A PROBLEM FORMULATION

Remark. In the main text we have compared the cross-task loss 1 and the meta-learning loss 2. For the cross-task loss, the data is presented as samples from the joint distribution $(\mathcal{T}, \mathcal{D}^\tau)$, and standard results from computational learning theory apply. For the meta-learning loss, we are given m tasks presented through $(n_i)_{i=1}^m$ datapoints for each task, whereas the true task parameter is hidden. When both m and n_i tend to infinity uniformly (i.e. there is some $C > 0$ such that $m > C$, and $n_i > C$ and $C \to \infty$), one may recover the convergence behaviour to the optimal parameter of the empirical risk. However, this cannot be claimed if this convergence is non-uniform. To see this, if $n_i = 1$ for all i, and $m \to \infty$, as the error in estimating each $\mathcal{L}^i$ may prevent convergence. Intuitively, if the growth rate of the n_i's relative to the growth rate of m controls the asymptotics of the model performance. In this paper, we formulate this question in the easiest scenario where all n_i are equal, and we ask what is the optimal growth rate of n and m with respect to the budget constraint (see §3.3).

B DETAILS OF METHODOLOGY

We develop an experimental framework to recover the optimal data allocation $(m^*(b), n^*(b))$ for a variety of different meta-learning problems. Our results test the following intuitive properties of the meta-loss landscape across different allocations, which hold for a fixed meta-learning problem:

- $\mathcal{L}^{meta}(m, n; \mathcal{T}, \mathcal{D}^\tau) \leq \mathcal{L}^{meta}(m', n'; \mathcal{T}, \mathcal{D}^\tau)$ if $m \geq m'$ and $n \geq n'$
- Let $m_1, ...m_b$ be the ordered divisors of b. Then the piecewise linear curve with segments at $(m_i, \mathcal{L}^{meta}(m_i, b/m_i; \mathcal{T}, \mathcal{D}^\tau))$ and $(m_{i+1}, \mathcal{L}^{meta}(m_{i+1}, b/m_{i+1}; \mathcal{T}, \mathcal{D}^\tau))$ is convex.
- $m^*(b) \geq m^*(b')$ if $b \geq b'$

These properties can be summarized as: the meta-loss decreases with increasing the number of tasks or the data per task, and for each budget the loss values sit on a convex curve with a unique minimum. Furthermore, as the budget increases, the value of this minimum decreases and the number allocation corresponding to this minimum increases in both task and data per task.

B.1 MAML

Throughout all experiments in this text, the MAML algorithm assumes *a single* step of gradient descent in the adaptation procedure. This is a constraint on the adaptation step required to evaluate the quality of meta-training, which would not be possible if the number of gradient steps in the adaptation procedure is different at training and test times, or if this number is large enough for no meta-learning to be needed, i.e. at test time, the model fully fits the test dataset through adaptation, not relying at all on the meta-learned meta-parameter.

B.2 META-TESTING

At different allocations, the models expect a different number of points for adaptation during training. Because we are estimating the impact of the allocation of the quality of training, it is nevertheless important to evaluate all models fairly across different allocations. This is why, when meta-testing, we always construct meta-datasets $\mathcal{M}^{test}(\mathcal{T}, \mathcal{D}^\tau; m^{test}, n^{test})$, with m^{test} and n^{test} as large as possible, in order to minimize the variance of Monte Carlo estimating the model's performance.

B.3 THE LOSS-ALLOCATION LANDSCAPE THROUGH GRID SEARCH

Our first method for exploring the optimal data allocation problem is to compute $\mathcal{L}^{meta}(m, n; \mathcal{T}, \mathcal{D}^\tau)$ as a function of m for various fixed budgets b, effectively exhausting a particular budget interval. We refer to the dependence of $\mathcal{L}^{meta}(m, n; \mathcal{T}, \mathcal{D}^\tau)$ on (m, n) or (m, b) as the *loss-allocation landscape*. Usually, in our experiments, we choose a spread of budgets that covers one or more orders of magnitude in order to track changes in $(m^*(b), n^*(b))$.

