AdaStop: adaptive statistical testing for sound comparisons of Deep RL agents

Anonymous authors
Paper under double-blind review

Abstract

Recently, the scientific community has questioned the statistical reproducibility of many empirical results, especially in the field of machine learning. To solve this reproducibility crisis, we propose a theoretically sound methodology to compare the overall performance of multiple algorithms with stochastic returns. We exemplify our methodology in Deep Reinforcement Learning (Deep RL). Indeed, the performance of one execution of a Deep RL algorithm is a random variable. Therefore, several independent executions are needed to accurately evaluate its performance. When comparing algorithms with random performance, a major question concerns the number of executions to perform to ensure that the results of the comparison are theoretically sound. Researchers in Deep RL often use less than 5 independent executions to compare algorithms: we claim that this is not enough in general. Moreover, when comparing several algorithms at once, we have to use a multiple tests procedure to preserve low error guarantees. We introduce AdaStop, a new statistical test based on multiple group sequential tests. When comparing algorithms, AdaStop adapts the number of executions to stop as early as possible while ensuring that we have enough information to distinguish algorithms that perform better than the others in a statistically significant way. We prove theoretically that AdaStop has a low probability of making a (family-wise) error. Finally, we illustrate the effectiveness of AdaStop in multiple Deep RL use-cases, including toy examples and challenging Mujoco environments.

AdaStop is the first statistical test fitted to this sort of comparisons: AdaStop is both a significant contribution to statistics, and an important contribution to computational studies performed in reinforcement learning and in other domains.

1 Introduction

In many fields of computer science, it is customary to perform an experimental investigation to compare the practical performance of two or more algorithms. When the behavior of an algorithm is non-deterministic (for instance because the algorithm is non-deterministic, or because the data it is fed upon is random), the performance of the algorithm is a random variable. Then, the way to do this comparison is not clear. Usually, one executes the algorithm (or rather an implementation of the algorithm: this point will soon be clarified) several times in order to obtain an average performance and its variability. How much "several" is depends on the authors, and some contingencies: thanks to the law of large numbers, one may think that the larger the better, the more accurate the estimates of the average and the variability. This may be a satisfactory answer, but when a single execution lasts days or even weeks or months (such as LLM training), performing a large enough number of executions is impossible. To illustrate this point, let us consider the field of deep reinforcement learning (Deep RL), that is reinforcement learning algorithms that use a neural network to represent what they learn. We surveyed all deep RL papers published in the proceedings of the International Conference on Machine Learning in 2022 (see Fig. 1a). In the vast majority of these papers, only a few executions have been performed: among the 18 papers using the Mujoco tasks, only 3 papers performed more than 10 runs, and 11 papers used only from 3 to 5 runs. This begs the question: if we can be confident that 3 executions is not enough, how many executions would be enough to draw statistically significant conclusions? Conversely, are 80 executions not too much? This is not only a matter of computation time

and computational resources occupation. Each execution contributes to pollute our planet and to the climate change. There is no answer to these questions today. Moreover, if someone redoes the comparison of the same algorithms on the same task, the conclusions should be the same: this concept is known as *statistical reproducibility* (Agarwal et al., 2021; Colas et al., 2019; Goodman et al., 2016) and should be a feature of any good experimental design. This paper provides an answer: we introduce AdaStop, a new statistical test to provide a statistically sound answer to this question, and we demonstrate its usage in practice. This paper is accompanied by a software that implements the test which is very easy to use. In short, in this paper, we provide a methodological approach and its actual implementation to compare the performance of algorithms having random performance in a statistically significant way, while trying to minimize the computational effort to reach such a conclusion. Such experimental investigations arise in various fields of computational AI such as machine learning, and computational optimization. We will use AdaStop in the field of reinforcement learning to illustrate this paper, but its application to other fields of computational AI is straightforward.

This paper is organized as follows: in an informal way, we detail the requirements that has to fulfil a statistical test to fit our expectations in Section 2. Section 3 introduces the main ingredients of our test, AdaStop. Section 4 in the formal analysis of the properties of AdaStop. Proofs are established in appendices C to F. We illustrate the use of AdaStop in Section 5 before concluding. Appendix A lists all the notations used in this paper. Appendix B provides a minimal exposition of the main concepts of hypothesis testing for readers who are not trained in statistics.

To reproduce the experiments of this paper, the python code is freely available on GitHub at `https://anonymous.4open.science/r/Adaptive_stopping_MC_RL-3434`. In addition, we provide a library and command-line tool that can be used independently: the AdaStop Python package is available at `https://anonymous.4open.science/r/adastop-EC1F`.

2 Informal presentation of AdaStop

2.1 Definitions

First, we define a few terms as we will use them in the rest of this paper.

As raised above, we do not compare algorithms but a certain implementation of an algorithm using a certain set of values for its hyperparameters. In D. Knuth's spirit (Knuth, 1968), we use the term **algorithm** in its usual meaning in computer science as the description of the basic operations required to transform a certain input into a certain output. By basic operations we mean the use of variables and simple operations (arithmetical, logical, etc), along with assignments to variables, sequences of instructions, tests and loops. Such an algorithm typically has some hyperparameters that control its behavior (a threshold, the dimension of the input domain, how a certain variable decays in time, the architecture of a neural network, etc). In this regard, we may say that as presented in (Schulman et al., 2017), PPO is an algorithm. However, one should be cautious that many aspects of PPO may be defined in various ways, and that the notion of "the PPO algorithm" is not as clearly defined as e.g. "the quicksort algorithm". The same may be said for all "Deep RL algorithms" of which there exist many variations.

Let us make clear the distinction between parameters and hyperparameters: parameters are learned from data, while hyperparameters are set a priori. For instance, the weights of a neural network are parameters, while the architecture of the network is a hyperparameter (unless this architecture is also learned during the training of the agent, which is far from being a common practice in Deep RL).

In experiments, we never compare algorithms: we compare a certain implementation of an algorithm, along a certain set of values for its hyperparameters, to other (implementation, hyperparameter values) pairs. We use the term **agent** to refer to a certain implementation of an algorithm along the value of its hyperparameters. If an agent is run several times, the value of the hyperparameters is the same at the beginning of each run. A special case concerns the seed of the pseudo-random number generator. Deep RL algorithms typically use a different seed to initialize each run. So the seed is not part of the definition of an agent. In the context of

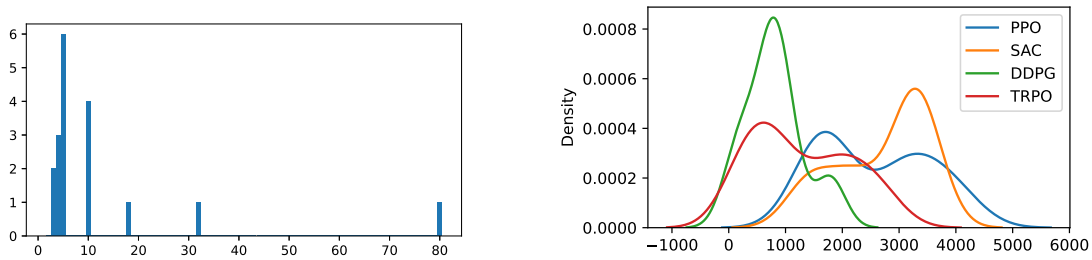
reinforcement learning, we call **policy** a trained agent. A policy is a decision function that maps observations to actions.

The result of one execution of a policy provides a **score**: this is a numerical value. To compare the performance of two agents, we compare their scores. We are free to define what the score is in a given experiment: this definition has to be aligned with the property of the agents we want to compare. For instance, in Deep RL, the common practice is as follows: an agent is trained. After training, the policy is run a given number of episodes to evaluate its performance. The score may be the mean of these evaluations, but the user is free to define the score according to her own interest. We suppose that the user has used enough evaluations to handle the possible stochasticity of both the policy and the evaluation environment.

2.2 Ingredients for an appropriate statistical test

Our goal is to provide a statistical test to decide whether one agent performs better than some others. Let us list the requirements and difficulties we have to face in the design of such a statistical test.

First, many tests in statistics are defined for Gaussian random variables. However, one quickly figures out that the observed performance of agents is usually not Gaussian. Fig. 1b illustrates this point: we represent the distribution of the performance of 4 agents implementing 4 different RL algorithms (PPO, SAC, DDPG, TRPO): the performance is usually multi-modal, and it is not even a mixture of a few Gaussian distributions as may seem. This leads us to a nonparametric test. Second, we would like to perform the minimal number of runs. This leads us to the use of a sequential adaptive test that tells us if we need to run the agents one or a few more times, or if we can take a decision in a statistically significant way with the already collected data. Third, we want to be able to compare more than 2 agents which leads us to multiple testing. Fourth, we want the conclusions of the test to be statistically reproducible, that is, if someone reproduces the execution of the agents and applies the test in the same way as someone else, the conclusion is the same. Fifth, we want to keep the number of runs within reasonable limits: for that purpose, we set a maximum number of scores to collect: if no decision can be made using this budget, the test can not decide whether an agent is better than the others. The first 3 requirements call for a nonparametric, sequential, multiple test. A candidate statistical test that may verify all these properties can be found in group sequential permutation test (see the textbook (Jennison & Turnbull, 1999) on general group sequential tests). We use these ingredients to construct AdaStop.



(a) (b)

Figure 1: Motivations for AdaStop. (a): a census of the number of scores (N_{score}) used in RL papers using Mujoco environments published in the proceedings of ICML 2022. (b): estimations of the score distributions of 4 well-known Deep RL agents on Hopper, a Mujoco environment, with $N_{score} = 30$. We see from the census that most experiments used 5 or less scores ($N_{score} \leq 5$) to draw conclusions. Those conclusions are most likely statistically wrong as they are equivalent to drawing 5 samples from distributions similar to the right plot to draw conclusions about the empirical means.

