i$
    Possible future parameters: $\bar{\mathbf{w}}_i' = \bar{\mathbf{w}}_i - \alpha\nabla_{\bar{\mathbf{w}}}\mathcal{L}_{\mathcal{T}_i}(\mathcal{M}(\bar{\mathbf{w}}))$
  end
  Update: $\bar{\mathbf{w}} \leftarrow \bar{\mathbf{w}} - \beta\nabla_{\bar{\mathbf{w}}}\sum\mathcal{L}_{\mathcal{T}_i}(\mathcal{M}(\bar{\mathbf{w}}_i'))$
end

Algorithm 1: MAML[6] Algorithm



(a) Linear Problem (b) Nonlinear Problem

Fig. 1: Visualizing the MAML intermediate model. 100 sample tasks drawn from $p_{\mathcal{T}}$ for both linear ($m = 0$ and $c = 1$) and nonlinear ($c_1 = 2$ and $c_2 = 2$) problems shown transparent and intermediate model trained (obtained from MAML algorithm) for 1D cases of the experimentation given with solid dark orange line. The linear and nonlinear problems are provided in Section 4.

3 Related Work

MAML inquired multiple developments that followed the same approach of finding a warm initialization for all the tasks coming from a certain task distribution. Subsequent works highlight some of its limitations. Authors in [8,3] improved on MAML's need for task similarity making the meta-learning more task family robust. The sensitivity of MAML to architectural details is noted and circumvented in [1]. The need for a second order term in MAML is questioned and shown that first order approximations can give equally good results in [15]. Deeper changes to the MAML method are presented in [9] to convert it to a method of probabilistic inference. Moreover, a continual learning extension where tasks are introduced sequentially is proposed in [7,17].

Besides most of the above-mentioned developments, some of the attention went to improving the quick adaptation stage from the learned initialization. In [12] not just the initialization, but also the update direction and the learning rate are optimized. Moreover, in [2] the learning rate for the adaptation is learned with hypergradient descent. Both methods aim to limit the adaptation steps needed.

Therefore, MAML's developments also focus on quick adaptation. However, the above-mentioned articles use the Omniglot dataset [11] as a supervised classification benchmark with MAML as the baseline and some of the works recreate the sinusoidal regression task as shown in the original MAML paper [6]. Both of these settings where MAML and its variants are tested do not have the time or memory restrictions as few-shot learning problems. Yet, quick adaptation methods are still being benchmarked with these datasets.

4 Experimental Setting

Throughout this work uppercase bold letters (*e.g.* $\mathbf{X}$), lowercase bold letters (*e.g.* $\mathbf{x}$), and lowercase letters (*e.g.* x) are used for matrices, vectors, and scalars respectively. Moreover, the vectors are assumed to be stored in columns. Finally, the $\mathbf{I}_D$ represents a $D \times D$ identity matrix, the $\mathbf{1}_D$ and $\mathbf{0}_D$ represents $D \times 1$ vector of ones and zeros respectively.

4.1 Learning Problems

In this work, two families of regression tasks are considered; a linear and a nonlinear regression task family.

Consider the conventional linear regression problem

$$y = \mathbf{x}^{\mathrm{T}}\mathbf{a} + \varepsilon, \tag{1}$$

where $y \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^D$, $\mathbf{a} \in \mathbb{R}^D$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Each realization of the slope $\mathbf{a}$ corresponds to a task $\mathcal{T}$ and the collection of N observations is represented by $\mathcal{Z} := \{\mathbf{x}, y\}_{i=1}^{N}$.

For the nonlinear problem family, the regression function is defined as a weighted combination of sinusoidal functions:

$$y = \sin(\mathbf{x} + \boldsymbol{\phi})^{\mathrm{T}}\mathbf{a} + \varepsilon, \tag{2}$$

where $y \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^D$, $\mathbf{a} \in \mathbb{R}^D$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Assuming that the each realization of scale term $\mathbf{a}$ and $\boldsymbol{\phi}$ corresponds to a task observed in the environment $\mathcal{T}$ and each set of observed N input ($\mathbf{x}$) and its corresponding label (y) is represented by a dataset $\mathcal{Z} := \{\mathbf{x}_i, y_i\}_{i=1}^{N}$.

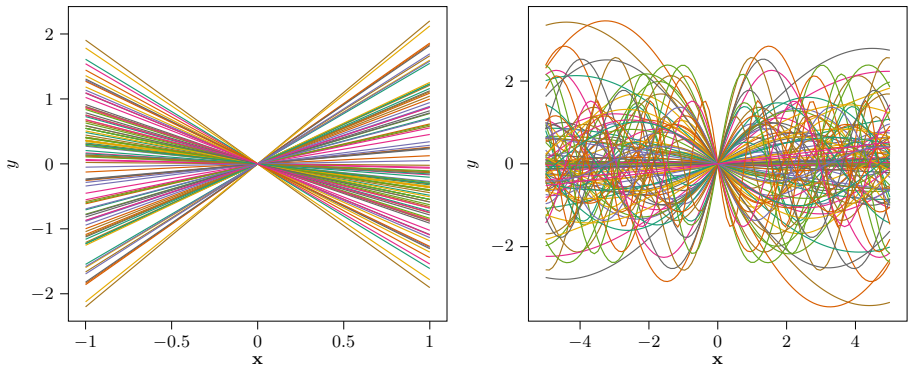
A model parameterized by $\bar{\mathbf{w}}$ is represented by $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}) : \mathbf{x} \rightarrow y$. A model $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}})$ that is trained with $\mathcal{Z}$ obtained from task $\mathcal{T}$ is represented by $\hat{\mathcal{M}}(\mathbf{x})$. Noting that $\hat{\mathcal{M}}(\mathbf{x})$ for a base learner is only exposed to a single task $\mathcal{T}$ and a single dataset $\mathcal{Z}$, whereas a meta learner, in this case, MAML, are exposed to