We choose values of $m(b)$ between 1 and b itself, however, we do not, in most cases check that $m(b)|b$ instead opting for adjusting some tasks randomly to fit an almost uniform allocation, or adjusting the budget itself. For each point estimate of $\mathcal{L}^{meta}(m, n; \mathcal{T}, \mathcal{D}^{\tau})$ we perform several runs on different meta-datasets $\mathcal{M}(\mathcal{T}, \mathcal{D}^{\tau}; m, n)$, and average the values of $\mathcal{L}^{meta}(\boldsymbol{\omega}^{*}(\mathcal{M}(\mathcal{T}, \mathcal{D}^{\tau}; m, n)); \mathcal{M}^{test})$. The individual number of point estimates for each budget used vary according to our computational constraints.

For high budgets, especially, it has been prohibitive to run a large number of such estimates, and hence the variance typically exhibited by these experimental results is higher.

B.4 Sequential decision making allocation procedure

Increasing the values of K_0 and K_1 gives rise to prohibitively large runtimes, however, in practice we choose small values for these integers. There is a tradeoff in the number of erroneous decisions made due to the increased noise when evaluating the payoffs with runtime which we do not explore in this paper.

B.5 Full-batch vs mini-batch gradient descent

In our experiments, wherever possible, we employ full-batch gradient descent, both in the adaptation step of MAML as well as in the gradient step on the meta-parameter $\boldsymbol{\omega}$, in order to control the exact amount of model update steps per datapoint. However, for natural datasets such as CIFAR100, this is infeasible for large budgets, either imposing memory constraints, or introducing unreasonably long runtimes, and in such cases we appeal to standard SGD. When a dataset $\mathcal{M}$ is augmented to $\mathcal{M}'$, there is an expected number of gradient descent steps before new data is sampled sufficiently many times to have a noticeable difference on the gradient estimates. This is a well known problem in the context of active learning.

When performing grid search all models are retrained from scratch, and this does not present a problem. However, for the SDM process, as the budget increases and learning rates decrease through training, it takes more and more time for new data points to *burn in*. In our experiments we stop the allocation process when we hit a predefined budget, regardless of the relative increases in performance at each decision, and keep a fixed number of update steps between any two allocations. A more sophisticated training regime which addresses this imbalance is the subject of more work.

C Details of experiments

C.1 Sinusoid regression experiments

The task parameters in equation 10 are sampled as follows: the phase a is sampled uniformly in the interval $(0.1, 5)$ while ϕ is sampled uniformly from $(0, \pi)$. The independent variable in each task is sampled uniformly from $(-5, 5)$ and labelled according to equation 10.

The neural network used in all sinusoid experiments is a MLP with 2 layers of 40 nodes each. During training we use full batch gradient descent. The initial learning rate for performing gradient updates in the outer loop of the MAML algorithm is 0.001, while the learning rate for the adaptation procedure (inner loop of MAML) is 0.01, except for experiments whose results are depicted in Figure 3 (where the adaptation learning rate varies). The training is done using full batches of data and the Adam optimizer (Kingma & Ba (2014)).

In all experiments we use a learning rate annealing schedule upon reaching a plateau in the training loss. This is done three times with the fourth incurring termination of training. Additionally, we only start the learning rate scheduler after a predetermined number of meta-iterations which depends on the budget.

Testing is performed on 500 adaptation data points.

To train the data allocation algorithm, we first sample a dataset of 5 tasks from the task distribution and 5 datapoints per task. After each subsequent decision to add more tasks or more data points per task, we augment the dataset by sampling additional data corresponding to the decision. We

then train on the augmented dataset for 100 epochs at which point the algorithm makes a subsequent decision. When the budget is depleted, we train the model to convergence using the learning rate annealing schedule. The number of epochs used in training while performing the grid search experiments roughly corresponds to the number of epochs used in training the corresponding data allocation algorithm. This is done to match the performance of the grid search to the performance of the policy. We found that performing shorter training for grid search experiments, worsens the performance by roughly a constant and in particular does not affect the shape of the curves in Figure 2, or the location of their minimum.

C.2 CIFAR-FS EXPERIMENTS

For the CIFAR-FS grid search and SDM allocations we used a MAML with a base learner given by a convolutional neural network (CNN) with architecture as described in Finn et al. (2017), with small modifications to run on the CIFAR data. We use a network with 4 convolutional blocks. Each block consists of a sequence of 2D convolution with kernel size 3, stride 1, same padding, batch normalization, a ReLU nonlinearity and MaxPooling to half size, stride 1. The predictor head is a softmax linear layer applied to the flattened resulting features.

For each budget limited run of MAML, data was presampled and arranged into the required number of tasks and data points per task. Whether from the training, testing or validation meta-datasets, a task has been sampled by selecting 5 random classes with replacement from the available pool. Thus independent samples of tasks may occasionally contain the same class of images.

