On Modulating the Gradient for Meta-Learning (Supplementary Material)

Christian Simon[†,§] Piotr Koniusz[†,§]
Richard Nock[†,‡,§] Mehrtash Harandi[♣,§]

[†]The Australian National University, [♣]Monash University,
[‡]The University of Sydney, [§] Data61-CSIRO
first.last@{anu.edu.au,monash.edu,data61.csiro.au}

Keywords: Meta Learning, Few Shot Learning, Adaptive Gradients

Below, we provide the implementation and training details of our method. The modulation generator is described in detail. Moreover, we provide a step-by-step algorithm for reinforcement learning. Finally, we perform a theoretical analysis, simulations and real-data experiments on the Hadamard-based modulator $\boldsymbol{M}(\Psi, r) \odot \nabla\mathcal{L}$ of the ModGrad unit.

1 Details of Experiments

For experiments on the few-shot image classification, we use three backbones to evaluate the accuracy and the convergence rate of our method, namely 4-convolutional blocks (Conv-4) [1], WRN-28-10 [2], and ResNet-34 [3]. Conv-4 has 64 filters in each layer. The model is trained on episodes without any data augmentation. WRN-28-10 uses the pre-trained model from the *mini*-ImageNet training set following [4]. We trained the ModGrad on WRN-28-10 using episodes with data augmentation. On ResNet-34, following [5], we initialized the weights randomly and trained the model on episodes. Learning rates were fixed to 0.1 and 0.001 for the inner- and the outer-loop on ResNet-34, respectively (though heavy fine-tuning the learning rates could potentially improve the results marginally). The models are trained with 100K episodes by the Adam [6] optimizer in the outer-loop.

2 Implementation Details of the Gradient Modulation Generator

Below we detail how the gradient modulation generator can be incorporated into a Fully Connected (FC) layer or a convolutional layer. The preconditioner generator is attached to a layer of a neural network. During training, this generator is fixed in the inner-loop but it is updated in the outer-loop.

FC Layer. An FC layer consists of a weight matrix ($\boldsymbol{W} \in \mathbb{R}^{D_1 \times D_2}$) and an additional bias ($\boldsymbol{b} \in \mathbb{R}^{D_2}$). A ModGrad cell ($\psi_i$) consists of two small networks

called *sister networks* which produce the weight and the bias, collectively. For instance, in the case of an FC layer, the weight matrix is preconditioned by the sister networks (ϕ_1, ϕ_2) using two tall matrices. Subsequently, the outer product of these matrices is vectorised to produce the modulator. Technically speaking, ϕ_j generates a vector which is then reshaped/partitioned to $\boldsymbol{\omega}_j \in \mathbb{R}^u$ and $\boldsymbol{h}_j \in \mathbb{R}^{D_j \times u}$ where u is also the number of attention weights. If a bias is also used then we also have $\boldsymbol{\phi}_3$ producing $\boldsymbol{\omega}_3 \in \mathbb{R}^u$ and $\boldsymbol{h}_3 \in \mathbb{R}^{D_2 \times u}$.

Convolutional Layer. Consider a weight matrix ($\boldsymbol{W} \in \mathbb{R}^{D_1 \times D_2 \times c_1 \times c_2}$) of a convolutional block where $c_1 \times c_2$ denotes the kernel size. To generate the modulator for $\boldsymbol{W}$, we again use the *sister networks* (ϕ_1, ϕ_2) as in the case of FC layer but we consider ϕ_1 yielding $\boldsymbol{\omega}_1 \in \mathbb{R}^u$, and $\boldsymbol{h}_1 \in \mathbb{R}^{D_1 \times u}$ and ϕ_2 yielding $\boldsymbol{\omega}_2 \in \mathbb{R}^u$ and $\boldsymbol{h}_2 \in \mathbb{R}^{(D_2 c_1 c_2) \times u}$. Thus, the outer product can be reshaped to the dimension of the weight matrix $\boldsymbol{W}$. For a bias term, the same approach as for the FC layer is applied.

Implementation Details. Every sister network ϕ_j consists of two FC layers with weight sizes of $\mathbb{R}^{d \times D_j}$ and $\mathbb{R}^{D_j \times (u + uD_j)}$. As shown in Fig. A1, a ReLU activation function is inserted between these two layers.