multiple tasks from $p_{\mathcal{T}}$ and multiple datasets $\mathcal{Z}$ in the meta-learning stage and then the adaptation is done as in the case of a base learner with just a single task $\mathcal{T}$ and a single dataset $\mathcal{Z}$. The discrepancy between the prediction of the estimator $\hat{\mathcal{M}}$ and y is measured in terms of squared loss $\mathcal{L} := (\hat{\mathcal{M}}(\mathbf{x}) - y)^2$. The main loss that this paper tries to investigate is the *Expected Squared Loss* of an estimator $\hat{\mathcal{M}}$ over the $p_{\mathcal{T}}$. Then the expected squared loss can be represented as

$$\mathcal{E} := \iiint (\hat{\mathcal{M}}(\mathbf{x}) - y)^2 p(\mathbf{x}, y) p_{\mathcal{Z}} p_{\mathcal{T}} d\mathbf{x} dy d\mathcal{Z} d\mathcal{T}. \tag{3}$$

This performance measure gives rise to the *Bayes Error* to be given by σ^2 that is coming from the noise term, which represents a model that is the perfect estimator, referred to as oracle in some of the meta-learning literature.

For all the problems the input distribution is given by $p_{\mathbf{x}} \sim \mathcal{N}(0, k\mathbf{I})$ where k is a parameter for the variance of the inputs. For the linear problem the $p_{\mathcal{T}} := p(\mathbf{a}) \sim \mathcal{N}(m\mathbf{1}_D, c\mathbf{I}_D)$ and for nonlinear problem the task distribution takes the form of a joint distribution $p_{\mathcal{T}} := p(\mathbf{a}, \boldsymbol{\phi})$ where $p_{\mathbf{a}} \sim \mathcal{N}(\mathbf{1}_D, c_1\mathbf{I}_D)$ and $p_{\boldsymbol{\phi}} \sim \mathcal{N}(\mathbf{0}_D, c_2\mathbf{I}_D)$



(a) $y = \mathbf{x}^{\mathrm{T}}\mathbf{a}$ (b) $y = \sin(\mathbf{x} + \boldsymbol{\phi})^{\mathrm{T}}\mathbf{a}$

Fig. 2: 100 sample tasks drawn from $p_{\mathcal{T}}$ for both linear ($m = 0$ and $c = 1$) and nonlinear ($c_1 = 1$ and $c_2 = 1$) problems.

4.2 Models

For the linear problems $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}) := \bar{\mathbf{x}}\bar{\mathbf{w}}$ with $\mathbf{x} \in \mathbb{R}^{1\times D}$, $\mathbf{w} \in \mathbb{R}^{D\times 1}$, $\bar{\mathbf{x}} \in \mathbb{R}^{1\times D+1}$ and $\bar{\mathbf{w}} \in \mathbb{R}^{D+1\times}$ where $\bar{\mathbf{w}} := [\mathbf{w}, b]^{\mathrm{T}}$ and $\bar{\mathbf{x}} := [\mathbf{x}, 1]$ is utilized. The optimum parameters ($\hat{\bar{\mathbf{w}}}$) for different linear models are obtained as follows:

Linear Estimator is given by the least-squares solution, $\hat{\bar{\mathbf{w}}} := (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$, where $\mathbf{X} \in \mathbb{R}^{N\times D}$ is the design matrix where the observed input data is stored in rows.

Ridge Estimator is given by $\hat{\bar{\mathbf{w}}} := (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I}_D)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$ which is obtained by minimizing the squared loss with the additional term of $\lambda||\bar{\mathbf{w}}||_2^2$.

Kernel Ridge Estimator is given by $\hat{\bar{\mathbf{w}}} = \mathbf{X}^{\mathrm{T}}\boldsymbol{\alpha}$ where $\boldsymbol{\alpha} := (\mathbf{K} + \lambda\mathbf{I}_N)^{-1}y$ where $\mathbf{K} \in \mathbb{R}^{N\times N}$ is the *Gram Matrix* obtained by replacing $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ inner product by a kernel $\kappa(\mathbf{X}, \mathbf{X})$. Then, the prediction of the estimator takes the form $\hat{\mathcal{M}}(\mathbf{x}^*, \bar{\mathbf{w}}) = \boldsymbol{\alpha}^{\mathrm{T}}\kappa(\mathbf{x}^*, \mathbf{X})$ where $\mathbf{x}^* \in \mathbb{R}^{D\times 1}$.