For the grid search algorithm, independent runs have been performed by independently sampling the initial conditions of the algorithm and independently sampling the meta-training and meta-validation and meta-test datasets. We perform between 3 and 10 independent runs this way for each point on the grid. The meta-training dataset consists of a number of tasks and data per task corresponding to the current allocation for which estimates of $\mathcal{L}^{meta}(m, n; \mathcal{T}, \mathcal{D}^{\tau})$ are required. For the meta-test datasets, a large number of of tasks and datapoints per task is sampled to minimize variance in these estimates (1000 tasks, 500 datapoints per task).

During training, the test-train split for each meta-training task dataset is 0.5. We run the Adam Algorithm in the meta-update of the MAML parameter $\boldsymbol{\omega}$ with batch size between 5 and 25 and initial learning rate 0.001. We anneal this learning rate on a plateau of the training loss for at least 250 meta-updates. We perform 3 annealing steps and instead of the fourth we stop training. For the adaptation step we run a single step of gradient descent with learning rate 0.01. The batch size varies between experiments between 10 and 50.

For the SDM algorithm we set $K_0 = 1$, $K_1 = 200$ and $\Delta t = 250$. We also performed the experiment with $(\Delta m, \Delta n) \in \{(0, 10), (10, 0)\}$ to speed-up runtime and to avoid the noise in the minibatch estimates drowning out the influence of the new quanta of data. After the final allocation we allow the algorithm to train to convergence and report the final test loss. Because of the algorithm's decisions, given a certain budget constraint, it may not be possible to realize the budget exactly with the actions specified, in which case we stop allocating data just short of overshooting the budget.

Remark. For the CIFAR dataset a more natural budget constraint is undoubtedly the number of labeled images, which has been used as a standard constraint in image classification active learning methods Sener & Savarese (2018). Nevertheless, this constraint can only be applied when different tasks share some portion of the underlying data, which is common in abstract scenarios, but relatively infrequent in production settings. We maintain here the definition of the budget given in §3.3, as it presents more generality and practical relevance.

D THE LOSS FUNCTION FOR MIXED LINEAR REGRESSION

We consider the problem of mixed linear regression $\mathbf{y} = X\mathbf{w} + \mathbf{z}$ with squared loss, where X is a $n \times p$ matrix of input data, each row is one of n data vectors of dimension p, $\mathbf{z}$ is a $n \times 1$ noise vector, $\mathbf{w}$ is a $p \times 1$ vector of ground truth parameters and $\mathbf{y}$ is a $n \times 1$ output vector. Data is collected for m tasks, each with a different value of the parameters $\mathbf{w}$ and a different realization of the input X and noise $\mathbf{z}$. We denote by $\mathbf{w}^{(i)}$ the parameters for task i, for $i = 1, \ldots, m$. For a given task i, we

denote by $X^{t(i)}, X^{v(i)}$ the input data for, respectively, the training and validation sets, by $\mathbf{z}^{t(i)}, \mathbf{z}^{v(i)}$ the corresponding noise vectors and by $\mathbf{y}^{t(i)}, \mathbf{y}^{v(i)}$ the output vectors.

Thus, for a given task i, the training output is equal to

$$\mathbf{y}^{t(i)} = X^{t(i)}\mathbf{w}^{(i)} + \mathbf{z}^{t(i)} \tag{11}$$

Similarly, the validation output is equal to

$$\mathbf{y}^{v(i)} = X^{v(i)}\mathbf{w}^{(i)} + \mathbf{z}^{v(i)}. \tag{12}$$

The meta-training loss is equal to

$$\mathcal{L}^{meta} = \frac{1}{2nm}\sum_{i=1}^{m}\left|\mathbf{y}^{v(i)} - X^{v(i)}\boldsymbol{\theta}^{(i)}(\boldsymbol{\omega})\right|^2 \tag{13}$$

where vertical brackets denote euclidean norm, and the estimated parameters $\boldsymbol{\theta}^{(i)}(\boldsymbol{\omega})$ are equal to the one-step gradient update on the single-task training loss $\mathcal{L}^{(i)} = |\mathbf{y}^{t(i)} - X^{t(i)}\boldsymbol{\theta}^{(i)}|^2/2n$, with initial condition given by the meta-parameter $\boldsymbol{\omega}$. The single gradient update is equal to

