NO TITLE

Author1, Author2
Affiliation
Univ
City
{Author1, Author2}email@email

Author3
Affiliation
Univ
City
email@email

ABSTRACT

***K*eywords** 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 the context of this work if a machine learning model's performance on a task is improving with training experiences it is said to be learning. In the light of this definition a machine learning model is said to be learning to learn if the performance on each task improves with training experience obtained from each task and with the number of tasks [1]. Meta-learning recently is being used 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 the one supervisory and one sub-ordinate model that interacts with each other for meta-learning. On one hand, sub-ordinate models try to improve the performance with training examples and on the other hand, the supervisory model tries to increase the performance over the family of tasks. MAML (Model-Agnostic Meta-Learning) [2] is a model that circumvents the need for supervisory and subordinate models. This method tries to tackle meta-learning by training any (As the name suggests MAML applies to all learners that improve performance by SGD (Stochastic Gradient Descent).) machine learning models parameters in a way to maximize the performance on a new learning task with few experiences through one or more gradient steps.

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

As mentioned above, MAML aims to improve the generalization of a model for a certain learning task from a given family of tasks, with little data and minimal training. Minimal training indicates quick adaptation capabilities. This feature can prove useful in certain settings, for instance, in robotics research, where the reaction/adaptation time of the agents to dynamic environments bestow an inherent time limitation. However, this limitation is not present for the supervised learning problems, where MAML or its variants are utilized as a baseline. (*e.g.*[3, 4, 5, 6, 7] etc.) Most of the unsupervised problem benchmark is image detection problem, where N-way K-shot classification problem (N different classes with K labeled training data) is tried to be tackled. Given the nature of the problem, most of the time memory or time limitation does not constitute a major issue in the given problem setting.

The main aim of this paper is to investigate the MAML under the 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 the MAML under 2 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.

2 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.

2.1 Learning Problems

In this work one linear and one nonlinear problem constitute the family of tasks. In the linear case a realization of slope of a linear function will be used for the definition of a single task, and for the nonlinear case a single task is represented by a realization of amplitude and phase for the sine function.

2.1.1 Linear Problem

Consider the conventional linear regression problem in $\mathbb{R}^D$ is given by

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

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 $\mathbf{a}$ corresponds to a task $\mathcal{T}$ and collection of N observations is represented by $\mathcal{Z}_j := (\mathbf{x}, y)_{i=1}^N$.

2.1.2 Nonlinear Problem

Consider a nonlinear problem in $\mathbb{R}^D$ is given by

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

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 $\mathcal{Z}_j := (\mathbf{x}_i, y_i)_{i=1}^N$.

2.1.3 General Problem Setting

For both problems presented in Sections 2.1.1 and 2.1.2 sample distribution is given by $p_{\mathcal{Z}}$ for a given $\mathcal{T}$ and the task distribution is represented by $p_{\mathcal{T}}$. A model parameterized by $\bar{\mathbf{w}}$ is represented by $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}) : \mathbf{x} \rightarrow y$. An estimator that is trained with $\mathcal{Z}$ that is obtained from the $\mathcal{T}$ 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, is exposed to multiple tasks T's 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 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 $\mathcal{L} := (\hat{\mathcal{M}}(x) - y)^2$. 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

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

For the defined expected error and the problem definitions, the *Bayes Error* is given by σ^2 that is coming from the noise term, which represents a model that is the perfect estimator, which is referred as oracle in some of the meta-learning literature.

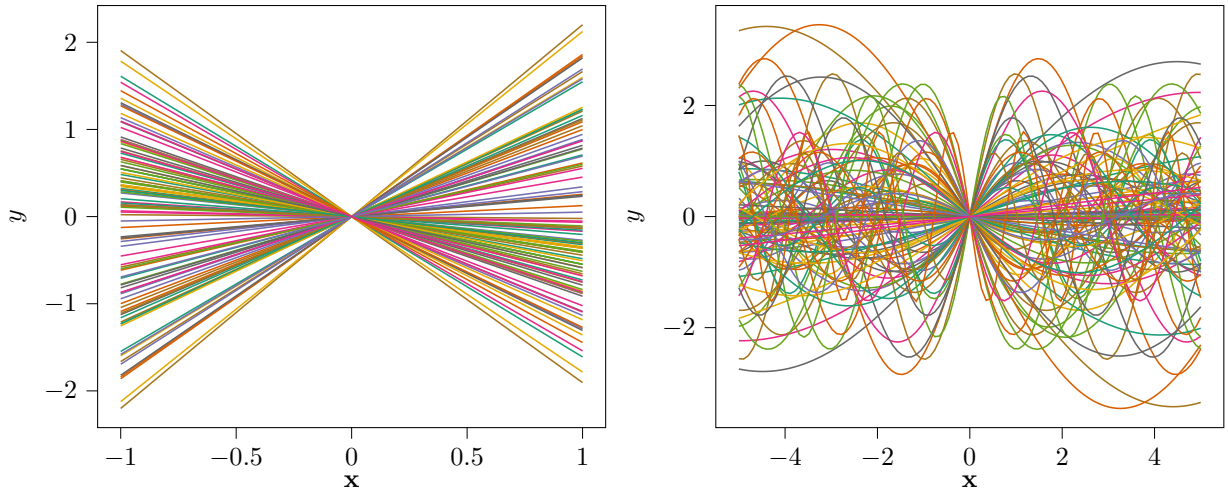
2.1.4 Experimental Assumptions

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 $p_{\boldsymbol{\phi}} \sim \mathcal{N}(\mathbf{0}_D, c_2\mathbf{I}_D,)$

2.2 Models

2.2.1 Single Task Learning Models

Assuming a linear model in the form of $\mathcal{M}(\mathbf{x}, \mathbf{w}, b) := \mathbf{x}\mathbf{w} + b$ or $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}) := \bar{\mathbf{x}}\bar{\mathbf{w}}$ 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 $\bar{\mathbf{x}} := [\mathbf{x}, 1]$. The optimum parameters ($\hat{\bar{\mathbf{w}}}$) for different models are obtained as follow:



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

Figure 1: 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.