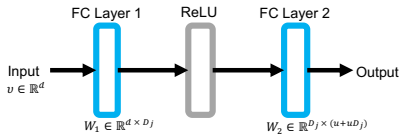


Fig. A1: The architecture of our sister network ϕ.

3 Analysis of the modulator $\boldsymbol{M}(\boldsymbol{\Psi}, \boldsymbol{r}) \odot \boldsymbol{\nabla \mathcal{L}}$

Below, we analyze the properties of the ModGrad modulator and its robustness.

Denoising Property of Modulation. Our $\boldsymbol{M}(\boldsymbol{\Psi}, r) \in \mathbb{R}_+^n$ follows in fact an imposed row-column structure due to Eq. (8) (main manuscript). For brevity, we drop the input arguments of $\boldsymbol{M}$ and we assume it to be already of size $n_1 \times n_2 = n$, that is, $\boldsymbol{M} \in \mathbb{R}_+^{n_1 \times n_2}$. Furthermore, let gradient $\nabla \mathcal{L}(\cdot)$ (reshaped to $n_1 \times n_2$) be decomposed into a noise-free gradient term $\boldsymbol{G} \in \mathbb{R}^{n_1 \times n_2}$ and the noise matrix $\hat{\boldsymbol{\epsilon}} \in \mathbb{R}^{n_1 \times n_2}$ such that $\nabla \mathcal{L} = \boldsymbol{G} + \hat{\boldsymbol{\epsilon}}$. Finally, consider the SVD factorization of each matrix, that is, $\boldsymbol{M} = \boldsymbol{U} \boldsymbol{\lambda} \boldsymbol{V}^T$, $\boldsymbol{G} = \boldsymbol{U}' \boldsymbol{\beta} \boldsymbol{V}'^T$ and $\hat{\boldsymbol{\epsilon}} = \boldsymbol{U}^* \boldsymbol{\epsilon} \boldsymbol{V}^{*T}$. Then, consider the factorized version of the modulated gradient $\widetilde{\boldsymbol{G}} = \boldsymbol{M} \odot (\boldsymbol{G} + \hat{\boldsymbol{\epsilon}})$ given as:

$$\begin{aligned}\widetilde{\boldsymbol{G}} &= \sum_{i=1}^{r} \sum_{j=1}^{r'} \lambda_i \boldsymbol{u}_i \boldsymbol{v}_i^T \odot \left(\beta_j \boldsymbol{u}'_j \boldsymbol{v}'^T_j + \epsilon_j \boldsymbol{u}^*_j \boldsymbol{v}^{*T}_j\right) \\ &= \sum_{i=1}^{r} \sum_{j=1}^{r'} \left[\sqrt{\lambda_i} \boldsymbol{u}_i \odot \left(\sqrt{\beta_j} \boldsymbol{u}'_j + \sqrt{\epsilon_j} \boldsymbol{u}^*_j\right)\right] \left[\sqrt{\lambda_i} \boldsymbol{v}_i \odot \left(\sqrt{\beta_j} \boldsymbol{v}'_j + \sqrt{\epsilon_j} \boldsymbol{v}^*_j\right)\right]^T, \quad (1)\end{aligned}$$

where $\odot$ is the Hadamard product, r and r' are the rank numbers for the modulator and the gradient $\nabla\mathcal{L}$, respectively. From Eq. (1) it is clear that expression $\lambda_i \boldsymbol{u}_i \boldsymbol{v}_i^T$ can select useful signal in $\beta_j \boldsymbol{u}'_j \boldsymbol{v}'^T_j$ while separating the noise in $\epsilon_j \boldsymbol{u}^*_j \boldsymbol{v}^{*T}_j$ as long as their principal eigenvectors do not overlap fully. This idea is consistent with the PCA in which the largest and smallest eigenvalues of the scatter matrix are assumed to correspond to the signal and noise, respectively.

