Multi-task Linear Bandits

Marta Soare *
INRIA Lille - Nord Europe, Team SequeL

Ouais Alsharif †
McGill University & Google

Alessandro Lazaric *
INRIA Lille - Nord Europe, Team SequeL

Joelle Pineau ‡
McGill University

1 Introduction

For a learning algorithm to be able to learn from only a few samples, it is necessary to exploit some structural bias in the environment of the task at hand. Such bias has been defined and included in the learning algorithms in many different ways, from Bayesian priors to regularization (e.g., smoothness, sparsity). One particularly successful way to define such bias is to learn it directly from multiple different, yet related tasks. While this approach, known as multi-task or transfer learning (see [7] for a survey), has made significant gains in supervised learning scenarios (see e.g., [3]), it received less attention for sequential decision making problems with limited feedback. In a recent work [5], it is shown that sequential transfer may indeed have a positive impact also in the multi-armed bandit problem with a significant reduction in the regret. In multi-arm bandit, the goal is to learn the reward (e.g., a click through rate or a star ranking) of different arms (e.g., items) and to sequentially select the arms that lead to the highest reward.

In this paper we propose an alternative multi-task approach and we move the focus from the multi-armed bandit scenario to the *linear bandit* setting. In the particular setting of linear bandit, each arm is characterized by a feature vector and the reward function is assumed to be a linear combination of the feature vector with an unknown parameter vector θ (e.g., a vector characterizing the preference of a user). Solving this problem requires to find a suitable balance between choosing arms that can contribute to better learn the parameter vector (e.g., learn user's preference) and selecting arms which are more rewarding. The introduction of multi-task learning in this particular setting would allow to exploit the fact that if two users have similar parameter vectors, then knowledge of one's interactions can be exploited to minimize "regret" of a learning algorithm while interacting with the other. In this paper, we focus on this setting and we explore how multi-task learning can contribute to boosting the performance of linear bandit algorithms.

2 Preliminaries

Multi-task linear bandits. We formalize the multi-task learning problem in the linear stochastic bandit setting. The learner is given a finite set of arms $\mathcal{X} \subset \mathbb{R}^d$ with $||x||_2 \leq L$ for any $x \in \mathcal{X}$. We assume that the learner has to solve a sequence of tasks. At each task j in the sequence, the learner is given a limited sampling budget n_j and it can sequentially choose n_j arms $x \in \mathcal{X}$. At any time step $s \leq n_j$, after an arm $x_{j,s} \in \mathcal{X}$ is chosen, the learner observes a noisy realization $r_{j,s}$ corresponding to the selected arm. In particular, here we assume that the reward is given by a linear function, characterized by an unknown parameter $\theta_j \in \mathbb{R}^d$, where $||\theta_j||_2 \leq S$, $\forall j$. More precisely, the noisy realization (or *reward*), denoted $r_{j,s}$, is given by the following model:

$$r_{j,s} = x_{j,s}^\top \theta_j + \eta_{j,s}, \tag{1}$$

where η is a random R-sub-Gaussian noise. Typically at each task j, the learner's objective is to maximize the sum of expected rewards $\sum_{s=1}^{n_j} x_{j,s}^\top \theta_j$. Since the parameter θ_j is unknown, the learner faces the so-called *exploration-exploitation dilemma*, where the exploration of the arms improves

*{marta.soare,alessandro.lazaric}@inria.fr
†ouais.alsharif@gmail.com
‡jpineau@cs.mcgill.ca

the estimate of the unknown parameter θ_j, while the exploitation of the estimated best-arms would supposedly maximize the sum of rewards. The performance of the learner is evaluated with respect to the sum of rewards obtained by an oracle-algorithm, which knows the value of θ_j. Denoting $x_j^* = \arg\max_{x \in \mathcal{X}} x^\top \theta_j$ the best arm for task j, the *pseudo-regret* suffered by the learner is given by the difference between the two sums of rewards. More precisely, for a task j, we compute the pseudo-regret at the end of task j as follows:

$$R_{n_j} = \sum_{s=1}^{n_j} {x_j^*}^\top \theta_j - \sum_{s=1}^{n_j} x_s^\top \theta_j = \sum_{s=1}^{n_j} (x_j^* - x_s)^\top \theta_j. \tag{2}$$

In any multi-task/transfer scenario, it is crucial to define a suitable notion of *similarity* between tasks that can be exploited by a multi-task learning algorithm. Similarly to [4], we define the similarity between tasks with respect to the similarity between their corresponding parameters (or targets). Thus, whenever the characterizing vectors of two tasks are close to each other (in a ℓ_2-norm sense), we say the two tasks are similar.

