Learning to Forget for Meta-Learning
Supplementary Materials

Sungyong Baik Seokil Hong Kyoung Mu Lee
ASRI, Department of ECE, Seoul National University
{dsybaik, hongceo96, kyoungmu}@snu.ac.kr

A. Loss Landscape

In [11], they analyze the stability and smoothness of the optimization landscape by measuring Lipschitzness and the "effective" β-smoothness of loss. We use these measurements to analyze learning dynamics for both MAML and our proposed method during training on 5-way 5-shot mini-ImageNet classification tasks. In Figure A, we start with investigating fast-adaptation (or inner-loop) optimization. At each inner-loop update step, we measure variations in loss (Figure A(a)), the l_2 difference in gradients (Figure A(b)), and the maximum difference in gradient over the distance (Figure A(c)), as we move to different points along the computed gradient for that gradient descent. We take an average of these values over the number of inner-loop updates and plot them against training iterations. With a similar approach, we also analyze the optimization stability of fast adaptation to validation tasks at every epoch (Figure B). The measurements were averaged over (the number of validation tasks $\times$ the number of inner-loop update steps).

At the initial stages of training, L2F appears to struggle more, while optimization of MAML seems more stable. This may seem contradictory at first but this actually validates our argument about conflicts between tasks even further. At the beginning, the MAML initialization is not trained enough and thus does not have sufficient prior knowledge of task distribution yet. As training proceeds, the initialization encodes more information about task distribution and encounters conflicts between tasks more frequently.

As for L2F, the attenuator network g_ϕ initially does not have enough knowledge about the task distribution and thus generates meaningless attenuation γ, deteriorating the initialization. But, the attenuator network increasingly encodes more information about the task distribution, generating more appropriate attenuation γ that corresponds to tasks well. The generated γ accordingly allows for a learner to *forget* the irrelevant part of prior knowledge to help fast adaptation, as illustrated by increasing stability and smoothness of landscape in Figure A. The similar observation can be made from B, illustrating the generalizability and the robustness of the proposed method to unseen tasks.

We also investigate the optimization landscape of learning the initialization θ itself for both MAML and L2F in Figure C. The figure demonstrates that the more stable and smoother landscape is realized by L2F. Because the task-dependent layer-wise attenuation allows for *forgetting* the irrelevant or conflicting part of prior knowledge present in the initialization θ, it lifts a burden of trying to resolve conflicts between tasks from θ, allowing for more stable training of the initialization itself.

B. Extended Experiments on Classification

To further validate that our method consistently provides benefits regardless of scenarios, we compare our method against the baseline on additional datasets that have been recently introduced: FC100 (Fewshot-CIFAR100) [7] and CIFAR-FS (CIFAR100 few-shots) [1]. Both aim for creating more challenging scenarios by using low resolution images (32×32, compared to 84×84 in miniImageNet [8] and tieredImageNet [9]) from CIFAR100 [4]. These two datasets differ in how they create the train/val/test splits of CIFAR100. While CIFAR-FS follows the procedure that

	FC100		
	1-shot	5-shot	10-shot
MAML*	$35.98 \pm 0.48\%$	$51.40 \pm 0.50\%$	$56.13 \pm 0.50\%$
MAML+L2F	$39.46 \pm 0.49\%$	$53.12 \pm 0.50\%$	$59.72 \pm 0.49\%$

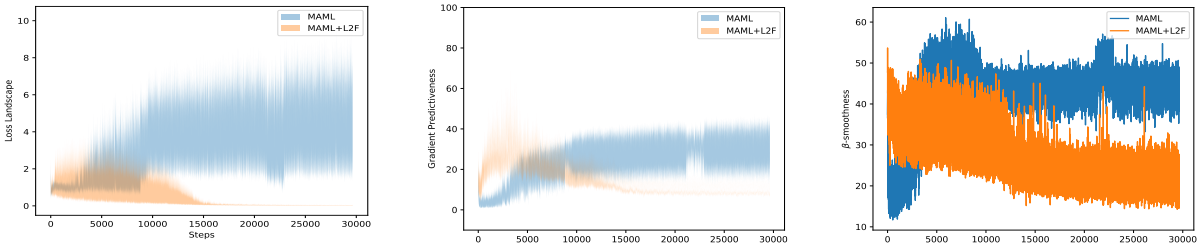
* Our reproduction.

