# GENERALISATION GUARANTEES FOR CONTINUAL LEARNING WITH ORTHOGONAL GRADIENT DESCENT

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In Continual Learning settings, deep neural networks are prone to Catastrophic Forgetting. Orthogonal Gradient Descent (Farajtabar et al., 2019) was proposed to tackle the challenge. However, no theoretical guarantees have been proven yet. We present a theoretical framework to study Continual Learning algorithms in the NTK regime. This framework comprises closed form expression of the model through tasks and proxies for transfer learning, generalisation and tasks similarity. In this framework, we prove that OGD is robust to Catastrophic Forgetting then derive the first generalisation bound for SGD and OGD for Continual Learning. Finally, we study the limits of this framework in practice for OGD and highlight the importance of the NTK variation for Continual Learning.

## 1 INTRODUCTION

Continual learning is a setting in which an agent is exposed to multiples tasks sequentially Kirkpatrick et al. (2016). The core challenge lies in the ability of the agent to learn the new tasks while retaining the knowledge acquired from previous tasks. Too much plasticity (Nguyen et al., 2018) will lead to catastrophic forgetting, which means the degradation of the ability of the agent to perform the past tasks (McCloskey & Cohen 1989, Ratcliff 1990, Goodfellow et al. 2014). On the other hand, too much stability will hinder the agent from adapting to new tasks.

While there is a large literature on Continual Learning (Parisi et al., 2019), few works have addressed the problem for a theoretical perspective. The recent findings on the Neural Tangent Kernel Jacot et al. (2018) and on the convergence of Stochastic Gradient Descent for overparameterized neural networks (Arora et al., 2019) have unlocked powerful tools to analyze the training dynamics of over-parameterized neural networks. We leverage these theoretical findings in order to to propose a theoretical framework for Continual Learning in the NTK regime then prove convergence and generalisation properties for the algorithm Orthogonal Gradient Descent for Continual Learning (Farajtabar et al., 2019).

Our contributions are summarized as follows:

1. We present a theoretical framework to study Continual Learning algorithms in the NTK regime. This framework frames Continual Learning as a recursive kernel regression and comprises proxies for transfer learning, generalisation, tasks similarity and curriculum learning. (Thm. 1, Lem. 1 and Thm. 3).
2. In this framework, we prove that OGD is robust to forgetting with respect to an arbitrary number of tasks under an infinite memory (Sec . 5, Thm. 2).
3. We prove the first generalisation bound for Continual Learning with SGD and OGD. We find that generalisation through tasks depends on a task similarity with respect to the NTK. As a side result, we also find that the ordering of tasks impacts generalisation, highlighting the importance of the task Curriculum. (Sec. 5, Theorem 3)
4. We study the limits of this framework in practical settings, through the importance of the NTK variation. We propose and use OGD+ as a tool for this analysis. We find that OGD+ is significantly more robust to Catastrophic Forgetting than OGD. (Sec. 6)

## 2 RELATED WORKS

Continual Learning addresses the Catastrophic Forgetting problem, which refers to the tendency of agents to "forget" the previous tasks they were trained on over the course of training. It's an active area of research, several heuristics were developed in order to characterise it (Ans & Rousset 1997, Ans & Rousset 2000, Goodfellow et al. 2014, French 1999, McCloskey & Cohen 1989, Robins 1995, Nguyen et al. 2019). Approaches to Continual Learning can be categorised into: regularization methods, memory based methods, and dynamic architectural methods. We refer the reader to the survey by Parisi et al. (2019) for an extensive overview on the existing methods. The idea behind memory-based methods is to store data from previous tasks in a buffer of fixed size, which can then be reused during training on the current task (Chaudhry et al. 2019, Van de Ven & Tolias 2018). While dynamic architectural methods rely on growing architectures which keep the past knowledge fixed and store new knowledge in new components, such as new nodes, or layers. (Lee et al. 2018, Schwarz et al. 2018) Finally, regularization methods regularize the objective in order to preserve the knowledge acquired from the previous tasks (Kirkpatrick et al. 2016, Aljundi et al. 2018, Farajtabar et al. 2019, Zenke et al. 2017).

While there is a large literature on the field, there is a limited number of theoretical works on Continual Learning. Alquier et al. (2017) define a compound regret for lifelong learning, as the regret with respect to the oracle who would have known the best common representation for all tasks in advance. Knoblauch et al. (2020) show that optimal CL algorithms generally solve an NP-HARD problem and will require perfect memory not to suffer from catastrophic forgetting. Benzing (2020) presents mathematical and empirical evidence that the two methods – Synaptic Intelligence and Memory Aware Synapses – approximate a rescaled version of the Fisher Information.

Continual Learning is not limited to Catastrophic Forgetting, but also closely related to Transfer Learning. A desirable property of a Continual Learning algorithm is to enable the agent to carry the acquired knowledge through his lifetime, and transfer it to solve new tasks. A new theoretical study of the phenomena was presented by Liu et al. (2019). They prove how the task similarity contributes to generalisation, in a two tasks setting and for over-parameterised 2 layer RELU neural networks.

The recent findings on the Neural Tangent Kernel (Jacot et al., 2018) and on the properties of overparameterized neural networks (Du et al. 2018, Arora et al. 2019) provide powerful tools to analyze their training dynamics. We build up on these advances to construct a theoretical framework for Continual Learning and study the generalisation properties of Orthogonal Gradient Descent.

## 3 PRELIMINARIES

**Notation** We use bold-faced characters for vectors and matrices. We use $\|\cdot\|$ to denote the Euclidian norm of a vector or the spectral norm of a matrix, and $\|\cdot\|_{\mathrm{F}}$ to denote the Frobenius norm of a matrix. We use $\langle\cdot,\cdot\rangle$ for the Euclidian dot product, and $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ the dot product in the Hilbert space $\mathcal{H}$. We index the task ID by $\tau$. The $\leq$ operator if used with matrices, corresponds to the partial ordering over symetric matrices. We denote $\mathbb{N}$ the set of natural numbers, $\mathbb{R}$ the space of real numbers and $\mathbb{N}^\star$ for the set $\mathbb{N} \setminus \{0\}$. We use $\oplus$ to refer to the direct sum over Euclidean spaces.

### 3.1 CONTINUAL LEARNING

Continual learning considers a series of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \ldots\}$, where each task can be viewed as a separate supervised learning problem. Similarly to online learning, data from each task is revealed only once. The goal of continual learning is to model each task accurately with a single model. The challenge is to achieve a good performance on the new tasks, while retaining knowledge from the previous tasks (Nguyen et al., 2018).

We assume the data from each task $\mathcal{T}_\tau$, $\tau \in \mathbb{N}^\star$, is drawn from a distribution $\mathcal{D}_\tau$. Individual samples are denoted $(\boldsymbol{x}_{\tau,i}, y_{\tau,i})$, where $i \in [n_\tau]$. Also, we only consider the binary classification setting for the sake of simplicity: $\boldsymbol{x}_{\tau,i} \in \mathbb{R}^d$ and $y_{\tau,i} \in \{-1, +1\}$. We note that it does not restrict the scope of the analysis, which can be easily extended to multiclass settings.

### 3.2 OGD FOR CONTINUAL LEARNING

Let $\mathcal{T}_T$ the current task, where $T \in \mathbb{N}^\star$. For all $i \in [n_T]$, let $\boldsymbol{v}_{T,i} = \nabla_{\boldsymbol{w}} f^\star_{T-1}(\boldsymbol{x}_{T-1,i})$, which is the Jacobian of task $\mathcal{T}_T$. We define $\mathbb{E}_\tau = \mathrm{vec}(\{\boldsymbol{v}_{\tau,i}, i \in [n_\tau]\})$, which is the subspace induced by the Jacobian. The idea behind OGD (Farajtabar et al., 2019) is to update the weights along the projection of the gradient on the orthogonal space induced by the Jacobians over the previous tasks $\mathbb{E}_1 \oplus \ldots \oplus \mathbb{E}_{\tau-1}$. The update rule for the task $\mathcal{T}_T$ is as follows :

$$\boldsymbol{w}_T(t+1) = \boldsymbol{w}_T(t) - \eta \Pi_{\mathbb{E}^\perp_{T-1}} \nabla_{\boldsymbol{w}} \mathcal{L}^T(\boldsymbol{w}_T(t)).$$

The intuition behind OGD is to "preserve the previously acquired knowledge by maintaining a space consisting of the gradient directions of the neural networks predictions on previous tasks" (Farajtabar et al., 2019). Throughout the paper, we only consider the OGD-GTL variant which stores the gradient with respect to the ground truth logit.

For a task $\mathcal{T}_\tau$, we write the corresponding loss as :

$$\mathcal{L}^\tau(\boldsymbol{w}) = \sum_{i=1}^{n_\tau} (f_\tau(\boldsymbol{x}_{\tau,i}) - y_{\tau,i})^2.$$

### 3.3 NEURAL TANGENT KERNEL

In their seminal paper, Jacot et al. (2018) established the connection between deep networks and kernel methods by introducing the Neural Tangent Kernel (NTK). They showed that at the infinite width limit, the kernel remains constant throughout training. Lee et al. (2019) also showed that a network evolves as a linear model in the infinite width limit when trained on certain losses under gradient descent.

Throughout our analysis, we make the assumption that the neural network is overparameterized, and consider the linear approximation of the neural network around its initialisation:

$$f^{(t)}(\boldsymbol{x}) \approx f^{(0)}(\boldsymbol{x}) + \nabla_{\boldsymbol{w}} f^{(0)}(\boldsymbol{x})^T (\boldsymbol{w}(t) - \boldsymbol{w}(0)).$$

## 4 CONVERGENCE - CONTINUAL LEARNING AS A RECURSIVE KERNEL REGRESSION

In this section, we derive a closed form expression for the Continual Learning models through tasks. We find that Continual Learning models can be expressed with recursive kernel ridge regression across tasks. We also find a that the NTK of OGD is recursive with respect to the projection of its feature map on the tasks' spaces. The result is presented in Theorem 1, a stepping stone towards proving the generalisation bound for OGD in Sec. 5.

### 4.1 CONVERGENCE THEOREM

**Theorem 1 (Continual Learning as a recursive Kernel Regression)**

*Given $\mathcal{T}_1, \ldots, \mathcal{T}_T$ a sequence of tasks. Fix a learning rate sequence $(\eta_\tau)_{\tau \in [T]}$. If, for all $\tau$, the learning rate satisfies $\eta_\tau < \frac{1}{\|\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I}\|}$, then for all $\tau$, $\boldsymbol{w}_\tau(t)$ converges linearly to a limit solution $\boldsymbol{w}^\star_\tau$ such that*

$$f^\star_\tau(\boldsymbol{x}) = f^\star_{\tau-1}(\boldsymbol{x}) + \kappa_\tau(\boldsymbol{x}, \mathbf{X}_\tau)^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau,$$

*where*

$$\begin{aligned}
\kappa_\tau(\boldsymbol{x}, \boldsymbol{x}') &= \widetilde{\phi}_\tau(\boldsymbol{x}) \widetilde{\phi}_\tau(\boldsymbol{x}')^T, \\
\tilde{\mathbf{y}}_\tau &= \boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1 \to \tau}, \\
\boldsymbol{y}_{\tau-1 \to \tau} &= f^\star_{\tau-1}(\mathbf{X}_\tau), \\
\widetilde{\phi}_\tau(\boldsymbol{x}) &= \begin{cases} \nabla_{\boldsymbol{w}} f^\star_0(\boldsymbol{x}) \in \mathbb{R}^d & \text{for SGD}, \\ \mathbf{T}_\tau \nabla_{\boldsymbol{w}} f^\star_0(\boldsymbol{x}) \in \mathbb{R}^{d - M_\tau} & \text{for OGD}. \end{cases}
\end{aligned}$$

*and* $\{\mathbf{T}_\tau \in \mathbb{R}^{d-M_\tau,d}, \tau \in [T]\}$ *are projection matrices from* $\mathbb{R}^d$ *to* $(\bigoplus_{k=1}^{\tau} \mathbb{E}_k)^\perp$ *and* $M_\tau$ *= dim(* $\bigoplus_{k=1}^{\tau} \mathbb{E}_k$*)*
.

The theorem describes how the model $f_\tau^\star$ evolves across tasks. The theorem is recursive because the learning is incremental. For a given task $\mathcal{T}_\tau$, $f_{\tau-1}^\star(\boldsymbol{x})$ is the knowledge acquired by the agent up to the task $\mathcal{T}_{\tau-1}$. At this stage, the model only fits the residual $\tilde{\mathbf{y}}_\tau = \boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau}$, which complements the knowledge acquired through previous tasks. This residual is also a proxy for task similarity. If the tasks are identical, the residual is equal to zero. The knowledge increment is captured by the term: $\kappa_\tau(\boldsymbol{x}, \mathbf{X}_\tau)^T(\kappa_{\tau+1}(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau$. Finally, task similarity is computed with respect to the most recent feature map $\widetilde{\phi}_\tau$, and $k_\tau$ is the NTK with respect to the feature map $\widetilde{\phi}_\tau$.

**Remark 1** *The recursive relation from Theorem 1 can also be written as a linear combination of kernel regressors as follows:*

$$f_\tau^\star(\boldsymbol{x}) = \sum_{k=1}^{\tau} \tilde{f}_k^\star(\boldsymbol{x}),$$

*where*

$$\tilde{f}_k^\star(\boldsymbol{x}) = \kappa_k(\boldsymbol{x}, \mathbf{X}_k)^T(\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_k.$$

### 4.2 Distance from initialisation through tasks

As described in Sec. 4.1, $\tilde{\mathbf{y}}_\tau$ is a residual. It is equal to zero if the model $f_{\tau-1}^\star$ makes perfect predictions on the next task $\mathcal{T}_\tau$. The more the next task $\mathcal{T}_\tau$ is different, the further the neural network needs to move from its previous state in order to fit it. Corollary 1 tracks the distance from initialisation as a function of task similarity.