For both linear and nonlinear models, gradient descent can be utilized to update the parameters of a model $\mathcal{M}$. Then,

Gradient Descent Estimator for any given model $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}})$ and a given number of iterations n_{iter} the gradient descent estimator is given by $\{\bar{\mathbf{w}}_{j+1} = \bar{\mathbf{w}}_j - \eta\sum_i^N \mathbf{x}_i(\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}_j) - y_i)\}_{j=0}^{n_{iter}}$. In other words for any given value of $\bar{\mathbf{w}}$ one gradient update is given by the gradient with respect to $\bar{\mathbf{w}}$ with a scaling parameter η.

All the models investigated can be divided into two sub-categories the models that have information regarding the task space and the ones that have not. The labels used in Section 5 of the models that have information regarding the task space are as follows:

- **GD**: corresponds to gradient descent with n_{iter} with adjustable parameters obtained from the mean of the tasks $\mathbb{E}[p_{\mathcal{T}}]$. (*e.g.* considering the linear problem the with $p_{\mathbf{a}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the initial starting point of the linear model is $\bar{\mathbf{w}} = \mathbf{0}$.)
- **MAML** (for linear problem): corresponds to gradient descent with n_{iter} with the adjustable parameters obtained from the mean of the tasks $\mathbb{E}[p_{\mathcal{T}}]$ with small perturbation $\delta \sim \mathcal{N}(\mathbf{0}, 0.1\mathbf{I})$. This is done to simulate the effect of various learning rates and stopping points during the meta-learning stage.
- **MAML** (for nonlinear problem): MAML algorithm trained with the information given in [6] for the sinusoidal regression problem. It should be noted that network architecture and all the other hyper-parameters are taken from the paper exactly.
- **random GD**: corresponds to a gradient descent of a randomly initialized model.

Finally, the following model labels have information from a single task:

- **Linear**: standard least squares solution.
- **Ridge**: standard least squares solution with $L_2 - regularization$.
- **random GD**: gradient descent with n_{iter} with the adjustable parameters starting from random initialization.
- **Kernel Ridge**: kernelized (with Radial Basis Function Kernel)

For the linear problem setting, it should be noted that the optimum can always be reached when the gradient descent is utilized with an infinitely small learning rate and an infinite number of gradient steps. Thus, allowing us to investigate the difference between taking limited steps or allowing the model to go towards the optimum directly.

It should be noted that the hyper-parameters of the utilized models, if there are any, are selected with a simple grid search with 20 different values and only the one with the lowest mean expected performance over the parameter under investigation is presented in the results.

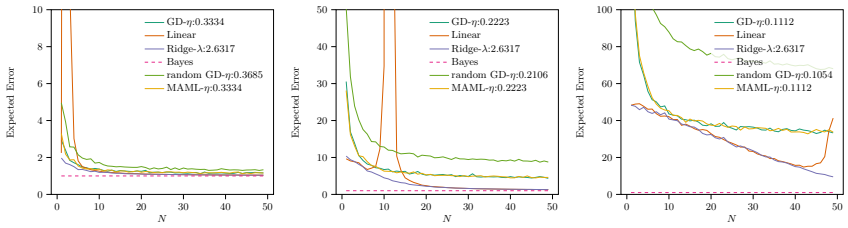
5 Results and Discussion

This section is dedicated to the expected performance results of a meta-learning model after adaptation and conventional base learners (*e.g.* Linear, Ridge, and Kernel Ridge Regression models), with the aim to see their performance differences under certain scenarios induced by the experimental assumptions (*e.g.* task variance, input variance, noise, and dimensionality).

Linear Problem The linear problem introduced in Section 4 is characterized by the dimensionality D, number of training samples N, number of gradient steps n_{iter}, the task variance c and task mean m, and the variance of the input samples k. For the sake of brevity, only some of the parameters are discussed in this section. Unless the parameter in the configuration is under investigation, the default values are utilized. And, the default values for the experimentation $\sigma = 1$, $m = 0$, $k = 1$, $c = 1$, $n_{iter} = 1$. Moreover, the number of tasks drawn ($N_{\mathcal{T}}$), and dataset draws ($N_{\mathcal{Z}}$) for approximating the expected error given in Equation 3 are taken to be 100 each. Finally, the test set size is taken as 1000.

Effect of the Number of Training Samples N: The results of this experiment can be found in Figure 3 for increasing problem dimensionality. It can be seen that the Linear model suffers from singularities, for instance in Figure 3b when the number of samples equals dimensionality $N = D$. However, it is able to have comparable performance over all the selected problem dimensionalities. For the increasing training samples case, the Ridge model performs much better as all the cases converge towards the Bayes error. However, the gradient descent variant models which have meta task information (*e.g.* MAML, GD), are unable to converge towards the Bayes error. This can be attributed to the regularizing effect of the limited gradient steps (n_{iter}) allowed for the models. Overall, the improvement that the additional task-related information brings to the gradient-based models is not visible, as the random GD model is orders of magnitude higher than expected performance. Although task information provides gain over the random initialization, the expected performance is hindered for the gradient-based models.