$$\boldsymbol{\theta}^{(i)}(\boldsymbol{\omega}) = \left(I_p - \frac{\alpha}{n}{X^{t(i)}}^T X^{t(i)}\right)\boldsymbol{\omega} + \frac{\alpha}{n}{X^{t(i)}}^T\mathbf{y}^{t(i)} \tag{14}$$

where I_p is the $p\times p$ identity matrix and α is the learning rate. We seek to minimize the meta-training loss with respect to the meta-parameter $\boldsymbol{\omega}$, namely

$$\boldsymbol{\omega}^\star = \arg\min_{\boldsymbol{\omega}} \mathcal{L}^{meta} \tag{15}$$

We evaluate the solution $\boldsymbol{\omega}^\star$ by calculating the meta-test loss

$$\mathcal{L}^{test} = \mathop{\mathbb{E}}_{\mathbf{w}'}\mathop{\mathbb{E}}_{\mathbf{z}^s}\mathop{\mathbb{E}}_{X^s}\mathop{\mathbb{E}}_{\mathbf{z}^r}\mathop{\mathbb{E}}_{X^r}\frac{1}{2n}\left|\mathbf{y}^s - X^s\boldsymbol{\theta}^\star\right|^2 \tag{16}$$

Note that the test loss is calculated over test data $X^s, \mathbf{z}^s$, and test parameters $\mathbf{w}'$, namely

$$\mathbf{y}^s = X^s\mathbf{w}' + \mathbf{z}^s \tag{17}$$

Furthermore, the estimated parameters $\boldsymbol{\theta}^\star$ are calculated on a separate set of target data $X^r, \mathbf{z}^r$, namely

$$\boldsymbol{\theta}^\star = \left(I_p - \frac{\alpha}{n}X^{rT}X^r\right)\boldsymbol{\omega}^\star + \frac{\alpha}{n}X^{rT}\mathbf{y}^r \tag{18}$$

$$\mathbf{y}^r = X^r\mathbf{w}' + \mathbf{z}^r \tag{19}$$

We are interested in calculating the average test loss, that is averaged over all possible realizations of meta-training data, equal to

$$\overline{\mathcal{L}}^{test} = \mathop{\mathbb{E}}_{\mathbf{w}}\mathop{\mathbb{E}}_{\mathbf{z}^t}\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{\mathbf{z}^v}\mathop{\mathbb{E}}_{X^v}\mathcal{L}^{test} = \mathop{\mathbb{E}}_{\mathbf{w}}\mathop{\mathbb{E}}_{\mathbf{z}^t}\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{\mathbf{z}^v}\mathop{\mathbb{E}}_{X^v}\mathop{\mathbb{E}}_{\mathbf{w}'}\mathop{\mathbb{E}}_{\mathbf{z}^s}\mathop{\mathbb{E}}_{X^s}\mathop{\mathbb{E}}_{\mathbf{z}^r}\mathop{\mathbb{E}}_{X^r}\frac{1}{2n}\left|\mathbf{y}^s - X^s\boldsymbol{\theta}^\star\right|^2 \tag{20}$$

E DEFINITION OF PROBABILITY DISTRIBUTIONS

We assume that all random variables are Gaussian. In particular, we assume that the rows of the matrix X are independent, and each row, denoted by $\mathbf{x}$, is distributed according to a multivariate Gaussian with zero mean and unitary covariance

$$\mathbf{x} \sim \mathcal{N}(0, I_p) \tag{21}$$

where I_p is the $p \times p$ identity matrix. Similarly, the noise is distibuted following a multivariate Gaussian with zero mean and variance equal to σ^2, namely

$$\mathbf{z} \sim \mathcal{N}\left(0, \sigma^2 I_n\right) \tag{22}$$

Finally, the ground truth parameters are also distributed according to a multivariate Gaussian of variance ν^2, namely

$$\mathbf{w} \sim \mathcal{N}\left(0, \frac{\nu^2}{p}I_p\right) \tag{23}$$

All of the above distributions apply independently to each task and dataset (training, validation, target, test). In order to perform the calculations in the next section, we need the following results for the moments of Wishart and inverse Wishart distributions (see Anderson (1962)):

$$\mathbb{E}\left[X^T X\right] = nI_p \tag{24}$$

$$\mathbb{E}\left[\left(X^T X\right)^2\right] = \left(n^2 + np + n\right) I_p \tag{25}$$