AdaStop, our proposed statistical test, meets all these expectations. Before diving into the technical details, let us briefly explain how AdaStop is used in practice. Fig. 2 illustrates an execution of AdaStop on a small example. Let us suppose that we want to compare the performance of 2 agents, a green agent and

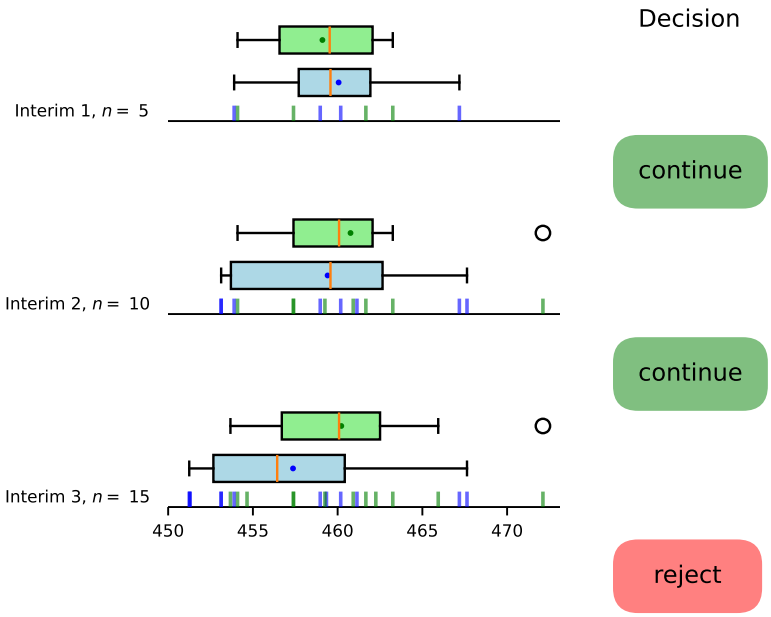


Figure 2: Schematic of Group Sequential Tests with AdaStop.

a blue agent. Fig. 2 illustrates the sequential nature of the test from top to bottom. Initially, each agent is executed $N = 5$ times. This yields 5 scores for each agent: these 10 initial scores are shown on the top-leftmost part of Fig. 2 labelled `Interim 1, n= 5`, along with their barplots. After the test statistics are computed, AdaStop decides that this information is not enough to conclude, and more scores are needed. This set of actions (collection of 5 scores for each agent, computation of the statistics, and decision) is known as an **interim**. As no conclusion can be drawn, a second interim is performed: both agents are run 5 more times, yielding 10 more scores. In the middle of Fig. 2, these additional scores are combined with the first 5 ones of each agent: these 20 scores are represented, as well as the barplot for each agent. The test statistics are computed using these 20 scores. Again, the difference is not big enough to make a decision given available information, so we make a third interim: each agent is executed 5 more times. At the bottom of Figure 2, these 30 scores are represented as well as the barplot for each agent. The test statistics are computed again on these 15 scores per agent. Now, AdaStop decides to reject the null hypothesis, which means that the two set of scores of the two agents are indeed different, and terminates: the compared agents do not perform similarly, and the green agent performs better than the blue one.

2.3 Current approaches for RL agents comparison

In the RL community, different approaches currently exist to compare agents. In (Colas et al., 2018; 2019), the authors show how to use hypothesis testing to test the equality between agents. Compared to our work, their approach is non-adaptive and only compares two agents. In Patterson et al. (2023), the authors explore a similar workflow with added steps specialized to deep RL (hyperparameter optimization, choice of the testing environment...). Another line of works can be found in (Agarwal et al., 2021) in which the authors compare many agents using confidence intervals.

In this section, we summarize some of the problems we identify with the current approaches used to compare two or more RL agents in research articles.

How many scores should we use? The number of scores used in practice in RL is quite arbitrary and often quite small (see Figure 1a). An intuition comes from the law of large numbers. As the performance of an agent is represented by the true mean of its scores, the more scores, the more precise the estimation of

its performance. However, this does not tell us anything about what is a sufficient number of scores to draw a statistically significant conclusion.

Theoretically sound comparison of multiple agents.

According to statistical theory, in order to compare more than 2 agents, we need more samples from each agent than when we compare only two agents. The basic idea is that there is a higher chance to make an error when we perform multiple comparisons than when we compare only two agents, hence we need more data to have a lower probability of error at each comparison. This informal argument is formalized in the theory of multiple testing. However, the theory of multiple testing has almost never been used to compare RL agents (with the notable exception of (Patterson et al., 2023, Section 4.5)). In this paper, we remedy this with AdaStop giving a theoretically sound workflow to compare 2 or more agents.

Theoretically sound study when comparing agents on a set of tasks.

Atari environments (Bellemare et al., 2013) are famous benchmarks in Deep RL. Due to time constraints, when using these environments, it is customary to use very few scores for one given game (typically 3 scores) and compare the agents on many different games. The comparisons are then aggregated: agent A_1 is better than agent A_2 on more than 20 games out of the 26 games consider in the experiments. In terms of rigorous statistics, this kind of aggregation is complex to analyse properly because reward distributions are not the same in all games. A_2 may be better than A_1 only on some easy games: does this mean that A_1 is better than A_2? Up to our knowledge, there is not any proper statistical guarantee for this kind of comparison.

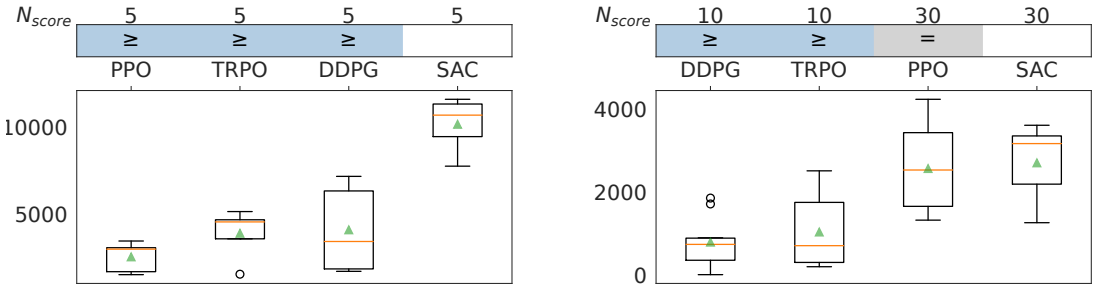
Advances have been made in (Agarwal et al., 2021) to interpret and visualize the results of RL agents in Atari environment. In particular the authors advise plotting confidence intervals and using the interquartile mean instead of the mean as aggregation functions. Correctly aggregating the comparisons on several games in Atari is still an *open problem*, and it is *beyond the scope of this article*. In this article, we suppose that we compare the agents on a single task, and we leave the comparison on a set of different tasks for future work. A discussion on these methods and the challenges of aggregating the results from several Atari environments can be found in the Appendix G.

2.4 Some methodologies for comparison of RL agents

Figure 1 of Patterson et al. (2023) defines a workflow for the meaningful comparison of two RL agents given an environment and a performance measure. In addition to the usual considerations regarding which statistics to compare (Colas et al., 2018; Agarwal et al., 2021), (Patterson et al., 2023) also include hyperparameter choice in the workflow. For that, they recommend to use 3 scores per algorithm per set of hyperparameters to identify a good choice of hyperparameters for a given algorithm. When this is done, a fixed number of scores (set using expert knowledge on the environment) are computed for each of these fully-specified agents. These scores are then used for statistical comparison. AdaStop fits at the end of this workflow.

In Fig 3, we showcase the use of AdaStop to compare SAC (Haarnoja et al., 2018) to other Deep RL algorithms on HalfCheetah and Hopper Mujoco tasks. One can imagine a scenario in which SAC inventors follow (Patterson et al., 2023) methodology. After finding the best hyperparameters for TRPO, PPO and DDPG (Schulman et al., 2015; 2017; Lillicrap et al., 2015) agents are compared with AdaStop using a minimal number of scores to get significant statistics. Patterson et al. (2023) recommends 15 scores per agent per environment for a maze environment but this number of scores should vary in other environments, and it is not clear how many scores should be used for HalfCheetah and Hopper. This is why we need to use AdaStop.

Using AdaStop, we collect scores in an adaptive manner. This allows us to state that we used enough scores to conclude that the SAC agent is better than other agents on HalfCheetah, and better than DDPG and TRPO agents on Hopper. As in all statistical tests, the conclusion holds up to a certain confidence level. A more in-depth study of the agents performance on Mujoco environments is given in Section 5.3.



(a) Comparisons on HalfCheetah.

(b) Comparisons on Hopper.

Figure 3: Example of the use of AdaStop to benchmark SAC in practice. We set the maximum number of runs B of each agent to 30. The upper row tables represent the conclusions when comparing SAC to the agents in the column using N_{score} scores. For example, on HalfCheetah, AdaStop concludes that SAC is better than PPO using 5 scores for SAC and 5 scores for PPO. On Hopper, 10 scores are enough to conclude that SAC outperforms DDPG and TRPO, and AdaStop concludes that SAC is as good as PPO using the maximum budget of B scores for both SAC and PPO.

2.5 Limitations of AdaStop

Before presenting the theory behind AdaStop, we want to make clear that doing a statistically sound comparison with very few samples is a hard problem and AdaStop is one way to partially answer this problem. In particular, (i) we do not claim that AdaStop is optimal, and (ii) more work is still necessary to prove theoretical guarantees for AdaStop to support the empirical performances exhibited in the experiments (see in particular Section 5.2 and discussions on the power of the test).

The main theoretical limitation of AdaStop comes from the fact that AdaStop is distribution-free: it does not make any assumption on the distributions of the scores. As a consequence, the only non-asymptotic guarantees that we have is on **the comparison of score distributions and not on the comparisons of their means**. Formally this means that we have control on the error (in Theorem 1) when doing the tests

$$H_j : P_{l_1} = P_{l_2}, \quad \text{against} \quad H'_j : P_{l_1} \neq P_{l_2}$$

where P_i is the distribution of scores for agent i, and (l_1, l_2) are the indices of the agents we want to compare.

Doing a comparison of distribution is not ideal, but it is the norm when doing distribution-free tests. In order to compare the means, some assumptions have to be made in order to have a good estimation of the mean. If we suppose a finite second moment (which correspond to a very weak hypothesis), we give asymptotic guarantees on the comparison of the means (see Theorem 2). Asymptotic guarantees are a sanity check for us but remain unsatisfactory because we are targeting small sample sizes. Non-asymptotic guarantees are harder to obtain and are left for future works.

3 Hypothesis testing for agents comparisons

In this section, we provide the background material on the statistical tests that we use to construct AdaStop. First we review the statistical evaluation methods found in the literature, and then we describe our methodology, and we express some results on AdaStop.

3.1 Literature overview of evaluation methods

In this section, we present some relevant references connected to statistical evaluation methodology.

Nonparametric (and non-sequential) hypothesis testing. One of our main challenges is to deal with the nonparametric nature of the data at hand. In the literature there has been a lot of work on nonparametric testing (see (Lehmann et al., 2005) for a comprehensive overview). Traditionally, the focus has been on asymptotic results due to challenges in deriving optimality for a nonparametric model. For most nonparametric tests, only rather weak (exact) theoretical results can be given, mostly on type I error (Romano, 1989; Shapiro & Hubert, 1979). Recent work on sequential nonparametric tests (Shin et al., 2021; Howard et al., 2021) show strong non-asymptotic results using concentration inequalities. However, these results often involve non-optimal constants, failing to explain non-asymptotic efficiency. In contrast, we use permutation tests due to their empirical (Ludbrook & Dudley, 1998) and theoretical (Kim et al., 2022) efficiency for small sample sizes.

Sequential tests. A closely related method for adaptive hypothesis testing consists in sequential tests. Two commonly used sequential tests are the Sequential Probability Ratio test (Wald, 1945) and the Generalized Likelihood Ratio test (Kaufmann & Koolen, 2021). In sequential testing, the scores are compared one after the other in a completely online manner. This is not adapted to our situation because in RL practice, one often trains several agents in parallel, obtaining a batch of scores at once. This motivates the use of group sequential tests (Jennison & Turnbull, 1999).