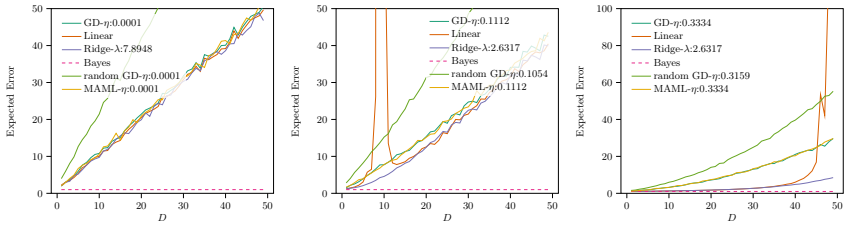
Effect of Dimensionality D: These results are quite similar to the ones obtained with the experiments investigating the effect of training samples. It can be observed from Figure 4 that, aside from singularities the Linear model variants yield lower expected performance compared to the gradient descent variants and this gap increases as the dimensionality of the problem or the number of training samples increases. Looking at Figure 3b it can be observed that for the number of



(a) $D = 1$ (b) $D = 10$ (c) $D = 50$

Fig. 3: The expected error for the increasing number of training samples and problem dimensionality.

training samples around the dimensionality of the problem Ridge model performs much better, but as the dimensionality of the problem increases so does the gap between the performances of the Ridge model and the gradient descent variants.



(a) $N = 1$ (b) $N = 10$ (c) $N = 50$

Fig. 4: The expected error for increasing problem dimensionality and the number of training points.

Effect of Task Variance c: The results of increasing task variance for various problem dimensions and various numbers of training samples can be found in Figure 5. The most obvious observation is that for all the models that utilize gradient descent, expected performance increases, whereas the Linear model and the Ridge model are only affected by this phenomenon for problem dimensionality $D \geq N$. In light of this observation, another important result is that for $N \geq D$ and for small task variance the gradient descent variants (except the randomly initialized model) have lower expected performance than the Ridge model. However, this performance diminishes with increasing problem dimensionality and the increasing number of training points. It is interesting to see a better performance from just one gradient step. That is why an extra mini-experimentation is done for the GD and MAML models to investigate if there is a performance improvement with the additional gradient steps. The results of this experimentation are given in Table 1. It is observed from the table that there exists a point at which the

gradient steps are hurting the expected performance one would get in this range after the second gradient step. Then, it can be conjectured that the number of gradient steps has a regularizing effect on the task distributions with small variance. Despite, the surprising results of MAML-like algorithms, the Ridge model is much more stable and performs better than gradient-based methods for $N \geq D$.

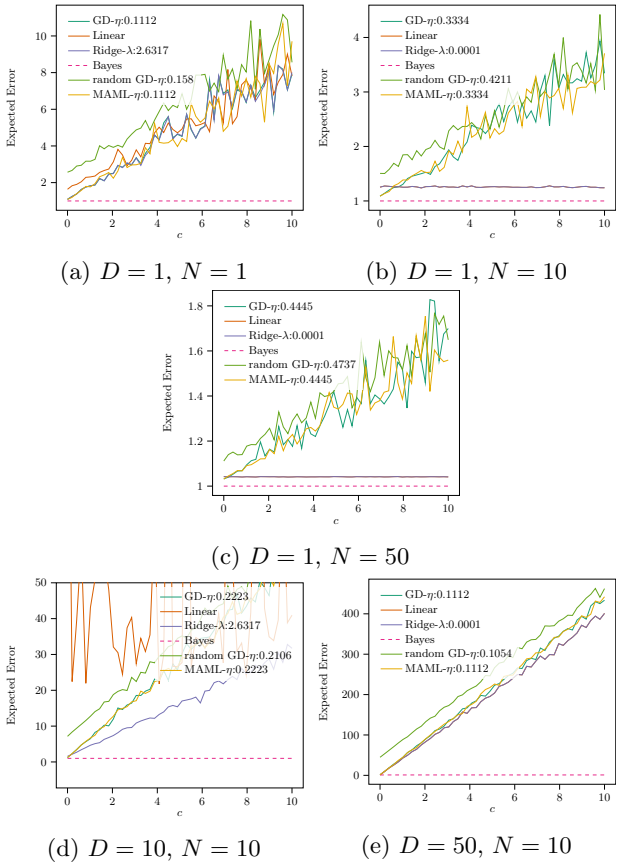
Table 1: Mean expected performance for the range $c : [0, 1]$ range with various gradient steps for the MAML and GD models with $\eta = 0.334$. Note that only $D = 1$, $N = 10$ case (see Figure 5b) is presented.

	n_{iter}									
	1	2	3	4	5	6	7	8	9	10
GD	1.2152	**1.1936**	1.2048	1.2140	1.2268	1.2390	1.2604	1.2970	1.3825	1.5748
MAML	1.2132	**1.1938**	1.2067	1.2171	1.2318	1.2476	1.2773	1.3330	1.4622	1.7556

Effect of Number of Gradient Steps n_{iter}: Looking at the interesting results observed from the task variance c the effect of the number of gradient steps taken becomes more relevant. These results can be seen in Figure 6. It can be observed from Figure 6a that for a low number of training samples the gradient steps taken have little to no influence. But as the number of training samples increases for a given problem dimensionality the effect n_{iter} on the expected performance gets much more prominent. It is evident that compared to single task learning with gradient descent from a random initialization, starting from a more informative point (*e.g.* near the task mean) decreases the number of gradient steps until convergence. Moreover, for $D = N$ case it even improves generalization after convergence too (see Figures 6d and 6a). Overall, it can be observed that the increasing n_{iter} converges towards the Ridge model variants with the exception of the $D = 1$ and $N = 1$ cases.