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 $\lambda||\bar{\mathbf{w}}||_2^2$. Thus, overall loss takes the form $\mathcal{L} + \lambda||\bar{\mathbf{w}}||_2^2$.

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 $\mathbf{x}^* \in \mathbb{R}^{D \times 1}$.

Gradient Descent Estimator for the Linear Model for a given number of iterations n_{iter} the gradient descent estimator is given by $\{\bar{\mathbf{w}}_{j+1} = \mathbf{w}_j - \eta\sum_i^N \mathbf{x}_i(\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}}_j) - y_i)\}_{j=0}^{n_{iter}}$. 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 ν.

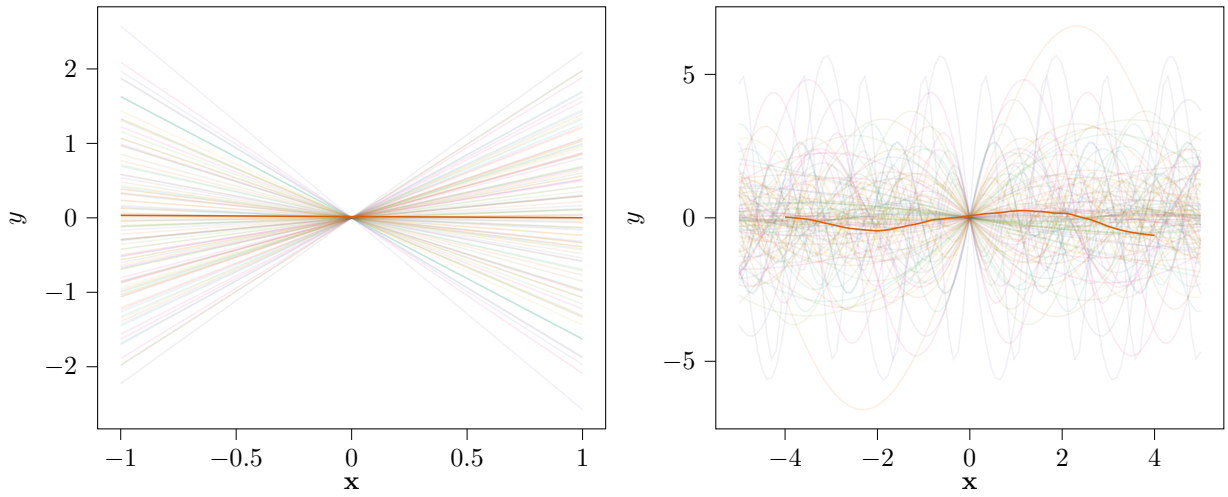
2.2.2 Meta Learning Models

Model-Agnostic Meta-Learning (MAML) aims to obtain an intermediate model that can generalize well after adaptation to a dataset $\mathcal{Z}$ observed from a new and unseen task $\mathcal{T}_i$ drawn from $p_{\mathcal{T}}$ where the number of training points K and the number of iterations n_{iter} is limited. The general procedure for supervised learning problems is given in Algorithm 1.

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. $\mathcal{Z}_K$
    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[2] Algorithm

As stated before after employing this algorithm the model $\mathcal{M}(\mathbf{x}, \bar{\mathbf{w}})$ is left at a place that is in optimal position for quick adaptation. At the time for inference for a particular dataset $\mathcal{Z}$ obtained from a particular task $\mathcal{T}$, simple gradient descent as mentioned above is utilized for the quick adaptation.



(a) Linear Problem

(b) Nonlinear Problem

Figure 2: 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.

3 Related Work

After the introduction of the MAML, multiple papers followed its backbone idea, which is finding a warm initialization for all the tasks coming from a certain task distribution. In the follow-up works, the deficiencies of MAML are tried to be tackled. [3, 6] try to tackle the task similarity needed for MAML, in other words, make the meta-learner more task family robust. The sensitivity of MAML to architectural details is tried to be circumvented in [8]. And, the need for the second order term needed by MAML is questioned and shown that first order approximations can give as good results as MAML in [4]. Deeper changes to the MAML method are presented in [9] Moreover, an extension for the continual learning where tasks are introduced sequentially is proposed in [10, 5].

Besides most of the above-mentioned architectural and problem tweaks, some of the attention went to improving the quick adaptation stage from the learned initialization. In [11] not just the initialization but the update direction and the learning rate are tried to be found. Moreover, in [12] the learning rate for the adaptation is tried to be learned with hypergradient descent. Both of the methods aim to limit the adaptation steps needed.

It can be seen from the derivative work of MAML that the extra aim introduced with MAML to meta-learning, which is a quick adaptation, is the common denominator in most of the work that can be found. There seems to be a case for quick adaptation in dynamic problems, in other problems, this seems not to be the case. All of the above-mentioned articles use the Omniglot dataset [13] as a supervised classification benchmark with the MAML as the baseline. Moreover, some of the work use the sinusoidal regression task as shown in the original MAML paper [2]. Both of these settings where MAML and its variants are tested do not have the time or the memory restrictions as a few-shot learning problem. However, quick adaptation methods are still being benchmarked with these methods. Although there is quite a lot of effort into improving MAML it is still not clear the effect of adaptation step limitation in a problem that does not necessarily suffer from the quick adaptation constraint.