Parametric group sequential tests. In traditional hypothesis testing, data is analysed as a whole once it has been collected. Conversely, a Group Sequential Test (GST) data are collected sequentially, the tests being performed at *interim* time points. At each interim, a new set of N scores is collected (see (Jennison & Turnbull, 1999; Gordon Lan & DeMets, 1983; Pocock, 1977; Pampallona & Tsiatis, 1994) for references on GST). GST are often used in clinical trials to minimize the amount of data needed to conclude and this makes them well adapted for our purpose. The decision to continue sampling or conclude (with a controlled probability of error) depends on pre-defined stopping criteria used to define the tests. GST often makes strong assumptions on the data. In particular, it is often assumed that the data are i.i.d. and drawn from a Gaussian distribution (Jennison & Turnbull, 1999). This contrasts with our approach which is nonparametric.

Bandits (Best arm identification or ranking). Our objective is close to the one of bandit algorithms (Lattimore & Szepesvári, 2020): we *minimize the stopping time* (as in the fixed-confidence setting) of the test, and we have a *fixed maximum budget* (as in a fixed-budget setting). In our test, we allow a type I error with probability $\alpha \in (0, 1)$, which is similar to the fixed confidence setting while still having a fixed budget. In practice, our approach is more sample efficient than fixed budget bandit algorithms because these algorithms will always exhaust their budget and thereby achieve lower error rates. In contrast, AdaStop allows larger error rate in exchange to higher sample efficiency.

3.2 Background material on hypothesis testing

This section describes the basic building blocks used to construct AdaStop: group sequential testing, permutation tests, and step-down method for multiple hypothesis testing. We explain these items separately, and then we combine them to create AdaStop in Section 4. We also provide a small recap on hypotheses testing in the Appendix B for readers unfamiliar with these notions.

In order to **perform the minimal number of runs**, we propose to use group sequential testing (GST) with a nonparametric approach using permutation tests. Our approach is similar to (Mehta et al., 1994) but for multiple hypothesis testing. Compared to GST, we keep the i.i.d. assumption, but we do not assume that the data are drawn from a specific family of parametric distribution. In (Mehta et al., 1994), the authors use rank tests with group-sequential testing. In contrast with our work, (Mehta et al., 1994) does not provide theoretical guarantees and considers only the case of 2 agents.

3.2.1 Permutation tests

Permutation tests are **nonparametric tests** that are exact for testing the equality of distributions. This means that the type I error of the test (i.e. the probability to make a mistake and reject the equality of two agents when their scores are statistically the same) is controlled by the parameter of the test α, and that this is true for any fixed sample size N. Permutation tests are also well-known to work well in practice on

very small sample sizes and are used extensively in biology. They were originally introduced by Pitman (1937) and Fisher (1936), see (Lehmann et al., 2005, Chapter 17) for a textbook introduction. More recently Chung & Romano (2013) have studied asymptotic properties of this class of tests, while Romano & Wolf (2003) have focused on stepdown methods for multiple hypothesis testing.

Let us recall the basic formulation of a two-sample permutation test. Let $X_1, \dots, X_N$ be i.i.d. sampled from a law P and $Y_1, \dots, Y_N$ i.i.d sampled from a law Q. We want to test $P = Q$ against $P \neq Q$. Let $Z_i = X_i$ if $i \leq N$ and $Z_i = Y_{i-N}$ if $i > N$, $Z_1, \dots, Z_{2N}$. The test proceeds as follows: we reject $P = Q$ if $T(\text{id}) = \left|\frac{1}{N}\sum_{i=1}^{N}(Z_i - Z_{N+i})\right|$ is larger than a proportion $(1-\alpha)$ of the values $T(\sigma) = \left|\frac{1}{N}\sum_{i=1}^{N}(Z_{\sigma(i)} - Z_{\sigma(N+i)})\right|$ where σ enumerates all possible permutations of $\{1, \dots, 2N\}$ and id is the identity permutation ($\text{id}(i) = i$ for all i). Formally, we define the $(1-\alpha)$-quantile as

$$B_N = \inf\left\{b > 0 : \frac{1}{N!}\sum_{\sigma \in \mathbf{S}_N} \mathbb{1}\{T(\sigma) \geq b\} \leq \alpha\right\}$$

and we reject $P = Q$ when $T(\text{id}) \geq B_N$. The idea is that if $P \neq Q$, then $T(\text{id})$ should be large, and due to compensations, most $T(\sigma)$ should be smaller than $T(\text{id})$. Conversely, if $P = Q$, the difference of mean $T(\sigma)$ will be closer to zero. It is then sufficient to compute $T(\text{id})$ and B_N in order to compute the decision of the test. Please note that this is a fairly usual simplification in the nonparametric tests literature to test the equality in distribution instead of the equality of the mean, because equality between distributions is easier to deal with. It can be shown that permutation tests are nonetheless a good approximation of doing a comparison on the means (see Appendix E).

3.2.2 GST comparison of two agents

In this section, we compare two agents A_1 and A_2 through a group sequential test that can be seen as a particular case of AdaStop for two agents (see Section 4). In this simplified case, the testing procedure is presented in Algorithm 1. We leave the case with more than 2 agents to compare and the full version of AdaStop, including multiple hypothesis testing, for Section 4. Algorithm 1 uses a permutation test, where, at each interim, the boundary deciding the rejection is derived from the permutation distribution of the statistics observed across all previously obtained data. In what follows, we use N to denote the number of scores per interim and k to denote the current interim number.

We denote by $\mathbf{S}_{2N}$ the set of permutations of $\{1, \dots, 2N\}$, $\sigma \in \mathbf{S}_{2N}$ one permutation and $\sigma(n)$ the n-th element of σ for $n \in \{1, \dots 2N\}$. In the GST setting, we perform a permutation test at each interim k, and $\sigma_k \in \mathbf{S}_{2N}$ denotes the permutation at interim k. For $\sigma_1, \sigma_2, \dots, \sigma_k \in \mathbf{S}_{2N}$, we denote $\sigma_{1:k} = \sigma_1 \cdot \sigma_2 \cdot \ldots \cdot \sigma_k$ the concatenation of the permutation σ_1 done in interim 1 with σ_2 done on interim 2,..., and σ_k on interim k. Then, $e_{\sigma_i(n),i}$ denotes the score corresponding to the n-th element of the permuted sample at interim i, permuted by σ_i (in the notations of Section 3.2.1 this corresponds to $Z_{\sigma(i)}$ but now, we specify the interim number in the notation). We denote:

$$T_{N,k}(\sigma_{1:k}) = \left|\sum_{i=1}^{k}\left(\sum_{n=1}^{N} e_{\sigma_i(n),i} - \sum_{n=N+1}^{2N} e_{\sigma_i(n),i}\right)\right|, \tag{1}$$

and the decision boundary:

$$B_{N,k} \in \inf\left\{b > 0 : \frac{1}{((2N)!)^k}\sum_{\sigma_{1:k} \in \widehat{\mathcal{S}}_{N,k}} \mathbb{1}\{T_{N,k}(\sigma_{1:k}) \geq b\} \leq \frac{\alpha}{K}\right\}, \tag{2}$$

where K is the total number of interims and $\widehat{\mathcal{S}}_{N,k}$ is the set of permutations $\sigma_{1:k} \in (\mathbf{S}_{2N})^k$ such that the test was not rejected before interim k, e.g.

$$\widehat{\mathcal{S}}_{N,k} = \{\sigma_{1:k} \in (\mathbf{S}_{2N})^k : \quad \forall m < k, \quad T_{N,m}(\sigma_{1:m}) \leq B_{N,m}\}. \tag{3}$$

One can show that this is equivalent to choosing $B_{N,k}$ such that

$$\mathbb{P}_{\sigma_{1:k}}\left(T_{N,m}(\sigma_{1:m}) > B_{N,k}, \forall m < k, \quad T_{N,m}(\sigma_{1:m}) \leq B_{N,m}\right) \leq \frac{\alpha}{K}.$$

We will see in Theorem 1 that this allows us to have control on the type I error of the test.

Algorithm 1: Adaptive stopping to compare two agents.

Parameters: Agents A_1, A_2, environment $\mathcal{E}$, number of interims $K \in \mathbb{N}^*$, size of an interim N, level of the test $\alpha \in (0,1)$.

```
for k = 1, ..., K do
    for l = 1, 2 do
        Train agent A_l on environment E. This generates the policy π_l.
        Collect scores e_{1,k}(A_l), ..., e_{N,k}(A_l) by running N times the policy π_l.
    end
    Compute the boundary B_{N,k} using Equation (2).
    if T_{N,k}(id) ≥ B_{N,k} then
        return reject
    else
        if k = K then
            return accept
        end
    end
end
```

3.2.3 Multiple hypothesis testing

In order to compare more than two agents we need to perform multiple comparisons which calls for a **multiple simultaneous statistical tests** (Lehmann et al., 2005, Chapter 9). The idea is that the probability to mistakenly reject a null hypothesis (type I error) generally applies only to each test considered individually. On the other hand, in order to conclude on all the tests at once, it is desirable to have an error controlled over the whole family of simultaneous tests. For this purpose, we use the family-wise error rate (Tukey, 1953) which is defined as the probability of making at least one type I error.

Definition 1 (Family-Wise Error (Tukey, 1953))**.** *Given a set of hypothesis H_j for $j \in \{1, \dots, J\}$, its alternative H'_j, and $\boldsymbol{I} \subset \{1, \dots, J\}$ the set of the true hypotheses among them, then the family-wise error (FWE) is defined by:*

$$\text{FWE} = \mathbb{P}_{H_j, j \in \boldsymbol{I}}(\exists j \in \boldsymbol{I}: \quad \textit{reject } H_j)\,.$$

where the notation $\mathbb{P}_{H_j, j \in \boldsymbol{I}}$ denotes the probability distribution for which all hypotheses $j \in \boldsymbol{I}$ hold true[1]*. We say that an algorithm has a weak FWE control at a joint level $\alpha \in (0,1)$ if the FWE is smaller than α when all the hypotheses are true, that is $\boldsymbol{I} = \{1, \dots, J\}$ but not necessarily otherwise. We say it has strong FWE control if FWE is smaller than α for any non-empty set of true hypotheses $\boldsymbol{I} \neq \emptyset$ (while the notation $\boldsymbol{I}^c$ refers to false hypotheses).*

If we want to test the equality of L distributions $P_1, P_2, \dots, P_L$, the straightforward way is to do a pairwise comparison. This creates $J = \frac{L(L-1)}{2}$ hypotheses. We let $\mathbf{C} = \{\mathbf{c}_1, \dots, \mathbf{c}_J\}$ be the set of all possible comparisons between the distributions, where $\mathbf{c}_j = (l_1, l_2) \in \mathbf{C}$ denotes a comparison between distributions P_{l_1} and P_{l_2} for $l_1, l_2 \in \{1, 2, \dots, L\}$. Therefore, for $\mathbf{c}_j = (l_1, l_2)$, $H_{\mathbf{c}_j}$ denotes a hypothesis stating that P_{l_1} and P_{l_2} are equal and its alternative $H'_{\mathbf{c}_j}$ is that P_{l_1} and P_{l_2} are different. On the other hand, please note that if one wants to compare an agent to agents whose ranking does not interest us, it may be sufficient to only rank this agent compared to all the others yielding the $L-1$ comparisons $c_1 = (1, l_2)$ for $2 \leq l_2 \leq L$.