Effect of Task Mean m: The results can be seen in Figure 7. The most important observation from this experimentation is that the Ridge model has increasing expected performance for the cases of $N \leq D$ cases (see Figures 7a, 7d and 7e) and mostly the best λ from the trials is found to be the lowest value, which makes the Ridge model behave similar to the Linear model. Furthermore, other models which have prior task information do not seem to be affected by the task mean shifting in the task space, as expected. Again, the superiority of including information from the task space is evident as the conventional regularization cannot deal with the changing task distribution mean for $N \leq D$.

Nonlinear Problem The nonlinear problem introduced in Section 4 has the parameters controlling the dimensionality D, number of training samples N, number of gradient steps n_{iter}, the task variances and means m_1 and m_2, c_1 and



(a) $D = 1$, $N = 1$ (b) $D = 1$, $N = 10$

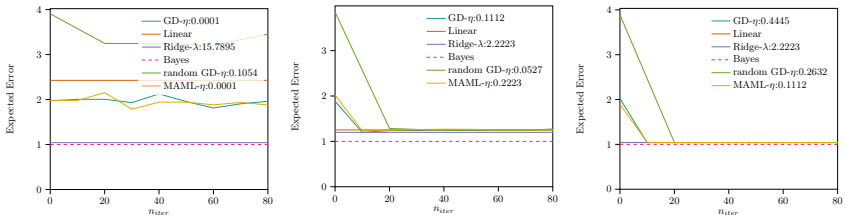
(c) $D = 1$, $N = 50$

(d) $D = 10$, $N = 10$ (e) $D = 50$, $N = 10$

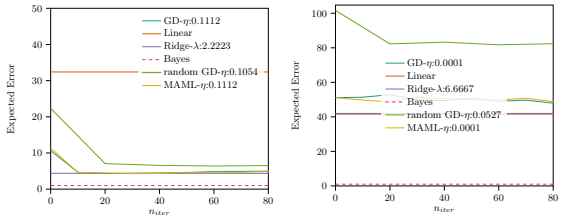
Fig. 5: The expected error for increasing task variance c when changing the number of training samples for various problems of different dimensions.

c_2, and the variance of the input samples k. Note that only some of the parameters are discussed in this section. Unless the parameter in the configuration is under investigation, the default values are utilized. And, the default values are given as $\sigma = 1$, $m_1 = 1$, $m_2 = 0$, $c_1 = 2$, $c_2 = 2$, $k = 1$, $n_{iter} = 5$. Moreover, the number of tasks drawn ($N_{\mathcal{T}}$), and dataset draws ($N_{\mathcal{Z}}$) for approximating the expected error given in Equation 3 are taken to be 50 each. Finally, the test set size is taken as 1000.

Effect of Training Samples N: By looking at Figure **??** it can be seen that for all the given dimensionalities there exists a training sample amount where the expected error of the Kernel Ridge model is higher than MAML. The most notable behavior for this experiment is that Kernel Ridge models tend towards the Bayes error while the MAML converges to a certain value and stays there. This might be again attributed to the limitation affiliated with the number of gradient

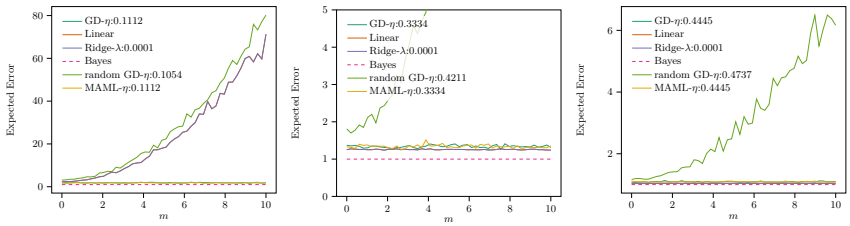


(a) $D = 1,\ N = 1$ (b) $D = 1,\ N = 10$ (c) $D = 1,\ N = 50$

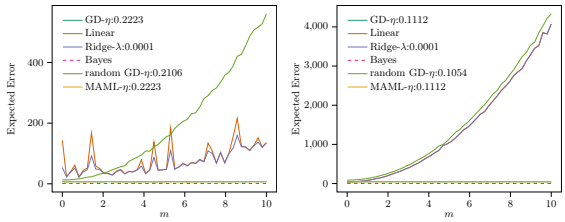


(d) $D = 10,\ N = 10$ (e) $D = 50,\ N = 10$

Fig. 6: The expected error for the increasing number of gradient steps n_{iter} used for adaptation when changing the number of training samples for various problems of different dimensions.



(a) $D = 1,\ N = 1$ (b) $D = 1,\ N = 10$ (c) $D = 1,\ N = 50$

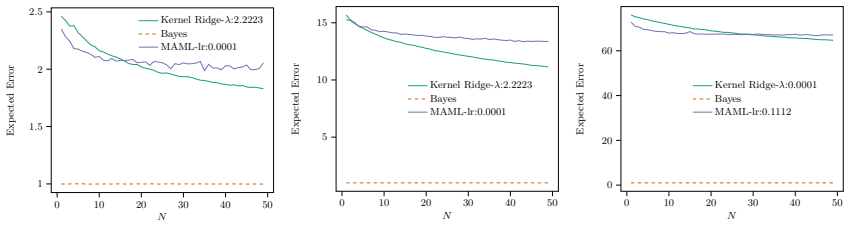


(d) $D = 10,\ N = 10$ (e) $D = 50,\ N = 10$

Fig. 7: The expected error for increasing task mean m when changing the number of training samples for various problems of different dimensions.

steps. Although the expected error is quite high for increasing dimensionality, the results obtained for $D = 1$ (Figure 8a) effectively show that for a small number

of training samples with limited gradient steps MAML will outperform a convex model Kernel Ridge.