Table A: Test accuracy on FC100 5-way classification

	CIFAR-FS	
	1-shot	5-shot
MAML*	$53.91 \pm 0.50\%$	$70.16 \pm 0.46\%$
MAML+L2F	$57.28 \pm 0.49\%$	$73.94 \pm 0.44\%$

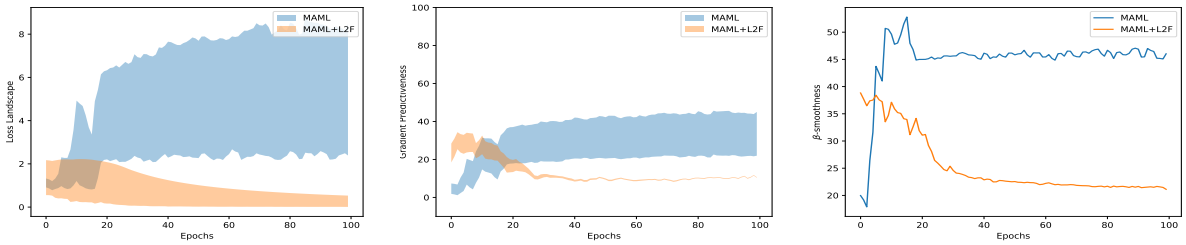
* Our reproduction.

Table B: Test accuracy on CIFAR-FS 5-way classification



(a) loss landscape (b) gradient predictiveness (c) "effective" β smoothness

Figure A: Analysis of the optimization landscape of the fast adaptation to tasks from the meta-training set. In each subfloat, averaged values are shown for each training iteration.



(a) loss landscape (b) gradient predictiveness (c) "effective" β smoothness

Figure B: Analysis of the optimization landscape of the fast adaptation to tasks from the meta-validation set. In each subfloat, averaged values are shown for each training epoch.

was used for miniImageNet, FC100 aligns more with the goal of tieredImageNet in that they try to minimize the amount overlap between splits by splitting based on super-classes. Table A presents results for FC100 and Table B for CIFAR-FS. We also perform additional experiments on Meta-Dataset [14], which is a combination of diverse datasets and hence poses more challenging scenarios, where conflicts can occur among tasks more frequently. Table C shows that our proposed method resolves conflicts better than MAML even under challenging scenarios.

C. Regression

C.1. Additional Qualitative results

In Figure D and E, we show a random sample of qualitative results from the k-shot sinusoid regression, where $k \in [5, 10]$. The target function (or true function) is a sine curve $y(x) = Asin(wx + b)$ with the amplitude A, frequency ω, phase b, and the input range $[-5.0, 5.0]$. The sampling range of amplitude, frequency, and phase defines a task distribution. In Figure D, we follow the general settings in [2, 5], where amplitude A, frequency ω, and phase b are sampled from the uniform distribution on intervals $[0.1, 5.0]$, $[0.8, 1.2]$, and $[0, \pi]$, respectively. MAML+L2F demonstrates more accurate regression for both 5 and 10-shot cases, compared to the baseline, MAML. To further stress the generalization of the MAML+L2F initialization, we extensively increase the degree of conflicts between new tasks and the prior knowledge. To that end, we modify the setting such that amplitude, frequency, and phase are sampled from the non-overlapped ranges for training and evaluation. In training, amplitude A, frequency ω, and phase b are sampled from the uniform distribution on intervals $[0.1, 3.0]$, $[0.8, 1.0]$, and $[0, \pi/2]$, respectively. In evaluation, amplitude A, frequency ω, and phase b are sampled from the uniform distribution on intervals $[3.0, 5.0]$, $[1.0, 1.2]$, and $[\pi/2, \pi]$, respectively. In Figure E, our method(MAML+L2F) exhibits better fitting and thus claims the better generalization than MAML for both 5 and 10-shot regression.