4 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). Experiments conducted have various models some of which have information about the family of tasks

(either due to meta-training or the initialization point being selected suitable to the task distribution), and others have only information from a single task at every step. Models that have information regarding the task space are as follows:

- **GD**: corresponds to gradient descent with n_{iter} with the 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 [2] 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 models 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)

It should be noted that the hyper-parameters of the utilized models, if there are any, are not tuned, properly as it would increase the computational burden of the problem to another level. However, a simple grid search is employed with 20 different values and only the one with the lowest mean expected performance over the parameter under investigation is presented for the results.

4.1 Results

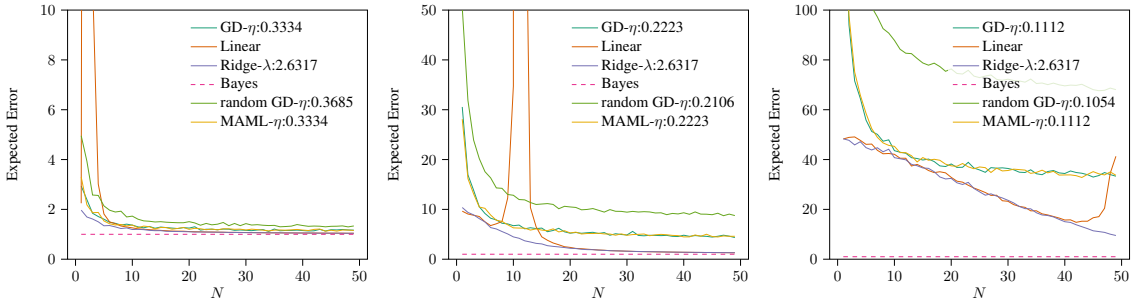
4.1.1 Linear Problem

The linear problem introduced in Section 2.1.1 has the parameters controlling 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 configuration is under investigation, the default values are utilized. And, the default values are given as,

- $\sigma = 1$,
- $m = 0$,
- $k = 1$,
- $c = 1$,
- $n_{iter} = 1$.

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.

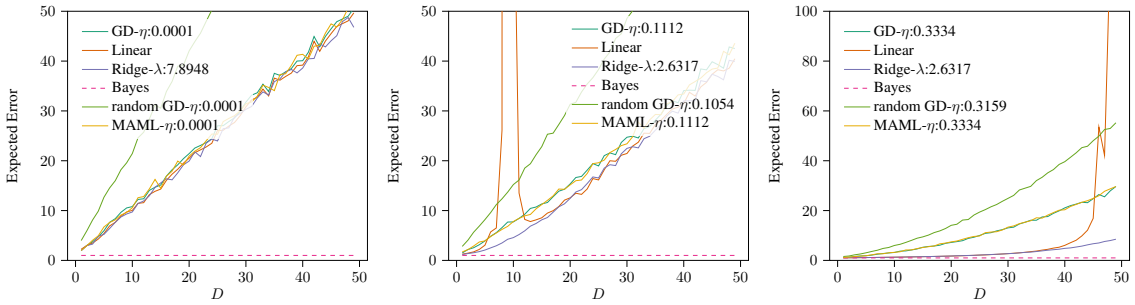
Effect of Training Samples N: The results of this experiment can be found in Figure 3 for increasing problem dimensionality. It can be seen that the Linear model suffers from singularities, however, it is able to have comparable performance over all the selected problem dimensionalities. As one might expect as the dimensionality increases the difference between the models with analytical optimum and gradient descent utilizing methods. Moreover, for the increasing training samples case the Ridge regression variants perform much better as for all the cases they Converge towards the Bayes error. However, for the gradient descent variant models whether there exists task information (*e.g.*MAML, GD) is 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, as the random GD model is orders of magnitude higher than expected performance. Although, task information inclusion decreases the expected performance as the number of gradient step limitations hinders the gradient-based models' capability to decrease the expected performance further.



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

Figure 3: The expected performance for changing number of training samples for various dimensional problems.

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, 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 number of 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) $N = 50$

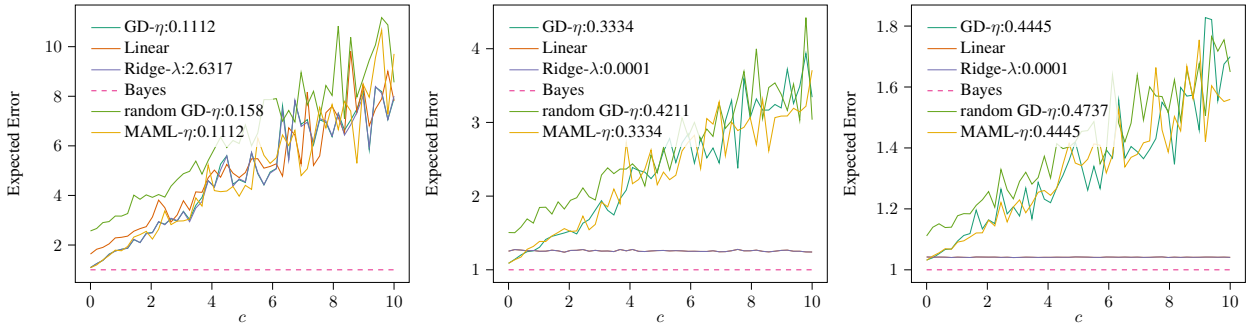
Figure 4: The expected performance for changing number of dimensions for various number of training points.