**Corollary 1** *For SGD, and for OGD under the additional assumption that* $\{\mathbf{T}_\tau, \tau \in [T]\}$ *are orthonormal,*

$$\left\|\boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star\right\|^2 = \tilde{\mathbf{y}}_{\tau+1}^T(\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1})(\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1},$$

**Remark 2** *Corollary 1 can be applied to get a similar result to Theorem 3 by Liu et al. (2019). In this remark, we consider mostly their notations. Their theorem states that under some conditions, for 2-layer neural networks with a RELU activation function, with probability no less than* $1-\delta$ *over random initialisation,*

$$\|\mathbf{W}(P) - \mathbf{W}(Q)\|_{\mathrm{F}} \leq \sqrt{\tilde{\mathbf{y}}_{P\to Q}^T H_P^{\infty-1}\tilde{\mathbf{y}}_{P\to Q}} + \epsilon,$$

*where, in their work:*

$$\boldsymbol{y}_{P\to Q} = H_{PQ}^{\infty,T} H_P^{\infty-1}\boldsymbol{y}_P,$$
$$\tilde{\mathbf{y}}_{P\to Q} = \boldsymbol{y}_Q - \boldsymbol{y}_{P\to Q}.$$

*Note that* $H_P^\infty$ *is a Gram matrix, which also corresponds to the NTK of the neural network they consider. We see an analogy with our result, where we work directly with the NTK, with no assumptions on the neural network. One important observation is that, to our knowledge, since there are no guarantees for the invertibility of our Gram matrix, we add a ridge regularisation to work with a regularised matrix, which is then invertible. In our setting, by considering* $\lambda \to 0$*, and with the additional assumption of invertibility of* $\mathbf{H}_{\tau,0}$*,which is valid in the two-layer overparametrised RELU neural network considered in the setting of Liu et al. (2019), we can recover a similar approximation.*

## 5 OGD : Learning without forgetting, provably

In this section, we study the generalisation properties of OGD building-up on Thm. 1. First, we prove that OGD is robust to catastrophic forgetting with respect to all previous tasks (Theorem 2). Then, we

present the main generalisation theorem for OGD (Thm. 3). The theorem provides several insights on the relation between task similarity and generalisation. Finally, we present how the Rademacher complexity relates to task similarity across a large number of tasks (Lemma 1). The lemma states that the more dissimilar tasks are, the larger the class of functions explored by the neural network, with high probability. This result highlights the importance of the curriculum for Continual Learning.

### 5.1 Memorisation property of OGD

The key to obtaining tight generalisation bounds for OGD is Theorem 2.

**Theorem 2 (No-forgetting Continual Learning with OGD)** *Given a task $\mathcal{T}_\tau$, for all $\boldsymbol{x}_{k,i} \in D_k$, a sample from the training data of a previous task $\mathcal{T}_k$, given that $\boldsymbol{x}_{k,i}$ is in the memory, it holds that:*

$$f^\star_\tau(\boldsymbol{x}_{k,i}) = f^\star_k(\boldsymbol{x}_{k,i}).$$

As motivated by Farajtabar et al. (2019), the orthogonality of the gradient updates aims to preserve the acquired knowledge, by not altering the weights along relevant dimensions when learning new tasks. Theorem 2 states that, given an infinite memory, the training error on all previous tasks is unchanged, when training with OGD.

### 5.2 Generalisation properties of SGD and OGD

Now, we state the main generalisation theorem for OGD, which provides generalisation bounds on the data from all the tasks, for SGD and OGD.

**Theorem 3 (Generalisation of SGD and OGD for Continual Learning)**

*Let $\{\mathcal{T}_1, \dots \mathcal{T}_T\}$ be a sequence of tasks. Let be $\{\mathcal{D}_1, \dots, \mathcal{D}_T\}$ the respective distributions over $\mathbb{R}^d \times \{-1, 1\}$. Let $\{(\boldsymbol{x}_{\tau,i}, y_{\tau,i}), i \in [n_t], \tau \in [T]\}$ be i.i.d. samples from $\mathcal{D}_\tau, \tau \in [T]$. Denote $\mathbf{X}_\tau = (\boldsymbol{x}_{\tau,1}, \dots, \boldsymbol{x}_{\tau,n_\tau})$, $\boldsymbol{y}_\tau = (y_{\tau,1}, \dots, y_{\tau,n_\tau})$. Consider the kernel ridge regression solution $f^\star_T$, then, for any loss function $\ell : \mathbb{R} \times \mathbb{R} \to [0, c]$ that is c-Lipchitz in the first argument, with probability at least $1 - \delta$,*

$$\mathcal{L}_{D_\tau}(f^\star_T) \leq \begin{cases} \frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}_\tau(k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau + R_T + 3c\sqrt{\frac{\log(2/\delta)}{2n_T}}, & \text{for OGD}, \tau \in [1, T], \\ \frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}_\tau(k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau + R_T + 3c\sqrt{\frac{\log(2/\delta)}{2n_T}}, & \text{for SGD}, \tau = T\ , \\ \frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}^T_\tau(k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau + \frac{1}{n_T}\sum_{k=\tau+1}^{T} H_{k,\tau} + R_T + 3c\sqrt{\frac{\log(2/\delta)}{2n_T}}, & \text{for SGD}, \tau < T. \end{cases}$$

*where*

$$\begin{aligned} \tilde{\mathbf{y}}_\tau &= \boldsymbol{y}_\tau - f^\star_{\tau-1}(\mathbf{X}_\tau), \\ R_T &= \sum_{k=1}^{T}\sqrt{\frac{\mathrm{tr}(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))}{n_\tau^2}\tilde{\mathbf{y}}^T_\tau(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau}, \\ H_{k,\tau} &= \tilde{\mathbf{y}}^T_k(k_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda\mathbf{I})^{-1}k_k(\mathbf{X}_k, \mathbf{X}_\tau)k_k(\mathbf{X}_\tau, \mathbf{X}_k)(k_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_k. \end{aligned}$$

Theorem 3 shows that the generalisation bound for OGD is tighter than SGD. This results from Thm. 2, which states that OGD is robust to Catastrophic Forgetting. The bound is looser for SGD over the past tasks through the forgetting term that appears.

The bound of Theorem 3 comprises the following main terms :

- $\frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}_\tau(k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau$ : this term is due to the regularisation and describes that the empirical loss is not equal to zero due to the regularisation term. This term tends to zero in case there is no regularisation. For interpretation purposes, in App. D.2.7, we show that the empirical loss tends to zero in the no-regularisation case.
- $R_T$ captures the impact of task similarity on generalisation which we discuss in Sec. 5.3.
- $\sum_{k=\tau+1}^{T} H_{k,\tau}$ is a residual term that appears for SGD only. It is due to the catastrophic forgetting that occurs with SGD. It also depends on the tasks similarity.

These bounds share some similarities with the bounds derived by Arora et al. (2019), Liu et al. (2019) and Hu et al. (2019), where in these works, the bounds were derived for supervised learning settings, and in some cases for two-layer RELU neural networks. Similarly, the bounds depend on the Gram matrix of the data, with the feature map corresponding to the NTK.

### 5.3 THE IMPACT OF TASK SIMILARITY ON GENERALISATION

Now, we state Lemma 1, which tracks the Rademacher complexity through tasks.

**Lemma 1** *Keeping the same notations and setting as Theorem 3, the Rademacher Complexity can be bounded as follows:*

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \sum_{\tau=1}^{T} \mathcal{O}\left(\sqrt{\frac{\mathrm{tr}(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))}{n_\tau^2} \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau}\right);$$

*where $\mathcal{F}_T$ is the function class covered by the model up to the task $\mathcal{T}_\tau$*

The right hand side term $R_T$ in the upper bound of the generalisation theorem follows directly from Lemma 1, and it draws a **complexity measure for Continual Learning**. It states that the upper bound on the Rademacher complexity increases when the tasks are dissimilar. We define the *NTK task dissimilarity* between two subsequent tasks $\mathcal{T}_{\tau-1}$ and $\mathcal{T}_\tau$ as $\bar{S}_{\tau-1\to\tau} = \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau$. This dissimilarity is a generalisation of the term that appears in the upper bound of Thm. 2 by Liu et al. 2019. The knowledge from the previous tasks is encoded in the kernel $\kappa_\tau$, through the feature map $\phi_\tau$. As an edge case, if two successive tasks are identical, $\bar{S}_{\tau-1\to\tau} = 0$ and the upper bound does not increase.

**Implications for Curriculum Learning** We also observe that the upper bound depends on the task ordering, which may provide a theoretical explanation on the importance of learning with a curriculum (Bengio et al. (2009)). In the following, we present an edge case which provided an intuition on how the bound captures the importance of the order. Consider two dissimilar tasks $\mathcal{T}_1$ and $\mathcal{T}_2$. A sequence of tasks alternating between $\mathcal{T}_1$ and $\mathcal{T}_2$ will lead to a large upper bound, as explained in the first paragraph, while a sequence of tasks concatenating two sequences of $\mathcal{T}_1$ then $\mathcal{T}_2$ will lead to a lower upper bound.

## 6 THE IMPORTANCE OF THE NTK VARIATION FOR CONTINUAL LEARNING

In the previous section, we demonstrated that under the NTK regime and an infinite memory, OGD is provably robust to catastrophic forgetting. In practical settings, these two assumptions do not hold, therefore, in this section, we study the limits of the NTK regime. We present OGD+, a variant of OGD we use to study the importance of the NTK variation for Continual Learning in practice.

In order to decouple the NTK variation phenomena in our experiments, we propose then study the OGD+ algorithm, which is designed to be more robust to the NTK variation, through updating its orthonormal basis with respect to all the tasks **at the end of each task**, as opposed to OGD.

Algorithm 1 presents the OGD+ algorithm, we highlight the differences with OGD in red. The main difference is that OGD+ stores the feature maps with respect to the samples from previous tasks, in addition to the feature maps with respect to the samples from the current task, as opposed to OGD. This small change is motivated by the variation of the NTK in practice. In order to compute the feature maps with respect to the previous samples, OGD+ saves these samples in a dedicated memory, we call this storage the *samples memory*. This memory comes in addition to the orthonormal *feature maps memory*. The only role of the *samples memory* is to compute the updated feature maps at the beginning of each task.

**Algorithm 1:** OGD+ for Continual Learning

**Input** : A task sequence $\mathcal{T}_1, \mathcal{T}_2, \ldots$, learning rate $\eta$

1. Initialize $S_J \leftarrow \{\}$ ; $S_D \leftarrow \{\}$; $\boldsymbol{w} \leftarrow \boldsymbol{w}_0$
2. **for** *Task ID* $\tau = 1, 2, 3, \ldots$ **do**

**repeat**

$\boldsymbol{g} \leftarrow$ Stochastic Batch Gradient for $\mathcal{T}_\tau$ at $\boldsymbol{w}$;

$\tilde{\boldsymbol{g}} = \boldsymbol{g} - \sum_{\boldsymbol{v} \in \mathcal{S}_J} \text{proj}_{\boldsymbol{v}}(\boldsymbol{g})$;

$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \tilde{\boldsymbol{g}}$

**until** *convergence*;

Sample $\mathcal{S} \subset \mathcal{S}_D$;

**for** $(\boldsymbol{x}, y) \in \mathcal{D}_\tau \bigcup \mathcal{S}$ *and* $k \in [1, c]$ *s.t.* $y_k = 1$ **do**

$\mathbf{u} \leftarrow \nabla f_\tau(\boldsymbol{x}; \boldsymbol{w}) - \sum_{\boldsymbol{v} \in \mathcal{S}_J} \text{proj}_{\boldsymbol{v}}(\nabla f_\tau(\boldsymbol{x}; \boldsymbol{w}))$ $\mathcal{S}_J \leftarrow \mathcal{S}_J \bigcup \{\mathbf{u}\}$

**end**

Sample $\mathcal{D} \subset \mathcal{D}_\tau$ ;

Update $\mathcal{S}_D \leftarrow \mathcal{S}_D \bigcup \mathcal{D}$

**end**

## 7 Experiments

Now, we study the validity Theorem 2 which states that under the given conditions, OGD is perfectly robust to Catastrophic Forgetting (Sec. 7.1). Then we benchmark the applicability of our framework in non-overparameterised settings, using the OGD+ algorithm we presented (Sec. 7.2).

**Benchmarks** We use the standard benchmarks similarly to Goodfellow et al. (2014) and Chaudhry et al. (2019). Permuted MNIST Goodfellow et al. (2014) consists of a series of MNIST supervised learning tasks, where the pixels of each task are permuted with respect to a fixed permutation. Rotated MNIST Farajtabar et al. (2019) consists of a series of MNIST classification tasks, where the images are rotated with respect to a fixed angle, monotonically. We increment the rotation angle by 5 degrees at each new task. Split CIFAR-100 Chaudhry et al. (2019) is constructed by splitting the original CIFAR-100 dataset Krizhevsky (2009) into 20 disjoint subsets, where each subset is formed by sampling without replacement 5 classes out of 100. In order to assess the robustness to catastrophic forgetting over long tasks sequences, we increase the length of the tasks streams from 5 to 15 for the MNIST benchmarks, and consider all the 20 tasks for the Split CIFAR-100 benchmark. Reproducibility details are presented in App. F.

### 7.1 Memorisation property of OGD (Thm. 2)

Thm. 2 states that in the NTK regime, given an infinite memory, the train error of OGD is unchanged. We study the impact the two assumptions on Catastrophic Forgetting through ablation studies.

**Experiment** We train a model on the full task sequence on the Rotated MNIST benchmark, then measure the variation of the train accuracy of the memorised samples from the first task. The reason we consider the samples in the memory is the applicability of Thm. 2 to these samples only. Then, we study how increasing overparameterisation or increasing the memory size impacts this variation.