Complexity of Modulating Patterns. To further illustrate the modulation property of our approach, assume $\boldsymbol{M} \in \{0,1\}^{n_1 \times n_2}$ rather than just $\boldsymbol{M} \geq 0$. Furthermore, let $\boldsymbol{u}_i \in \{0,1\}^{n_1}$ and $\boldsymbol{v}_i \in \{0,1\}^{n_2}$. Then, expression $\bigvee_{i=1}^{r} \boldsymbol{u}_i \boldsymbol{v}_i^T$ becomes a binary modulator (switch or selector) for which r controls the complexity of selection pattern and $\bigvee$ performs the element-wise logical 'or' operation on matrices. For such a constrained problem, we have:

$$1+(2^{n_1}-1)(2^{n_2}-1) = \text{patt}(\boldsymbol{u}_1 \boldsymbol{v}_1^T) \leq \text{patt}(\textstyle\bigvee_{i=1}^{r} \boldsymbol{u}_i \boldsymbol{v}_i^T) \leq \text{patt}(\boldsymbol{M}) = 2^{n_1 n_2}, \quad (2)$$

where $\text{patt}(\cdot)$ is the total number of unique patterns that can be generated for a given input matrix. For instance, if $n_1 = n_2 = 4$, we can generate 226, 8776, 49432, 65536 unique patterns for $r = 1, \cdots, 4$. Therefore, r controls the pattern complexity of our modulator. While for $\boldsymbol{M} \geq 0$ and $\boldsymbol{u}_i \in \mathbb{R}^{n_1}$ and $\boldsymbol{v}_i \in \mathbb{R}^{n_2}$ the expression for the pattern complexity may be more elaborate, the general principle of controlling the complexity of the modulating pattern via r holds.

Low-pass Filtering Property. The rank and singular values of the Hadamard product of two matrices are known to be upper-bounded as follows:

$$\begin{aligned} &\text{rank}(\boldsymbol{M} \odot \nabla\mathcal{L}) \leq \text{rank}(\boldsymbol{M}) \cdot \text{rank}(\nabla\mathcal{L}), \\ &\lambda_{max}(\boldsymbol{M} \odot \nabla\mathcal{L}) \leq \lambda_{max}(\boldsymbol{M}) \cdot \lambda_{max}(\nabla\mathcal{L}). \end{aligned} \quad (3)$$

These popular bounds do only tell that $\text{rank}(\boldsymbol{M} \odot \nabla\mathcal{L})$ could be lower than $\text{rank}(\nabla\mathcal{L})$ (the same holds for the largest singular values). However, by just simply looking at the rank-1 factorisation $\boldsymbol{M} = \boldsymbol{u}_1 \boldsymbol{v}_1^T$, we obtain:

$$(\boldsymbol{u}_1 \boldsymbol{v}_1^T) \odot \nabla\mathcal{L} = \text{diag}(\boldsymbol{u}_1) \cdot \nabla\mathcal{L} \cdot \text{diag}(\boldsymbol{v}_1), \quad (4)$$

where $\text{diag}(\cdot)$ is a diagonal matrix created from a vector. Thus, coefficients of $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$ scale rows and columns of $\nabla\mathcal{L}$, and they can nullify entire rows and/or columns thus effectively decrease the rank so that $\text{rank}(\boldsymbol{M} \odot \nabla\mathcal{L}) < \text{rank}(\boldsymbol{M}) \cdot \text{rank}(\nabla\mathcal{L})$. Specifically, let $\nabla\mathcal{L}$ have r' singular values $\gamma_1 \leq \gamma_2 \leq \cdots \leq \gamma_{r'}$ and $\text{diag}(\boldsymbol{u}_1) \cdot \nabla\mathcal{L} \cdot \text{diag}(\boldsymbol{v}_1)$ have $k = r' - 1$ singular values $\gamma^*_1 \leq \gamma^*_1 \leq \cdots \leq \gamma^*_k$ due to the row and/or column deletion. Then, by the Cauchy's interlacing theorem, we have:

$$0 = \gamma^*_0 \leq \gamma_1 \leq \gamma^*_1 \leq \gamma_2 \leq \gamma^*_2 \leq \cdots \leq \gamma^*_k \leq \gamma_{r'}. \quad (5)$$

Now, $p_A = [\gamma_1, \cdots, \gamma_{r'}] / \sum_i \gamma_i$ and $p_B = [\gamma^*_1, \cdots, \gamma^*_k] / \sum_i \gamma^*_i$ can be thought of as two probability mass functions. As the support of p_B equals the support of p_A minus one, that is $\sup(p_B) = \sup(p_A) - 1$, and $\gamma^*_{i-1} \leq \gamma_i$ for $i = 1, \cdots, r'$, we have the following relation between the means and variances of both PMFs: (i) $\mu(p_B) < \mu(p_A)$ and (ii) $\sigma^2(p_B) < \sigma^2(p_A)$.