Definition 1. *Let $\theta, \theta' \in \mathbb{R}^d$ be the parameters characterizing the linear functions corresponding to two different tasks. If $||\theta - \theta'||_2 \leq \varepsilon$, then tasks i and j are **similar**.*

Given the assumption that all the tasks considered by the learner are similar, the goal of this paper is to study how the overall performance (the sum of regrets for all tasks) can be improved if the task similarity is exploited by transferring the information collected from the rewards observed in the previous tasks. Before providing a multi-task version of the LinUCB algorithm, we introduce additional technical tools.

Tools. For a single-task j, at each time step $s = 1, \ldots, n_j$ the parameter θ_j is estimated using the available sample-rewards obtained during task j, through the regularized least-squares solution. Thus, the estimate of θ_j, after observing $t \leq n_j$ rewards is given by:

$$\widehat{\theta}_{j,t}^{\lambda} = (A_{j,t} + \lambda \mathbb{I})^{-1} b_{j,t} = (A_{j,t}^{\lambda})^{-1} b_{j,t}, \tag{3}$$

where $A_{j,t} = \sum_{s=1}^{t} x_{j,s} x_{j,s}^\top$ is the design matrix, λ is the regularization parameter ($A_{j,t}^{\lambda}$ is a shorthand for the regularized design-matrix), and $b_{j,t} = \sum_{s=1}^{t} x_{j,s} r_{j,s}$ is the sum of all observed rewards. For the typical linear-bandit (single-task) setting, the regret is given by [1, Theorem 2].

Preposition 1 (Theorem 2 in [1])**.** *Let $\widehat{\theta}_{j,t}$ be the single-task regularized least-squares estimate, for any $\delta \geq 0$, with probability at least $1 - \delta$ and , for any $t \geq 1$, it holds that:*

$$|x^\top \widehat{\theta}_{j,t}^{\lambda} - x^\top \theta_j| \leq ||x||_{\left(A_{j,t}^{\lambda}\right)^{-1}} \left(R \sqrt{2 \log \left(\frac{\det \left(A_{j,t}^{\lambda}\right)^{1/2} \det(\lambda \mathbb{I})^{-1/2}}{\delta} \right)} + \lambda^{1/2} S \right) = B_{j,t}(x). \tag{4}$$

3 Multi-task confidence bounds

We now extend the previous single-task result to the multi-task scenario where at task $m+1$ the learner uses all the rewards observed from the past tasks $(\theta_1, \ldots, \theta_j, \ldots, \theta_m)$. Using the similarity assumption between tasks, we define the multi-task estimate, $\widetilde{\theta}_{m+1,t}^{\lambda}$ computed as the "global" regularized least-squares solution. More precisely, suppose that we are at time step t in task $m+1$. By denoting A_j the design matrix for task j and b_j the vector of rewards observed during task j, we can compute the the multi-task estimate of θ_{m+1} as follows:

$$\begin{aligned}
\widetilde{\theta}_{m+1,t}^{\lambda} &= \big(\sum_{j=1}^{m} A_j + A_{m+1,t} + \lambda \mathbb{I} \big)^{-1} \big(\sum_{j=1}^{m} b_j + b_{m+1,t} \big) \\
&= \big(\sum_{j=1}^{m} \sum_{s=1}^{n_j} x_s x_s^\top + \sum_{s=1}^{t} x_{m+1,s} x_{m+1,s}^\top + \lambda \mathbb{I} \big)^{-1} \big(\sum_{j=1}^{m} \sum_{s=1}^{n_j} x_s y_{j,s} + \sum_{s=1}^{t} x_{m+1,s} y_{m+1,s}^\top \big) \\
&= (\widetilde{A}_{m+1,t} + \lambda \mathbb{I})^{-1} \cdot \widetilde{b}_{m+1,t} = (\widetilde{A}_{m+1,t}^{\lambda})^{-1} \cdot \widetilde{b}_{m+1,t}
\end{aligned} \tag{5}$$