There are several procedures that can be used to control the FWE. The most famous one is Bonferroni's procedure (Bonferroni, 1936) recalled in the Appendix (Section B). As Bonferroni's procedure can be very

[1] See also Appendix B for further explanations on this concept.

conservative in general, we prefer a *step-down* method (Romano & Wolf, 2003) that performs better in practice because it implicitly estimates the dependence structure of the test statistics. The step-down method is detailed in the next section.

3.2.4 Step-down procedure

Romano & Wolf (2003) proposed the step-down procedure to solve a multiple hypothesis testing problem. It is defined as follows (for a *non group-sequential test*): for a permutation $\sigma \in \mathbf{S}_{2N}$ and for $(e_n(j))_{1\leq n\leq 2N}$ the random variables being compared in hypothesis j, the permuted test statistic of hypothesis j is defined by:

$$T_N^{(j)}(\sigma) = \left| \sum_{n=1}^{N} e_{\sigma(n)}(j) - \sum_{n=N+1}^{2N} e_{\sigma(n)}(j) \right| . \tag{4}$$

This test statistic is extended to any subset of hypothesis $\mathbf{C} \subset \{1, \ldots, J\}$ with the following formula:

$$\overline{T}_N^{(\mathbf{C})}(\sigma) = \max_{j\in\mathbf{C}} T_N^{(j)}(\sigma). \tag{5}$$

To specify the test, one compares $\overline{T}_N^{(\mathbf{C})}(\mathrm{id})$ to some threshold value $B_N^{(\mathbf{C})}$, that is: we accept all hypotheses in $\mathbf{C}$ such that $\overline{T}_N^{(\mathbf{C})}(\mathrm{id}) \leq B_N^{(\mathbf{C})}$. The threshold of the test $B_N^{(\mathbf{C})}$ is defined as the quantile of order $1-\alpha$ of the permutation law of $\overline{T}_N^{(\mathbf{C})}(\sigma)$:

$$B_N^{(\mathbf{C})} = \inf \left\{ b > 0 : \left(\frac{1}{(2N)!} \sum_{\sigma\in\mathbf{S}_{2N}} \mathbb{1}\{\overline{T}_N^{(\mathbf{C})}(\sigma) \geq b\} \right) \leq \alpha \right\}. \tag{6}$$

In other words, $B_N^{(\mathbf{C})}$ is the real number such that an α proportion of the values of $\overline{T}_N^{(\mathbf{C})}(\sigma)$ exceeds it, when σ enumerates all the permutations of $\{1, \ldots, 2N\}$. The permutation test is summarized in Algorithm 2.

Algorithm 2: Multiple testing by step-down permutation test.

Parameters: $\alpha \in (0, 1)$
Input: $e_n(j)$ for $1 \leq n \leq 2N$ and $j \in \mathbf{C}_0 = \{\mathbf{c}_1, \ldots, \mathbf{c}_J\}$.

```
Initialize C ← C_0.
while C ≠ ∅ do
    Compute T_N^(C)(σ) for every j and every σ using Equation (5).
    Compute B_n^(C) using Equation (6).
    if T̄_N^(C)(id) ≤ B_N^(C) then
        Accept all the hypotheses H_j, j ∈ C and exit the loop.
    else
        Reject H_{c_{j_max}} where c_{j_max} = arg max_{c_j ∈ C} T_N^(j)(id).
        Define C = C \ {c_{j_max}}
    end
end
```

Algorithm 2 is initialized with $\mathbf{C} = \mathbf{C}_0$ containing all the comparisons we want to test. Then, it enters a loop where the test decides to reject or not the most extreme hypothesis in $\mathbf{C}$, *i.e.* $H_{j_{\max}}$ where $j_{\max} = \arg\max_{j\in\mathbf{C}} T_N^{(j)}(\mathrm{id})$. Here, $\mathbf{C}$ is the current set of neither yet rejected nor accepted hypotheses. If the test statistic $T_N^{(j)}(\mathrm{id})$ for the most extreme hypothesis in $\mathbf{C}$ (i.e. $\overline{T}_N^{\mathbf{C}}(\mathrm{id})$) does not exceed the given threshold $B_n^{(\mathbf{C})}$, then all hypotheses in $\mathbf{C}$ are accepted, and the loop is exited. Otherwise, the most extreme hypothesis is discarded from the set $\mathbf{C}$ and another iteration is performed until either all remaining hypothesis are accepted, or the set of remaining hypotheses is empty.

The maximum of the statistics in Equation (5) for $\sigma = \text{id}$ allows us to test intersections of hypotheses, while the threshold $B_n^{(\mathbf{C})}$, under the null hypotheses of equality of distribution, allows for strong control on the FWE (i.e. FWE $\leq \alpha$). This last result follows from (Romano & Wolf, 2003, Corollary 3) and is a particular case of Theorem 1 for AdaStop. In fact, this procedure is not specific to permutation tests, and it can be used for other tests provided some properties hold on the thresholds $B_n^{(\mathbf{C})}$.

4 AdaStop: adaptive stopping for nonparametric group-sequential multiple tests

In this section, we present the construction and the theoretical properties of AdaStop (see Algorithm 3). AdaStop compares the scores of multiple agents in an *adaptive* rather than a fixed way. We consider $L \geq 2$ agents $A_1, \ldots, A_L$. As above, we let $\mathbf{C}_0 = \{\mathbf{c}_1, \ldots, \mathbf{c}_J\} \subseteq \{1, \ldots, L\}^2$ be the set of all the comparisons to make between the agents. $\mathbf{I}$ denotes the set of indices of the true hypotheses among $\{1, \ldots, J\}$.

Algorithm 3 specifies AdaStop. AdaStop relies on the test statistic $\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_{1:k})$ defined in equation (8) below, and the boundary thresholds $B_{N,k}^{\mathbf{C}}$ defined in Equation (9). We discuss a few implementation details in the rest of this section.

Definition of the test statistics. Let $e_{1,i}(j), \ldots, e_{2N,i}(j)$ denote the $2N$ scores used at interim i. They are obtained through execution of the policies trained from the two agents A_{l_1} and A_{l_2} for which $\mathbf{c}_j = (l_1, l_2)$. We also consider permutations of these scores to define the test statistics $T_{N,k}^{(j)}$ below. For a comparison j, we consider a permutation $\sigma_i \in \mathbf{S}_{2N}$ at interim i that reshuffles the order of the scores mapping $n \in \{1, \ldots, 2N\}$ on $\sigma_i(n) \in \{1, \ldots, 2N\}$. Note that if $n \in \{1, \ldots, N\}$ and $\sigma_i(n) \in \{N+1, \ldots, 2N\}$, we are exchanging a score of the first agent with a score of the second agent in the comparison. It can also happen that instead, we permute two scores of the same agents. The difference between the two cases is important for the definition of the following permutation statistic:

$$T_{N,k}^{(j)}(\sigma_{1:k}) = \left| \sum_{i=1}^{k} \left(\sum_{n=1}^{N} e_{\sigma_i(n),i}(j) - \sum_{n=N+1}^{2N} e_{\sigma_i(n),i}(j) \right) \right| . \tag{7}$$

In other words, $T_{N,k}^{(j)}(\sigma_{1:k})$ is the absolute value of the sum of differences of all scores until interim k after consecutive permutations of the concatenation of the two agents scores by $\sigma_1, \ldots, \sigma_k \in \mathbf{S}_{2N}$. Let $\mathbf{C} \subseteq \mathbf{C}_0$ be a subset of the set of considered hypothesis and let us denote:

$$\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_{1:k}) = \max_{j \in \mathbf{C}} T_{N,k}^{(j)}(\sigma_{1:k}) \,, \tag{8}$$

$\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_{1:k})$ is the test statistic used in AdaStop. The construction of the test is inspired by the permutation tests of Equation (4) used to test intersection of hypotheses as from Equation (5) in the step-down method presented in Section 3.2. Still, it also incorporates group sequential tests from Section 3.2.2 and its test statistics introduced in Equation (1).

Choice of permutations. Instead of using all the permutations as we did for now, one may use a random subset among all permutations $\mathcal{S}_{N,k} \subset \{\sigma_{1:k}, \quad \forall i \leq k, \sigma_i \in \mathbf{S}_{2N}\}$ to speed-up computations. The theoretical guarantees persist as long as the choice of the permutations is made independent on the data. Using a small number of permutations will decrease the total power of the test, but with a sufficiently large number of random permutations (typically for the values of N and K we consider, 10^4 permutations are sufficient) the loss in power is acceptable. Please note that we need to include the identity in addition to the random permutations to keep the type I error guarantee (Phipson & Smyth, 2010).

$T_{N,k}$ does not change when the permutation does not exchange any index from $\{1, \ldots, N\}$ with an index of $\{N+1, \ldots, 2N\}$. In essence, choosing a permutation is equivalent to choosing the signs in $\sum_{n=1}^{N} e_{\sigma_i(n),i}(j) - \sum_{n=N+1}^{2N} e_{\sigma_i(n),i}(j)$. And because we take the absolute value, there are $\frac{1}{2}\binom{2N}{N}$ possible unique values to $T_{N,1}$ (up to ties when the score distributions are discrete). Then, by enumerating all the permutations for the other interims, there are $\frac{1}{2}\binom{2N}{N}^k$ possible unique values to $T_{N,k}$.

In practice, we use a parameter $B \in \mathbb{N}$ and the number of permutations used at interim k will be $|\mathcal{S}_{N,k}| = m_k = \min\left(B, \frac{1}{2}\binom{2N}{N}^k\right)$, i.e. whenever possible, we use all the permutations and if this is too much, we use permutations drawn at random.

Definition of the boundaries. With these permutations, we define the boundary thresholds $B_{N,k}^{(\mathbf{C})}$ by:

$$B_{N,k}^{(\mathbf{C})} = \inf\left\{ b > 0 : \frac{1}{m_k} \sum_{\sigma \in \widehat{\mathcal{S}}_{N,k}} \mathbb{1}\{\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_{1:k}) \geq b\} \leq q_k \right\}. \tag{9}$$

where $\sum_{j=1}^{k} q_j \leq \frac{k\alpha}{K}$ and where $\widehat{\mathcal{S}}_{N,k}$ is the subset of $\mathcal{S}_{N,k}$ such that the statistic associated to the permutation would not have been rejected before. Formally, $\widehat{\mathcal{S}}_{N,k}$ is the following set of permutations:

$$\widehat{\mathcal{S}}_{N,k} = \left\{ \sigma_{1:k} : \forall m < k, \quad \overline{T}_{N,m}^{(\mathbf{C})}(\sigma_{1:m}) \leq B_{N,m}^{(\mathbf{C})} \right\}. \tag{10}$$

Note that q_1 is not equal to α/K. Due to discreetness (we use an empirical quantile over a finite number of values), q_1 is chosen equal to $\lfloor \frac{\alpha}{2K}\binom{2N}{N} \rfloor / (\frac{1}{2}\binom{2N}{N})$. Similarly, q_2 is chosen to be as large as possible while having $q_1 + q_2 \leq 2\alpha/K$. And so on and so forth: in practice this means taking $q_i = \lfloor \frac{\alpha i}{2K}\binom{2N}{N} \rfloor / (\frac{1}{2}\binom{2N}{N}) - q_{i-1}$ for $2 \leq i \leq K$.