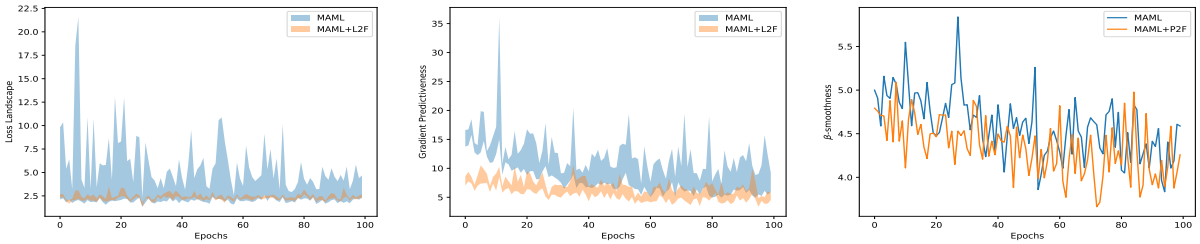
C.2. Additional Quantitative results

In Table D, we compare the proposed method against other advanced MAML-based methods, which are generalizable across domains, specifically MuMoMAML and MAML++. As with results on classification, our method consistently outperforms in regression task.

D. Reinforcement Learning

D.1. Additional Qualitative results

The qualitative results for the 2D navigation experiments are shown in Figure F. In training, the position of starting point is fixed at $[0, 0]$ and the position of destination is randomly sampled from space $[-0.5 \times 0.5, -0.5 \times 0.5]$,



(a) loss landscape (b) gradient predictiveness (c) "effective" β smoothness

Figure C: Analysis of the optimization landscape of the initialization learning dynamics.

Model	ILSVRC	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic signs	MSCOCO
MAML	19.35 ± 0.84	66.14 ± 1.47	40.20 ± 1.05	40.61 ± 1.79	38.94 ± 1.69	42.46 ± 1.54	13.80 ± 1.19	61.07 ± 1.50	23.38 ± 1.12	13.29 ± 1.11
Ours	25.93 ± 1.10	72.26 ± 1.63	53.31 ± 1.48	42.62 ± 1.30	49.57 ± 1.03	50.28 ± 1.67	20.20 ± 1.08	64.23 ± 1.31	31.71 ± 1.45	19.75 ± 0.93

Table C: Test accuracy (%) of MAML (our reproduction) vs MAML+L2F on Meta-Dataset

which is the same experiment procedure from [2]. Velocity is clipped to be in the range $[0.2, 0.2]$. In evaluation, we performed experiments with four different task distributions. In Figure F(a), the task distribution for evaluation is the same as for training. On the other hand, as in regression experiment E, we perform additional 3 experiments (Figure F(b), (c), (d) that evaluate models under extreme conditions, where the task distribution for evaluation is chosen to be different from the task distribution for training. In Figure F(b), the staring point is no longer fixed but rather sampled from space $[-0.5 \times 0.5, -0.5 \times 0.5]$. In Figure F(c), the position of starting point is fixed at $[0, 0]$. However, the position of the ending point is sampled from a larger space $[-2.0 \times 2.0, -2.0 \times 2.0]$. In Figure F(d), both the starting and destination positions are sampled from space $[-2.0 \times 2.0, -2.0 \times 2.0]$. Overall, our proposed method demonstrates more accurate and robust navigation, compared to the baseline MAML.

E. Implementation Details

E.1. classification

E.1.1 Experiment Setup

We use the standard settings [2] for N-way k-shot classification in both miniImageNet[8] and tieredImageNet[9]. When calculating gradients for fast adaptation to each task, the number of examples $\mathcal{D}$ used is either N. The fast adaptation is done via 5 gradient steps with the fixed step size, $\alpha = 0.01$ for all models, except LEO and LEO+L2F during both training and evaluation. Gradients for meta-updating the networks f_θ and g_ϕ are calculated with 15 number of examples $\mathcal{D}'$ at each iteration. The MAML and its variants were trained for 50000 iterations in miniImageNet and 125000 in tieredImageNet to account for the larger number of examples as in [6]. The meta batch size of tasks is set to be 2 for 5-shot and 4 for 1-shot, with the exception that the batch size is 1 for ResNet12 in miniImageNet and tieredImageNet. This is due to the limited memory and the heavy computation load from the combination of second-order gradient computation, large image size, and a larger network. As for experiments with LEO, we follow the exact setup from LEO [10]. We only add attenuation process before adaptation in latent space and fine-tuning in parameter space for LEO+L2F.