where in the last step we introduce the notation $\widetilde{A}_{m+1,t}$ for the global design matrix (containing all observed rewards), its regularized version $\widetilde{A}_{m+1,t}^{\lambda}$, and $\widetilde{b}_{m+1,t}$ the vector of all observations, from all tasks.

Also, we introduce the "average" task implicitly defined in computing $\widetilde{\theta}^{\lambda}_{m+1,t}$ characterized by the multi-task parameter vector $\widetilde{\theta}^{*}_{m+1,t}$:

$$\mathbb{E}\big[\widetilde{\theta}_{m+1,t}\big] = \widetilde{\theta}^{*}_{m+1,t}. \tag{6}$$

In the following, we discuss the construction of confidence ellipsoids around the empirical multi-task estimate $\widetilde{\theta}^{\lambda}_{m+1,t}$, such that with high probability the true parameter vector of task $m+1$ is included in the confidence set at each time step. Intuitively, while one can expect the confidence sets to be much tighter compared to the corresponding bounds for the single-task estimate $\widehat{\theta}^{\lambda}_{m+1,t}$ thanks to the much larger number of samples transferred from past tasks, one must also consider the fact that these rewards come from tasks with θs different from θ_{m+1}, thus potentially introducing a bias.

We proceed by bounding the error in estimating the expected reward $x^\top\theta_{m+1}$ of any arm x. We decompose the error of the multi-task least-squares estimate (Eq. 5) into the estimation error due to the use of random rewards and the approximation error caused by the difference between the tasks:

$$\big|x^\top\widetilde{\theta}^{\lambda}_{m+1,t} - x^\top\theta_{m+1}\big| \le \underbrace{\big|x^\top\widetilde{\theta}^{\lambda}_{m+1,t} - x^\top\widetilde{\theta}^{*}_{m+1,t}\big|}_{①} + \underbrace{\big|x^\top\widetilde{\theta}^{*}_{m+1,t} - x^\top\theta_{m+1}\big|}_{②}.$$

① While the following is true for any $t \in [1, n_{m+1}]$, to simplify the notation, we suppose that we are at the end of task $m+1$. Thus, we use A_{m+1} instead of $A_{m+1,t}$ and t becomes n_{m+1}.

$$\begin{aligned}\widetilde{\theta}^{\lambda}_{m+1} &= \big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1}\Big(\sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_{j,s}y_{j,s}\Big) = \big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1}\Big(\sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_{j,s}(x_{j,s}^\top\theta_j + \eta_{j,s})\Big)\\ &= \big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1}\cdot\Big(\sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_{j,s}\eta_{j,s}\Big) + \widetilde{\theta}^{*}_{m+1} - \lambda\big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1}\cdot\widetilde{\theta}^{*}_{m+1}.\end{aligned}$$

Then, following the derivation detailed in Appendix A, we obtain that w.p. $\ge 1-\delta$ it holds that:

$$\big|x^\top\widetilde{\theta}^{\lambda}_{m+1,t} - x^\top\widetilde{\theta}^{*}_{m+1,t}\big| \le \big|\big|x\big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}}\left(R\sqrt{2\log\left(\frac{\det\big(\widetilde{A}^{\lambda}_{m+1}\big)^{1/2}\det(\lambda\mathbb{I})^{-1/2}}{\delta}\right)} + \lambda^{1/2}S\right).$$

Thus, assuming that the R-sub-Gaussianity condition holds for the noise in any task, we recover the same type of bound as in the single-task setting (Prop. 1). The only difference is that the single-task matrix A^{λ}_{j} is replaced here with the multi-task matrix $\widetilde{A}^{\lambda}_{m+1}$.