Effect of Task Variance c: The results of increasing task variance for various problem dimensions and various number 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 Ridge model are only affected by this phenomenon only for problem dimensionality $D \geq N$. In the light of this, observation another important result is that for $N \geq D$ for small task variance the gradient descent variants, other than the randomly initialized model, the expected performance is lower than the Ridge model. Although 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 is 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. Although, this surprising performance of MAML-like algorithms, the performance of the Ridge model is much more stable and performs reasonably better than gradient-based methods for $N \geq D$.

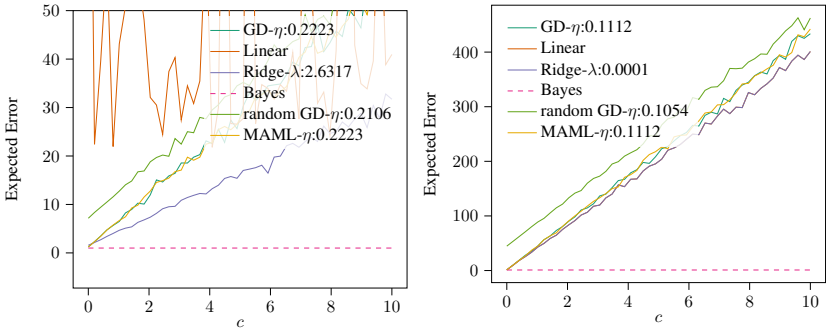
Effect of Number of Gradient Steps n_{iter}: Looking at the interesting results observed from the task variance c the effect of number of gradient steps taken becomes more relevant. These results can be seen in Figure 6. It can be observed from Figure 6a that for a low number of training samples the 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 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

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



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



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

Figure 5: The expected performance for changing number of training samples for various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 5a, 5b, 5c and the effect of increasing dimensionality can be seen by looking at Figures 5b, 5d, 5e.

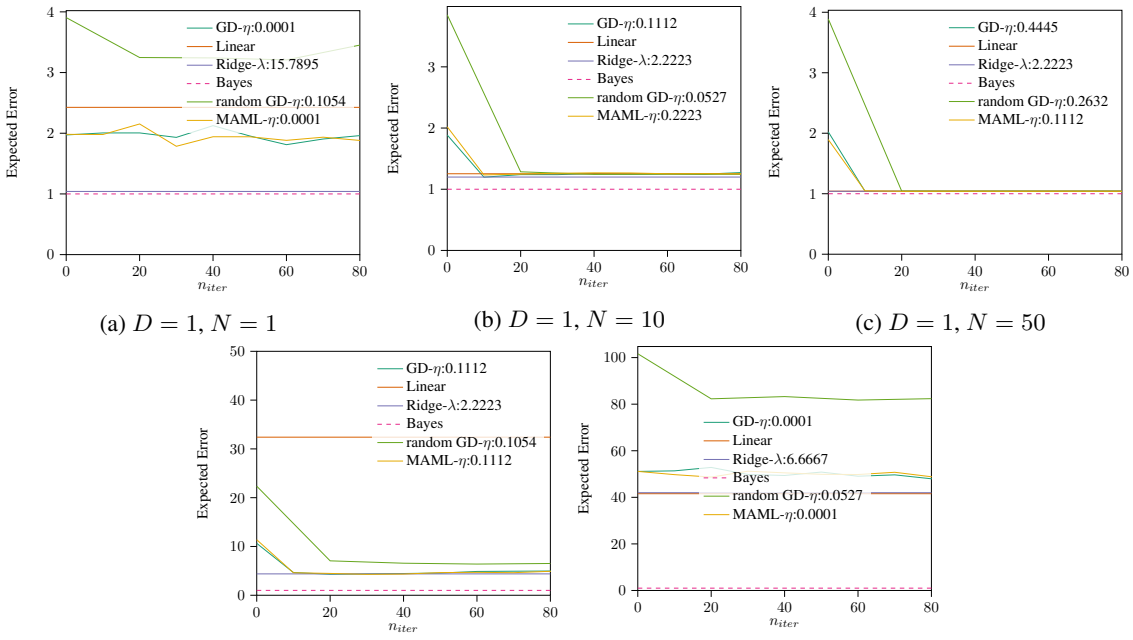
steps for the convergence. Moreover, for the $D = N$ case it even improves generalization after convergence too (see Figures 6d and 6a). Overall, it can be observed that the increasing n_{iter} converges towards the Ridge model variants with the exception of $D = 1$ and $N = 1$ case.

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 same as the Linear model. Furthermore, other models which have prior task information do not seem to be affected from 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 $N \leq D$.

4.1.2 Nonlinear Problem

The nonlinear problem introduced in Section 2.1.2 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 c_2, 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 configuration is under investigation, the default values are utilized. And, the default values are given as,