**Results** The figures 1, 3 and 4 show that the train error variation decreases with overparameterisation and with the memory size for OGD, on all benchmarks. This result concurs with Thm. 2.

### 7.2 The importance of the NTK variation for Continual Learning

In Sec. 6, we presented the OGD+ algorithm, a variant of the OGD algorithm we use to study the importance of accounting for the NTK variation in standard Continual Learning settings, where the overparameterisation assumption may not hold.

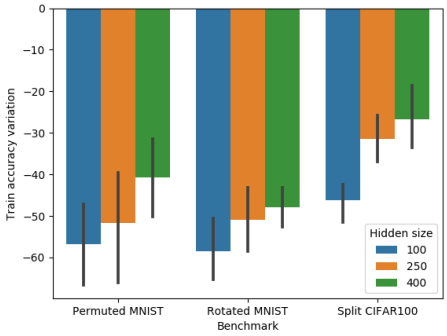



Figure 1: The variation of the memory accuracy on the first task as a function of overparameterisation (higher is better). The forgetting decreases with overparameterisation, as stated in Theorem 2.

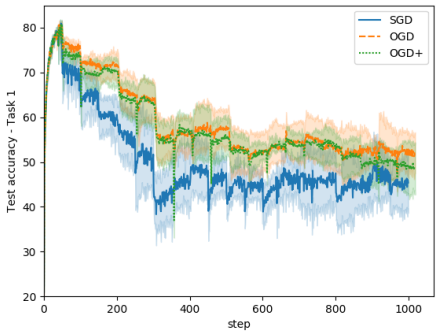



Figure 2: Test accuracy on the first task of Split CIFAR-100, for SGD, OGD and OGD+. The recurring positive backward transfer highlights the Transfer Learning property of this benchmark.

**Experiment** We track the test error through tasks on the standard Continual Learning benchmarks with the standard parameters in the litterature, which correspond to the non-NTK regime.

**Results** Tables 1, 5 and 6 show that the OGD+ outperforms OGD with a significant margin on the Rotated MNIST and Permuted MNIST benchmarks, on the oldest tasks. On the most recent tasks, the model is exposed to less diverse data implying a smaller weight change therefore a smaller feature map variation in which case OGD+ loses its competitive advantage.

While Table 1 shows that OGD outperforms OGD+ on the CIFAR-100 benchmark on the first task, Fig. 2 shows that this difference is very small overall. One probable reason behind this small performance gain is the small feature map variation. This is due to the underlying stucture of the benchmark which implies more Transfer learning between tasks than the MNIST benchmarks. Figures 2 and 5 confirm this hypothesis : while backward transfer occurs frequently on CIFAR-100 where the test error increases frequently, no backward transfer occurs on Rotated MNIST where the test accuracy decreases steadily.

For completion, we present in App. A.1 connections between the OGD, OGD+ and A-GEM algorithms on the robustness to the NTK variation.

| | **Split MNIST** | **Rotated MNIST** | **Permuted MNIST** | **Split CIFAR 100** |
|---|---|---|---|---|
| OGD+ | **96.8±0.9** | **65.4±1.2** | **72.8±1.6** | 48.8±6.9 |
| OGD | 96.2±1.7 | 41.7±1.6 | 33.1±9.2 | **56.2±4.3** |
| SGD | 83.8±4.1 | 28.6±1.3 | 34.8±2.4 | 45.4±8.3 |

Table 1: The test accuracy on the first task after the model was trained on all tasks in sequence, for a given benchmark. The best continual learning results are highlighted in **bold**.

## 8 Conclusion

We presented a theoretical framework to study Continual Learning algorithms in the NTK regime. We leveraged this framework to study the convergence and generalisation properties of the SGD and OGD algorithms. Then, we highlighted with experiments the importance of the NTK variation through the OGD+ algorithm. Our analysis highlights multiple connections to neighbouring fields such as Transfer Learning and Curriculum Learning. Extending the analysis to other Continual Learning algorithms is a promising future direction, which would provide a better understanding of the various properties of these algorithms and how they could be improved. It would also be interesting to investigate the applicability of the theory to neighbouring fields such as meta-learning or multitask learning. We hope this framework provides new keys to investigate these directions.

## REFERENCES


Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning, 02 2019.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018.

Pierre Alquier, The Tien Mai, and Massimiliano Pontil. Regret Bounds for Lifelong Learning. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 261–269, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

Bernard Ans and Stéphane Rousset. Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Comptes Rendus de l'Académie des Sciences - Series III - Sciences de la Vie*, 320 (12):989 – 997, 1997.

Bernard Ans and Stéphane Rousset. Neural networks with a self-refreshing memory: Knowledge transfer in sequential learning tasks without catastrophic forgetting. *Connection Science*, 12:1–19, 04 2000. doi: 10.1080/095400900116177.

Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *CoRR*, abs/1901.08584, 2019.

Natalia D í az Rodr í guez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning. *CoRR*, abs/1810.13166, 2018.

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482, March 2003.

Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In Bernhard Schölkopf and Manfred K. Warmuth (eds.), *Learning Theory and Kernel Machines*, pp. 567–580, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. ISBN 978-3-540-45167-9.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380.

Frederik Benzing. Understanding regularisation methods for continual learning. *ArXiv*, abs/2006.06357, 2020.

Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. In *NeurIPS*, 2019.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a- GEM . In *International Conference on Learning Representations*, 2019.

Giulia Denevi, Carlo Ciliberto, Dimitris Stamos, and Massimiliano Pontil. Incremental learning-to-learn with statistical guarantees, 2018a.

Giulia Denevi, Carlo Ciliberto, Dimitris Stamos, and Massimiliano Pontil. Learning to learn around a common mean. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 10169–10179. Curran Associates, Inc., 2018b.

Lee H. Dicker, Dean P. Foster, and Daniel Hsu. Kernel ridge vs. principal component regression: Minimax bounds and the qualification of regularization operators. *Electron. J. Statist.*, 11(1): 1022–1047, 2017. doi: 10.1214/17-EJS1258.

Simon S. Du, Xiyu Zhai, Barnab á s P ó czos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *CoRR*, abs/1810.02054, 2018.

Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. *ArXiv*, abs/2002.09434, 2020.

Sayna Ebrahimi, Mohamed Elhoseiny, Trevor Darrell, and Marcus Rohrbach. Uncertainty-guided continual learning with bayesian neural networks. In *International Conference on Learning Representations*, 2020.

Cong Fang, Hanze Dong, and Tong Zhang. Over parameterized two-level neural networks can learn near optimal feature representations, 2019.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. *ArXiv*, abs/1910.07104, 2019.

Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3: 128–135, 05 1999. doi: 10.1016/S1364-6613(99)01294-2.

S. Goldt, M. M é zard, F. Krzakala, and L. Zdeborov á . Modelling the influence of data structure on learning in neural networks, 2020.

Sebastian Goldt, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborov´ a . Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 6979–6989. Curran Associates, Inc., 2019.

Ian J. Goodfellow, Mehdi Mirza, Xia Da, Aaron C. Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. *CoRR*, abs/1312.6211, 2014.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. Mean-field behaviour of neural tangent kernel for deep neural networks, 2019.

Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL `http://arxiv.org/abs/1502.03167`.

Arthur Jacot, Franck Gabriel, and Clé ment Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 8580–8589, Red Hook, NY, USA, 2018. Curran Associates Inc.

Kaiyi Ji, Junjie Yang, and Yingbin Liang. Multi-step model-agnostic meta-learning: Convergence and improved algorithms, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114, 12 2016. doi: 10.1073/pnas.1611835114.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114.

Jeremias Knoblauch, H. Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. *ArXiv*, abs/2006.05188, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent, 2019.

Jeongtae Lee, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *ArXiv*, abs/1708.01547, 2018.

Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. *CoRR*, abs/1903.11680, 2019.

Hong Liu, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Towards understanding the transferability of deep representations, 2019.

Nicolas Masse, Gregory Grant, and David Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115, 02 2018. doi: 10.1073/pnas.1803839115.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

Cuong V Nguyen, Alessandro Achille, Michael Lam, Tal Hassner, Vijay Mahadevan, and Stefano Soatto. Toward understanding catastrophic forgetting in continual learning. *ArXiv*, abs/1908.01091, 2019.

Pingbo Pan, Alexander Immer, Siddharth Swaroop, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past, 2020.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54 – 71, 2019.

Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97 2:285–308, 1990.

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 3742–3752, Red Hook, NY, USA, 2018. Curran Associates Inc.

A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7:123–, 01 1995.

Nikunj Saunshi, Yi Zhang, Mikhail Khodak, and Sanjeev Arora. A sample complexity separation between non-convex and convex meta-learning, 2020.

Jonathan Schwarz, Jelena Luketina, Wojciech Czarnecki, Agnieszka Grabska-Barwinska, Yee Teh, Razvan Pascanu, and Raia Hadsell. Progress and compress: A scalable framework for continual learning, 05 2018.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *CoRR*, abs/1801.01423, 2018.

Alex J. Smola, Bernhard Schö lkopf, and Klaus-Robert Mü ller. The connection between regularization operators and support vector kernels. *Neural Netw.*, 11(4):637–649, June 1998. doi: 10.1016/S0893-6080(98)00032-X.

Gido M Van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.

Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Ruosong Wang, Simon S. Du, Lin F. Yang, and Sham M. Kakade. Is long horizon reinforcement learning more difficult than short horizon reinforcement learning?, 2020.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets v.s. their induced kernel. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 9709–9721. Curran Associates, Inc., 2019.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning, 2020.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 3987–3995. JMLR.org, 2017.

# A Complementary discussion

## A.1 NTK variation - The importance of orthogonality for the OGD, OGD+ and A-GEM algorithms

In the main article, we focused only on the SGD, OGD and OGD+ algorithms. In this section, we highlight a connection of these algorithms to the A-GEM algorithm. All these algorithms perform orthogonal projections during the weight update step, however these projections differ in the following ways :

- the span of the space the updates are orthogonal to.
- the rate at which the projection space is updated.

Additionally, the A-GEM algorithm present the additional property of allowing backward positive transfer by design.

**Similarities of the algorithms** Table 2 presents an overview of the connections between the algorithm. In practice, the models are not trained in the NTK regime. Since the feature map changes through training, the orthogonality constraint of OGD becomes less relevant and may harm the learning. On the other hand, while A-GEM performs a projection on a smaller subspace which may not protect all the subspaces, the subspace is updated as it is computed at each training step, the updates are therefore expected to be more relevant. Finally OGD+ lies at the intersection of both, while it doesn't update the feature maps at each gradient step, the projection constraint is with respect to a larger space.

| | Projection Space | | | |
|---|---|---|---|---|
| Algorithm | Span | Granularity | Update rate | Backward Transfer |
| SGD | None | None | None | No |
| OGD | Full | Sample | Task | Unclear |
| OGD+ | Full | Sample | Task | Unclear |
| A-GEM | Sample | Sample | Gradient step | Yes |

Table 2: Overview of the properties of the SGD, OGD, OGD+, A-GEM and A-GEM-NT algorithms.

**On the theoretical connection between OGD and A-GEM** The A-GEM algorithm (Chaudhry et al., 2019) is a state of the art Continual Learning algorithm on the standard benchmarks. The idea behind the algorithm is to perform a gradient descent if an estimate of the loss over the previous tasks increases or is unchanged. Otherwise the gradient is projected orthogonally to the gradient over the loss estimate over the previous tasks.

We find that OGD draws a upper bound in comparison to A-GEM-NT in terms of generalisation error, as stated in Proposition 1.

**Proposition 1** *In the NTK regime, OGD implies A-GEM with no backward positive transfer.*

The proof is presented in App. E.1.

**Experiments** The extended experiments on the importance of the NTK variation for Continual Learning (Sec. F.4), show that A-GEM outperform OGD and OGD+ on most benchmarks. A probable reason behind this difference is the rate of update of the feature map for the A-GEM algorithm, even though it spans a smaller space than OGD and OGD+.

### A.2 EXPRESSIONS OVERVIEW

For convenience, we present in Table 3 an overview of the notions that were mentioned in the main article, with their respective mathematical expressions.

| **Name** | **Expression** |
|---|---|
| Residual | $\tilde{\mathbf{y}}_\tau = \boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau}$ |
| Knowledge increment | $\tilde{f}^\star_\tau(\boldsymbol{x}) = \kappa_\tau(\boldsymbol{x}, \mathbf{X}_\tau)^T(\kappa_{\tau+1}(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau$ |
| Projection matrix | $\mathbf{T}_\tau$ |
| Feature map | $\phi(\boldsymbol{x})$ |
| Effective feature map | $\widetilde{\phi}_\tau(\boldsymbol{x})$ |
| Effective kernel | $\kappa_\tau(\boldsymbol{x}, \boldsymbol{x}')$ |
| Complexity measure | $\sum_{\tau=1}^T \sqrt{\frac{\mathrm{tr}(\kappa_\tau(\mathbf{X}_\tau,\mathbf{X}_\tau))}{n_\tau^2}\tilde{\mathbf{y}}_\tau^T(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau}$ |
| NTK task dissimilarity | $\bar{S}_{\tau-1\to\tau} = \tilde{\mathbf{y}}_\tau^T(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau$ |

Table 3: Overview of the notions presented in the main manuscript with their respective notations.

# B PROOFS OVERVIEW

## B.1 CONVERGENCE :

**Theorem 1 : Convergence of SGD and OGD for Continual Learning** We prove Theorem 1 by induction. We rewrite the loss function as a regression on the residual $\tilde{\boldsymbol{y}}_\tau$ instead of $\boldsymbol{y}_\tau$. Then, we rewrite the optimisation objective as an unconstrained strongly convex optimisation problem. Finally, we compute the unique solution in a closed form. The full proof is presented in App. C.1.

**Remark 1 : Continual Learning as a recursive Kernel Regression** The remark follows directly from the recursive form of Theorem 1.