Due to points (i) and (ii), the Hadamard-based modulator $\boldsymbol{M} \odot \nabla\mathcal{L}$ can act as a low-pass filter.

Moreover, as our generic modulator goes over $i=1,\cdots,r$, we have:

$$\boldsymbol{M} \odot \nabla\mathcal{L} = \sum_{i=0}^{r} \text{diag}(\boldsymbol{u}_i) \cdot \nabla\mathcal{L} \cdot \text{diag}(\boldsymbol{v}_i). \tag{6}$$

The higher the rank r of $\boldsymbol{M}$, the higher the rank of the entire modulated gradient can be. Specifically, we have:

$$\text{rank}(\boldsymbol{M}(\Psi,1) \odot \nabla\mathcal{L}) \leq \text{rank}(\boldsymbol{M}(\Psi,2) \odot \nabla\mathcal{L}) \leq \cdots \leq \text{rank}(\boldsymbol{M}(\Psi,r) \odot \nabla\mathcal{L}), \tag{7}$$

because when we iterate over i in Eq. (6) and aggregate, the consecutive contributions $\text{diag}(\boldsymbol{u}_i) \cdot \nabla\mathcal{L} \cdot \text{diag}(\boldsymbol{v}_i)$ can either fully/partially/not-at-all lie in the span of $\sum_{i'=1}^{i-1} \text{diag}(\boldsymbol{u}_{i'}) \cdot \mathcal{L} \cdot \text{diag}(\boldsymbol{v}_{i'})$.

Consider $\text{diag}(\boldsymbol{u}_i) \cdot \nabla\mathcal{L} \cdot \text{diag}(\boldsymbol{v}_i)$ and their corresponding PMFs p_{B_i} for $i=1,\cdots,r$. Assume that each $\boldsymbol{u}_i\boldsymbol{v}_i^T$ is arranged as previously to have the ability to remove some row and/or column of $\nabla\mathcal{L}$. Based on Eq. (7) and Eq. (5), observe that combining random variables B_i by aggregation $\sum_{i=1}^{r} B_i$ (as Eq. (6) dictates) yields the following inequality which holds with a high probability if $\boldsymbol{u}_i \perp \boldsymbol{u}_j$ and $\boldsymbol{v}_i \perp \boldsymbol{v}_j$ for $i \neq j$:

$$\mu(p_{B_1}) \leq \mu(p_{B_1+B_2}) \leq \cdots \leq \mu(p_{B_1+B_2+\cdots+B_r}) \tag{8}$$

and

$$\sigma^2(p_{B_1}) \leq \sigma^2(p_{B_1+B_2}) \leq \cdots \leq \sigma^2(p_{B_1+B_2+\cdots+B_r}). \tag{9}$$

Synthetic experiment. Figure A2 presents a synthetic experiment with the following setting. We generate $\nabla\mathcal{L} \in \mathbb{R}^{10\times10}$ from the uniform distribution, that is $\nabla\mathcal{L}_{mn} \sim \mathcal{U}(0,1)$. Moreover, we generate the modulation matrix $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{V}^T$ with $\boldsymbol{U} \in \mathbb{R}^{10\times r}$ and $\boldsymbol{V} \in \mathbb{R}^{10\times r}$ being drawn from the normal distribution *e.g.*, $U_{mn} \sim \mathcal{N}(0,1)$ and $V_{mn} \sim \mathcal{N}(0,1)$. Importantly, we note that the choice of the uniform and normal distributions can be swapped in an arbitrary way but it does not affect the conclusions of the following experiment.

Figure A2a shows that the lower the rank r is, the quicker the probability mass function (the trace normalized spectrum) decays. This illustrates that $\boldsymbol{M}(\cdot,r) \odot \nabla\mathcal{L}$ acts in fact as a low-pass filter on the gradient matrix $\nabla\mathcal{L}$. As the smallest singular values corresponding to the largest i represent the noise (by analogy to PCA), low-pass filtering helps us filter out the gradient noise.

Figures A2b and A2c show the corresponding mean and variance w.r.t. the rank r. Our ModGrad acts as a low-pass filter on the spectrum because the mean of the probability mass function for rank $r \leq 3$ is below the mean of the original PMF of $\nabla\mathcal{L}$. Similarly, the variance of the ModGrad-modulated spectrum is below the original variance for $\nabla\mathcal{L}$. These two figures validate the theoretical analysis/assertions of Eq. (8) and (9).