Algorithm 3: AdaStop (main algorithm).

Parameters: Agents $A_1, A_2, \ldots, A_L$, environment $\mathcal{E}$, comparison pairs $(c_j)_{j \leq J}$ where c_i is a couple of agents that we want to compare. Integers $K, N \in \mathbb{N}^*$, test parameter α.

```
Set C = {1, ..., J} the set of indices for the comparisons to perform.
for k = 1, ..., K do
    for l = 1...L do
        Train N times agent A_l on environment E. This generates N policies.
        Collect the N scores of the N policies.
    end
    while True do
        Compute the boundaries B^(C)_{N,k} using Equation (9).
        if T^(C)_{N,k}(id) > B^(C)_{N,k} then
            Reject H_{j_max} where j_max = arg max ( T̄^(j)_{N,k}(id),  j ∈ C ).
            C = C \ {j_max}
        else
            Exit the while loop.
        end
    end
    if C = ∅ then  Exit the loop and return the decision of the test.
    if k = K then  Exit the loop and accept all hypotheses remaining in C.
end
```

4.1 Theoretical guarantees of AdaStop

One of the basic properties of two-sample permutation tests is that when the null hypothesis is true, then all permutations are as likely to give a certain value and permuting the sample should not change the test statistic too much. Following our choice of $\overline{B}_{N,k}$ as a quantile of the law given the data, the algorithm has a probability to wrongly reject the hypothesis bounded by α. This informal statement is made precise in the following theorem.

Theorem 1 (Controlled family-wise error)**.** *Suppose that $\alpha \in (0,1)$, and consider the multiple testing problem $H_j : P_{l_1} = P_{l_2}$ against $H'_j : P_{l_1} \neq P_{l_2}$ for all the couples $\boldsymbol{c}_j = (l_1, l_2) \in \{\boldsymbol{c}_1, \dots, \boldsymbol{c}_J\}$. Then, the test resulting from Algorithm 3 has a strong control on the family-wise error for the multiple test, i.e. if we suppose that all the hypotheses $H_i, i \in \mathbf{I}$ are true and the others are false, then*

$$\mathbb{P}\left(\exists j \in \mathbf{I} : \quad \text{reject } H_j\right) \leq \alpha.$$

The proof of Theorem 1 is given in the Appendix (Section C).

Hypotheses of the test: in Theorem 1 we show that Algorithm 3 tests the equality of the distributions $P_{l_1} = P_{l_2}$ versus $P_{l_1} \neq P_{l_2}$, whereas in practice we would prefer to compare the means of the distribution $\mu_{l_1} = \mu_{l_2}$ versus $\mu_{l_1} \neq \mu_{l_2}$. Doing the test on distributions is something common in nonparametric tests (Lehmann et al., 2005). It is justified because without strong concentration assumptions on the distributions, we lack control on the mean of the distributions. On the other hand, we show in the Appendix E that for large N, the test comparing the means $\mu_{l_1} = \mu_{l_2}$ versus $\mu_{l_1} \neq \mu_{l_2}$ has the right guarantees (FWE smaller than α). This shows that even though we test the distributions, we also have an approximate test on the means. More precisely, we show the following in the Appendix E for the comparison of the means of two distributions.

Theorem 2. *Suppose that $\alpha \in (0,1)$, suppose that X has a finite variance, and consider the two-sample testing problem $H_0 : \mathbb{E}_P[X] = \mathbb{E}_Q[X]$ against $H'_0 : \mathbb{E}_P[X] \neq \mathbb{E}_Q[X]$. Then, the test resulting from Algorithm 3 has an asymptotic level of α*

$$\lim_{N \to \infty} \mathbb{P}_{H_0}\left(\textit{reject } H_0\right) = \alpha.$$

The proof of Theorem 2 is given in the Appendix (Section E).

Power of the test: in Theorem 1, there is no information on the power of the test. For any N, K, we show that, the test is of level α. In the appendix E, we establish Theorem 3 that can be used to get a control on the asymptotic power. On the other hand, having non-asymptotic information on the power would allow us to give a rule for the choice of N and K. However, non-asymptotic power analysis in a nonparametric setting is in general hard, and it is beyond the scope of this article. Instead, we compute empirically the power of our test and show that it performs well empirically compared to non-adaptive approaches. See Section 5.2 for the empirical power study on a Mujoco environment.

5 Experimental study

This section demonstrates AdaStop from a practitioner perspective. First, we illustrate the statistical properties of AdaStop on toy examples in which the scores of the agents are sampled from known distributions. Then, we compare empirically AdaStop to non-adaptive approaches. Finally, we exemplify the use of AdaStop on a real case to compare a set of Deep RL agents.

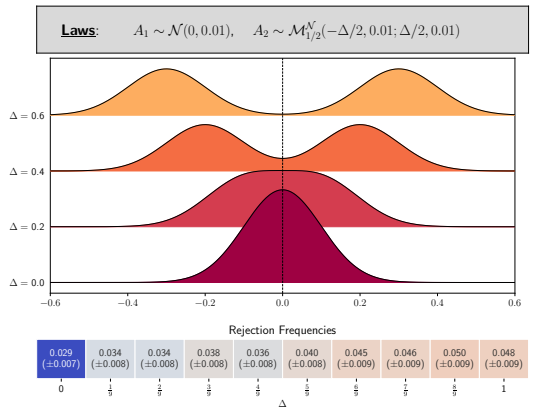
5.1 Toy examples

To start with, let us follow the execution of our algorithm on toy examples. In what follows, let us denote (i) $\mathcal{N}(\mu, \sigma^2)$ the normal distribution with mean μ and standard deviation σ, (ii) $t(\mu, \nu)$ the t-Student distribution with mean μ and ν degrees of freedom, (iii) $\mathcal{M}^{\mathcal{N}}_{\frac{1}{2}}(\mu_1, \sigma_1^2; \mu_2, \sigma_2^2)$ the mixture of 2 Gaussian distributions $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$, and (iv) $\mathcal{M}^{t}_{\frac{1}{2}}(\mu_1, \nu_1; \mu_2, \nu_2)$ the mixture of 2 t-Student distributions $t(\mu_1, \nu_1)$ and $t(\mu_2, \nu_2)$.

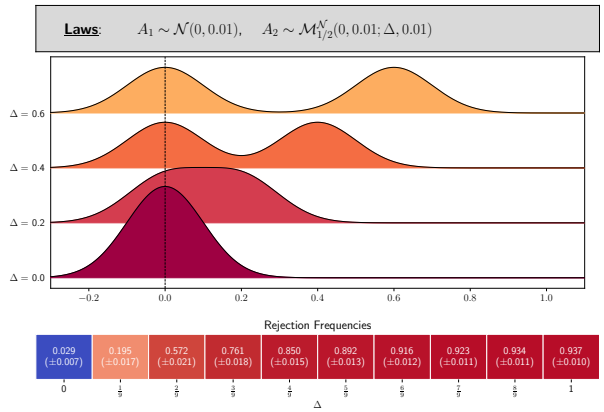
We compare two agents A_1 and A_2 for which we know the distributions of their scores. We consider three settings in Fig. 4. Δ is the distance between two modes of the mixtures ($\Delta = |\mu_1 - \mu_2|$). In all three cases, we run AdaStop with $K = 5$, $N = 5$ and $\alpha = 0.05$. We also limit the maximum number of permutations to $B = 10^4$. At the bottom of each subfigure of Figure 4, a color bar indicates the rejection rate of the null hypothesis that the compared distributions are the same. By varying Δ from 0 to 1, we observe the evolution of the power of the test, i.e. the probability of rejecting the null hypothesis when it is indeed

false. Figure 4a shows that the power of the test remains around 0.05 level for all Δ (it is at most 0.1 for the most extreme case). Indeed, even though the distributions in the comparison are different, their means remain the same. If the null hypothesis states that the means are the same, then AdaStop will return the correct answer with type I error not larger than 0.095 (see Figure 4a) for $\alpha = 0.05$. This is an illustration of the fact that in addition to performing a test on the distributions, AdaStop approximates the test on the means as shown theoretically in the asymptotic result in Appendix Section E and as discussed at the end of Section 4.1. In contrast, Figure 4b demonstrates an increasing trend, reaching a level close to 1 when $\Delta > 0.6$, which corresponds to the case where the two modes are separated by 3 standard deviations from both sides. Finally, in Figure 4c, we use a distribution made of an unbalanced mixture of 2 normal distributions. This is meant to model an agent does not perform optimally except in some rare cases (e.g. the agent has a score around 0 for 90% of the time). As in Case 1, the mean of the scores distributions for the two agents are 0 and AdaStop accepts the equality most of the time, with an error of 0.2 in the most extreme case which we think is acceptable given the difficulty of this setting.

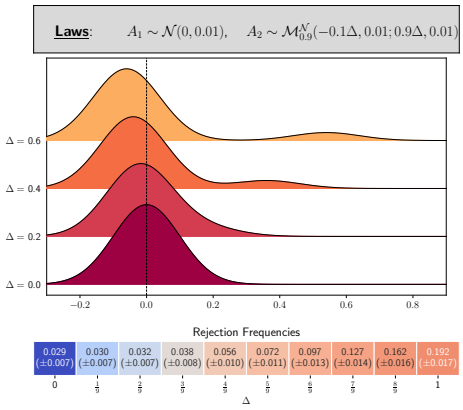
To obtain an estimation of the error, we have executed each comparison $M = 5 \cdot 10^3$ times, and we plot confidence intervals corresponding to $3\sigma/\sqrt{M}$ (confidence larger than 99%) where σ is the standard deviation of the test decision. In addition to cases 1, 2 and 3, we also provide a fourth experiment with a comparison of 10 agents in Appendix H.1.



(a) Case 1



(b) Case 2



(c) Case 3

Figure 4: Toy examples. Fr each of these 3 settings, we plot the distribution of scores of each of the 2 agents that are compared, as well as the frequency of rejection of the null hypothesis. Please, refer to the text for details.