E.1.2 Network Architecture for f_θ

4 conv As with most algorithms [15, 8, 12, 13] that use 4-layer CNN as a backbone, we use 4 layers each of which contains 64-filter 3×3 convolution filters, a batch normalization [3], a Leaky ReLU nonlinearity, and a 2×2 max pooling. Lastly, the classification linear layer and softmax are placed at the end of the network.

ResNet12 As for the ResNet12 architecture, the network consists of 4 residual blocks. each residual block consists of three 3×3 convolution layers. The first two convolution layers are followed by a batch normalization and a Leaky ReLU nonlinearity. The last convolution layer is followed by a batch normalization and a skip connection that contains a 1×1 convolution layer and a batch normalization. After a skip connection, a Leaky ReLU nonlinearity and a max 2×2 are placed at the end of each residual block.

The number of convolution filters for 4 residual blocks is set to be 64, 128, 256, 512 for 4 residual blocks in the increasing order of depth.

E.1.3 Network Architecture for g_ϕ

g_ϕ is a 3-layer MLP, with each layer of l hidden units, where l is the number of layers of the main network, f_θ. Activation

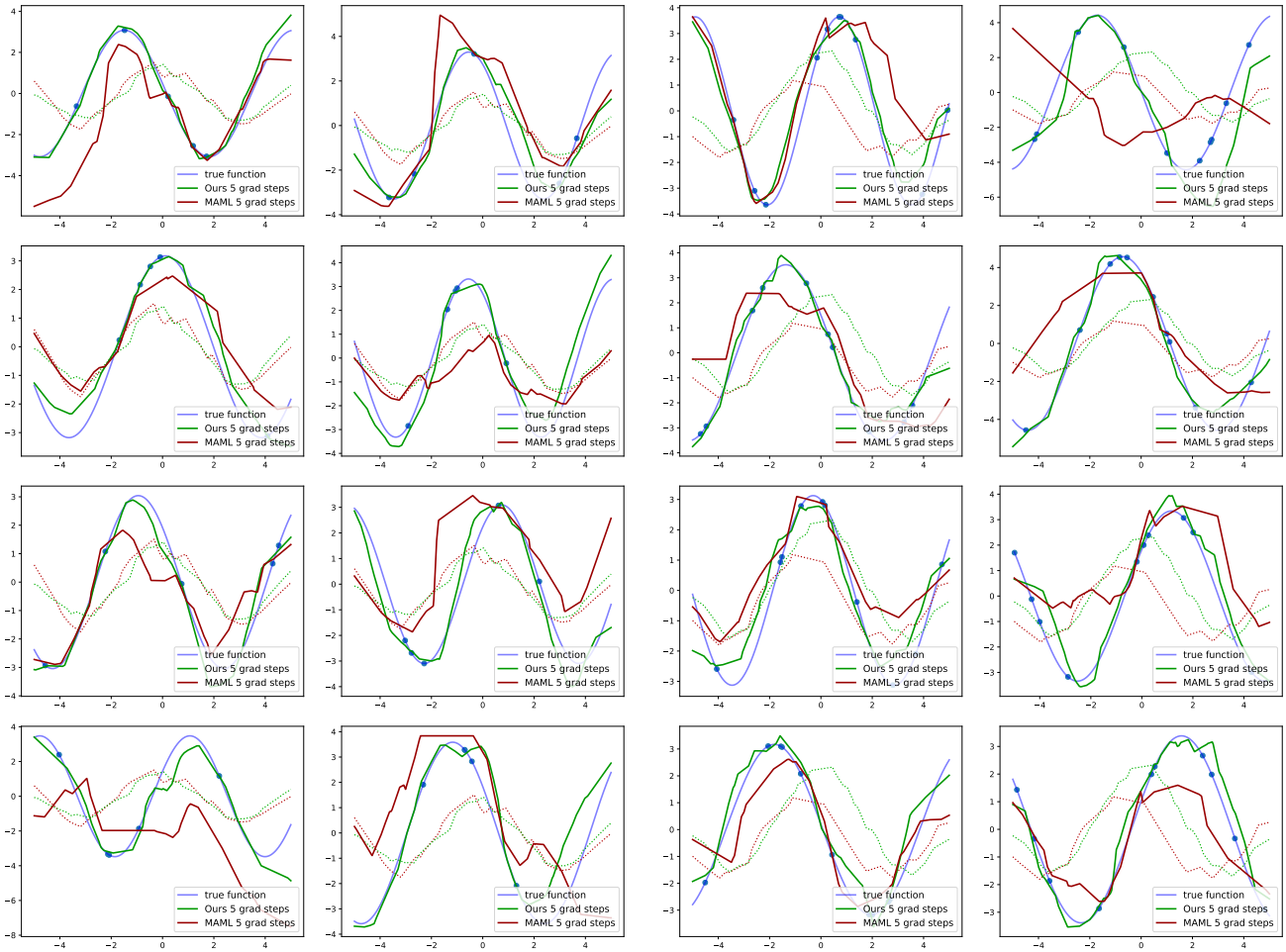
	Models	1 step	2 steps	5 steps
5-shot	MAML	1.2247	1.0268	0.8995
	MuMoMAML	1.1010	0.9291	0.8615
	MAML++	1.2028	0.9268	0.7547
	Ours	**1.0537**	**0.8426**	**0.7096**