**Corollary 1 : Distance from initialisation through tasks** The proof follows immediatly from the proof of Theorem 1, in which we compute a closed form of the weights variation. It is presented in App. C.2.

## B.2 GENERALISATION :

**Theorem 2 : No-forgetting Continual Learning with OGD** The proof relies on the orthogonality property of the OGD update, which implies that the subspaces spanned by the samples in the memory would not have any changes incurred. The proof is presented in App . D.1

**Theorem 3 : Generalisation of SGD and OGD for Continual Learning** The proof or Theorem 3 is presented in App. D.2. The proof follows the following structure :

- Bounding the Rademacher complexity (App. D.2.3) : First, we state the technical Lemma 3 that upper bounds the Rademacher complexity of the function class that corresponds to the set of linear combinations of kernel regressors. Then we apply the lemma to the function class obtained through Theorem 1.
- Bounding the empirical loss for SGD (App. D.2.5) : the error is the same as OGD on the last task. While on the previous tasks, Catastrophic Forgetting can be incurred which implies the appearance of a residual term that corresponds to the forgetting.
- Bounding the empirical loss for OGD (App. D.2.4) : The proof techniques are very similar to the SGD case, however we leverage Theorem 2 in order to derive tighter bounds due to the absence of Catastrophic Forgetting.
- Wrap-up (App. D.2.6) : Finally, we wrap-up all the lemmas to derive the bound of Theorem 3.

**Lemma 1 - Implications for Curriculum Learning** This results follows from the technical Lemma 3, which upper bounds the Rademacher complexity of the function class that corresponds to the set of linear combinations of kernel regressors.

## B.3 THE IMPORTANCE OF THE NTK VARIATION FOR CONTINUAL LEARNING :

**Proposition 1 - OGD implies A-GEM-NT** The proof relies mainly on Theorem 2 stating the robustness of OGD to Catastrophic Forgetting. We derive the A-GEM update then apply Theorem 2, which leads to the implication. The full proof is presented in App. D.2.3.

# C MISSING PROOFS OF SECTION 4 - CONVERGENCE

## C.1 PROOF OF THEOREM 1

**Stochastic Gradient Descent** We start by proving the Stochastic Gradient Descent (SGD) case of Theorem 1.

**Proof**

We prove the Theorem 1 by induction.

Our induction hypothesis $H_\tau$ is the following : $\mathcal{H}_\tau$ : For all $k \leq \tau$, Theorem 1 holds.

First, we prove that $\mathcal{H}_1$ holds.

The proof is straightforward. For the first task, since there were no previous tasks, OGD on this task is the same as SGD.

Therefore, it is equivalent to minimising the following objective :

$$\arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left\| f_0(\mathbf{X}_1) + \phi(\mathbf{X}_1)^T(\boldsymbol{w} - \boldsymbol{w}_0^\star) - \boldsymbol{y}_1 \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_0^\star\|^2 \tag{1}$$

where $\phi(\boldsymbol{x}) = \nabla_{\boldsymbol{w}_0^\star} f_0^\star(\boldsymbol{x})$.

We replace into the objective with $\tilde{\mathbf{y}}_1$ applying $\tilde{\mathbf{y}}_1 = \boldsymbol{y}_1 - f_0(\mathbf{X}_1)$ :

$$\arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left\| \phi(\mathbf{X}_1)^T(\boldsymbol{w} - \boldsymbol{w}_0^\star) - \tilde{\mathbf{y}}_1 \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_0^\star\|^2 \tag{2}$$

The objective is quadratic and the Hessian is positive definite, therefore the minimum exists and is unique :

$$\boldsymbol{w}_1^\star - \boldsymbol{w}_0^\star = \phi(\mathbf{X}_1)(\phi(\mathbf{X}_1)^T\phi(\mathbf{X}_1) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{3}$$

Under the NTK regime assumption :

$$f_1^\star(\boldsymbol{x}) = f_0^\star(\boldsymbol{x}) + \nabla_{\boldsymbol{w}} f_0^\star(\boldsymbol{x})^T(\boldsymbol{w}_1^\star - \boldsymbol{w}_0^\star) \tag{4}$$

Then, by replacing into $\boldsymbol{w}_1^\star - \boldsymbol{w}_0^\star$ :

$$f_1^\star(\boldsymbol{x}) = f_0^\star(\boldsymbol{x}) + \nabla_{\boldsymbol{w}} f_0^\star(\boldsymbol{x})^T\phi(\mathbf{X}_1)(\phi(\mathbf{X}_1)^T\phi(\mathbf{X}_1) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{5}$$

$$f_1^\star(\boldsymbol{x}) = f_0^\star(\boldsymbol{x}) + \kappa_1(\boldsymbol{x}, \mathbf{X}_1)(\kappa_1(\mathbf{X}_1, \mathbf{X}_1) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{6}$$

Finally :

$$f_1^\star(\boldsymbol{x}) - f_0^\star(\boldsymbol{x}) = \kappa_1(\boldsymbol{x}, \mathbf{X}_1)(\kappa_1(\mathbf{X}_1, \mathbf{X}_1) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{7}$$

Which completes the proof of $\mathcal{H}_1$.

Let $\tau \in \mathbb{N}^\star$, we assume that $\mathcal{H}_\tau$ is true, then we show that $\mathcal{H}_{\tau+1}$ is true.

On the task $\mathcal{T}_{\tau+1}$, we can write the loss $\mathcal{L}^{\tau+1}$ as :

$$\mathcal{L}^{\tau+1}(\boldsymbol{w}) = \left\| \phi_\tau(\mathbf{X}_{\tau+1})^T(\boldsymbol{w} - \boldsymbol{w}_\tau^\star) - \tilde{\mathbf{y}}_{\tau+1} \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_\tau^\star\|^2 \tag{8}$$

We recall that the optimisation problem on the task $\mathcal{T}_{\tau+1}$ is :

$$\arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \left\| \phi_\tau(\mathbf{X}_{\tau+1})^T(\boldsymbol{w} - \boldsymbol{w}_\tau^\star) - \tilde{\mathbf{y}}_{\tau+1} \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_\tau^\star\|^2 \tag{9}$$

The optimisation objective is quadratic, unconstrainted, with a positive definite hessian. Therefore, an optimum exists and is unique :

$$\boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star = \phi(\mathbf{X}_{\tau+1})(\phi(\mathbf{X}_{\tau+1})^T\phi(\mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{10}$$

We define the kernel $\kappa_{\tau+1} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as :

$$\kappa_{\tau+1}(\boldsymbol{x}, \boldsymbol{x}') = \phi_\tau(\boldsymbol{x})^T \phi_\tau(\boldsymbol{x}') \quad \text{for all } \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d \tag{11}$$

Finally, we recover a closed form expression for $f^\star_{\tau+1}$ :

First, we use the induction hypothesis $\mathcal{H}_\tau$ :

$$\begin{aligned} f^\star_{\tau+1}(\boldsymbol{x}) &= f^\star_\tau(\boldsymbol{x}) + \langle \nabla_{\boldsymbol{w}} f^\star_\tau(\boldsymbol{x}), \boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau \rangle && (12)\\ &= f^\star_\tau(\boldsymbol{x}) + \phi_\tau(\boldsymbol{x})\phi(\mathbf{X}_{\tau+1})(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} && (13)\\ &= f^\star_\tau(\boldsymbol{x}) + \kappa_{\tau+1}(\boldsymbol{x}, \mathbf{X}_{\tau+1})(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} && (14) \end{aligned}$$

At this stage, we have proven $\mathcal{H}_{t+1}$.

$$(15)$$

**Orthogonal Gradient Descent** Now, we prove the Orthogonal Gradient Descent (OGD) case of Theorem 1. The key difference in the proof is that the OGD optimisation objective is constrained. The constraints correspond to the orthogonality to the subspace spanned by the memorised feature maps from the previous tasks. Another key difference occurs during the regularisation, as opposed to the SGD case where the regularisation of the weights spans on the whole space, the regularisation of OGD only applies to the "learnable" space. This property follows from the orthogonality constraint, which enforces that the space spanned by the previous tasks is unchanged.

**Proof**

We prove the Theorem 1 by induction. Our induction hypothesis $\mathcal{H}_\tau$ is the following : $\mathcal{H}_\tau$ : For all $k \leq \tau$, Theorem 1 holds.

First, we prove that $\mathcal{H}_1$ holds.

The proof is straightforward. For the first task, since there were no previous tasks, OGD on this task is the same as SGD.

Therefore, it is equivalent to minimising the following objective :

$$\underset{\boldsymbol{w} \in \mathbb{R}^d}{\arg\min} \left\| f_0(\mathbf{X}_1)\phi(\mathbf{X}_1)^T(\boldsymbol{w} - \boldsymbol{w}^\star_0) - \boldsymbol{y}_1 \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}^\star_0\|^2 \tag{16}$$

We replace into the objective by the residual term $\tilde{\mathbf{y}}_1 = y_1 - f_0(\mathbf{X}_1)$

$$\underset{\boldsymbol{w} \in \mathbb{R}^d}{\arg\min} \left\| \phi(\mathbf{X}_1)^T(\boldsymbol{w} - \boldsymbol{w}^\star_0) - \tilde{\mathbf{y}}_1 \right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_0\|^2 \tag{17}$$

where $\phi(\boldsymbol{x}) = \nabla_{\boldsymbol{w}^\star_0} f^\star_0(\boldsymbol{x})$.

The objective is quadratic and its Hessian is positive definite, therefore the minimum exists and is unique :

$$\boldsymbol{w}^\star_1 - \boldsymbol{w}^\star_0 = \phi(\mathbf{X}_1)(\phi(\mathbf{X}_1)^T\phi(\mathbf{X}_1) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{18}$$

Under the NTK regime assumption :

$$f^\star_1(\boldsymbol{x}) = f^\star_0(\boldsymbol{x}) + \nabla_{\boldsymbol{w}} f^\star_0(\boldsymbol{x})^T(\boldsymbol{w}^\star_1 - \boldsymbol{w}^\star_0) \tag{19}$$

Then, by replacing into $\boldsymbol{w}^\star_1 - \boldsymbol{w}^\star_0$ :

$$\begin{aligned} f^\star_1(\boldsymbol{x}) &= f^\star_0(\boldsymbol{x}) + \nabla_{\boldsymbol{w}} f^\star_0(\boldsymbol{x})^T\phi(\mathbf{X}_1)(\phi(\mathbf{X}_1)^T\phi(\mathbf{X}_1) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_1 && (20)\\ f^\star_1(\boldsymbol{x}) &= f^\star_0(\boldsymbol{x}) + \kappa_1(\boldsymbol{x}, \mathbf{X}_1)(\kappa_1(\mathbf{X}_1, \mathbf{X}_1) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_1 && (21) \end{aligned}$$

Finally :

$$f^\star_1(\boldsymbol{x}) - f^\star_0(\boldsymbol{x}) = \kappa_1(\boldsymbol{x}, \mathbf{X}_1)(\kappa_1(\mathbf{X}_1, \mathbf{X}_1) + \lambda \mathbf{I})^{-1}\tilde{\mathbf{y}}_1 \tag{22}$$

Which completes the proof of $\mathcal{H}_1$.
Let $\tau \in \mathbb{N}^\star$, we assume that $\mathcal{H}_\tau$ is true, then we show that $\mathcal{H}_{\tau+1}$ is true.
On the task $\mathcal{T}_{\tau+1}$, we can write the loss $\mathcal{L}^{\tau+1}$ as :

$$\mathcal{L}^{\tau+1}(\boldsymbol{w}) = \left\|\phi_\tau(\mathbf{X}_{\tau+1})^T(\boldsymbol{w} - \boldsymbol{w}_\tau^\star) - \tilde{\mathbf{y}}_{\tau+1}\right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_\tau^\star\|^2 \tag{23}$$

We recall that the optimisation problem on the task $\mathcal{T}_{\tau+1}$ is :

$$\underset{\boldsymbol{w}\in\mathbb{R}^d}{\arg\min}\left\|\phi_\tau(\mathbf{X}_{\tau+1})^T(\boldsymbol{w} - \boldsymbol{w}_\tau^\star) - \tilde{\mathbf{y}}_{\tau+1}\right\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{w} - \boldsymbol{w}_\tau^\star\|^2 \tag{24}$$

$$\text{u.c.} \quad \mathbf{V}_{\tau+1}(\boldsymbol{w} - \boldsymbol{w}_\tau^\star) = 0 \tag{25}$$

where $\mathbf{V}_{\tau+1}$ is a projection matrix on $(\bigoplus_{k=1}^{\tau} \mathbb{E}_k)^\perp$, the euclidian space induced by the feature maps of the tasks $\{\mathcal{T}_k, k \in [\tau]\}$.
Let $\mathbf{T}_{\tau+1} \in \mathbb{R}^{d\times(d-K_{\tau+1})}$ and $\tilde{\mathbf{w}} \in \mathbb{R}^{d-K_{\tau+1}}$ such as :

$$\boldsymbol{w} - \boldsymbol{w}_\tau^\star = \mathbf{T}_{\tau+1}\tilde{\mathbf{w}} \tag{26}$$

$$K_\tau = \dim(\bigoplus_{k=1}^{\tau} \mathbb{E}_k) \tag{27}$$

We rewrite the objective by plugging in the variables we just defined. The two objectives are equivalent :

$$\underset{\tilde{\mathbf{w}}\in\mathbb{R}^{d-K_{\tau+1}}}{\arg\min} \left\|\phi_\tau(\mathbf{X}_{\tau+1})^T\mathbf{T}_{\tau+1}\tilde{\mathbf{w}} - \tilde{\mathbf{y}}_{\tau+1}\right\|_2^2 + \frac{\lambda}{2}\|\mathbf{T}_{\tau+1}\tilde{\mathbf{w}}\|^2 \tag{28}$$