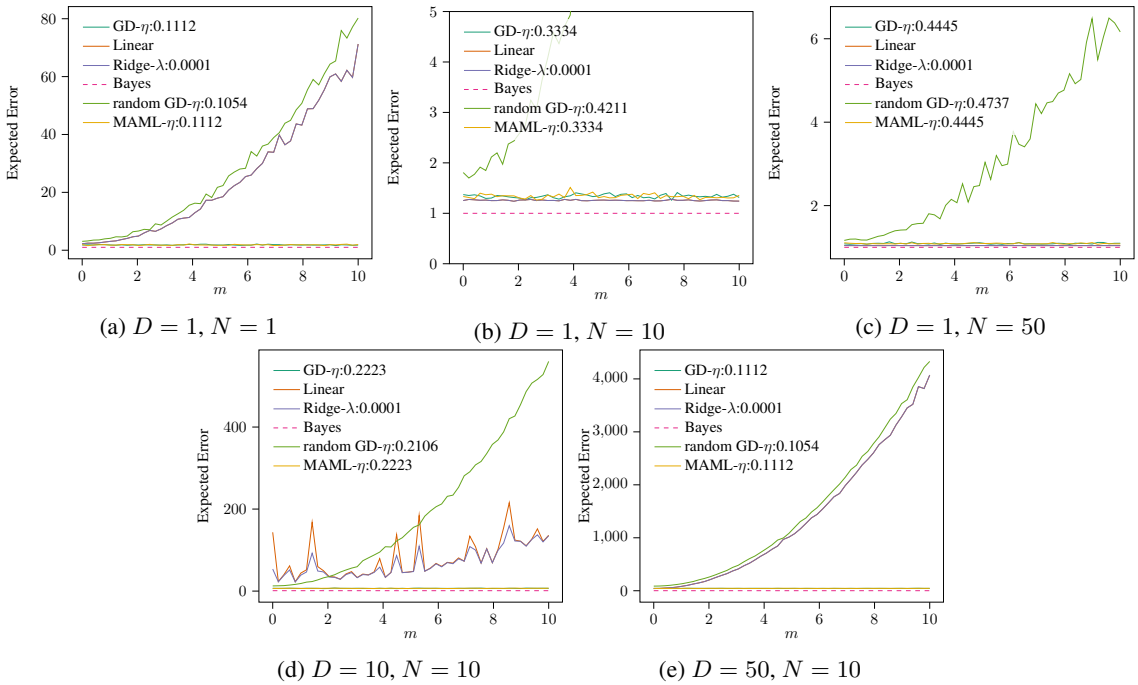
- $\sigma = 1$,



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

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

Figure 6: The expected performance for changing number of gradient steps n_{iter} with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 6a, 6b, 6c and the effect of increasing dimensionality can be seen by looking at Figures 6b, 6d, 6e.



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

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

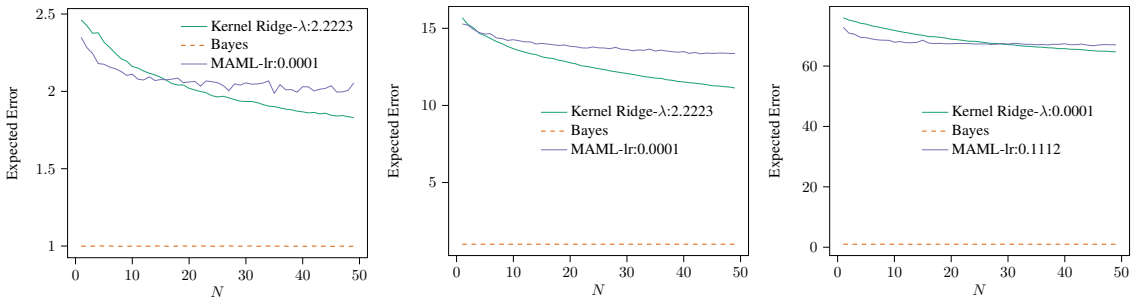
Figure 7: The expected performance for changing task mean m with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 7a, 7b, 7c and the effect of increasing dimensionality can be seen by looking at Figures 7b, 7d, 7e.

- $m_1 = 1$,
- $m_2 = 0$,

- $c_1 = 2$,
- $c_2 = 2$,
- $k = 1$,
- $n_{iter} = 5$.

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 8 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 the 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 fact that the number of gradient steps limitation. Although the expected error is quite high for increasing dimensionality, the results obtained for $D = 1$ (Figure 8a) is still an effective result that shows 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) $D = 50$

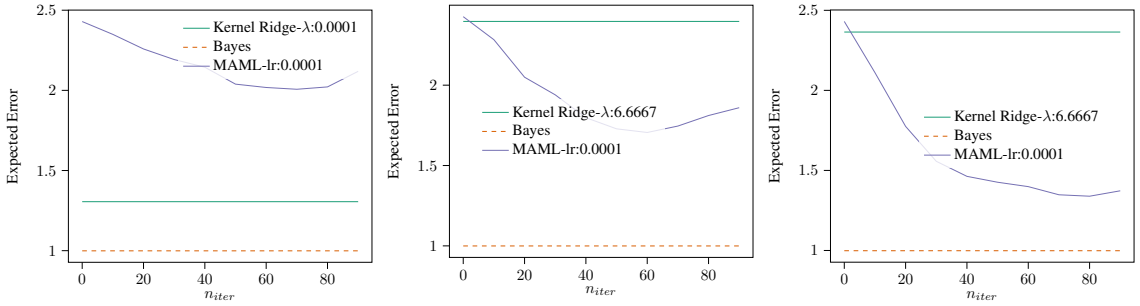
Figure 8: The expected performance for changing number of training samples for various dimensional problems.

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.

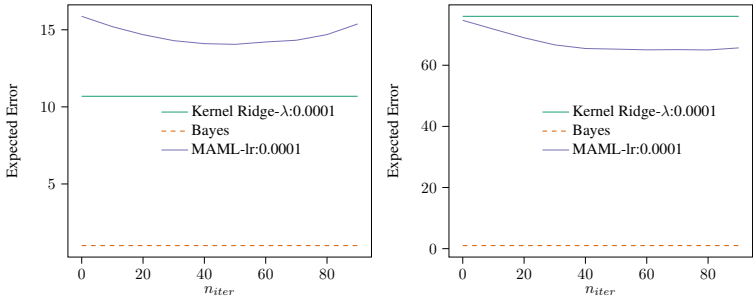
Effect of Phase Task Variance c_2: Remembering that the task variance effect for the linear problem had some interesting properties where even a single gradient step resulted in better expected performance. One might wonder if that is the case for the nonlinear problem as well. As can be seen in Figure 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".