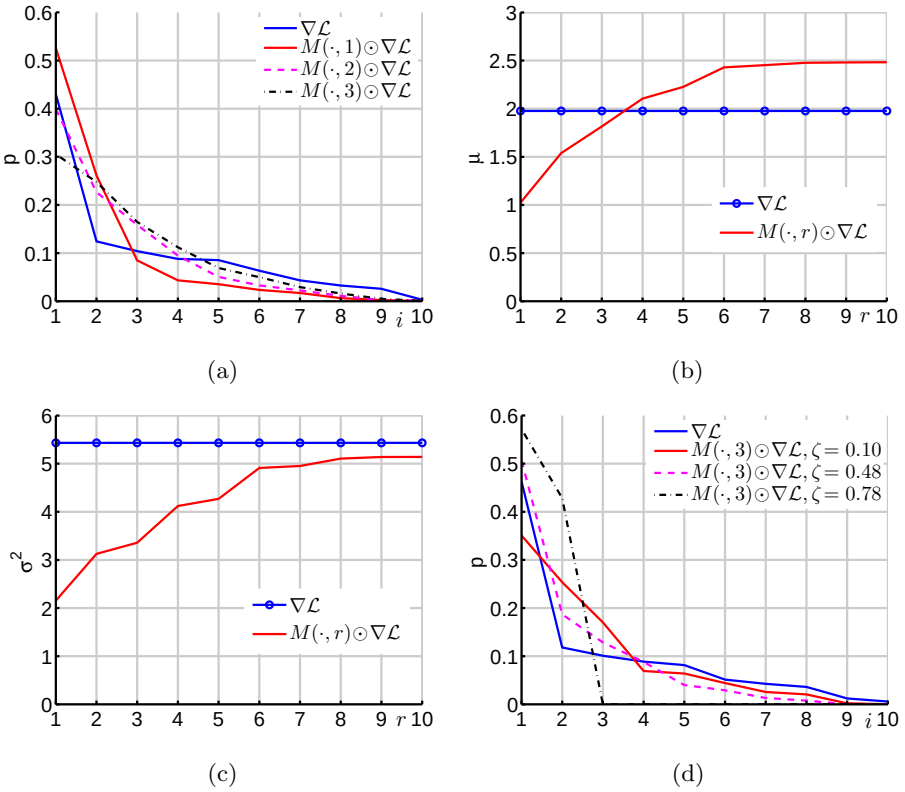
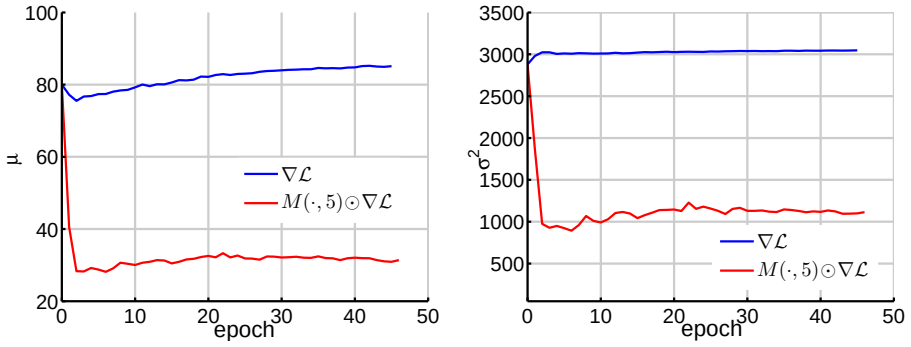


Fig. A2: Synthetic experiments on the Hadamard-based modulator (see Section 3 for details). Figure A2a shows the probability mass function (the SVD singular values γ_i normalized by the trace and connected by line segments) for $\nabla\mathcal{L}$ and ModGrad of rank $r = 1, 2, 3$, respectively. Figures A2b and Figure A2c show the corresponding mean and variance w.r.t. the rank r. Figure A2d shows the probability mass function for ModGrad of rank $r=3$ w.r.t. the sparsity level ζ.

Figure A2d analyses the impact of sparsity of $\boldsymbol{U}$ and $\boldsymbol{V}$ on low-pass filtering. Specifically, we nullify entries U_{mn} and V_{mn} with the values below threshold τ^* which yields sparsity ζ (the number of nullified entries divided by the total number of coefficients of $\boldsymbol{U}$ and $\boldsymbol{V}$). Figure A2d shows that for $r=3$ the sparsity has a strong impact of lowering the rank of ModGrad. This finding is again consistent with the analysis in Eq. (5) regarding the ability of row and/or column scaling to lower the rank of $\boldsymbol{M}\odot\nabla\mathcal{L}$.