$$\mathbb{E}\left[\left(X^T X\right)^3\right] = \left(n^3 + 3n^2 p + np^2 + 3n^2 + 3np + 4n\right) I_p \tag{26}$$

$$\mathbb{E}\left[X^T X A X^T X\right] = n\left(n+1\right) A + n\mathrm{Tr}\left(A\right) I_p \tag{27}$$

$$\mathbb{E}\left[X^T X\ \mathrm{Tr}\left(X^T X\right)\right] = \left(n^2 p + 2n\right) I_p \tag{28}$$

$$\mathbb{E}\left[\left(X^T X\right)^2 \mathrm{Tr}\left(X^T X\right)\right] = \left(n^3 p + n^2 p^2 + n^2 p + 4n^2 + 4np + 4n\right) I_p \tag{29}$$

F Derivation of formula

We calculate the average test loss as a function of the hyperparameters $n, p, m, \alpha, \sigma, \nu$. We will assume that m is large and $p/(nm)$ is small. Using the expression 17 for the test output, we rewrite the test loss 20 as

$$\overline{\mathcal{L}}^{test} = \mathbb{E}\,\frac{1}{2n}\left|X^s\left(\mathbf{w}' - \boldsymbol{\theta}^\star\right) + \mathbf{z}^s\right|^2 \tag{30}$$

We start by averaging this expression with respect to $X^s, \mathbf{z}^s$, noting that $\boldsymbol{\theta}^\star$ does not depend on test data. We further average with respect to $\mathbf{w}'$, but note that $\boldsymbol{\theta}^\star$ depends on test parameters, so we average only terms that do not depend on $\boldsymbol{\theta}^\star$. Using 24, the result is

$$\overline{\mathcal{L}}^{test} = \frac{\sigma^2}{2} + \frac{\nu^2}{2} + \mathbb{E}\left[\frac{\left|\boldsymbol{\theta}^\star\right|^2}{2} - \mathbf{w}'^T \boldsymbol{\theta}^\star\right] \tag{31}$$

The second term in the expectation is linear in $\boldsymbol{\theta}^\star$ and can be averaged over $X^r, \mathbf{z}^r$, using 18 and noting that $\boldsymbol{\omega}^\star$ does not depend on target data. The result is

$$\mathop{\mathbb{E}}_{X^r}\mathop{\mathbb{E}}_{\mathbf{z}^r}\ \boldsymbol{\theta}^\star = (1-\alpha)\boldsymbol{\omega}^\star + \alpha\mathbf{w}' \tag{32}$$

Furthermore, using 38, we find that the following average holds

$$\mathop{\mathbb{E}}_{\mathbf{w}}\mathop{\mathbb{E}}_{\mathbf{z}^t}\mathop{\mathbb{E}}_{\mathbf{z}^v}\ \boldsymbol{\omega}^\star = 0 \tag{33}$$

Combining 32, 33, we can calculate the second term in the expectation of 31 and find

$$\overline{\mathcal{L}}^{test} = \frac{\sigma^2}{2} + (1-2\alpha)\frac{\nu^2}{2} + \mathbb{E}\frac{\left|\boldsymbol{\theta}^\star\right|^2}{2} \tag{34}$$

We start averaging the third term of this expression over $\mathbf{z}^r, \mathbf{w}'$, using 18 and noting that $\boldsymbol{\omega}^\star$ does not depend on target data and test parameters. The result is

$$\mathop{\mathbb{E}}_{\mathbf{w}'}\mathop{\mathbb{E}}_{\mathbf{z}^r}\left|\boldsymbol{\theta}^\star\right|^2 = \boldsymbol{\omega}^{\star T}\left(I - \frac{\alpha}{n}X^{rT}X^r\right)^2\boldsymbol{\omega}^\star + \frac{\alpha^2\sigma^2}{n^2}\mathrm{Tr}\left[X^r X^{rT}\right] + \frac{\alpha^2\nu^2}{n^2 p}\mathrm{Tr}\left[\left(X^r X^{rT}\right)^2\right] \tag{35}$$

We now average over $\mathbf{X}^r$, again noting that $\boldsymbol{\omega}^\star$ does not depend on target data. Using 24, 25, we find

$$\mathop{\mathbb{E}}_{X^r}\mathop{\mathbb{E}}_{\mathbf{w}'}\mathop{\mathbb{E}}_{\mathbf{z}^r}\left|\boldsymbol{\theta}^\star\right|^2 = \left[1 - 2\alpha + \alpha^2\left(1 + \frac{p+1}{n}\right)\right]\left|\boldsymbol{\omega}^\star\right|^2 + \frac{\alpha^2\sigma^2 p}{n} + \alpha^2\nu^2\left(1 + \frac{p+1}{n}\right) \tag{36}$$