4.2 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. This investigation 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 behaviour is investigated in [14] as well, where the large learning rate in training phase acts as a regularizer due to the discrepancy

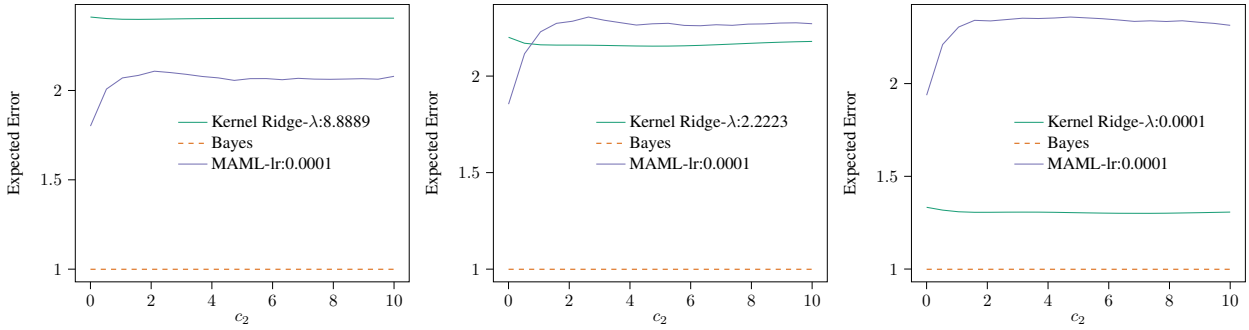


(a) $D = 1, N = 1$ (b) $D = 1, N = 10$ (c) $D = 1, N = 50$

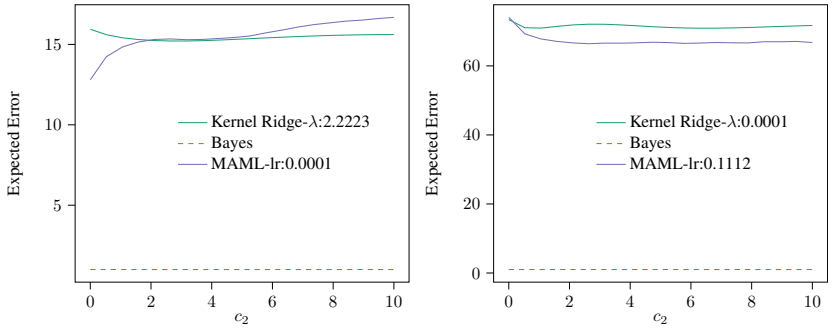


(d) $D = 10, N = 10$ (e) $D = 50, N = 10$

Figure 9: The expected performance for changing number of gradient steps n_{iter} with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 9a, 9b, 9c and the effect of increasing dimensionality can be seen by looking at Figures 9b, 9d, 9e.



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



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

Figure 10: The expected performance for changing number of training samples for various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 10a, 10b, 10c and the effect of increasing dimensionality can be seen by looking at Figures 10b, 10d, 10e.

between train and test loss landscapes. A similar behaviour can be observed for the MAML with nonlinear problem too due to having a non-convex problem.

This limitation of adaptation steps is increased in [12, 15] that try to improve MAML adaptation step so that the adaptation is limited to fewer gradient steps preferably one. Our findings suggests that the expected performance of these methods should be investigated as well as some of the generalization power of the 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 the MAML is feature reuse instead of rapid learning.

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 regularization parameter is selected for the whole expected performance curves individually. We showed that even in the case of a general regularization when enough data is present a single task learner can outperform on expectation a meta learner when the tasks observed start to deviate from each other. This indicates that the regularization based meta-learners similar to ones presented in [17], 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 [7] for MAML can prove useful to understand and study MAML.

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.

5 Conclusion

The use of quick adaptation constraint under a supervised learning setting where this constraint is not imposed by the problem is investigated. It is found that there exists a regularization effect 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.

Acknowledgments

References

[1] Sebastian Thrun and Lorien Pratt, editors. *Learning to Learn*. Springer US, Boston, MA, 1998.

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:1703.03400 [cs]*, July 2017.

[3] Sebastian Flennerhag, Pablo G Moreno, Neil D Lawrence, and Andreas Damianou. TRANSFERRING KNOWLEDGE ACROSS LEARNING PROCESSES. page 23, 2019.

[4] Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms. *arXiv:1803.02999 [cs]*, October 2018.

[5] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. iTAML: An Incremental Task-Agnostic Meta-learning Approach. *arXiv:2003.11652 [cs, stat]*, March 2020.

[6] Liam Collins, Aryan Mokhtari, and Sanjay Shakkottai. Task-Robust Model-Agnostic Meta-Learning. *arXiv:2002.04766 [cs, math, stat]*, June 2020.

[7] Simon Guiroy, Vikas Verma, and Christopher Pal. Towards Understanding Generalization in Gradient-Based Meta-Learning. *arXiv:1907.07287 [cs, stat]*, July 2019.

[8] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your MAML, March 2019.

[9] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting Gradient-Based Meta-Learning as Hierarchical Bayes. *arXiv:1801.08930 [cs]*, January 2018.

[10] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic Model-Agnostic Meta-Learning. *arXiv:1806.02817 [cs, stat]*, October 2019.