Real-data experiment. Figure A3 presents the results with and w/o ModGrad on *mini*-ImageNet. We reshape the gradient into matrices and we average them over entire epoch. Similarly, we average the ModGrad-modulated matrices. We compute the SVD for each variant to obtain the singular values and we trace-normalize them to obtain the probability mass functions.



(a) (b)

Fig. A3: Experiments on the Hadamard-based modulator (see Section 3 for details) given the *mini*-ImageNet dataset. We analyze the probability mass function (the SVD singular values normalized by the trace and connected by line segments) for $\nabla\mathcal{L}$ and ModGrad of rank $r=5$. Figures A3a and A3b show the mean and variance w.r.t. the epoch number.

Figure A3a analyses the mean for the basic $\nabla\mathcal{L}$ *vs*. $\boldsymbol{M}\odot\nabla\mathcal{L}$. It is clear from the plot that our approach immediately reduces the mean of PMF and it keeps adjusting it over epochs for the best performance (see Table 5 in the main submission for the accuracy *vs*. the rank number). In contrast, the mean of PMF without ModGrad remains high throughout epochs.

Figure A3b analyses the variance for the basic $\nabla\mathcal{L}$ *vs*. $\boldsymbol{M}\odot\nabla\mathcal{L}$. Again, our GradMod immediately reduces the variance compared to the variance of PMF without ModGrad which remains high.

We note that while we analyse theoretically the ability of ModGrad to act as a low-pass spectral filter based on the Hadamard product, our design also benefits from the generator which produces matrix $\boldsymbol{M}$. The generator is exposed implicitly to gradients during the minimization of the loss function, therefore it is able to implicitly observe the directions directions helpful in minimizing the objective. The generative approach paired together with the adaptable low-pass filtering appear a robust combination to deal with noisy gradients.

Spectrum of row-wise scaled matrix $\boldsymbol{X}$. Analyzing the spectrum of row- and/or column-wise scaling is a somewhat complex matter but it can shed some light on the scaling properties of (the Hadamard product. Below we consider the toy case performing the row-wise scaling $\mathrm{diag}(\boldsymbol{u})\cdot\boldsymbol{X}$ where $\boldsymbol{u}\in\mathbb{R}^2_+$, $\mathrm{diag}(\boldsymbol{u})\in\mathcal{S}^2_+$ and $\boldsymbol{X}\in\mathcal{S}^2_{++}$. In what follows, we assume that eigenvalues we deal with are simple, that is $\lambda_1\neq\lambda_2$. Then, from the following system of equations

$$\begin{cases}\prod_{i=1}^2\lambda_i=\det(\mathrm{diag}(\boldsymbol{u})\cdot\boldsymbol{X})=\det(\boldsymbol{X})\prod_{i=1}^2 u_i,\\ \sum_{i=1}^2\lambda_i=\mathrm{tr}(\mathrm{diag}(\boldsymbol{u})\cdot\boldsymbol{X})=\sum_{i=1}^2 u_i\cdot X_{ii}\end{cases}\tag{10}$$

we obtain readily the following quadratic equation:

$$\lambda^2-\mathrm{tr}(\mathrm{diag}(\boldsymbol{u})\cdot\boldsymbol{X})\cdot\lambda+\det(\boldsymbol{X})\prod_{i=1}^2 u_i=0.\tag{11}$$

Solving Eq. (11) readily yields the following two eigenvalues for the row-scaled matrix $\boldsymbol{X}$:

$$\begin{cases} \lambda_1^u = \frac{1}{2}\text{tr}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X}) + \frac{1}{2}\sqrt{\text{tr}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X})^2 - 4\cdot\det(\boldsymbol{X})\prod_{i=1}^{2} u_i}, \\ \lambda_2^u = \frac{1}{2}\text{tr}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X}) - \frac{1}{2}\sqrt{\text{tr}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X})^2 - 4\cdot\det(\boldsymbol{X})\prod_{i=1}^{2} u_i}. \end{cases} \tag{12}$$