This objective is equivalent to :

$$\underset{\tilde{\mathbf{w}}\in\mathbb{R}^{d-K_{\tau+1}}}{\arg\min} \left\|\phi_\tau(\mathbf{X}_{\tau+1})^T\mathbf{T}_{\tau+1}\tilde{\mathbf{w}} - \tilde{\mathbf{y}}_{\tau+1}\right\|_2^2 + \frac{\lambda}{2}\|\tilde{\mathbf{w}}\|^2 \tag{29}$$

For clarity, we define $\mathbf{Z}_{\tau+1}\mathbb{R}^{n_{\tau+1}\times(d-K_{\tau+1})}$ as :

$$\mathbf{Z}_{\tau+1} = \phi_\tau(\mathbf{X}_{\tau+1})^T\mathbf{T}_{\tau+1} \tag{30}$$

By plugging in $\mathbf{Z}_{\tau+1}$, we rewrite the objective as :

$$\underset{\tilde{\mathbf{w}}\in\mathbb{R}^{d-K_{\tau+1}}}{\arg\min} \left\|\mathbf{Z}_{\tau+1}\tilde{\mathbf{w}} - \tilde{\mathbf{y}}_{\tau+1}\right\|_2^2 + \frac{\lambda}{2}\|\mathbf{T}_{\tau+1}\tilde{\mathbf{w}}\|^2 \tag{31}$$

The optimisation objective is quadratic, unconstrainted, with a positive definite hessian. Therefore, an optimum exists and is unique :

$$\tilde{\mathbf{w}}_{\tau+1}^\star = \mathbf{Z}_{\tau+1}^T(\mathbf{Z}_{\tau+1}\mathbf{Z}_{\tau+1}^T + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{32}$$

We recover the expression of the optimum in the original space :

$$\boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star = \mathbf{T}_{\tau+1}\mathbf{Z}_{\tau+1}^T(\mathbf{Z}_{\tau+1}\mathbf{Z}_{\tau+1}^T + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{33}$$

We define the kernel $\kappa_{\tau+1} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as :

$$\kappa_{\tau+1}(\boldsymbol{x}, \boldsymbol{x}') = \phi_\tau(\boldsymbol{x})^T\mathbf{T}_{\tau+1}\mathbf{T}_{\tau+1}^T\phi_\tau(\boldsymbol{x}') \quad \text{for all } \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d \tag{34}$$

Now we rewrite $\boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star$ :

$$\boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star = \mathbf{T}_{\tau+1}\mathbf{Z}_{\tau+1}^T(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{35}$$

Finally, we recover a closed form expression for $f_{\tau+1}^\star$ :
First, we use the induction hypothesis $\mathcal{H}_\tau$ :

$$f_{\tau+1}^\star(\boldsymbol{x}) = f_\tau^\star(\boldsymbol{x}) + \langle\nabla_{\boldsymbol{w}} f_\tau^\star(\boldsymbol{x}), \boldsymbol{w}_{\tau+1}^\star - \boldsymbol{w}_\tau^\star\rangle \tag{36}$$

$$= f_\tau^\star(\boldsymbol{x}) + \phi_\tau(\boldsymbol{x})\mathbf{T}_{\tau+1}\mathbf{Z}_{\tau+1}^T(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{37}$$

$$= f_\tau^\star(\boldsymbol{x}) + \kappa_{\tau+1}(\boldsymbol{x}, \mathbf{X}_{\tau+1})(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{38}$$

At this stage, we have proven $\mathcal{H}_{t+1}$.
We conclude the proof of Thm. 1.

$$\tag{39}$$

### C.2 Proof of the Corollary 1

**Stochastic Gradient Descent** The proof follows immediately from Thm. 1.

**Proof**

In the proof of Theorem 1 (App. C.1), for the SGD case, we showed that :

$$\boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau = \phi(\mathbf{X}_{\tau+1})(\phi(\mathbf{X}_{\tau+1})^T\phi(\mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{40}$$

This result implies that :

$$\left\|\boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau\right\|^2 = \tilde{\mathbf{y}}^T_{\tau+1}(\phi(\mathbf{X}_{\tau+1})^T\phi(\mathbf{X}_{\tau+1}) + \lambda\mathbf{I})\phi(\mathbf{X}_{\tau+1})^T\phi(\mathbf{X}_{\tau+1})(\phi(\mathbf{X}_{\tau+1})^T\phi(\mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{41}$$

$$= \tilde{\mathbf{y}}^T_{\tau+1}(\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1})(\kappa(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{42}$$

$$\tag{43}$$

**Orthogonal Gradient Descent** The proof is exatcly the same as the proof above, the difference lies in the kernel definition, which is implicit.

**Proof**

In the proof of Theorem 1 (App. C.1), for the OGD case, we showed that :

$$\boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau = \mathbf{T}_{\tau+1}\mathbf{Z}^T_{\tau+1}(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{44}$$

This result implies that :

$$\left\|\boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau\right\|^2 = \tilde{\mathbf{y}}^T_{\tau+1}(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\mathbf{Z}_{\tau+1}\mathbf{T}^T_{\tau+1}\mathbf{T}_{\tau+1}\mathbf{Z}^T_{\tau+1}(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{45}$$

$$= \tilde{\mathbf{y}}^T_{\tau+1}(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1})(\kappa_{\tau+1}(\mathbf{X}_{\tau+1}, \mathbf{X}_{\tau+1}) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_{\tau+1} \tag{46}$$

$$\tag{47}$$

# D MISSING PROOFS OF SECTION 5 - GENERALISATION

## D.1 PROOF OF THEOREM 2

The intuition behind the proof is : since the gradient updates were performed orthogonally to the feature maps of the training data of the source task, the parameters in this space are unchanged, while the remaining space, which was changed, is orthogonal to these features maps, therefore, the inference is the same and the training error remanis the same as at the end of training on the source task.

**Proof**

In the proof of Theorem 1, App. C.1, we showed that, for $\mathcal{T}_\tau$ a fixed task:

$$f^\star_{\tau+1}(\boldsymbol{x}) = f^\star_\tau(\boldsymbol{x}) + \langle \nabla_{\boldsymbol{w}} f^\star_\tau(\boldsymbol{x}), \boldsymbol{w}^\star_{\tau+1} - \boldsymbol{w}^\star_\tau \rangle. \tag{48}$$

We rewrite the recursive relation into a sum:

$$f^\star_{\tau+1}(\boldsymbol{x}) - f^\star_0(\boldsymbol{x}) = \sum_{k=1}^{\tau} \langle \nabla_{\boldsymbol{w}} f^\star_k(\boldsymbol{x}), \boldsymbol{w}^\star_{k+1} - \boldsymbol{w}^\star_k \rangle. \tag{49}$$

We observe that, for all $k \in [T]$:

$$\boldsymbol{w}^\star_{k+1} - \boldsymbol{w}^\star_k \in \mathbb{E}_{k'}. \tag{50}$$

On the other hand, for OGD+, given a sample $\boldsymbol{x}$ from $\mathcal{D}_\tau$, for all $k' \in [\tau+1, T]$ :

$$\nabla_{\boldsymbol{w}} f^\star_k(\boldsymbol{x}) \in \mathbb{E}_{k'} \tag{51}$$

Therefore, for all $k' \in [k+1, \tau]$ :

$$\langle \nabla_{\boldsymbol{w}} f^\star_{k'}(\boldsymbol{x}), \boldsymbol{w}^\star_{k'+1} - \boldsymbol{w}^\star_{k'} \rangle = 0 \tag{52}$$

Therefore :

$$f^\star_\tau(\boldsymbol{x}) = f^\star_k(\boldsymbol{x}) \tag{53}$$

We conclude.

$$\tag{54}$$

## D.2 PROOF OF THEOREM 3

### D.2.1 REMINDERS

**Reminder on RKHS norm**

Let $\kappa$ a kernel, and $\mathcal{H}$ the reproducing kernel Hilbert space (RKHS) corresponding to the kernel $\kappa$.

Recall that the RKHS norm of a function $f(\boldsymbol{x}) = \alpha^T \kappa(\boldsymbol{x}, \mathbf{X})$ is :

$$\|f\|_{\mathcal{H}} = \sqrt{\alpha^T \kappa(\mathbf{X}, \mathbf{X}) \alpha} \tag{55}$$

$$\tag{56}$$

**Reminder on Generalization and Rademacher Complexity** Consider a loss function $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. The population loss over the distribution $\mathcal{D}$, and the empirical loss over $n$ samples $D = \{(\boldsymbol{x}_i, y_i), i \in [n]\}$ from the same distribution $\mathcal{D}$ are defined as :

$$L_D(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[l(f(\boldsymbol{x}), y)] \tag{57}$$

$$L_S(f) = \frac{1}{n} \sum_{i=1}^{n} l(f(\boldsymbol{x}_i), y_i) \tag{58}$$

**Theorem 4** *Suppose the loss function is bounded in $[0, c]$ and is $\rho-$Lipchitz in the first argument. Then, with probability at least $1-\delta$ over sample $S$ of size $n$ :*

$$\sup_{f\in\mathcal{F}} \{L_D(f) - L_S(f)\} \leq 2\rho\hat{\mathcal{R}}(\mathcal{F}) + 3c\sqrt{\frac{\log(2/\delta)}{2n}} \tag{59}$$

### D.2.2 BOUNDING $\left\|\tilde{f}_\tau^\star\right\|_{\mathcal{H}_\tau}$ :

**Lemma 2** *Let $\mathcal{H}_\tau$ the Hilbert space associated to the kernel $\kappa_\tau$ .*

*We recall that :*

$$\tilde{f}_\tau^\star(\boldsymbol{x}) = \kappa_\tau(\boldsymbol{x}, \mathbf{X}_\tau)^T \boldsymbol{\alpha}_\tau \tag{60}$$

$$\boldsymbol{\alpha}_\tau = (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau \tag{61}$$

*Then :*

$$\left\|\tilde{f}_\tau^\star\right\|_{\mathcal{H}_\tau}^2 \leq \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau \tag{62}$$

$$\tag{63}$$

**Proof**

We start from the definition of the RKHS norm of $\tilde{f}_\tau^\star$ :

$$\left\|\tilde{f}_\tau^\star\right\|_{\mathcal{H}_\tau}^2 = \boldsymbol{\alpha}_\tau^T \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) \boldsymbol{\alpha}_\tau \tag{64}$$

$$= \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau \tag{65}$$

Since $(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \leq (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))^{-1}$

$$\leq \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)^{-1} \tilde{\mathbf{y}}_\tau \tag{66}$$

$$\leq \tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau \tag{67}$$

### D.2.3 BOUNDING THE RADEMACHER COMPLEXITY

The goal of this section is to upper bound the Rademacher Complexity. First, we derive a general upper bound for the Rademacher Complexity of a linear combination of kernels in Lemma 3 . Then, we apply this bound to the linear combination of kernels obtained through Thm. 1, which descrbies the model through the Continual Learning.

**A general bound for the Rademacher Complexity**

**Lemma 3 (Rademacher Complexity of a linear combination of kernels)** *Let $\kappa_t : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, t \in [T]$ kernels such that :*

$$\sup_{\boldsymbol{x} \in \mathcal{X}} \|\kappa_t(\boldsymbol{x}, \boldsymbol{x})\| < \infty$$

*To every kernel $\kappa_t$, we associate a feature map $\phi_t : \mathcal{X} \to \mathcal{H}_t$, where $\mathcal{H}_t$ is a Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_t}$, and for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$, $\kappa_t(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi_t(\boldsymbol{x}), \phi_t(\boldsymbol{x}') \rangle_{\mathcal{H}_t}$*

*For all $T \in \mathbb{N}^\star$, given a sequence $\{B_\tau, \tau \in [T]\}$ we define $\mathcal{F}_T$ as follows :*

$$\mathcal{F}_T = \{\boldsymbol{x} \to \sum_{t=1}^{T} f_t(\boldsymbol{x}), \quad f_t(\boldsymbol{x}) = \alpha_t^T \kappa_t(\boldsymbol{x}, \mathbf{X}_t) \quad \forall t \in [T], \|f_t\|_{\mathcal{H}_t} \leq B_t\} \tag{68}$$

*Let $X_1, ..., X_n$ be random elements of $\mathcal{X}$. Then for the class $\mathcal{F}$, we have :*

$$\hat{\mathcal{R}}(\mathcal{F}) \leq \sum_{t=1}^{T} \frac{2B_t}{n_t} (Tr(\kappa_t(\mathbf{X}_t, \mathbf{X}_t)))^{1/2}$$

**Proof**

Let $f \in \mathcal{F}$, and let $\boldsymbol{x} \in \mathcal{X}$ :

$$f(\boldsymbol{x}) = \sum_{\tau=1}^{T} \sum_{i=1}^{n_\tau} \alpha_i^\tau \kappa_\tau(\boldsymbol{x}, \boldsymbol{x}_i^\tau) \tag{69}$$

For all $\tau \in [T]$, we associate a feature map $\phi_\tau : \mathbf{X} \to \mathcal{H}_\tau$

$$\forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X} \quad \kappa_\tau(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi_\tau(\boldsymbol{x}), \phi_\tau(\boldsymbol{x}') \rangle_{\mathcal{H}_\tau} \tag{70}$$

Therefore :

$$f(\boldsymbol{x}) = \sum_{t=1}^{T} \sum_{i=1}^{n_\tau} \alpha_i^\tau \langle \phi_\tau(\boldsymbol{x}_i^\tau), \phi_\tau(\boldsymbol{x}) \rangle_{\mathcal{H}_\tau} \tag{71}$$

$$= \sum_{\tau=1}^{T} \langle \sum_{i=1}^{n_\tau} \alpha_i^\tau \phi_\tau(\boldsymbol{x}_i^\tau), \phi_\tau(\boldsymbol{x}) \rangle_{\mathcal{H}_\tau} \tag{72}$$