② We begin with a convenient rewriting of $\widetilde{\theta}^{*}_{m+1,t}$ (Eq. 6). Let $A_j = n_j\left(\frac{1}{n_j}\sum_{s=1}^{n_j} x_s x_s^\top\right) = n_j\cdot C_j$ and $N = \sum_{j=1}^{m} n_j + t$. Then we have $\widetilde{A}_{m+1,t} = N\big(\sum_{j=1}^{m}\frac{n_j}{N}C_j + \frac{t}{N}C_{m+1,t}\big) = N\widetilde{C}_{m+1,t}$ and

$$\widetilde{\theta}^{*}_{m+1,t} = \sum_{j=1}^{m}\frac{n_j}{N}\widetilde{C}^{-1}_{m+1,t}C_j\theta_j + \frac{t}{N}\widetilde{C}^{-1}_{m+1,t}C_{m+1,t}\theta_{m+1} = \widetilde{C}^{-1}_{m+1,t}\Big(\sum_{j=1}^{m}\frac{n_j}{N}C_j\theta_j + \frac{t}{N}C_{m+1,t}\theta_{m+1}\Big).$$

Then, using the steps described in Appendix B, it follows that

$$\big|\big|\widetilde{\theta}^{*}_{m+1,t} - \theta_{m+1}\big|\big| \le \big|\big|\sum_{j=1}^{m}\frac{n_j}{N}\widetilde{C}^{-1}_{m+1,t}C_j\big|\big|\cdot\big|\big|\theta_j - \theta_{m+1}\big|\big| \le \varepsilon.$$

This result shows that although the arms were chosen mostly according to the estimations of some parameters different from θ_{m+1}, the current notion of similarity ensures that the total approximation error is upper-bounded by ε. Keeping the same assumptions and notation as defined above, we can now use ① and ② to define the multi-task confidence bound.

Theorem 1. *Let $\widetilde{\theta}^{\lambda}_{m+1,t}$ be the multi-task regularized least-squares estimate defined in Eq. 5. Then, for any $\delta \ge 0$, for any $t \ge 1$, with probability at least $1-\delta$ it holds that:*

$$\big|x^\top(\widetilde{\theta}^{\lambda}_{m+1,t} - \theta_{m+1})\big| \le \tag{7}$$

$$\big|\big|x\big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1,t}\right)^{-1}}\left(R\sqrt{2\log\left(\frac{\det\big(\widetilde{A}^{\lambda}_{m+1,t}\big)^{1/2}\det(\lambda\mathbb{I})^{-1/2}}{\delta}\right)} + \lambda^{1/2}S\right) + x^\top\varepsilon = \widetilde{B}_{m+1,t}(x).$$

Relying on the multi-task confidence bound, we now introduce an algorithm that selects arms according to $\widetilde{B}_{m+1,t}$, whenever this is tighter than the corresponding single-task bound $B_{m+1,t}$.

4 Multi-task LinUCB Algorithm

Similarly to LINUCB[2, 6], MT-LINUCB relies on confidence bounds for the arms' values to select the arms to be pulled. The difference comes from the fact that here after each arm pull, both the single-task (Prop.1) and the multi-task (Th.1) bounds are updated and the arm selected at the next time step will depend on the two bounds. More precisely, for the first task $j = 1$ one can only construct the task-specific estimate and use the bound B_j in Eq. 4 (as in LINUCB). Starting with task $j = 2$, after each observation we can update simultaneously the task-specific estimate $\widehat{\theta}_{j,t}$, the multi-task estimate $\widetilde{\theta}_{j,t}$, and their corresponding confidence bounds ($B_{j,t}(x)$ and $\widetilde{B}_{j,t}(x)$). Since both bounds are valid upper-confidence bounds on the reward of the arm, we only retain the tightest (smallest) of them, that is, the one closest to the true value of the arm. Then, we select the arm $x \in \mathcal{X}$ with the largest retained confidence bound. The resulting algorithm is sketched in Fig.1.