5.2 Comparison with non-adaptive approach

N\K	2	3	4	5	6
1	0.0 (2.0)	0.0 (3.0)	0.277 (4.0)	0.465 (5.0)	0.56 (6.0)
2	0.005 (4.0)	0.33 (6.0)	0.531 (6.96)	0.602 (8.345)	0.704 (9.198)
3	0.213 (5.984)	0.506 (8.085)	0.627 (10.212)	0.689 (11.02)	0.785 (11.52)
4	0.371 (7.616)	0.611 (9.648)	0.744 (11.7)	0.82 (12.08)	0.845 (13.89)
5	0.465 (9.044)	0.691 (11.031)	0.78 (13.28)	0.853 (14.27)	0.884 (14.532)
6	0.534 (10.4)	0.73 (12.306)	0.837 (14.124)	0.89 (14.94)	0.911 (15.978)
7	0.599 (11.358)	0.779 (13.404)	0.879 (14.916)	0.92 (15.495)	0.939 (16.404)
8	0.635 (12.322)	0.818 (13.95)	0.885 (15.824)	0.942 (16.03)	0.961 (17.268)

Table 1: Average empirical statistical power and, in parentheses, effective number of scores used by AdaStop as a function of the total number of scores ($N \times K$) when comparing SAC and TD3 agents on Mujoco HalfCheetah task. The number of permutations B is set to 10^4 and α is set to 0.05. AdaStop is run 10^3 times for each (N, K) pair. The shades of blue are proportional to the power, a value in $[0, 1]$ (we use the same color scheme as in (Colas et al., 2018)).

Colas et al. (2019) share the same objective with us. However, they use non-adaptive tests unlike AdaStop. We follow their experimental protocol and compare AdaStop and non-adaptive approaches empirically in terms of statistical power as a function of the sample size (number of scores). In particular, we use the data[2] they provide for a SAC agent and for a TD3 agent evaluated on HalfCheetah (see Fig. 11 in the Appendix). Similarly to (Colas et al., 2019, Table 15), we compute the empirical statistical power of AdaStop as a function of the number of scores of the RL agents (Table 1). To compute the empirical statistical power for a given number of scores, we make the hypothesis that the distribution of SAC and TD3 agents scores are different, and we count how many times AdaStop decides that one agent is better than the other (number of true positives). As the test is adaptive, we also report the effective number of scores that are necessary to make a decision with 0.95 confidence level. For each number of scores, we have run AdaStop 10^3 times. For example, when comparing the scores of SAC and TD3 on HalfCheetah using AdaStop with $N = 4$ and $K = 5$, the maximum number of scores that could be used is $N \times K = 20$ without early stopping. However, we observe in Table 1 that when $N = 4$ and $K = 5$, AdaStop can make a decision with a power of 0.82 using only 12 scores. In (Colas et al., 2019, Table 15), the minimum number of scores required to obtain a statistical power of 0.8 when comparing SAC and TD3 agents is 15 when using either a t-test, or a Welch test, or a bootstrapping test. With this example, we first show that being an adaptive test, AdaStop may save computations we want to perform a "reasonable enough" amount of runs of their agents. We also show that as long as the scores of agents are made available, AdaStop can use them to provide a statistically sound conclusion, and as such, AdaStop may be used to assess the initial conclusions, hopefully strengthening them with a statistically significant argument.

Remark: when N and K are both small, it can happen that AdaStop does not in fact allow an error up to α but it will restrict itself to a smaller error and this will lead to a much lower power. For instance, if $N = 2, K = 2, \alpha = 0.05$ at first interim, we have $\lfloor \frac{1}{2}\binom{2N}{N}\alpha/2 \rfloor = 0$ hence the first boundary is necessarily infinity and never rejects, e.g. with the notations of Equation (2), $q_1 = 0$. Then at the second interim $\lfloor \frac{1}{2}(\binom{2N}{N})^2\alpha/2 \rfloor = 3$ so that in effect we only used an error $q_1 + q_2 = \lfloor \frac{1}{2}(\binom{2N}{N})^2\alpha/2 \rfloor / (\frac{1}{2}(\binom{2N}{N})^2) \simeq 0.041$. This is an extreme case happening in the first few interims when N is small explaining some values in Table 1 that may seem strange at first glance.

5.3 AdaStop for Deep Reinforcement Learning

In this section, we use AdaStop to compare four commonly-used Deep RL agents on the MuJoCo[3] (Todorov et al., 2012) benchmark for high-dimensional continuous control. We use the Gymnasium[4] implementation.

[2] available at `https://github.com/flowersteam/rl_stats/tree/master/data`.
[3] We use MuJoCo version 2.1, as required by https://github.com/openai/mujoco-py.
[4] https://github.com/Farama-Foundation/Gymnasium.

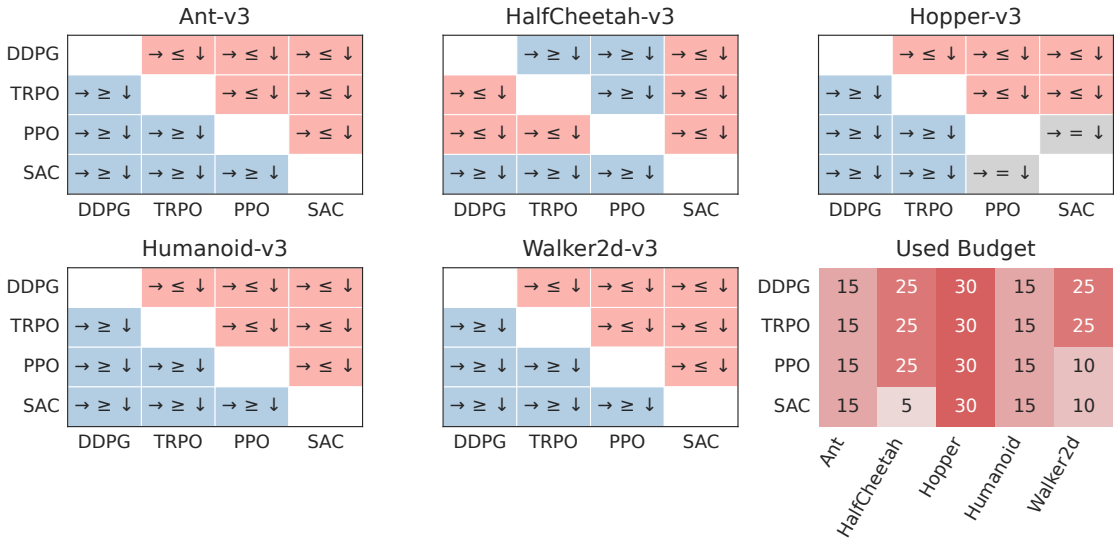


Figure 5: AdaStop decision tables for each MuJoCo environment, and the budget used to make these decisions (bottom right). See Appendix H.2 for further details.

More specifically, we train agents on the Ant-v3, HalfCheetah-v3, Hopper-v3, Humanoid-v3, and Walker-v3 environments using PPO from rlberry (Domingues et al., 2021), SAC from Stable-Baselines3 (Raffin et al., 2021), DDPG from CleanRL (Huang et al., 2022), and TRPO from MushroomRL (D'Eramo et al., 2021). PPO, SAC, DDPG, and TRPO are all Deep RL algorithms used for high-dimensional continuous control tasks. We choose these algorithms because they are commonly used and represent a diverse set of approaches from different RL libraries. We use different RL libraries in order to demonstrate the flexibility of AdaStop, as well as to provide examples on how to use these popular libraries with AdaStop.

On-policy algorithms, such as PPO and TRPO, update their policies based on the current data they collect during training, while off-policy algorithms, such as SAC and DDPG, can learn from any data, regardless of how it was collected. This difference may make off-policy algorithms more sample-efficient but less stable than on-policy algorithms. Furthermore, SAC typically outperforms DDPG in continuous control robotics tasks due to its ability to handle stochastic policies, while DDPG restricts itself to deterministic policies (Haarnoja et al., 2018). Finally, PPO is generally considered performing better than TRPO in terms of cumulative reward (Engstrom et al., 2020).

For each algorithm, we fix the hyperparameters to those used by the library authors in their benchmarks for one of the MuJoCo environments. Appendix H.2 lists the values that were used and we further discuss the experimental setup.

We compare the four agents in each environment using AdaStop with $N = 5$ and $K = 6$. Fig. 5 shows the AdaStop decision tables for each environment, as well as the number of scores per agent and environment. As expected, SAC ranks first in each environment. In contrast, DDPG is ranked last in four out of five environments; this may be due to the restriction to deterministic policies which hurts exploration in high-dimensional continuous control environments such as the MuJoCo benchmarks. Furthermore, we observe that the expected ordering between PPO and TRPO is generally respected, with TRPO outperforming PPO in only one environment. Finally, we note that PPO performs particularly well in some environments obtaining scores that are comparable to those of SAC, while also being the worst-performing algorithm on HalfCheetah-v3. Overall, the AdaStop rankings in these experiments are not unexpected.

Moreover, our experiments demonstrate that AdaStop can make decisions with fewer scores, thus reducing the computational cost of comparing Deep RL agents. For instance, as expected, SAC outperformed other agents on the environment HalfCheetah-v3. AdaStop required only five scores to make all decisions involving SAC. Additionally, we observed that the decisions requiring the entire budget of $NK = 30$ scores were the

ones in which AdaStop determined that the agents were equivalent in terms of their scores. With the early-accept heuristic detailed in the Appendix (Section F), this process can be sped-up. For instance in the Walker2d-v3 environment, early accept allows us to take all the decisions after only 10 scores have been collected for each agent.

6 Conclusion and future works

In this paper, we introduce AdaStop which is a sequential group test aiming at ranking agents based on their practical performance. AdaStop may be applied in various fields where the performance of algorithms are random, such as machine learning, or optimization. Our goal is to provide statistical grounding to define the number of times a set of agents should be run to be able to confidently rank them, up to some confidence level α. This is the first such test, and we think this is an important contribution to computational studies in reinforcement learning and other domains. From a statistical point of view, we have been able to demonstrate the soundness of AdaStop as a statistical test. Using AdaStop is simple. We provide open source software to use it. Experiments demonstrate how AdaStop may be used in practice, even in a retrospective manner using logged data: this allows one to diagnose prior studies in a statistically significant way, hopefully confirming their conclusions, possibly showing that the same conclusions could have been obtained with fewer computations.

Currently, AdaStop considers a set of agents facing one single task. Our next step will be to extend the test to experimental settings where a set of agents are compared on a collection of tasks, such as the set of Atari games or the set of Mujoco tasks in reinforcement learning. Properly dealing with such experimental settings requires a careful statistical analysis. Moreover, additional theoretical guarantees for the early accept and the non-asymptotic control of the power and FWE of AdaStop for the comparison of means would also greatly improve the interpretability of the conclusions of AdaStop. We are currently investigating these questions.

As this is illustrated in the experimental section, AdaStop can be run on already collected scores: we do not need to run anew the agents as long as scores are available. This remark calls for an effort of the community to make their scores publicly available so that it is easy for anyone to compare one's new agent with others already proposed.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Carlo Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, 8:3–62, 1936.