On the other hand, the following holds $\forall t \in [T]$ :

$$\left\| \sum_{i=1}^{n_\tau} \alpha_i^\tau \phi_\tau(\boldsymbol{x}_i^\tau) \right\|_{\mathcal{H}_\tau}^2 = \sum_{i,j} \alpha_i^\tau \alpha_j^\tau \kappa_\tau(\boldsymbol{x}_i^\tau, \boldsymbol{x}_j^\tau) \leq B_\tau^2 \tag{73}$$

Therefore :

$$\mathcal{F}_T \subset \{\boldsymbol{x} \to \sum_{t=1}^{T} \langle \boldsymbol{w}_\tau, \phi_\tau(\boldsymbol{x}) \rangle_{\mathcal{H}_\tau}, \|\boldsymbol{w}_\tau\|_2 \leq B_\tau \quad \forall t \in [T]\} := \tilde{\mathcal{F}_T} \tag{74}$$

Now, we derive an upper bound of the Rademacher complexity of $\mathcal{F}_T$ :

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \hat{\mathcal{R}}(\tilde{\mathcal{F}}_T) \tag{75}$$

$$= \mathbb{E}[\sup_{\|\boldsymbol{w}_\tau\|_2 \leq B_\tau, t \in [T]} \sum_{t=1}^{T} \langle \boldsymbol{w}_\tau, \frac{2}{n_\tau} \sum_{i=1}^{n_\tau} \epsilon_i \phi_\tau(\boldsymbol{x}_i^\tau) \rangle_{\mathcal{H}_\tau} | (\mathbf{X}_\tau)] \tag{76}$$

$$= \sum_{t=1}^{T} \mathbb{E}[\sup_{\|\boldsymbol{w}_\tau\|_2 \leq B_\tau} \langle \boldsymbol{w}_\tau, \frac{2}{n_\tau} \sum_{i=1}^{n_\tau} \epsilon_i \phi_\tau(\boldsymbol{x}_i^\tau) \rangle_{\mathcal{H}_\tau} | (\mathbf{X}_\tau)] \tag{77}$$

Now we apply on each function $f_\tau$ the upper bound from Lemma 22 by Bartlett & Mendelson 2003

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \sum_{t=1}^{T} \frac{2B_\tau}{n_\tau} (Tr(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)))^{1/2} \tag{78}$$

$$\tag{79}$$

**Bounding the Rademacher Complexity for Continual Learning**

**Lemma 4** *Keeping the same notations and setting as Theorem 3, the Rademacher Complexity can be bounded as follows:*

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \sum_{\tau=1}^{T} \mathcal{O}\left( \sqrt{\frac{\tilde{\mathbf{y}}_\tau^T (\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau}{n_\tau}} \right), \tag{80}$$

*where $\mathcal{F}_T$ is the function class spanned by the model up to the task $\mathcal{T}_\tau$.*

$$\tag{81}$$

**Proof**

For all $T \in \mathbb{N}^\star$, given a sequence $\{B_\tau, \tau \in [T]\}$ we define $\mathcal{F}_T$ as follows :

$$\mathcal{F}_T = \{\boldsymbol{x} \to \sum_{\tau=1}^{T} f_\tau(\boldsymbol{x}), \quad f_\tau(\boldsymbol{x}) = \alpha_\tau^T \kappa_\tau(\boldsymbol{x}, \mathbf{X}_\tau) \quad \forall \tau \in [T], \|f_\tau\|_{\mathcal{H}_\tau} \leq B_\tau\} \tag{82}$$

where we set $B_\tau$ as :

$$B_\tau = \sqrt{(\boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau})^T(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}(\boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau})} \tag{83}$$

Following Lemma 2 for all $\tau \in [T]$ :

$$f_\tau^\star \in \mathcal{F}_T \tag{84}$$

which means that the function class $\mathcal{F}_t$ contains the Continual Learning model up to task $\mathcal{T}_\tau$.

We apply Lemma 3 in order to upper bound the Rademacher complexity :

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \sum_{\tau=1}^{T} \frac{2B_\tau}{n_\tau}(Tr(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)))^{1/2} \tag{85}$$

We made the assumption that for all $\tau \in [T]$ $\mathrm{tr}(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)) = \mathcal{O}(n_\tau)$ :

$$\hat{\mathcal{R}}(\mathcal{F}_T) \leq \sum_{\tau=1}^{T} \frac{2B_\tau}{n_\tau}\mathcal{O}(\sqrt{n_\tau}) \tag{86}$$

$$\leq \sum_{\tau=1}^{T} \mathcal{O}(\frac{B_\tau}{\sqrt{n_\tau}}) \tag{87}$$

$$\leq \sum_{\tau=1}^{T} \mathcal{O}\left(\sqrt{\frac{(\boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau})^T(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}(\boldsymbol{y}_\tau - \boldsymbol{y}_{\tau-1\to\tau})}{n_\tau}}\right) \tag{88}$$

$$\tag{89}$$

#### D.2.4 Bounding the empirical loss for OGD

**Lemma 5** *Given a current task $\mathcal{T}_T$, the empirical losses on the data from all previous tasks ($\mathcal{T}_\tau, \tau \leq T$) can be bounded as follows :*

*Let $T \in \mathbb{N}$ fixed. Then, for all $\tau \in [T]$*

$$\mathcal{L}_{S_\tau}(f_T^\star) \leq \frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}_\tau(k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau \tag{90}$$

$$\tag{91}$$

**Case 1 -Task $\mathcal{T}_T$ Proof**

We start from the definition of the empirical loss :

$$\mathcal{L}_{S_T}(f_T^\star) = \frac{1}{n_T}\sum_{i=1}^{n_T}(f_T^\star(\boldsymbol{x}_{T,i}) - y_{T,i})^2 \tag{92}$$

$$= \frac{1}{n_T}\|f_T^\star(\mathbf{X}_T) - \boldsymbol{y}_T\|_2^2 \tag{93}$$

We replace into $f_T^\star$ by its expression from Theorem 1 :

$$= \frac{1}{n_T}\left\|\tilde{f}_T^\star(\mathbf{X}_T) + f_{T-1}^\star(\mathbf{X}_T) - \boldsymbol{y}_T\right\|_2^2 \tag{94}$$

$$= \frac{1}{n_T}\left\|f_{T-1}^\star(\mathbf{X}_T) - \tilde{\mathbf{y}}_T\right\|_2^2 \tag{95}$$

$$= \frac{1}{n_T}\left\|(k_T(\mathbf{X}_T, \mathbf{X}_T)^T(k_T(\mathbf{X}_T, \mathbf{X}_T) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T - \tilde{\mathbf{y}}_T\right\|_2^2 \tag{96}$$

$$= \frac{1}{n_T}\left\|(k_T(\mathbf{X}_T, \mathbf{X}_T) + \lambda\mathbf{I} - \lambda\mathbf{I})(k_T(\mathbf{X}_T, \mathbf{X}_T) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T - \tilde{\mathbf{y}}_T\right\|_2^2 \tag{97}$$

$$= \frac{1}{n_T}\left\|\tilde{\mathbf{y}}_T - \lambda(k_T(\mathbf{X}_T, \mathbf{X}_T) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T - \tilde{\mathbf{y}}_T\right\|_2^2 \tag{98}$$

$$= \frac{\lambda^2}{n_T}\left\|(k_T(\mathbf{X}_T, \mathbf{X}_T) + \lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T\right\|_2^2 \tag{99}$$

Now, we apply Lemma 2 in order to upper bound the right hand side norm :

$$\mathcal{L}_{S_T}(f_T^\star) \leq \frac{\lambda^2}{n_T}\tilde{\mathbf{y}}_T(k_T(\mathbf{X}_T,\mathbf{X}_T)+\lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T \tag{100}$$

$$\tag{101}$$

**Case 2 - Tasks $\{\mathcal{T}_\tau, \tau \in [1, T-1]\}$ :** The proof is very similar to Case 1, we apply Theorem 2, which is the key property of OGD in terms of no forgetting.

**Proof**

We start from the definition of the empirical loss :

$$\mathcal{L}_{S_\tau}(f_T^\star) = \frac{1}{n_\tau}\sum_{i=1}^{n_\tau}(f_T^\star(\boldsymbol{x}_{\tau,i}) - y_{\tau,i})^2 \tag{102}$$

$$= \frac{1}{n_\tau}\|f_T^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 \tag{103}$$

Now, applying Theorem 2 which implies that $f_\tau^\star(\mathbf{X}_\tau) = f_T^\star(\mathbf{X}_\tau)$ :

$$\mathcal{L}_{S_\tau}(f_T^\star) = \frac{1}{n_\tau}\|f_\tau^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 \tag{104}$$

Then with a similar analysis as Case 1, we get :

$$\mathcal{L}_{S_\tau}(f_T^\star) \leq \frac{\lambda^2}{n_\tau}\tilde{\mathbf{y}}_\tau(k_T(\mathbf{X}_\tau,\mathbf{X}_\tau)+\lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_\tau \tag{105}$$

$$\tag{106}$$

#### D.2.5 Bounding the empirical loss for SGD

As opposed to the analysis for OGD, forgetting can occur for OGD and Theorem 2 is no longer valid. Similarly to the OGD empirical loss analysis, we study the case on the data from the last task and the case on the data from all the other tasks.

**Lemma 6** *The empirical losses on the source and target tasks can be bounded as follows :*

*Let $T \in \mathbb{N}$ fixed. Then, for all $\tau \in [T]$*

$$\mathcal{L}_{S_T}(f_T^\star) \leq \frac{\lambda^2}{n_T}\tilde{\mathbf{y}}_T(k_T(\mathbf{X}_T,\mathbf{X}_T)+\lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_T \tag{107}$$

*For all $\tau \in [T-1]$*

$$\mathcal{L}_{S_\tau}(f_T^\star) \leq \frac{1}{n_\tau}(\lambda^2\tilde{\mathbf{y}}_\tau^T(k_\tau(\mathbf{X}_\tau,\mathbf{X}_\tau))^{-1}\tilde{\mathbf{y}}_\tau \tag{108}$$

$$+\sum_{k=\tau+1}^{T}\tilde{\mathbf{y}}_k^T(\kappa_k(\mathbf{X}_k,\mathbf{X}_k)+\lambda\mathbf{I})^{-1}\kappa_k(\mathbf{X}_\tau,\mathbf{X}_k)\kappa_k(\mathbf{X}_\tau,\mathbf{X}_k)^T(\kappa_k(\mathbf{X}_k,\mathbf{X}_k)+\lambda\mathbf{I})^{-1}\tilde{\mathbf{y}}_k) \tag{109}$$

**Case 1 -Task $\mathcal{T}_T$** For this case, the analysis is the same as for OGD, as no forgetting occurs with respect to the data the model is being trained on.

**Case 2 - Tasks $\{\mathcal{T}_\tau, \tau \in [1, T-1]\}$ :** Since SGD is not guaranteed to be robust to Catastrophic Forgetting, this case comprises additional residual terms that correspond to forgetting.

**Proof**

Let $\tau \in [T-1]$.

From Corr. 1, we recall that :

$$f_T^\star(\boldsymbol{x}) = f_\tau^\star(\boldsymbol{x}) + \sum_{k=\tau+1}^{T} \tilde{f}_k^\star(\boldsymbol{x}) \tag{110}$$

Then :

$$\|f_T^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 = \left\| f_\tau^\star(\mathbf{X}_\tau) + \sum_{k=\tau+1}^{T} \tilde{f}_k^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau \right\|_2^2 \tag{111}$$

$$\leq \|f_\tau^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 + \left\| \sum_{k=\tau+1}^{T} \tilde{f}_k^\star(\mathbf{X}_\tau) \right\|_2^2 \tag{112}$$

$$\leq \underbrace{\|f_\tau^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2}_{\text{A}} + \underbrace{\sum_{k=\tau+1}^{T} \left\| \tilde{f}_k^\star(\mathbf{X}_\tau) \right\|_2^2}_{\text{B}} \tag{113}$$

We can upper bound A similarly to the previous paragraphs :

$$\|f_\tau^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 \leq \lambda^2 \tilde{\mathbf{y}}_\tau^T (k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_\tau \tag{114}$$

Now, we upper bound B. Let $k \in [\tau+1, T]$ :

$$\left\| \tilde{f}_k^\star(\mathbf{X}_\tau) \right\|_2^2 = \tilde{\mathbf{y}}_k^T (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \underbrace{\kappa_k(\mathbf{X}_\tau, \mathbf{X}_k) \kappa_k(\mathbf{X}_\tau, \mathbf{X}_k)^T}_{\text{Similarity between the tasks } \mathcal{T}_\tau \text{ and } \mathcal{T}_k} (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_k \tag{115}$$

We conclude by plugging back the upper bounds of A and B

$$\|f_T^\star(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\|_2^2 \leq \lambda \tilde{\mathbf{y}}_\tau^T (k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))^{-1} \tilde{\mathbf{y}}_\tau \tag{116}$$

$$+ \sum_{k=\tau+1}^{T} \tilde{\mathbf{y}}_k^T (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \kappa_k(\mathbf{X}_\tau, \mathbf{X}_k) \kappa_k(\mathbf{X}_\tau, \mathbf{X}_k)^T (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_k \tag{117}$$

Therefore :

$$\mathcal{L}_{S_\tau}(f_T^\star) \leq \frac{1}{n_\tau} (\lambda^2 \tilde{\mathbf{y}}_\tau^T (k_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))^{-1} \tilde{\mathbf{y}}_\tau \tag{118}$$

$$+ \sum_{k=\tau+1}^{T} \tilde{\mathbf{y}}_k^T (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \kappa_k(\mathbf{X}_\tau, \mathbf{X}_k) \kappa_k(\mathbf{X}_\tau, \mathbf{X}_k)^T (\kappa_k(\mathbf{X}_k, \mathbf{X}_k) + \lambda \mathbf{I})^{-1} \tilde{\mathbf{y}}_k) \tag{119}$$

#### D.2.6 Proof of the Generalisation Theorem (Thm. 3)

Now, we prove the Theorem 3 by applying the lemmas we developed above.