Input: budgets $\{n_j\}_j$, arms $\mathcal{X} \subset \mathbb{R}^d$, regularizer λ
$j = 1$
$A = \lambda \mathbb{I}_d, \widetilde{b} = b = 0_d, \widetilde{A}_j = \lambda \mathbb{I}_d, \hat{\theta}_j = A^{-1} b$
for $t = 1, \dots, n_j$ **do**
  Choose: $x_t = \arg\max_{x \in \mathcal{X}} (x_t^\top \hat{\theta}_j + B_{j,t}(x))$
  Observe reward: $r_t = x_t^\top \theta_j + \eta_t$
  Update A, b and the estimate $\hat{\theta}_j = A^{-1} b$
end for
for $j = 2, \dots, m+1$ **do**
  $\widetilde{A}_j = \widetilde{A}_j + A - \lambda \mathbb{I}_d$, $\widetilde{b} = \widetilde{b} + b$, $\widetilde{\theta}_j = \widetilde{A}_j^{-1} \widetilde{b}$
  $A = \lambda \mathbb{I}_d, b = 0_d, \hat{\theta}_j = A^{-1} b$
  for $t = 1, \dots, n_j$ **do**
    $x_t = \arg\max_{x \in \mathcal{X}} \min\left(x^\top \hat{\theta}_j + B_{j,t}(x); x^\top \widetilde{\theta}_j + \widetilde{B}_{j,t}(x)\right)$
    Observe reward: $r_t = x_t^\top \theta_j + \eta_t$
    Update: $A, b, \hat{\theta}_j, B_{j,t}, \widetilde{A}_j, \widetilde{b}, \widetilde{\theta}_j, \widetilde{B}_{j,t}$
  end for
end for

Figure 1: Multi-task LinUCB

Numerical simulations. We illustrate the performance of the multi-task strategy in a setting consisting of 200 tasks, with parameters $\theta_1, \dots, \theta_{200} \in \mathbb{R}^2$, generated at random, but bounded, with $\max_\theta ||\theta||_2 = 1.1 \cdot \sqrt{2}$, and $||\varepsilon||_2 = 0.2 \cdot \sqrt{2}$. The decision set $\mathcal{X}$ consists of five arms with ℓ_2 norm smaller than 1. The regularization parameter λ is set to 0.02, the noise η is distributed uniformly on $(-0.5, 0.5)$ and $\delta = 0.05$. In Fig. 2 we report the regret for each task, averaged over 1000 runs, where the sampling budget is limited to 100 samples for each task.

We compare the per-task regret obtained by MT-LINUCB with the regret of LINUCB that only uses the samples from the current task for the arm selection. These preliminary numerical results show that while the regret for LINUCB remains constant over time, in the case of the multi-task algorithm the regret decreases with every additional task. Indeed, given the strong similarity between the vectors θ and the fact that the sampling is done on only five arms, we gain valuable knowledge by exploiting the similarity between tasks and transferring samples from the past, while the bias introduced does not penalize the regret performance. In fact, we can expect that the regret of MT-LINUCB will be reduced down to the task-similarity term ε with the increase in the number of tasks.

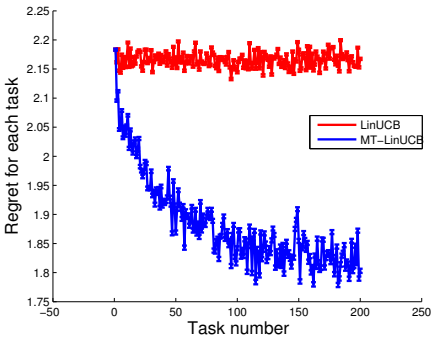

Figure 2: Per-task regret

5 Conclusion

This preliminary work shows the potential impact that multi-task learning could have in a sequence of decision-making tasks such as in the linear bandit setting. The transfer of samples strategy implemented in MT-LINUCB is relatively simple but already shows that the improvement with respect to single-task learning could be significant whenever the tasks at hand are similar. In the future we intend to provide a regret analysis for MT-LINUCB and test the similarity assumption through the empirical evaluation on recommendation system datasets. In addition, a number of interesting future challenges arise, such as defining a less restrictive notion of similarity between tasks or weighting the samples from past tasks according to their relevance for the current task.

Acknowledgments This work was supported by the French Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council, European Community's Seventh Framework Programme under grant agreement no 270327 (project CompLACS) and French National Research Agency (ANR) under project ExTra-Learn ANR-14-CE24-0010-01.