[11] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, Broader and Artier Domain Generalization. *arXiv:1710.03077 [cs]*, October 2017.

[12] Harkirat Singh Behl, Atılım Güneş Baydin, and Philip H. S. Torr. Alpha MAML: Adaptive Model-Agnostic Meta-Learning, May 2019.

[13] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. The Omniglot challenge: A 3-year progress report, May 2019.

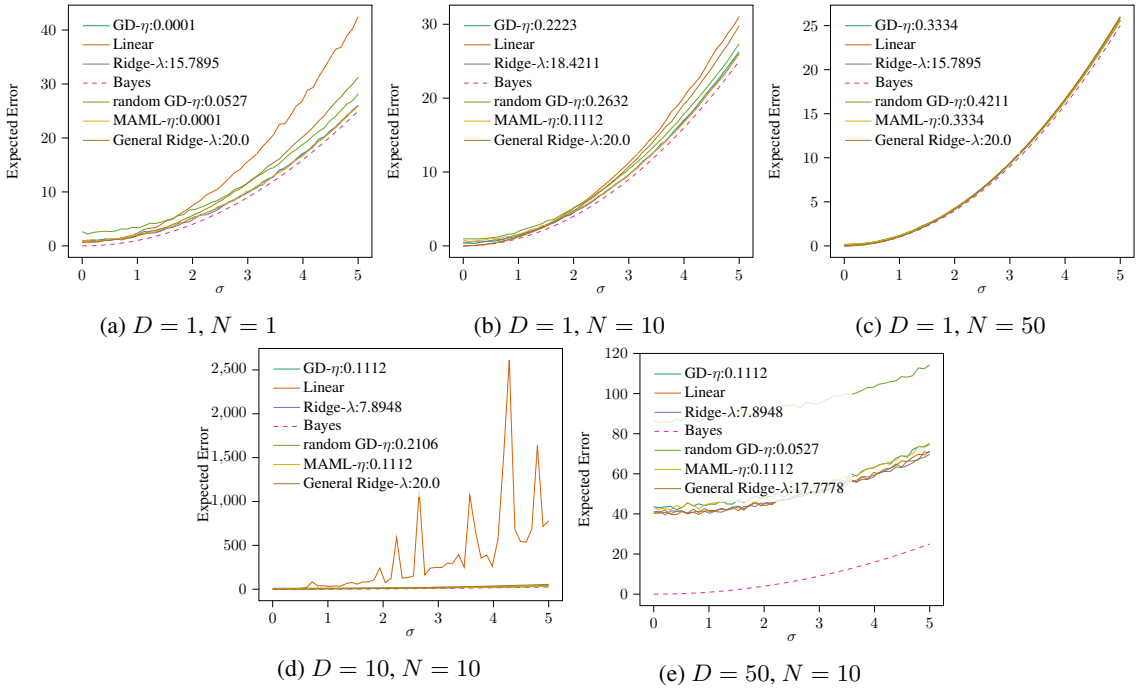
[14] Preetum Nakkiran. Learning Rate Annealing Can Provably Help Generalization, Even for Convex Problems. *arXiv:2005.07360 [cs, stat]*, May 2020.

[15] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-SGD: Learning to Learn Quickly for Few-Shot Learning. *arXiv:1707.09835 [cs]*, September 2017.

[16] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML, February 2020.

[17] Giulia Denevi, Carlo Ciliberto, Dimitris Stamos, and Massimiliano Pontil. Incremental Learning-to-Learn with Statistical Guarantees. *arXiv:1803.08089 [cs, stat]*, March 2018.

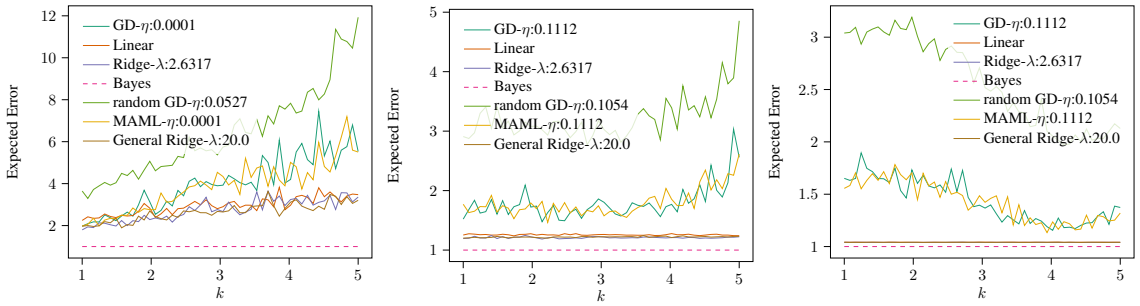
6 Appendix



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

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

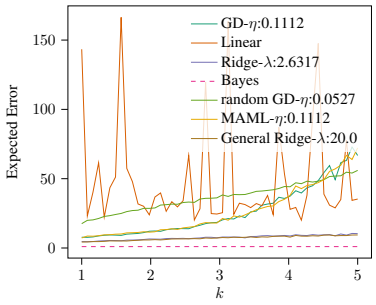
Figure 11: [**Linear Problem**] The Expected Error for changing noise standard deviation σ with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 11a, 11b, 11c and the effect of increasing dimensionality can be seen by looking at Figures 11b, 11d, 11e.