**Proof**

With probability $1-\delta$ we have :

$$\sup_{f\in\mathcal{F}_T}\{L_D(f)-L_S(f)\}\leq 2\rho\hat{\mathcal{R}}(\mathcal{F}_T)+3c\sqrt{\frac{\log(2/\delta)}{2n_T}} \tag{120}$$

$$\mathcal{L}_{D_\tau}(f_T^\star)\leq\mathcal{L}_{S_\tau}(f_T^\star)+2\rho\hat{\mathcal{R}}(\mathcal{F}_T)+3c\sqrt{\frac{\log(2/\delta)}{2n_T}} \tag{121}$$

$$\mathcal{L}_{D_\tau}(f_T^\star)\leq\mathcal{L}_{S_\tau}(f_T^\star)+\sum_{k=1}^{T}\mathcal{O}\left(\sqrt{\frac{\tilde{\mathbf{y}}_k^T(\kappa_k(\mathbf{X}_k,\mathbf{X}_k))^{-1}\tilde{\mathbf{y}}_k}{n_k}}\right)+3c\sqrt{\frac{\log(2/\delta)}{2n_T}} \tag{122}$$

Then, by replacing into $\mathcal{L}_{S_\tau}(f_T^\star)$ with Lemmas 5 and 6, we get the upper bounds of the theorem for the various cases of SGD and OGD.

$$\tag{123}$$

#### D.2.7 Alternative proof for the empirical loss bounds - No regularisation case

This section is complementary and aims to provide a better interpretation of the bounds in Theorem 3.

We prove that in the case where there is no regularisation, the training error converges to zero. This result illustrates better the intuition behind the $\lambda$ term that appears in the upper bound of the Theorem 3, which is under the regularisation assumption.

First, we derive the differential equation of the model's output dynamics in Lemma 7. Then, we apply this Lemma to prove the convergence to zero in Lemma 8. Finally, in Lemma 9, we show that for the past tasks, the training error equals to zero all the time.

**The training dynamics of OGD**

**Lemma 7 (Differential equation of the model's output dynamics)** *Let $T\in\mathbb{N}^\star$, the index of the task currently being trained on, with OGD.*

*For all $\tau<T$, defining $\widetilde{\mathbf{u}}_\tau$ as :*

$$\widetilde{\mathbf{u}}_\tau(t)=f_\tau^{(t)}(\mathbf{X}_\tau)-\sum_{k<\tau}\tilde{f}_k^\star(\mathbf{X}_\tau)=\phi_\tau(\mathbf{X}_\tau)^T(\boldsymbol{w}_\tau(t)-\boldsymbol{w}_{\tau-1}^\star) \tag{124}$$

*The model's output dynamics while training on the task $\mathcal{T}_\tau$ is as follows :*

$$\widetilde{\mathbf{u}}_\tau(t+1)-\widetilde{\mathbf{u}}_\tau(t)=-\eta\kappa_\tau(\mathbf{X}_\tau,\mathbf{X}_\tau)(\widetilde{\mathbf{u}}_\tau(t)-\tilde{\mathbf{y}}_\tau). \tag{125}$$

$$\tag{126}$$

**Proof**

Let $\tau\in\mathbb{N}^\star$, we derive the dynamics while training on the task $\mathcal{T}_\tau$, while training on this task.

First, we define :

$$\tilde{\tilde{\boldsymbol{y}}}_\tau=\tilde{\mathbf{y}}_\tau+\phi_\tau(\mathbf{X}_\tau)^T\boldsymbol{w}_{\tau-1}^\star \tag{127}$$

We also define the projection matrix $\mathbf{P}_\tau\in\mathbb{R}^{p\times p}$ as :

$$\mathbf{P}_\tau=\mathbf{T}_\tau^T\mathbf{T}_\tau \tag{128}$$

The matrix $\mathbf{P}_\tau$ performs the projection from the original weight space $\mathbb{R}^p$ to the trainable weight space during training on task $\mathcal{T}_\tau$, which corresponds to $(\bigoplus_{k=1}^{\tau}\mathbb{E}_k)^\perp$ .

Our starting point is the SGD update rule :

$$\boldsymbol{w}_\tau(t+1) = \boldsymbol{w}_\tau(t) - \eta \mathbf{P}_{\tau+1} \nabla_{\boldsymbol{w}} \mathcal{L}^\tau(\boldsymbol{w}_\tau(t)) \tag{129}$$

where :

$$\mathcal{L}^\tau(\boldsymbol{w}_\tau(t)) = \left\|\phi_\tau(\mathbf{X}_\tau)^T(\boldsymbol{w}_\tau(t) - \boldsymbol{w}^\star_{\tau-1}) - \tilde{\mathbf{y}}_\tau\right\|_2^2 \tag{130}$$

$$= \left\|\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - (\tilde{\mathbf{y}}_\tau + \phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}^\star_{\tau-1})\right\|_2^2 \tag{131}$$

$$= \left\|\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - \tilde{\tilde{\boldsymbol{y}}}_\tau\right\|_2^2 \tag{132}$$

Therefore, we get the following expression for the gradient of the loss :

$$\nabla_{\boldsymbol{w}} \mathcal{L}^\tau(\boldsymbol{w}_\tau(t)) = \phi_\tau(\mathbf{X}_\tau)(\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - \tilde{\tilde{\boldsymbol{y}}}_\tau) \tag{133}$$

Then, the following holds :

$$\tag{134}$$

$$\widetilde{\mathbf{u}}_\tau(t+1) - \widetilde{\mathbf{u}}_\tau(t) = \phi_\tau(\mathbf{X}_\tau)^T(\boldsymbol{w}_\tau(t+1) - \boldsymbol{w}_\tau(t)) \tag{135}$$

$$= \phi_\tau(\mathbf{X}_\tau)^T(-\eta \mathbf{P}_\tau \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}_\tau(t))) \tag{136}$$

$$= \phi_\tau(\mathbf{X}_\tau)^T(-\eta \mathbf{P}_\tau \phi_\tau(\mathbf{X}_\tau)(\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - \tilde{\tilde{\boldsymbol{y}}}_\tau)) \tag{137}$$

$$= (\phi_\tau(\mathbf{X}_\tau)\mathbf{T}_\tau)^T(-\eta \mathbf{T}_\tau \phi_\tau(\mathbf{X}_\tau)(\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - \tilde{\tilde{\boldsymbol{y}}}_\tau)) \tag{138}$$

$$= -\eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)(\phi_\tau(\mathbf{X}_\tau)^T \boldsymbol{w}_\tau(t) - \tilde{\tilde{\boldsymbol{y}}}_\tau) \tag{139}$$

$$= -\eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)(\widetilde{\mathbf{u}}_\tau(t) - \tilde{\mathbf{y}}_\tau) \tag{140}$$

$$= -\eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)(\widetilde{\mathbf{u}}_\tau(t) - \tilde{\mathbf{y}}_\tau) \tag{141}$$

It follows that :

$$\widetilde{\mathbf{u}}_\tau(t+1) - \tilde{\mathbf{y}}_\tau = (\mathbf{I} - \eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))(\widetilde{\mathbf{u}}_\tau(t) - \tilde{\mathbf{y}}_\tau) \tag{142}$$

$$\widetilde{\mathbf{u}}_\tau(t) - \tilde{\mathbf{y}}_\tau = -(\mathbf{I} - \eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))^t(\widetilde{\mathbf{u}}_\tau(0) - \tilde{\mathbf{y}}_\tau) \tag{143}$$

$$\tag{144}$$

**The training error converges to zero**

**Lemma 8** *For all tasks $\mathcal{T}_\tau$, $\tau \in [T]$, the training error $\mathcal{L}^\tau(\boldsymbol{w}_\tau(t))$ converges to $0$ when $t$ tends to infinity .*

**Proof**

We start with the definition of the training error on task T :

$$\mathcal{L}_S(f_\tau^{(t)}) = \left\|f_\tau^{(t)}(\mathbf{X}_\tau) - \boldsymbol{y}_\tau\right\|^2 \tag{145}$$

$$= \|\widetilde{\mathbf{u}}_\tau(t) - \tilde{\mathbf{y}}_\tau\|^2 \tag{146}$$

$$= \left\|(I - \eta \kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))^t(\widetilde{\mathbf{u}}_\tau(0) - \tilde{\mathbf{y}}_\tau)\right\| \tag{147}$$

Under the assumption that $\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau)$ is positive definite, for $\eta \leq \frac{1}{\|(\kappa_\tau(\mathbf{X}_\tau, \mathbf{X}_\tau))\|}$ the eigenvalues become less than 1, therefore :

$$\lim_{t \to \infty} \mathcal{L}_S(f_\tau^{(t)}) = 0 \tag{148}$$

**The training error on the past tasks is zero**

**Lemma 9** *For all tasks $\mathcal{T}_\tau$, $\tau \in [T]$, the training error $\mathcal{L}_{S_\tau}(f_T^\star) = 0$.*

**Proof** This lemma follow immediately from Lemma 8 which states that the training error converges to zero for OGD on all tasks and Theorem 2 which states that the training error on the past tasks is unchanged.

# E MISSING PROOFS OF APP A - THE IMPORTANCE OF THE NTK VARIATION FOR CONTINUAL LEARNING

## E.1 PROOF OF PROPOSITION 1 - OGD IMPLIES A-GEM-NT

We present the proof of Proposition 1, which relies mainly on Theorem 2 stating the robustness of OGD to Catastrophic Forgetting.

**Proof**

Let $\mathcal{T}_\tau$ a task fixed, we recall that in the NTK regime, the model can be expressed as :

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = f^\star_{\tau-1}(\boldsymbol{x}) + \nabla f^\star_0(\boldsymbol{x})(\boldsymbol{w} - \boldsymbol{w}^\star_{\tau-1}) \tag{149}$$

Given a task $\mathcal{T}_k$ and its associated memory $\mathcal{M}_k$, we recall that the loss can be expressed as :

$$l(f_{\boldsymbol{w}}, \mathcal{M}_k) = \sum_{(\boldsymbol{x},y)\in\mathcal{M}_k} (f^\star_{\tau-1}(\boldsymbol{x}) + \nabla f^\star_0(\boldsymbol{x})(\boldsymbol{w} - \boldsymbol{w}^\star_{\tau-1}) - y)^2 \tag{150}$$

Similarly to the A-GEM paper, we define the gradients vectors $\boldsymbol{g}$ and $\{\boldsymbol{g}_k, k \in [\tau-1]\}$ as :

$$\boldsymbol{g} = \nabla_{\boldsymbol{w}} l(f_{\boldsymbol{w}}, D_\tau) \tag{151}$$

$$\boldsymbol{g}_k = \nabla_{\boldsymbol{w}} l(f_{\boldsymbol{w}}, \mathcal{M}_k) \tag{152}$$

Now we extend the expressions :

$$\boldsymbol{g}_k = \nabla_{\boldsymbol{w}} l(f_{\boldsymbol{w}}, D_\tau) \tag{153}$$

$$= \nabla_{\boldsymbol{w}} \sum_{(\boldsymbol{x},y)\in\mathcal{M}_k} (f^\star_{\tau-1}(\boldsymbol{x}) + \nabla f^\star_0(\boldsymbol{x})(\boldsymbol{w} - \boldsymbol{w}^\star_{\tau-1}) - y)^2 \tag{154}$$

$$= \sum_{(\boldsymbol{x},y)\in\mathcal{M}_k} (f^\star_{\tau-1}(\boldsymbol{x}) + \nabla f^\star_0(\boldsymbol{x})(\boldsymbol{w} - \boldsymbol{w}^\star_{\tau-1}) - y)\nabla f^\star_0(\boldsymbol{x}) \tag{155}$$

For OGD, following Thm. 2 variant for finite memory :

$$= \sum_{(\boldsymbol{x},y)\in\mathcal{M}_k} (\nabla f^\star_0(\boldsymbol{x})(\boldsymbol{w} - \boldsymbol{w}^\star_{\tau-1}))\nabla f^\star_0(\boldsymbol{x}) \tag{156}$$

Finally, since the gradient updates are orthogonal to $\mathcal{M}_k$ :

$$\boldsymbol{g}_k = 0 \tag{157}$$

Therefore, for all $k \in [\tau-1]$ :

$$\boldsymbol{g}_k \cdot \boldsymbol{g} = 0 \tag{158}$$

We conclude, OGD implies learning with GEM and A-GEM with no positive backward transfer.

$$\tag{159}$$

# F EXPERIMENTS - REPRODUCIBILITY

## F.1 REPRODUCIBILITY

### F.1.1 HYPERPARAMETERS

We use the same architecture and the same hyperparameters as Farajtabar et al. (2019). We also keep a small learning rate, in order to preserve the locality assumption of OGD, and in order to verify the conditions of the theorems.

For the MNIST benchmarks, the neural network is a three-layer MLP with 100 hidden units in two layers, each layer uses RELU activation function. The model has either 2 logits or 10 logit outputs, which do not use any activation function. For the Split CIFAR-100 benchmark, the neural network is a multi-head LeNet Lecun et al. (1998) network with Batch Normalisation Ioffe & Szegedy (2015) and 200 hidden units for the penultimate layer. The optimiser is either SGD, OGD or OGD+ and the loss is Softmax cross-entropy. We report the hyperparameters in detail in Table 4.

| Hyperparameter | MNIST | CIFAR-100 |
|---|---|---|
| Epochs | 5 | 50 |
| Architecture | MLP | LeNet |
| Hidden dimension | 100 | 200 |
| A-GEM $\boldsymbol{g}_{\text{ref}}$ batch size | 256 | 1300 |
| Learning rate | 1e-03 | |
| Batch size | 32 | |
| Torch seeds | 1 to 5 | |
| Memory size per task | 100 | |
| Activation | RELU | |

Table 4: Hyperparameters used across experiments

### F.1.2 EXPERIMENT SETUP

We run each experiment 5 times, the seeds set is the same across all experiments sets. We report the mean and standard deviation of the measurements. The test error is measured every 250 mini-batch interval.