(a) Regression

	Models	1 step	2 steps	3 steps
2D Navi	MAML	-32.626	-25.746	-20.734
	MuMoMAML	-25.785	-23.705	-19.747
	MAML++	-36.281	-27.264	-18.620
	Ours	**-24.230**	**-19.598**	**-16.517**

(b) RL

Table D: Additional Quantitative Experiments



(a) 5 shot regression

(b) 10 shot regression

Figure D: Qualitative results for the K$\in [5, 10]$-shot sinusoid regression$(y(x) = Asin(x + b))$. Parameters are sampled from the same distribution for training and evaluation.

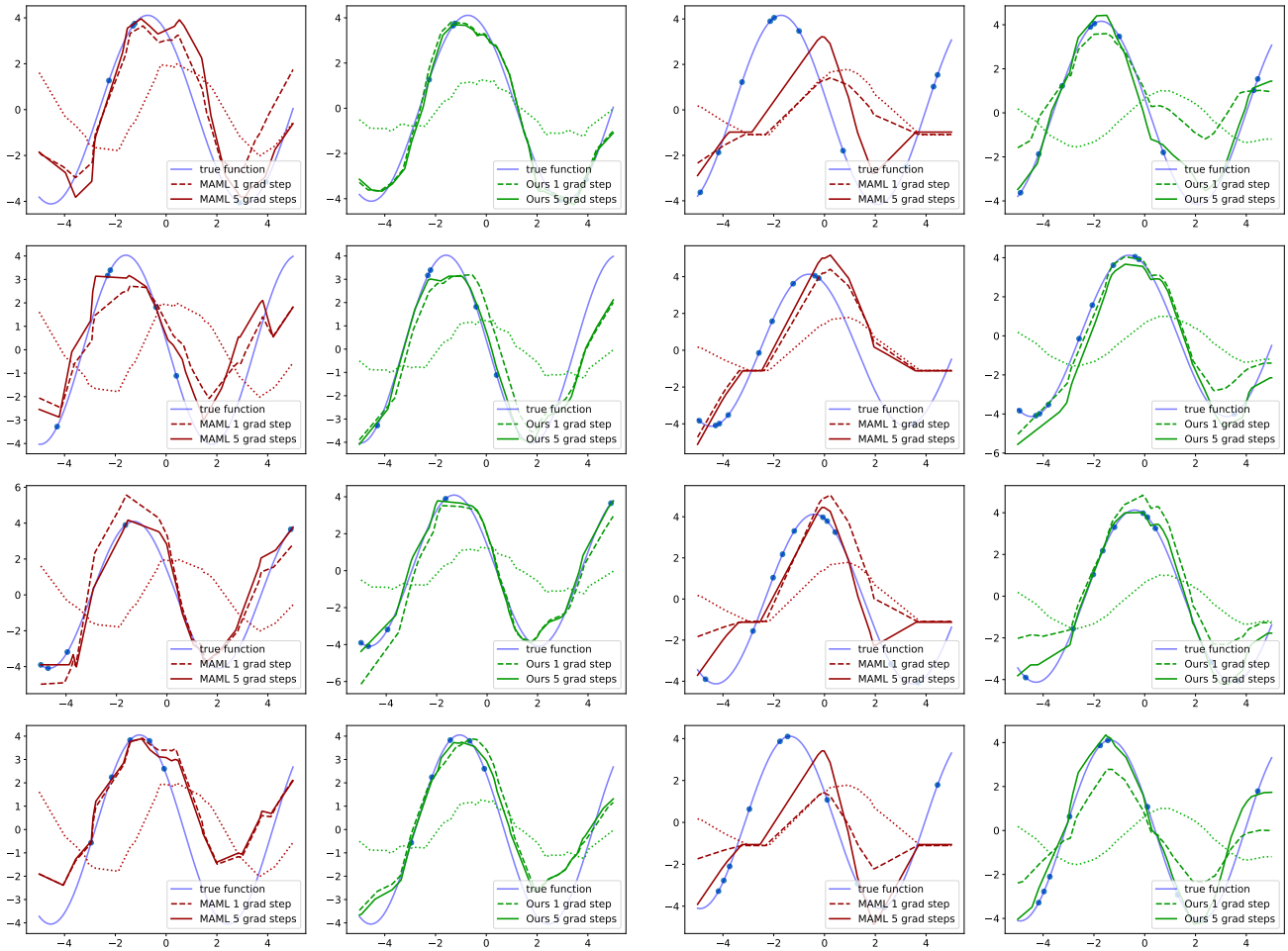
functions are ReLU in between and a sigmoid at the end. The input is layer-wise mean of gradients. As for LEO, the case is a bit different because they perform one adaptation in latent codes and one on decoded classifier weights. Thus, we introduce two 3-layer MLPs, one for each. Again, for each MLP, the numbers of hidden units for each layer is n, where n is the dimension of latent codes or the number of classifier weights.

E.2. Regression and RL

The details of the few shot regression and reinforcement learning experiments are listed in Table E.

E.3. System

All experiments were performed on a single NVIDIA GeForce GTX 1080Ti.



(a) Non-overlapped task distributions on 5-shot

(b) Non-overlapped task distributions on 10-shot

Figure E: Qualitative results for the K$\in [5, 10]$-shot sinusoid regression($y(x) = Asin(x + b)$). Parameters are sampled from non-overlapped ranges for training and evaluation.