References

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems 24 - NIPS*, pages 2312–2320, 2011.

[2] Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual Bandits with Linear Payoff Functions. 2011.

[3] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning - ICML*, 2008.

[4] Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9:1757–1774, 2008.

[5] Mohammad Gheshlaghi Azar, Alessandro Lazaric, and Brunskill Emma. Sequential transfer in multi-armed bandit with finite set of models. In *Advances in Neural Information Processing Systems 26 - NIPS*, 2013.

[6] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web - WWW*, pages 661–670, 2010.

[7] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

A. Derivation of results in ①

Following the same steps as in the proof of [1, Theorem 2], we obtain:

$$
\begin{aligned}
&\left|x^\top \widetilde{\theta}^{\lambda}_{m+1,t} - x^\top \widetilde{\theta}^{*}_{m+1,t}\right| \leq \left|x^\top \big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1} \cdot \big(\sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_s \eta_s\big) + \lambda x^\top \big(\widetilde{A}^{\lambda}_{m+1}\big)^{-1} \cdot \widetilde{\theta}^{*}_{m+1}\right| \\
&\leq \big|\big|x\big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} \Bigg(\big|\big| \sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_s \eta_s \big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} + \lambda \big|\big| \widetilde{\theta}^{\lambda\ *}_{m+1} \big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} \Bigg) \quad \text{(by Cauchy-Schwarz)} \\
&\leq \big|\big|x\big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} \Bigg(\big|\big| \sum_{j=1}^{m+1}\sum_{s=1}^{n_j} x_s \eta_s \big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} + \lambda^{1/2} \big|\big| \widetilde{\theta}^{\lambda\ *}_{m+1} \big|\big|_2 \Bigg) \quad \text{(by [1, Th.1])} \\
&\leq \big|\big|x\big|\big|_{\left(\widetilde{A}^{\lambda}_{m+1}\right)^{-1}} \Bigg(R\sqrt{2\log\left(\frac{\det\big(\widetilde{A}^{\lambda}_{m+1}\big)^{1/2} \det(\lambda \mathbb{I})^{-1/2}}{\delta}\right)} + \lambda^{1/2} S \Bigg), \ w.p. \geq 1-\delta.
\end{aligned}
$$

B. Derivation of results in ②

Using the notation introduced in Sec.3, we obtain:

$$
\begin{aligned}
&\big|\big| \widetilde{\theta}^{*}_{m+1,t} - \theta_{m+1} \big|\big| \\
&= \big|\big| \sum_{j=1}^{m} \frac{n_j}{N} \widetilde{C}^{-1}_{m+1,t} C_j \big(\theta_j - \theta_{m+1}\big) + \sum_{j=1}^{m} \frac{n_j}{N} \widetilde{C}^{-1}_{m+1,t} C_j \theta_{m+1} + C^{-1}_{m+1,t} \frac{t}{N} C_{m+1,t} \theta_{m+1} - \theta_{m+1} \big|\big| \\
&= \big|\big| \sum_{j=1}^{m} \frac{n_j}{N} \widetilde{C}^{-1}_{m+1,t} C_j \big(\theta_j - \theta_{m+1}\big) + \widetilde{C}^{-1}_{m+1,t} \theta_{m+1} \big(\sum_{j=1}^{m} \frac{n_j}{N} C_j + \frac{t}{N} C_{m+1,t} \big) - \theta_{m+1} \big|\big| \\
&= \big|\big| \sum_{j=1}^{m} \frac{n_j}{N} \widetilde{C}^{-1}_{m+1,t} C_j \big(\theta_j - \theta_{m+1}\big) + \theta_{m+1} - \theta_{m+1} \big|\big| \\
&\leq \big|\big| \sum_{j=1}^{m} \frac{n_j}{N} \widetilde{C}^{-1}_{m+1,t} C_j \big|\big| \cdot \big|\big| \theta_j - \theta_{m+1} \big|\big| \\
&\leq \big|\big| \mathbb{I} \big|\big| \cdot \varepsilon = \varepsilon.
\end{aligned}
$$