### F.1.3 OGD+ IMPLEMENTATION DETAILS

**Memory :** During the OGD+ memory update step, for each task, the new associated feature maps replace the memory slots of the previous feature maps for the same task. The goal is to ensure a balance of the feature maps from all tasks in the memory.

**Multi-head models :** For the dataset streams Split MNIST and Split CIFAR-100, we consider multi-headed neural networks. We only store the feature maps with respect to the shared weights, the projection step is not performed for the heads' weights.

### F.2 Complementary results - Overview

In this section, we present extended results for the experiments presented in the main article (Sec. 7).

**Memorisation property of OGD (Thm. 2)** We study the two main assumptions of Thm. 2 :

- overparameterisation in Sec. F.3.1 : we find that overparameterisation decreases Catastrophic Forgetting for OGD.
- infinite memory in Sec. F.3.2 : we find that increasing the memory size also descreases Catastrophic Forgetting for OGD.

**The importance of the NTK variation for Continual Learning** We track the test accuracy through tasks for the algorithms SGD, OGD, OGD+ and A-GEM, on the benchmarks Split MNIST, Rotated MNIST and Split CIFAR-100. (Sec. F.4)

We run the experiments on A-GEM, for completion since A-GEM is a state-of-the-art algorithm for Continual Learning, and as a supporting material for App. A.

### F.3 Experiments : Memorisation property of OGD (Thm. 2)

#### F.3.1 The importance of the overparameterisation assumption

Thm. 2 states that in the NTK regime, given an infinite memory, the train error of OGD is unchanged.

**Experiments** We track the variation of the **train** accuracy of the samples in the memory through tasks after being trained on all subsequent tasks. We consider the MLP hidden layer size as a proxy for overparameterisation. We run the experiments on the Permuted MNIST and Rotated MNIST benchmarks, with the OGD algorithm. We vary the hidden size from 100 to 400.

**Results** Figure 3 shows that the variation of the train accuracy of OGD+ decreases uniformly with the hidden size. This result concurs with the hypothesis that the train error decreases with overparameterisation.

#### F.3.2 The importance of the memory size assumption

Thm. 2 states that in the NTK regime, given an infinite memory, the train error of OGD is unchanged.

**Experiments** We track the variation of the **train** accuracy of the samples in the memory through tasks after being trained on all subsequent tasks, as a function of the memory size per task.

**Results** Figure 4 shows that the *mean* train accuracy variation decreases uniformly with the memory size. The difference is significant the most on the oldest tasks, which were exposed to Catastrophic Forgetting the most.

### F.4 The importance of the NTK variation for Continual Learning (Sec. 6)

In this section, we present extended results on the importance of the NTK variation for Continual Learning. For completion, we present the variation of the test accuracy of the model at the end of training for all tasks. Also, we run the benchmarks on the A-GEM algorithm, for the reference, and as a supporting materials for the extended discussion in App. A.

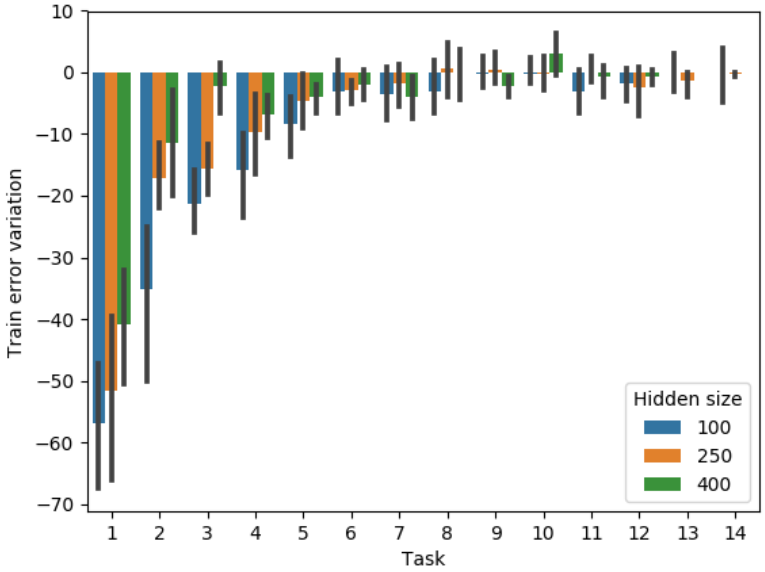



Figure 3: The variation of the train accuracy on the memorised samples from the first task, after the model was trained on all tasks in sequence (higher is better). We vary the hidden size as a proxy for overparameterisation.

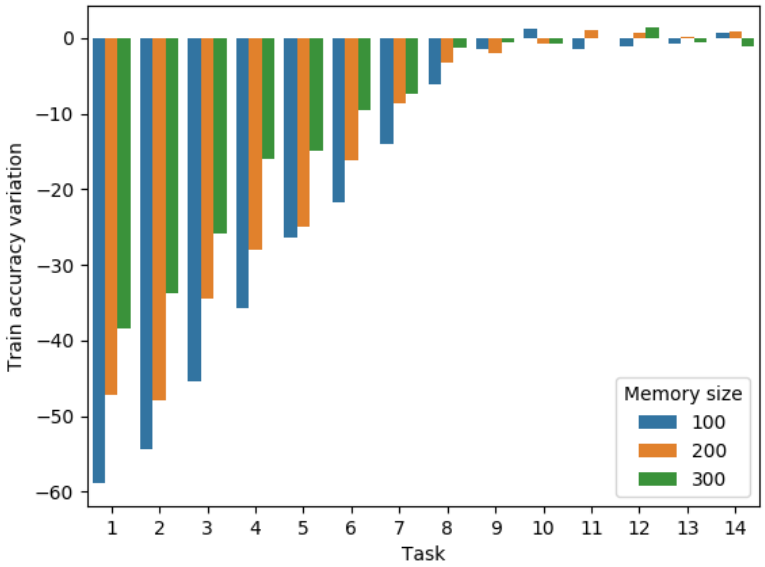



Figure 4: The variation of the train accuracy on the memorised samples from the first task, after the model was trained on all tasks in sequence (higher is better). We vary the memory size per task from 100 to 300.

#### F.4.1 PERMUTED MNIST

| | Accuracy ± Std. (%) | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Task 1** | **Task 2** | **Task 3** | **Task 4** | **Task 5** | **Task 6** | **Task 7** |
| AGEM | **76.0±1.6** | **78.4±1.7** | **80.3±1.4** | **82.0±1.6** | 82.1±1.5 | **84.6±0.9** | 86.2±0.9 |
| OGD+ | 72.8±1.6 | 75.2±2.9 | 78.0±1.8 | 79.8±1.4 | 81.4±1.4 | 84.3±1.8 | 85.4±1.9 |
| OGD | 33.1±9.2 | 62.8±1.6 | 75.5±2.8 | 77.1±4.4 | **82.2±1.4** | 83.5±1.9 | **86.5±1.3** |
| SGD | 34.8±2.4 | 41.5±2.6 | 42.4±5.6 | 48.1±8.9 | 56.9±7.9 | 58.7±5.8 | 66.6±6.5 |

| | Accuracy ± Std. (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Task 8** | **Task 9** | **Task 10** | **Task 11** | **Task 12** | **Task 13** | **Task 14** | **Task 15** |
| AGEM | 87.1±1.1 | **89.1±0.9** | 89.0±1.1 | 90.3±0.7 | 91.5±1.0 | 92.5±0.8 | 93.6±0.4 | **94.7±0.2** |
| OGD+ | **87.3±1.1** | 88.4±1.2 | 90.2±0.7 | 91.5±0.8 | 92.4±0.5 | **93.3±0.4** | **94.0±0.3** | 94.5±0.1 |
| OGD | 86.8±0.5 | 88.4±1.3 | **90.2±0.9** | **91.6±0.4** | **92.7±0.2** | 93.2±0.3 | 94.0±0.1 | 94.4±0.1 |
| SGD | 79.0±2.3 | 72.8±10.1 | 80.8±5.1 | 86.1±1.8 | 90.5±0.7 | 92.1±1.0 | 93.3±0.4 | 94.5±0.1 |

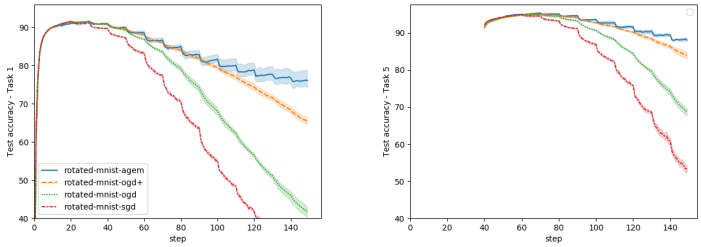
Table 5: Permuted MNIST : The test accuracy of models from the indicated task after being trained on all tasks in sequence. The best continual learning results are highlighted in **bold**.

#### F.4.2 ROTATED MNIST

| | Accuracy ± Std. (%) | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Task 1** | **Task 2** | **Task 3** | **Task 4** | **Task 5** | **Task 6** | **Task 7** |
| AGEM | **76.1±2.3** | **78.3±2.1** | **83.6±1.1** | **86.1±0.8** | **88.1±0.5** | **89.5±0.1** | **91.0±0.3** |
| OGD+ | 65.4±1.2 | 67.8±1.1 | 75.6±1.1 | 80.1±1.5 | 83.9±1.1 | 86.3±1.2 | 88.7±0.8 |
| OGD | 41.7±1.6 | 44.7±1.4 | 53.2±1.7 | 61.4±1.4 | 68.7±1.2 | 74.8±1.2 | 81.3±0.8 |
| SGD | 28.6±1.3 | 31.4±0.6 | 37.6±1.0 | 45.0±1.0 | 53.3±1.1 | 60.3±1.1 | 68.9±0.9 |

| | Accuracy ± Std. (%) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | **Task 8** | **Task 9** | **Task 10** | **Task 11** | **Task 12** | **Task 13** | **Task 14** | **Task 15** |
| AGEM | **91.6±0.2** | **93.0±0.2** | **94.0±0.2** | **95.1±0.2** | **95.9±0.0** | **96.4±0.1** | 96.3±0.1 | 96.0±0.1 |
| OGD+ | 90.3±0.7 | 92.3±0.4 | 93.9±0.2 | 94.9±0.1 | 95.9±0.1 | 96.1±0.1 | 96.1±0.1 | 95.6±0.1 |
| OGD | 86.0±0.7 | 89.7±0.5 | 92.5±0.2 | 94.4±0.1 | 95.7±0.1 | 96.1±0.2 | 96.2±0.2 | 95.8±0.2 |
| SGD | 76.8±1.0 | 83.4±0.5 | 88.3±0.4 | 92.7±0.1 | 94.9±0.1 | 96.1±0.2 | **96.5±0.1** | **96.2±0.1** |

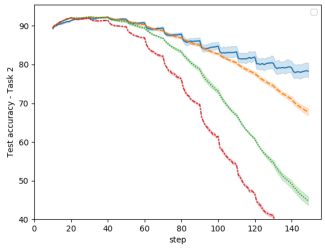
Table 6: Rotated MNIST : The test accuracy of models from the indicated task after being trained on all tasks in sequence. The best continual learning results are highlighted in **bold**.



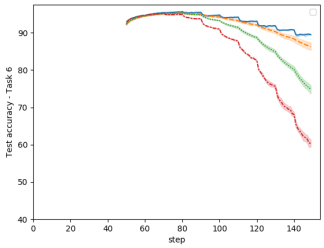

(a) Task 1



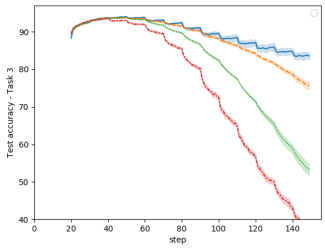

(b) Task 5



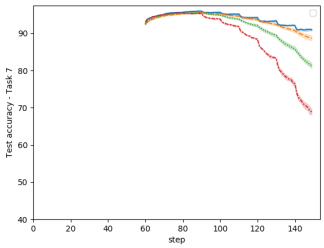

(c) Task 2



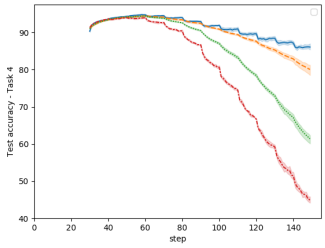

(d) Task 6



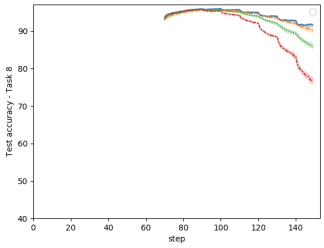

(e) Task 3



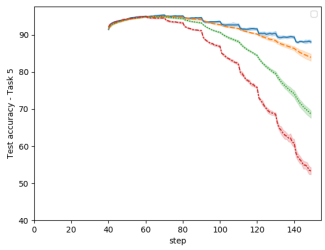

(f) Task 7



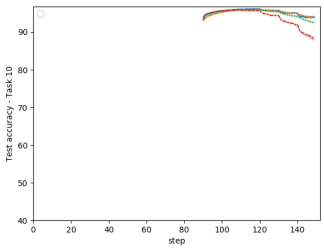

(g) Task 4




(h) Task 8




(i) Task 5




(j) Task 10

Figure 5: Test accuracy on the 10 first tasks of Rotated MNIST, for SGD, OGD, OGD+ and A-GEM. The y-axis is truncated for clarity. We report the mean and standard deviation over 5 independent runs. The test error is measured for every 250 mini-batch interval.