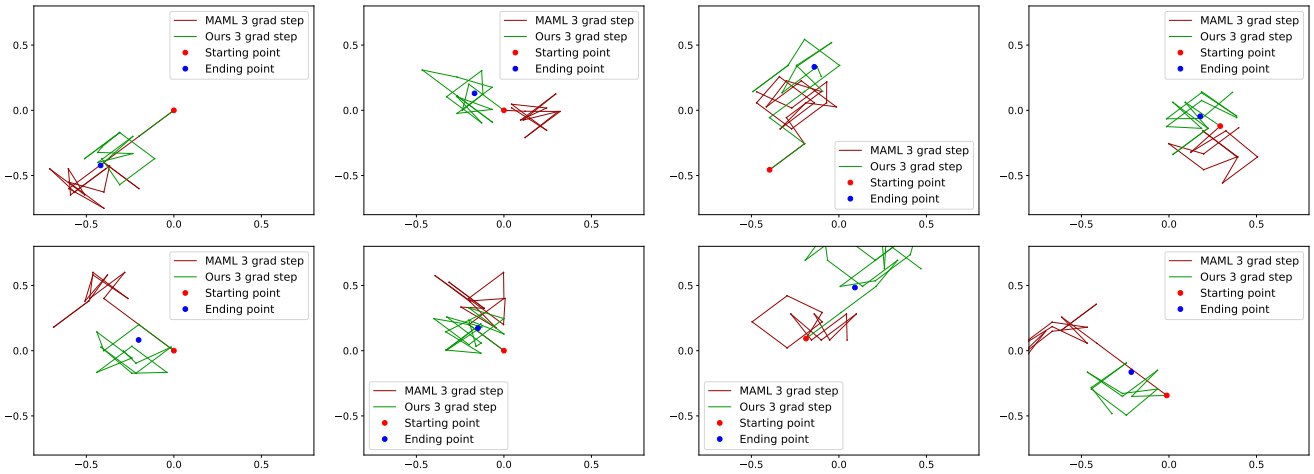
	Hyperparameters
policy network	2 hidden layers of size 40 with ReLU
training iterations	50000 epochs
inner update α	0.01
meta update optimizer	Adam
meta batch size	4
k shot	[5,10,20]
loss function	MSE loss
eval	randomly sample 100 sine curves, sample 100 examples (repeated 100 times)

(a) Regression

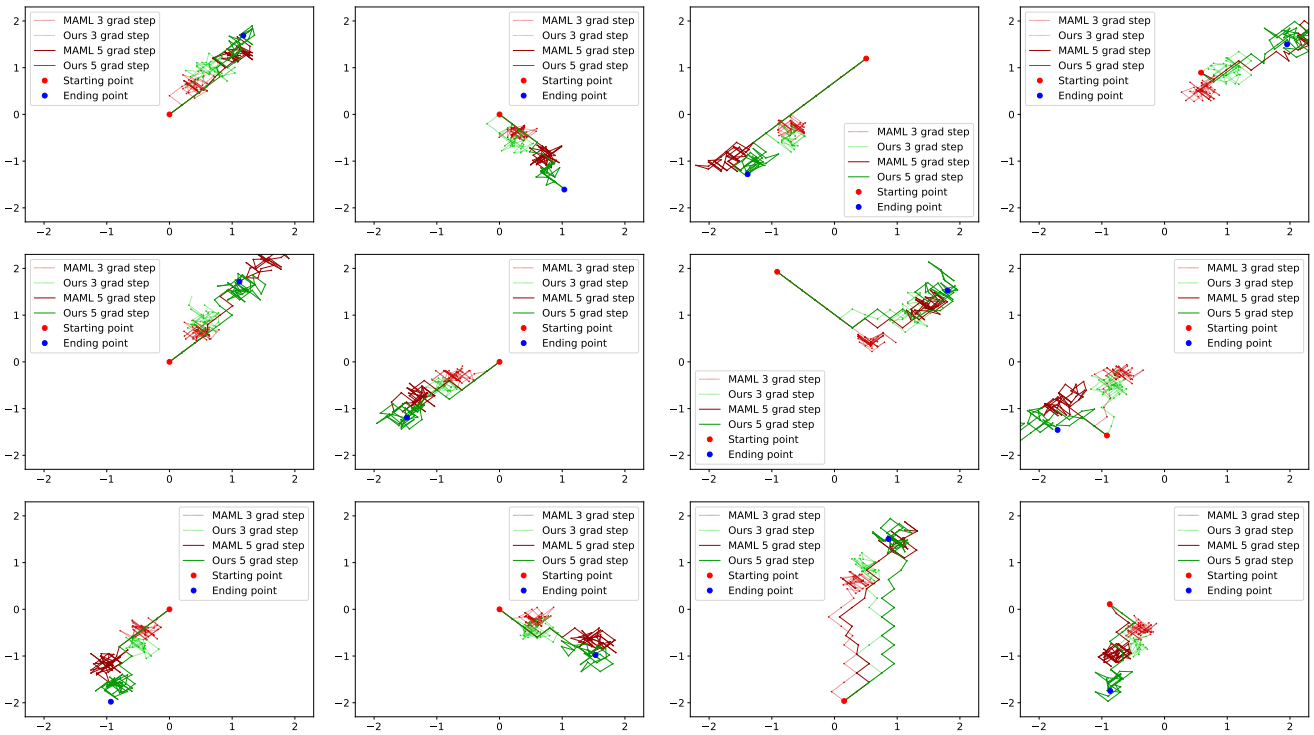
	Hyperparameters
policy network	2 hidden layers of size 100 with ReLU
inner update	vanilla policy gradient
inner update α	0.01
meta-optimizer	TRPO
training iterations	500 epochs, choose best model
MuJoCo horizon	200
MuJoCo batch size	40
MUJoCo evals	update 4 gradient updates, each with 40 samples for a task
2D navigation horizon	100
2D navigation batch size	20
2D navigation evals	update 4 gradient updates, each with 20 samples for a task