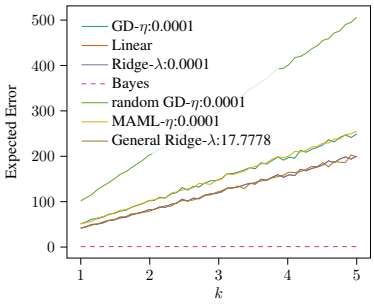
(a) $D = 1$, $N = 1$

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

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

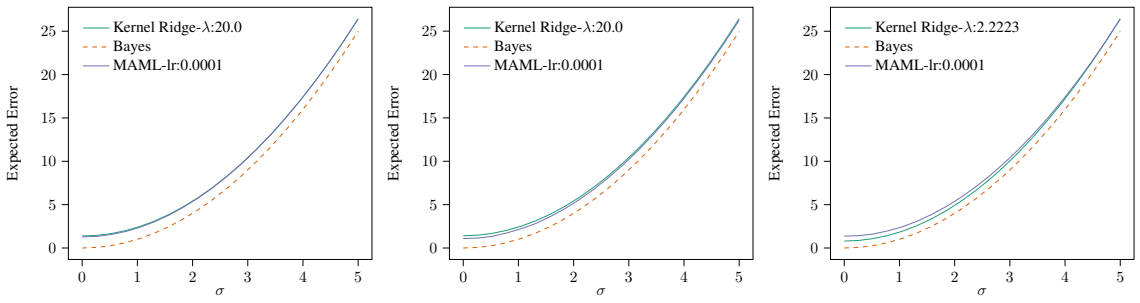


(d) $D = 10$, $N = 10$



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

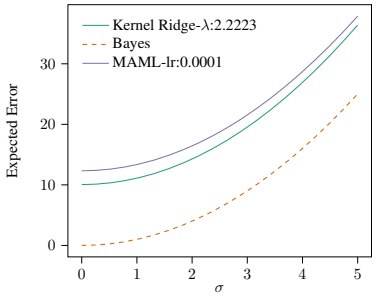
Figure 12: [**Linear Problem**] The Expected Error for changing input variance k with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 12a, 12b, 12c and the effect of increasing dimensionality can be seen by looking at Figures 12b, 12d, 12e.



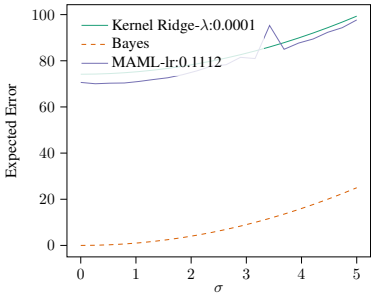
(a) $D = 1$, $N = 1$

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

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

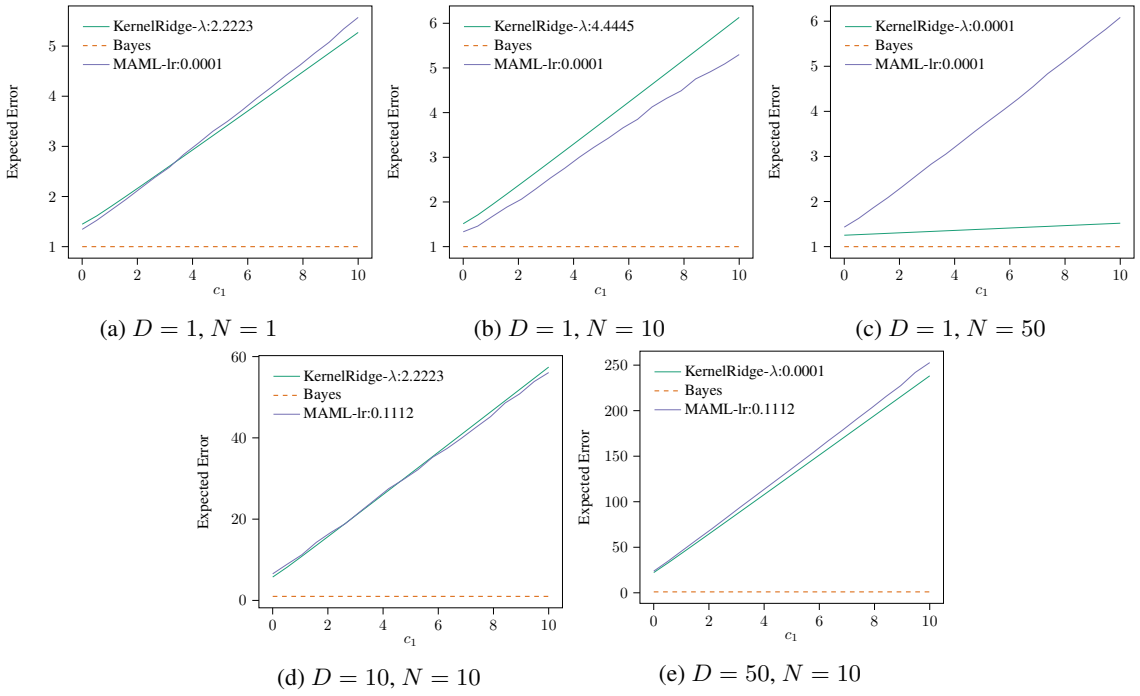


(d) $D = 10$, $N = 10$



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

Figure 13: [**Nonlinear Problem**] The Expected Error for changing noise standard deviation σ with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 13a, 13b, 13c and the effect of increasing dimensionality can be seen by looking at Figures 13b, 13d, 13e.



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

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

Figure 14: [**Nonlinear Problem**] The Expected Error for changing variance of amplitude of the task c_1 with various training samples and various dimensional problems. For the given parameter the effect of increasing number of training samples can be seen by looking at Figures 14a, 14b, 14c and the effect of increasing dimensionality can be seen by looking at Figures 14b, 14d, 14e.