EunYi Chung and Joseph P. Romano. Exact and asymptotically robust permutation tests. *The Annals of Statistics*, 41(2):484 – 507, 2013. doi: 10.1214/13-AOS1090. URL `https://doi.org/10.1214/13-AOS1090`.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. How many random seeds? statistical power analysis in deep reinforcement learning experiments. *arXiv preprint arXiv:1806.08295*, 2018.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. A hitchhiker's guide to statistical comparisons of reinforcement learning algorithms. *arXiv preprint arXiv:1904.06979*, 2019.

Geoff Cumming and Sue Finch. Inference by eye: confidence intervals and how to read pictures of data. *American psychologist*, 60(2):170, 2005.

Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. Mushroomrl: Simplifying reinforcement learning research. *Journal of Machine Learning Research*, 22(131):1–5, 2021. URL `http://jmlr.org/papers/v22/18-056.html`.

Omar Darwiche Domingues, Yannis Flet-Berliac, Edouard Leurent, Pierre Ménard, Xuedong Shang, and Michal Valko. rlberry - A Reinforcement Learning Library for Research and Education, 10 2021. URL `https://github.com/rlberry-py/rlberry`.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. *arXiv preprint arXiv:2005.12729*, 2020.

Ronald Aylmer Fisher. Design of experiments. *British Medical Journal*, 1(3923):554, 1936.

Yannis Flet-Berliac, Reda Ouhamma, Odalric-Ambrym Maillard, and Philippe Preux. Learning value functions in deep policy gradients using residual variance. In *ICLR 2021-International Conference on Learning Representations*, 2021.

Steven N Goodman, Daniele Fanelli, and John PA Ioannidis. What does research reproducibility mean? *Science translational medicine*, 8(341):341ps12–341ps12, 2016.

KK Gordon Lan and David L DeMets. Discrete sequential boundaries for clinical trials. *Biometrika*, 70(3): 659–663, 1983.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055 – 1080, 2021. doi: 10.1214/20-AOS1991. URL `https://doi.org/10.1214/20-AOS1991`.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL `http://jmlr.org/papers/v23/21-1342.html`.

Christopher Jennison and Bruce W Turnbull. *Group sequential methods with applications to clinical trials*. CRC Press, 1999.

Emilie Kaufmann and Wouter M Koolen. Mixture martingales revisited with applications to sequential tests and confidence intervals. *The Journal of Machine Learning Research*, 22(1):11140–11183, 2021.

Ilmun Kim, Sivaraman Balakrishnan, and Larry Wasserman. Minimax optimality of permutation tests. *The Annals of Statistics*, 50(1):225 – 251, 2022. doi: 10.1214/21-AOS2103. URL `https://doi.org/10.1214/21-AOS2103`.

D. Knuth. *The art of programming, volume 1, Fundamental Algorithms*. Addison-Wesley, 1968.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Erich Leo Lehmann, Joseph P Romano, and George Casella. *Testing statistical hypotheses*, volume 3. Springer, 2005.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

John Ludbrook and Hugh Dudley. Why permutation tests are superior to t and f tests in biomedical research. *The American Statistician*, 52(2):127–132, 1998.

Cyrus R Mehta, Nitin Patel, Pralay Senchaudhuri, and Anastasios Tsiatis. Exact permutational tests for group sequential clinical trials. *Biometrics*, pp. 1042–1053, 1994.

Sandro Pampallona and Anastasios A Tsiatis. Group sequential designs for one-sided and two-sided hypothesis testing with provision for early stopping in favor of the null hypothesis. *Journal of Statistical Planning and Inference*, 42(1-2):19–35, 1994.

Andrew Patterson, Samuel Neumann, Martha White, and Adam White. Empirical design in reinforcement learning, 2023.

Belinda Phipson and Gordon K Smyth. Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn. *Statistical applications in genetics and molecular biology*, 9(1), 2010.

Edwin JG Pitman. Significance tests which may be applied to samples from any populations. *Supplement to the Journal of the Royal Statistical Society*, 4(1):119–130, 1937.

Stuart J Pocock. Group sequential methods in the design and analysis of clinical trials. *Biometrika*, 64(2): 191–199, 1977.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

Joseph P Romano. Bootstrap and randomization tests of some nonparametric hypotheses. *The Annals of Statistics*, 17(1):141–159, 1989.

Joseph P. Romano and Michael Wolf. Exact and approximate stepdown methods for multiple hypothesis testing. *SSRN Journal Electronic Journal*, 2003. doi: 10.2139/ssrn.563267. URL `https://doi.org/10.2139%2Fssrn.563267`.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Connie P Shapiro and Lawrence Hubert. Asymptotic normality of permutation statistics derived from weighted sums of bivariate functions. *The Annals of Statistics*, 7(4):788–794, 1979.

Jaehyeok Shin, Aaditya Ramdas, and Alessandro Rinaldo. Nonparametric iterated-logarithm extensions of the sequential generalized likelihood ratio test. *IEEE Journal on Selected Areas in Information Theory*, 2 (2):691–704, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

John Wilder Tukey. The problem of multiple comparisons. *Multiple comparisons*, 1953.

A. Wald. Sequential Tests of Statistical Hypotheses. *The Annals of Mathematical Statistics*, 16(2):117 – 186, 1945. doi: 10.1214/aoms/1177731118. URL `https://doi.org/10.1214/aoms/1177731118`.

A Index of notations

- N_{score}: the total number of scores used for one agent.
- N: the number of scores per interim.
- K: the total number of interims.
- k: the current interim number.
- L: the number of agents to compare.
- $A_1, A_2, \ldots, A_L$: the agents to compare.
- $\mathbb{E}[X]$: expectation of the random variable X.
- α: type I error or Family-wise error of a test.
- β: type II error of a test.
- $\mathbf{S}_n$: set of all the permutations of $\{1, \ldots, n\}$.
- σ: generic notation for a permutation, $\sigma \in \mathbf{S}_n$ for some $n \in \mathbb{N}^*$.
- id: an identity permutation, *i.e.* $\mathrm{id}(k) = k$ for all $1 \leq k \leq n$ for $\mathrm{id} \in \mathbf{S}_n$.
- B: the number of random permutations used to approximate the test statistics (see discussion *choice of permutations* in Section 4).
- $\sigma_{1:k}$: shorthand for the concatenation of permutations $\sigma_1, \ldots, \sigma_k$, *i.e.* $\sigma_{1:k}(e_{n,i}) = \sigma_i(e_{n,i})$ for all $1 \leq i \leq k$.
- H_j: denotes hypothesis j in a multiple test, H'_j denotes the alternative of hypothesis H_j.
- $e_i(j)$ or $e_{i,k}(j)$: score corresponding to run number i when doing the test for comparison $\mathbf{c}_j$. Index k denotes interim k when the scores are used sequentially (e.g. in AdaStop).
- $T_N(\sigma)$ and $T_{N,k}^{(j)}(\sigma)$: test statistics. See Equation (5) and Equation (7).
- $\mathbf{c}_j$: denotes a comparison. This is a couple (l_1, l_2) in $\{1, \ldots, L\}^2$.
- j: shorthand for denoting comparison c_j.
- $\mathbf{C}_0$: set of all the comparisons done in AdaStop.
- $\mathbf{C}$: current set of undecided comparisons in AdaStop, a subset of $\mathbf{C}_0$.
- $\mathbf{C}_k$: state of $\mathbf{C}$ at interim k in AdaStop.
- $B_{N,k}^{(\mathbf{C})}$: boundary for test statistics $T_{N,k}^{(\mathbf{C})}$, *s.t.* if $T_{N,k}^{(\mathbf{C})}(\mathrm{id}) > B_{N,k}^{(\mathbf{C})}$, then reject the set of hypotheses associated to $\mathbf{C}$.
- $\mathbf{I}$: set of true hypotheses and $\mathbf{I}^c$ its complement.
- FWE: family-wise error, see Definition 1.
- $\mathcal{N}(\mu, \sigma^2)$: law of a Gaussian probability distribution with mean μ and variance σ^2.
- $t(\mu, \nu)$: law of a translated Student probability distribution with center of symmetry μ and ν degrees of freedom.
- $\mathcal{M}_{\frac{1}{2}}^{\mathcal{N}}(\mu_1, \sigma_1^2; \mu_2, \sigma_2^2)$: mixture of the two normal probability distributions $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$.
- $\mathcal{M}_{\frac{1}{2}}^{t}(\mu_1, \nu_1; \mu_2, \nu_2)$: mixture of the two Student probability distributions $t(\mu_1, \nu_1)$ and $t(\mu_2, \nu_2)$.
- $\mathbb{P}_{H_j, j \in \mathbf{I}}$: probability distribution when $H_j, j \in \mathbf{I}$ are true and $H_j, j \notin \mathbf{I}$ are false.

B Recap on hypothesis testing

To be fully understood, this paper requires the knowledge of some notions of statistics. In the hope of widening the audience of this paper, we provide a short recap of essential notions of statistics related to hypothesis testing.

B.1 Type I and type II error

In its simplest form, a statistical test is aimed at deciding whether a given collection of data $X_1, \dots, X_N$ adheres to some hypothesis H_0 (called the null hypothesis), or if it is a better fit for the alternative hypothesis H_1. Typically, the null hypothesis states that the mean of the distribution from which the X_i's are sampled is equal to some μ_0: $H_0 : \mu = \mu_0$ and $H_1 : \mu \neq \mu_0$ where μ is the mean of the distribution of $X_1, \dots, X_N$. Because μ is unknown, it has to be estimated using the data. Often this is done using the empirical mean $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} X_i$. $\widehat{\mu}$ is a random variable and some deviation from μ is to be expected. The theory of hypothesis tests is concerned in finding a threshold c such that if $|\widehat{\mu} - \mu_0| > c$ then we say that H_0 is false because the deviation is greater than what was expected by the theory.

A slightly more complex problem is to consider two samples $X_1, \dots, X_N$ and $Y_1, \dots, Y_N$ and do a two-sample test deciding whether the mean of the distribution of the X_i's is equal to the mean of the distribution of the Y_i's.

In both cases, the result of a test is either accept H_0 or reject H_0. This answer is not a ground truth: there is some probability that we make an error. However, this probability of error is often controlled and can be decomposed in type I error and type II error (often denoted α and β respectively, see Table 2). The problem

	H_0 is true	H_0 is false
We accept H_0	No error	Type II error β
We reject H_0	Type I error α	No error

Table 2: Type I and type II error.

is not symmetric: failing to reject the null hypothesis does not mean that the null hypothesis is true. It can be that there is not enough data to reject H_0. It has been shown that a test cannot simultaneously minimize α and β. It is customary to minimize β with a given value of α (typically α is set to 0.05). The probability to reject H_0 when it is false is $1 - \beta$ and is usually called the *power* of a test.

B.2 Multiple tests and FWE

When performing simultaneously L statistical tests for $L > 1$, one must be careful that the error of each test accumulates. If one is not cautious, the overall error may become non-negligible. As a consequence, multiple strategies have been developed to deal with multiple testing problem.