From Eq. (12), we readily note that (i) $\lambda_2^u = 0$ if $u_i = 0$ for any $i \in \{1, 2\}$, and (ii) as u_i is being gradually increased (step increases of u_i from 0 to ∞ for the chosen $i \in \{1, 2\}$) and $\boldsymbol{u} > 0$, λ_1^u in Eq. (12) grows faster than λ_2^u for the majority of these step increases as the second part of both expressions (containing the square root) in Eq. (12) is added to/subtracted from $\frac{1}{2}\text{tr}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X}$ for λ_1^u and λ_2^u, respectively. This can be clearly seen as we compute:

$$\lim_{u_i \to \infty} \frac{\lambda_1^u}{\lambda_2^u} = \infty \cdot \text{sign}(u_j \cdot \det(\boldsymbol{X})), \; i \neq j, \tag{13}$$

where $\text{sign}(\cdot)$ returns the sign of the input expression.

> As the ratio $\lambda_1^u/\lambda_2^u \to \infty$ in Eq. (13) for $u_i \to \infty$ if $u_j > 0$ and $\det(\boldsymbol{X}) > 0$, the mean and variance of the underlying PMF decrease down to 1 and 0, respectively. Thus, Eq. (13) shows that setting a sufficiently large row-wise multiplication coefficient u_i, $i \in \{1, 2\}$ for $\boldsymbol{X}$ makes the Hadamard product act as a low-pass filter.

The same observations hold for the singular values of rectangular matrix $\boldsymbol{X} \in \mathbb{R}^{2 \times n_2}$, $n_2 \geq 2$ as Eq. (12) (*ie.*, the bottom equation) can be written as:

$$\lambda_2^u = \frac{1}{2}\text{tr}(\boldsymbol{\lambda}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X})) - \frac{1}{2}\sqrt{\text{tr}(\boldsymbol{\lambda}(\text{diag}(\boldsymbol{u})\cdot\boldsymbol{X}))^2 - 4\cdot\det(\boldsymbol{X}\boldsymbol{X}^T)^{\frac{1}{2}}\prod_{i=1}^{2} u_i}, \tag{14}$$

where $\boldsymbol{\lambda}(\cdot)$ is a matrix of singular values.

4 Reinforcement Learning

Below, we explain how to train ModGrad for reinforcement learning. In the K-shot case, the network parameters (θ) are adapted based on K rollouts and their rewards. The gradients for both inner- and outer-loop (meta-optimization) are estimated using REINFORCE [7] and the trust-region policy optimization [8], respectively. The step-by-step ModGrad training procedure for reinforcement learning is provided in Algorithm A1.

References

1. J. Snell, K. Swersky, and Z. Richard, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017. 1

Algorithm A1 Train ModGrad for Reinforcement Learning

Require: Policy $\pi_{\boldsymbol{\theta},\boldsymbol{\Psi}}$, task distribution $p(\mathcal{T})$
$\boldsymbol{\theta}$, $\boldsymbol{\Psi} \leftarrow$ Random initialization
while not done **do**
  Sample $\mathcal{T}_i \dots \mathcal{T}_B$ from $p(\mathcal{T})$
  for b in $\{1, ..., B\}$ **do**
    Sample $\mathcal{D}^{trn}, \mathcal{D}^{tst}$ from $\mathcal{T}_b$
    for i in $\{0, ..., K-1\}$ **do**
      Collect rollout using $\pi_{\boldsymbol{\theta}^i,\boldsymbol{\Psi},\boldsymbol{\nu}}$
      Calculate $\boldsymbol{\nu}$
      Collect rollout using $\pi_{\boldsymbol{\theta}^i,\boldsymbol{\Psi},,\boldsymbol{\nu}'}$
      Calculate parameters update $\boldsymbol{\theta}^{i+1}$
    end for
  end for
  $\boldsymbol{\theta} \leftarrow$ OptimizerStep($\boldsymbol{\theta}^K, \mathcal{D}^{tst}$)
  $\boldsymbol{\Psi} \leftarrow$ OptimizerStep($\boldsymbol{\Psi}, \mathcal{D}^{tst}$)
end while

2. S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proceedings of the British Machine Vision Conference*, 2016, pp. 87.1–87.12. 1
3. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1
4. A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *International Conference on Learning Representations*, 2019. 1
5. W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," in *International Conference on Learning Representations*, 2019. 1
6. D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015. 1
7. R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992. 7
8. J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897. 7