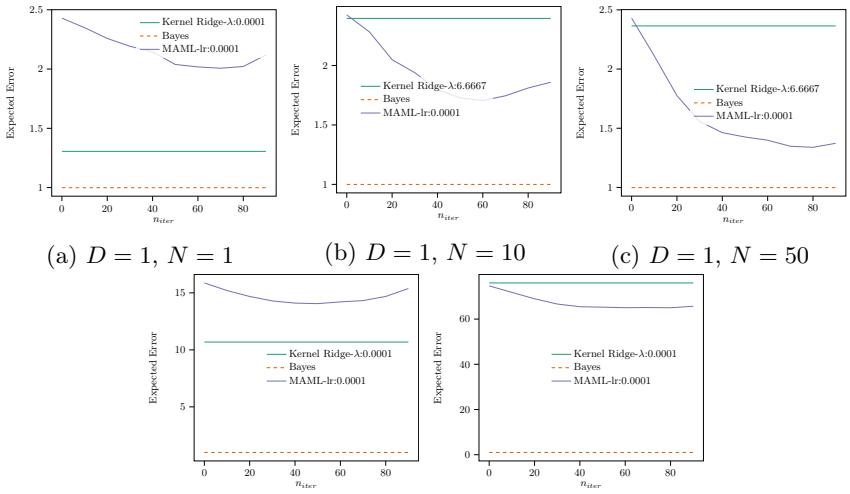
(a) $D = 1$ (b) $D = 10$ (c) $D = 50$

Fig. 8: The expected error for the increasing number of training samples and problem dimensionality.

Effect of Number of Gradient Steps n_{iter}: Presented learning curves until this point, beg the investigation of the effect of n_{iter}. As it can be seen from Figure 9a, the single realization of the Kernel Ridge model can have a lower expected error for an extreme value of 1 training sample. However, as the number of training samples increases for a given problem dimensionality MAML model starts showing a better-expected error. However, the lowest expected error is not realized for a fairly large n_{iter}. Moreover, it can be observed that for $N \leq D$ Kernel Ridge model can achieve a lower expected error, and for all the other cases one might find a better MAML model given that sufficient gradient steps are allowed. Finally, it should be noted that the number of gradient steps required to perform better than the Kernel Ridge model is fairly low.

Effect of Phase Task Variance c_2: Remembering that the task variance effect for the linear problem had some interesting properties where even a single gradient step resulted in better expected performance. One might wonder if that is the case for the nonlinear problem as well. As can be seen in Figure 10 a similar effect is observed for the nonlinear problem too for small training sample size N values. For problem dimensionality of 1 and 10 there is a clear expected error rise between task variance $[0, 2]$ as shown in Figures 10a,10b,10c and 10d. This indicates that the "MAML" even with a limited number of gradient steps provides a clear benefit compared to a model that has an analytical solution. However, mostly this superiority vanishes as the task variance increases as the increased values of variance lead to worse expected error compared to "Kernel Ridge".

Additional experimentation results for σ for linear and nonlinear problems can be found in the Appendix, it is observed that increasing noise has similar behavior with single task learning models. Moreover, the effect of input variance is investigated and found that the gradient descent based methods perform poorer for the linear problem.



(a) $D = 1$, $N = 1$ (b) $D = 1$, $N = 10$ (c) $D = 1$, $N = 50$

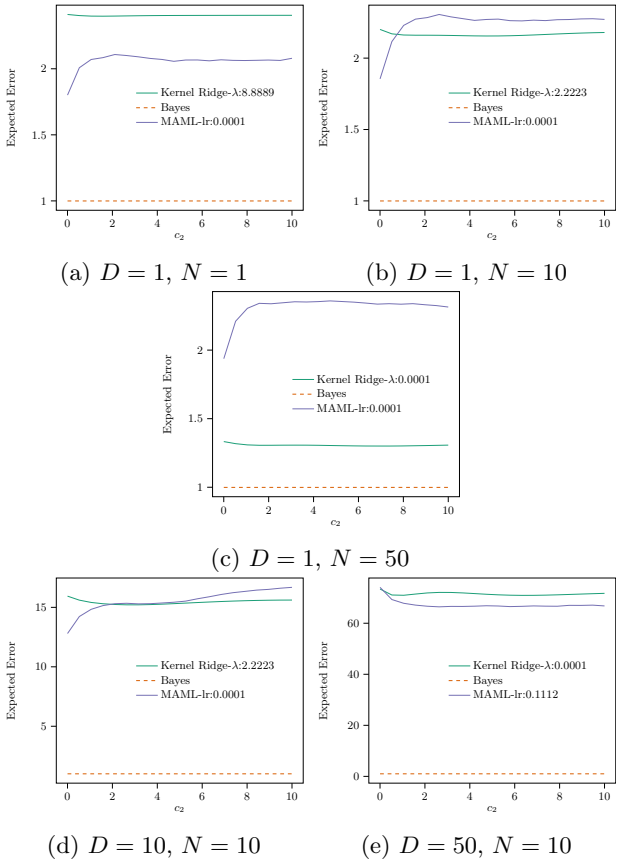
(d) $D = 10$, $N = 10$ (e) $D = 50$, $N = 10$

Fig. 9: The expected error for the increasing number of gradient steps n_{iter} used for adaptation when changing the number of training samples for various problems of different dimensions.