(b) RL

Table E: Implementation Details for Regression and RL



(a) Start $= [0.0]$, End $\in [-0.5, 0.5] \times [-0.5, 0.5]$ (b) Start, End $\in [-0.5, 0.5] \times [-0.5, 0.5]$



(c) Start $= [0.0]$, End $\in [-2.0, 2.0] \times [-2.0, 2.0]$ (d) Start, End $\in [-2.0, 2.0] \times [-2.0, 2.0]$

Figure F: Qualitative results for the 2D Navigation task with MAML vs MAML+L2F (Ours) comparison. Only (a) experiment, tasks are sampled from the same distribution for training and evalution, and (b), (c), and (d) experiments, tasks are sampled from the non-overlapped ranges for training and evaluation

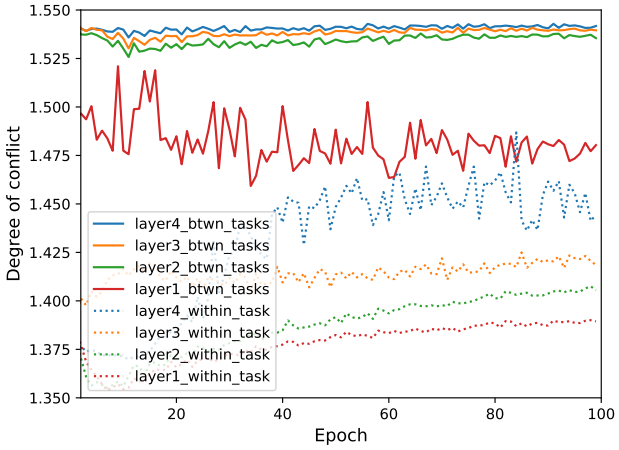


Figure G: Degree of conflict within a task: The degree of conflict within each task is averaged across tasks per each epoch. The degree of conflict within each task is observed to be lower than the degree of conflict between tasks. This is expected as the examples within a task are more similar than examples from different tasks. Thus, the gradients are more aligned within tasks.

F. Conflict Within a Task

In this section, we also measure the degree of conflict that exist within task (conflict between examples in the same task), as illustrated in Figure G. As expected, the degree of conflict within task is observed to be lower than the degree of conflict between tasks. This is because the examples within a task are more similar to each other than examples from different tasks. This leads to gradients being more aligned within a task, while gradients between tasks have a larger amount of disagreement.

References

[1] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019. 1

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 2, 3

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 3

[4] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. University of Toronto, 2009. 1

[5] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few shot learning. *CoRR*, abs/1707.09835, 2017. 2

[6] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *ICLR*, 2019. 3

[7] Boris N. Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NIPS*, 2018. 1

[8] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1, 3

[9] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 1, 3

[10] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *ICLR*, 2019. 3

[11] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *NIPS*, 2018. 1

[12] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017. 3

[13] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H.S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 3

[14] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *CoRR*, abs/1903.03096, 2019. 2

[15] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NIPS*, 2016. 3