We can now rewrite the average test loss 34 as

$$\overline{\mathcal{L}}^{test} = \frac{\sigma^2}{2}\left(1 + \frac{\alpha^2 p}{n}\right) + \frac{1}{2}\left[(1-\alpha)^2 + \alpha^2\frac{p+1}{n}\right]\left(\nu^2 + \mathbb{E}\left|\boldsymbol{\omega}^\star\right|^2\right) \tag{37}$$

In order to average the last term, we need an expression for $\boldsymbol{\omega}^\star$. We note that the loss 13 is quadratic in $\boldsymbol{\omega}$, therefore the solution 15 can be found using standard linear algebra. Under the assumption that the following matrix inverse exists (typically for $p < mn$), the solution is equal to

$$\boldsymbol{\omega}^\star = \left[\sum_{i=1}^{m}\left(I - \frac{\alpha}{n}X^{t(i)^T}X^{t(i)}\right)^T X^{v(i)^T}X^{v(i)}\left(I - \frac{\alpha}{n}X^{t(i)^T}X^{t(i)}\right)\right]^{-1} \cdot \tag{38}$$

$$\cdot\left\{\sum_{i=1}^{m}\left(I - \frac{\alpha}{n}X^{t(i)^T}X^{t(i)}\right)^T X^{v(i)^T}\left[X^{v(i)}\left(\mathbf{w}^{(i)} - \frac{\alpha}{n}X^{t(i)^T}\left(X^{t(i)}\mathbf{w}^{(i)} + \mathbf{z}^{t(i)}\right)\right) + \mathbf{z}^{v(i)}\right]\right\} \tag{39}$$

Using this expression we can derive 33. This expression can be squared and averaged over $\mathbf{w}^{(i)}$, $\mathbf{z}^{t(i)}$, $\mathbf{z}^{v(i)}$ to calculate the last average in 37. For ease of notation, we define $A^{t(i)} = I - \frac{\alpha}{n}X^{t(i)^T}X^{t(i)}$. The result is

$$\mathop{\mathbb{E}}_{\mathbf{w}}\mathop{\mathbb{E}}_{\mathbf{z}^t}\mathop{\mathbb{E}}_{\mathbf{z}^v}|\boldsymbol{\omega}^\star|^2 = \mathrm{Tr}\left\{\sigma^2\left[\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right]^{-1} + \right. \tag{40}$$

$$+\frac{\nu^2}{p}\left[\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right]^{-2}\sum_{i=1}^{m}\left(A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right)^2 + \tag{41}$$

$$\left.+\frac{\alpha^2\sigma^2}{n^2}\left[\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right]^{-2}\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}X^{t(i)^T}X^{t(i)}X^{v(i)^T}X^{v(i)}A^{t(i)}\right\} \tag{42}$$

The final step is to average this expression with respect to training and validation input data X^t, X^v. Here, we resort to the following approximation: we assume that the number of tasks m is large, and noting that data for different tasks i is independent, we use the law of large numbers to substitute each sum of m independent r.v.'s appearing in the expressions 40, 41 and 42, by its mean. Since 40 is the inverse of a sum over m terms, while 41 and 42 are products of a sum over m and the inverse square of another sum, the above approximation will yield terms of order $1/m$ in all three expressions. Although we do not attempt at a rigorous proof, this type of approximations should yield errors of smaller order than the mean we are approximating. This can be easily verified in particular 1-dimensional cases ($p = 1$) and is the subject of large deviation estimates in the general case.