5.1 Discussion

Upon our investigation, it is found empirically that meta-information about the task space can help the generalization performance in linear and nonlinear problem settings even with limited gradient steps. Increased generalization performance of MAML compared to single task learning models on expectation when the tasks that are in consideration are close to each other is observed, where the same observation is made theoretically in [5]. This observation suggests that there is a regularizing effect of limiting the gradient steps needed for adaptation. We conjecture that after the meta-learning stage intermediate model parameters $\bar{\mathbf{w}}$ are closer to the test set optimum compared to the proximity of train and test set optimums. This type of behavior is investigated in [14] as well, where the large learning rate in the training phase acts as a regularizer due to the discrepancy between train and test loss landscapes. Similar behavior can be observed for MAML with the nonlinear problem too due to being a non-convex problem.

This limitation of adaptation steps is noted in [2,13] that tries to improve the MAML adaptation step so that the adaptation is limited to fewer gradient steps, preferably one. Our findings suggest that the expected performance of these methods should be investigated as well as some of the generalization power of MAML might be coming from the regularization induced by not optimizing the training loss perfectly. This hypothesis is supported by the findings of [16] which concludes that the performance gain of MAML is about feature reuse instead of rapid learning.



(a) $D = 1$, $N = 1$ (b) $D = 1$, $N = 10$

(c) $D = 1$, $N = 50$

(d) $D = 10$, $N = 10$ (e) $D = 50$, $N = 10$

Fig. 10: The expected error for increasing task variance n_{iter} used for adaptation when changing the number of training samples for various problems of different dimensions.

6 Conclusion

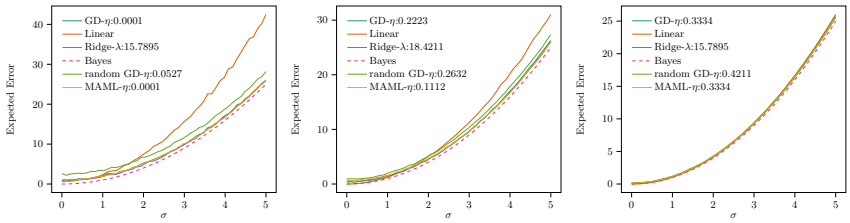
Although its minor superiority with regards to expected performance, single-task learners are able to compete with MAML. In all of our experiments single best learning rate and the regularization parameter are selected for the whole expected performance curves individually. We show that even in the case of general regularization, and when enough data is present a single task learner can outperform on expectation of a meta learner when the tasks observed start to deviate from each other. This indicates that the regularization-based meta-learners similar to the ones presented in [4], but also suitable for the nonlinear problem settings can be competitive and robust enough for much wider task variance. Moreover, regularization-based methods similar to the ones presented in [10] for MAML can prove useful to understand and study MAML.

The use of quick adaptation under a supervised learning setting where this constraint is not imposed by the problem is investigated. It is found that a regularization effect exists induced by quick adaptation which contributes to the generalization performance of MAML even in the convex problem setting. However, more in-depth expected performance should be done for the exact causes of Omniglot dataset and other widely used supervised few-shot learning benchmarks. We believe that understanding the effects of all the contributing assumptions is key to correct use cases of meta-learning methods.