We calculate the expectation of each term in the sum using 24, 25, finding

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\left[\left(I - \frac{\alpha}{n}X^{t^T}X^t\right)^T X^{v^T}X^v\left(I - \frac{\alpha}{n}X^{t^T}X^t\right)\right] = n\left[(1-\alpha)^2 + \alpha^2\frac{p+1}{n}\right]I_p \tag{43}$$

Substituting the sums in the the inverses appearing in 40, 41, 42 by their mean, we obtain

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\left[\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right]^{-1} \simeq \left[(1-\alpha)^2 + \alpha^2\frac{p+1}{n}\right]^{-1}(nm)^{-1}I_p \tag{44}$$

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\left[\sum_{i=1}^{m}A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right]^{-2} \simeq \left[(1-\alpha)^2 + \alpha^2\frac{p+1}{n}\right]^{-2}(nm)^{-2}I_p \tag{45}$$

Furthermore, we calculate the following expectation in 41

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\sum_{i=1}^{m}\left(A^{t(i)^T}X^{v(i)^T}X^{v(i)}A^{t(i)}\right)^2 \simeq \left[(1-\alpha)^2 + \alpha^2\frac{p+1}{n}\right]^2 m\left(n^2+np+n\right)I_p \tag{46}$$

Finally, we compute the following average, appearing in 42, using 24, 25, 26, 27, 28, 29

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\sum_{i=1}^{m} A^{t(i)^T} X^{v(i)^T} X^{v(i)} X^{t(i)^T} X^{t(i)} X^{v(i)^T} X^{v(i)} A^{t(i)} = \tag{47}$$

$$= \mathop{\mathbb{E}}_{X^t}\sum_{i=1}^{m}\left[n\left(n+1\right)A^{t(i)^T} X^{t(i)^T} X^{t(i)} A^{t(i)} + nA^{t(i)^T} A^{t(i)} \mathrm{Tr}\left(X^{t(i)^T} X^{t(i)}\right)\right] = \tag{48}$$

$$= m\left\{n\left(n+1\right)\left[n - \frac{2\alpha}{n}\left(n^2+np+n\right) + \frac{\alpha^2}{n^2}\left(n^3+3n^2p+np^2+3n^2+3np+4n\right)\right] + \right. \tag{49}$$

$$\left. +n\left[np - \frac{2\alpha}{n}\left(n^2p+2n\right) + \frac{\alpha^2}{n^2}\left(n^3p+n^2p^2+n^2p+4n^2+4np+4n\right)\right]\right\} I_p \tag{50}$$

Note that this expression is multiplied by α^2 in 42. We simplify this expression by neglecting all orders higher than α^2 in 42, meaning that we drop all α in this expression, obtaining

$$\mathop{\mathbb{E}}_{X^t}\mathop{\mathbb{E}}_{X^v}\sum_{i=1}^{m} A^{t(i)^T} X^{v(i)^T} X^{v(i)} X^{t(i)^T} X^{t(i)} X^{v(i)^T} X^{v(i)} A^{t(i)} \simeq mn^2\left(n+p+1\right) I_p \tag{51}$$

For the same reason, we drop all α terms in 45 when using it to compute 42. Putting everything together, 44, 45, 46, 49, 50 and applying the trace operator, we find the following expression for the meta-parameter variance

$$\mathbb{E}\left|\boldsymbol{\omega}^\star\right|^2 \simeq \sigma^2\left[\left(1-\alpha\right)^2 + \alpha^2\frac{p+1}{n}\right]^{-1}\frac{p}{nm} + \frac{\nu^2}{m}\left(1+\frac{p+1}{n}\right) + \frac{\alpha^2\sigma^2 p}{mn^2}\left(n+p+1\right) \tag{52}$$

We substitute this expression back into the average test loss 37, and we drop again all terms of order higher than α^2. The result is

$$\overline{\mathcal{L}}^{test} = \frac{\nu^2}{2}\left[\left(1-\alpha\right)^2 + \alpha^2\frac{p+1}{n}\right]\left[1+\frac{1}{m}\left(1+\frac{p+1}{n}\right)\right] + \tag{53}$$

$$\frac{\sigma^2}{2}\left[1+\frac{p}{nm}+\frac{\alpha^2 p}{n}\left(1+\frac{p+1}{nm}+\frac{1}{m}\right)\right] \tag{54}$$

We denote by $b = nm$ the total budget of data points, and rewrite the test loss as a function of m. To simplify the expression further, we substitute $p+1$ with p as the number if parameters is usually much larger than 1. The result is

$$\overline{\mathcal{L}}^{test} = \frac{\nu^2}{2}\left[\left(1-\alpha\right)^2 + \alpha^2\frac{pm}{b}\right]\left(1+\frac{1}{m}+\frac{p}{b}\right) + \frac{\sigma^2}{2}\left[1+\frac{p}{b}+\frac{\alpha^2 pm}{b}\left(1+\frac{p}{b}+\frac{1}{m}\right)\right] \tag{55}$$