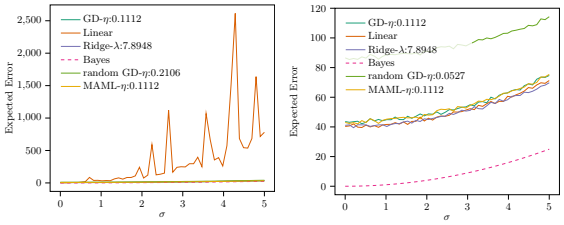
References

1. Antoniou, A., Edwards, H., Storkey, A.: How to train your MAML (Mar 2019)
2. Behl, H.S., Baydin, A.G., Torr, P.H.S.: Alpha MAML: Adaptive Model-Agnostic Meta-Learning (May 2019)
3. Collins, L., Mokhtari, A., Shakkottai, S.: Task-Robust Model-Agnostic Meta-Learning. arXiv:2002.04766 [cs, math, stat] (Jun 2020)
4. Denevi, G., Ciliberto, C., Stamos, D., Pontil, M.: Incremental Learning-to-Learn with Statistical Guarantees. arXiv:1803.08089 [cs, stat] (Mar 2018)
5. Fallah, A., Mokhtari, A., Ozdaglar, A.: Generalization of Model-Agnostic Meta-Learning Algorithms: Recurring and Unseen Tasks. arXiv:2102.03832 [cs, math, stat] (Nov 2021)
6. Finn, C., Abbeel, P., Levine, S.: Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. arXiv:1703.03400 [cs] (Jul 2017)
7. Finn, C., Xu, K., Levine, S.: Probabilistic Model-Agnostic Meta-Learning. arXiv:1806.02817 [cs, stat] (Oct 2019)
8. Flennerhag, S., Moreno, P.G., Lawrence, N.D., Damianou, A.: TRANSFERRING KNOWLEDGE ACROSS LEARNING PROCESSES p. 23 (2019)
9. Grant, E., Finn, C., Levine, S., Darrell, T., Griffiths, T.: Recasting Gradient-Based Meta-Learning as Hierarchical Bayes. arXiv:1801.08930 [cs] (Jan 2018)
10. Guiroy, S., Verma, V., Pal, C.: Towards Understanding Generalization in Gradient-Based Meta-Learning. arXiv:1907.07287 [cs, stat] (Jul 2019)
11. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: The Omniglot challenge: A 3-year progress report (May 2019)
12. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, Broader and Artier Domain Generalization. arXiv:1710.03077 [cs] (Oct 2017)
13. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-SGD: Learning to Learn Quickly for Few-Shot Learning. arXiv:1707.09835 [cs] (Sep 2017)
14. Nakkiran, P.: Learning Rate Annealing Can Provably Help Generalization, Even for Convex Problems. arXiv:2005.07360 [cs, stat] (May 2020)
15. Nichol, A., Achiam, J., Schulman, J.: On First-Order Meta-Learning Algorithms. arXiv:1803.02999 [cs] (Oct 2018)
16. Raghu, A., Raghu, M., Bengio, S., Vinyals, O.: Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML (Feb 2020)
17. Rajasegaran, J., Khan, S., Hayat, M., Khan, F.S., Shah, M.: iTAML: An Incremental Task-Agnostic Meta-learning Approach. arXiv:2003.11652 [cs, stat] (Mar 2020)
18. Thrun, S., Pratt, L. (eds.): Learning to Learn. Springer US, Boston, MA (1998). https://doi.org/10.1007/978-1-4615-5529-2

7 Appendix

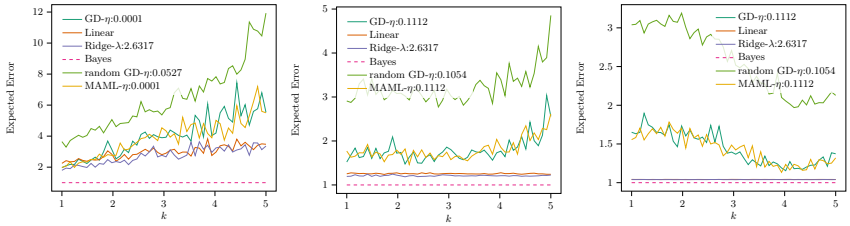


(a) $D = 1$, $N = 1$ (b) $D = 1$, $N = 10$ (c) $D = 1$, $N = 50$

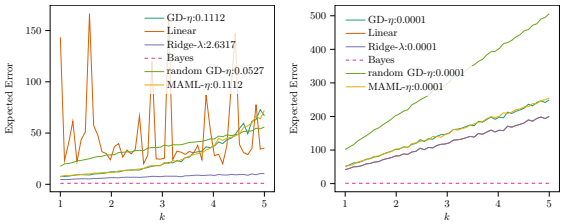


(d) $D = 10$, $N = 10$ (e) $D = 50$, $N = 10$

Fig. 11: [**Linear Problem**]: The expected error for increasing noise standard deviation σ when changing the number of training samples for various problems of different dimensions.

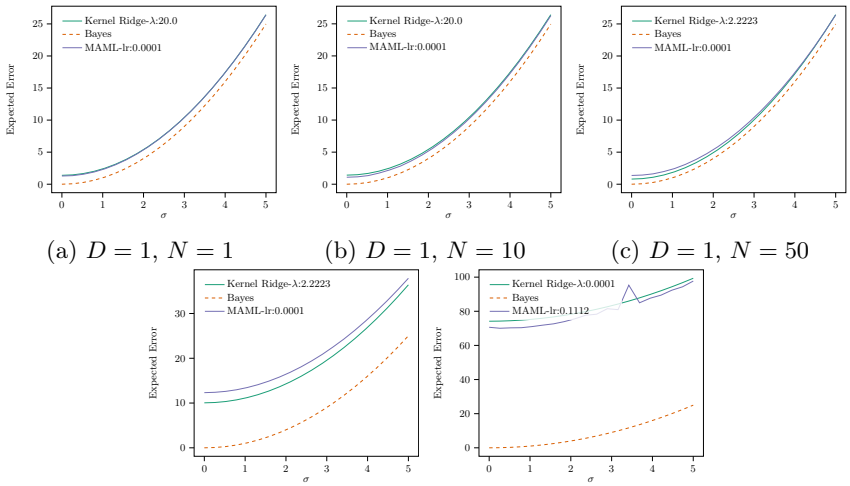


(a) $D = 1$, $N = 1$ (b) $D = 1$, $N = 10$ (c) $D = 1$, $N = 50$



(d) $D = 10$, $N = 10$ (e) $D = 50$, $N = 10$

Fig. 12: [**Linear Problem**]: The expected error for increasing input variance k when changing the number of training samples for various problems of different dimensions.



(a) $D = 1,\ N = 1$ (b) $D = 1,\ N = 10$ (c) $D = 1,\ N = 50$

(d) $D = 10,\ N = 10$ (e) $D = 50,\ N = 10$

Fig. 13: [**Nonlinear Problem**]: The expected error for increasing noise standard deviation σ when changing the number of training samples for various problems of different dimensions.