To deal with the multiple testing problem, the first step is to define what is an error. There are several definitions of error in multiple testing among which is the False Discovery Rate which measures the expected proportion of false rejections. Another possible measure of error is the Family-Wise Error (the error we use in this article) which is defined as the probability to make at least one false rejection:

$$\text{FWE} = \mathbb{P}_{H_j, j \in \mathbf{I}} \left(\exists j \in \mathbf{I} : \quad \text{reject } H_j \right),$$

where $\mathbb{P}_{H_j, j \in \mathbf{I}}$ is used to denote a probability. $\mathbf{I} \subset \{1, \dots, L\}$ is the set of indices of the hypotheses that are true (and $\mathbf{I}^c$ the set of hypotheses that are actually false). To construct a procedure with an FWE smaller than α, the simplest method is perhaps Bonferroni's correction (Bonferroni, 1936) in which a statistical test is applied on each of the J couples of hypotheses to be tested ($J = L(L-1)/2$). And then, one would tune each hypothesis test to have a type I error α/J. The union bound then implies that the FWE is bounded by α:

$$\text{FWE} \ = \mathbb{P}_{H_j, j \in \mathbf{I}} \Big(\bigcup_{j \in \mathbf{I}} \{\text{reject } H_j\} \Big) \leq \sum_{i \in \mathbf{I}} \mathbb{P}_{H_j, j \in \mathbf{I}} \left(\text{reject } H_j \right) \leq |\mathbf{I}| \frac{\alpha}{J} \leq \alpha \,.$$

Bonferroni's correction has the advantage of being very simple to implement. However, it is often very conservative and the FWE is most often a lot smaller than α. An alternative method that performs well in practice is the step-down method that we use in this article and is presented in Section 3.2.4.

C Proof of Theorem 1

The proof of Theorem 1 is based on an extension of the proof of the control of FWE in the non-sequential case and the proof of the step-down method (Romano & Wolf, 2003). The interested reader may refer to Lemma 1, in the Appendix C.1, where we reproduce the proof of the bound on FWE for simple permutation tests as it is a good introduction to permutation tests. The proof proceeds as follows: first, we prove weak control[5] on the FWE by decomposing the error as the sum of the errors on each interim and using the properties of permutation tests to show that the error done at each interim is controlled by α/K. Then, using the step-down method construction, we show that the strong control of the FWE is a consequence of the weak control because of monotony properties on the boundary values of a permutation test.

C.1 Simplified proof for $L = 2$ agents, and $K = 1$

The proof of this theorem is a bit technical. We begin by showing it in a very simplified case where $L = 2$ agents, and $K = 1$.

Lemma 1. *Let $X_1, \dots, X_N$ be i.i.d. from a distribution P and $Y_1, \dots, Y_N$ be i.i.d. from a distribution Q. Let $Z_1^{2N} = X_1, \dots, X_N, Y_1, \dots, Y_N$ be the concatenation of X_1^N and Y_1^N. Let $\alpha \in (0, 1)$ and define B_N as:*

$$B_N = \inf \left\{ b > 0 : \quad \frac{1}{(2N)!} \sum_{\sigma \in \boldsymbol{S}_{2N}} \mathbb{1} \left\{ \frac{1}{N} \sum_{i=1}^{N} (Z_{\sigma(i)} - Z_{\sigma(N+i)}) > b \right\} \leq \alpha \right\}.$$

Then, if $P = Q$, we have:

$$\mathbb{P} \left(\frac{1}{N} \sum_{i=1}^{N} (X_i - Y_i) > B_N \right) \leq \alpha.$$

Proof. Let us denote $T(\sigma) = \frac{1}{N} \sum_{i=1}^{N} (Z_{\sigma(i)} - Z_{\sigma(n+i)})$. Since $P = Q$, for any $\sigma, \sigma' \in \mathbf{S}_{2N}$ we have $T(\sigma) \overset{d}{=} T(\sigma')$. Then, because B_N does not depend on the permutation σ (but it depends on the values of Z_1^{2N}), we have, for any $\sigma \in \mathbf{S}_{2N}$

$$\mathbb{P}\left(T(\mathrm{id}) > B_N\right) = \mathbb{P}\left(T(\sigma) > B_N\right).$$

Now, we sum all the permutations:

$$\begin{aligned} \mathbb{P}\left(T(\mathrm{id}) > B_N\right) &= \frac{1}{(2N)!} \sum_{\sigma \in \mathbf{S}_{2N}} \mathbb{E}\left[\mathbb{1}\{T(\sigma) > B_N\}\right] \\ &= \mathbb{E}\left[\frac{1}{(2N)!} \sum_{\sigma \in \mathbf{S}_{2N}} \mathbb{1}\{T(\sigma) > B_N\} \right] \leq \alpha, \end{aligned}$$

which proves the result. □

Next, we prove weak control in the general case.

C.2 Proof of Theorem 1

In this section, we use the shorthand $\mathbb{P}$ instead of $\mathbb{P}_{H_j, j \in \mathbf{I}}$ and omit $H_{j, j \in \mathbf{I}}$ because $\mathbf{I}$ will always be the set of true hypotheses and the meaning should be clear from the context.

[5] A procedure controls has a weak control on the FWE if the FWE is smaller than α when all null hypotheses are true. On the other hand, strong control is when FWE is smaller than α whatever the set of true null hypotheses.

Weak control on FWE: First, we prove weak control on the FWE. This means that we suppose that all the hypotheses are true ($\mathbf{I} = \{1, \dots, J\}$), and we control the probability to make at least one rejection. We have:

$$\text{FWE} = \mathbb{P}\left(\exists j \in \mathbf{I} : \quad H_j \text{ is rejected}\right).$$

We decompose the FWE on the set of interims, using the fact that a rejection happens if and only if we reject a true hypothesis for the first time at interim k for some $k = 1, \dots, K$. These events are mutually exclusive therefore we have that their probabilities sum up and we can rewrite the FWE as:

$$\text{FWE} = \sum_{k=1}^{K} \mathbb{P}\left(\overline{T}_{N,k}^{(\mathbf{I})}(\text{id}) > B_{N,k}^{(\mathbf{I})}, \text{NR}_k(\text{id})\right), \tag{11}$$

where $\text{NR}_k(\text{id})$ is the event on which we did *Not Reject* (NR) before interim k. $\text{NR}_k(\text{id})$ can be defined directly for a concatenation of permutations $\sigma_{1:k}$ as $\text{NR}_k(\sigma_{1:k}) = \{\forall m < k, \overline{T}_{N,m}^{(\mathbf{I})}(\sigma_{1:k}) \leq B_{N,m}^{(\mathbf{I})}\}$. We introduced the definition for $\sigma_{1:k}$ and not only for id since this will be useful later on, for example in Equation (12).

Then, similarly as in the proof of Lemma 1, we want to use the invariance by permutation to make the link with the definition of $B_{N,k}^{(\mathbf{I})}$. For this purpose, we introduce the following lemma, that we prove in Appendix D.

Lemma 2. *For $k \leq K$, for any $\sigma_{1:k}$ concatenation of k permutations:*

$$(\overline{T}_{N,l}^{(\mathbf{I})}(\text{id}), B_{N,l}^{(\mathbf{I})})_{l \leq k} \overset{d}{=} (\overline{T}_{N,l}^{(\mathbf{I})}(\sigma_{1:l}), B_{N,l}^{(\mathbf{I})})_{l \leq k}.$$

Using Lemma 2, we have for any interim k and concatenation of permutations $\sigma_{1:k}$

$$\mathbb{P}\left(\overline{T}_{N,k}^{(\mathbf{I})}(\text{id}) > B_{N,k}^{(\mathbf{I})}, \text{NR}_k(\text{id})\right) = \mathbb{P}\left(\overline{T}_{N,k}^{(\mathbf{I})}(\sigma_{1:k}) > B_{N,k}^{(\mathbf{I})}, \text{NR}_k(\sigma_{1:k})\right). \tag{12}$$

Combing this equation with equation (11), we have:

$$\begin{aligned}
\text{FWE} &\leq \sum_{k=1}^{K} \frac{1}{m_k} \sum_{\sigma_{1:k} \in \mathcal{S}_{N,k}} \mathbb{P}\left(\overline{T}_{N,k}^{(\mathbf{I})}(\sigma_{1:k}) > B_{N,k}^{(\mathbf{I})}, \text{NR}_k(\sigma_{1:k})\right) \\
&= \sum_{k=1}^{K} \mathbb{E}\left[\frac{1}{m_k} \sum_{\sigma_{1:k} \in \mathcal{S}_{N,k}} \mathbb{1}\left\{\overline{T}_{N,k}^{(\mathbf{I})}(\sigma_{1:k}) > B_{N,k}^{(\mathbf{I})}, \text{NR}_k(\sigma_{1:k})\right\}\right].
\end{aligned}$$

Then, use that $\sigma_{1:k} \in \widehat{\mathcal{S}}_{N,k}$ if and only if $\sigma_{1:k} \in \mathcal{S}_{N,k}$ and $\text{NR}_k(\sigma_{1:k})$ is true. Hence,

$$\text{FWE} \leq \sum_{k=1}^{K} \mathbb{E}\left[\frac{1}{m_k} \sum_{\sigma_{1:k} \in \widehat{\mathcal{S}}_{N,k}} \mathbb{1}\left\{\overline{T}_{N,k}^{(\mathbf{I})}(\sigma_{1:k}) > B_{N,k}^{(\mathbf{I})}\right\}\right] \leq \sum_{k=1}^{K} q_k \leq \alpha,$$

where we use the definition of $B_{N,k}^{(\mathbf{I})}$ to make the link with α

Strong control of FWE: To prove strong control, it is sufficient to show the following Lemma (see Appendix D for a proof), which is an adaptation of the proof of step-down multiple-test strong control of FWE from (Romano & Wolf, 2003).

Lemma 3. *Suppose that $\mathbf{I} \subset \{1, \dots, J\}$ is the set of true hypotheses. We have:*

$$\text{FWE} = \mathbb{P}\left(\exists j \in \mathbf{I} : \; H_j \textit{ is rejected}\right) \leq \mathbb{P}\left(\exists k \leq K : \quad \overline{T}_{N,k}^{(\mathbf{I})}(\text{id}) > B_{N,k}^{(\mathbf{I})}\right).$$

Lemma 3 shows that to control the FWE, it is sufficient to control the probability to reject on $\mathbf{I}$ given by $\mathbb{P}\left(\exists k \leq K : \quad \overline{T}_{N,k}^{(\mathbf{I})}(\sigma_{1:k}) > B_{N,k}^{(\mathbf{I})}\right)$ and this quantity, in turn, is exactly the FWE of the restricted problem of testing $(H_j)_{j \in \mathbf{I}}$ against $(H'_j)_{j \in \mathbf{I}}$. In other words, Lemma 3 says that to prove strong FWE control for AdaStop, it is sufficient to prove weak FWE control, and we already did that in the first part of the proof.

D Proof of Lemmas 2 and 3

D.1 Proof of Lemma 2

In this section, for an easier understanding, we change the notation for the score $e^{(j)}_{n,k}(\sigma)$ to $e_{n,k}(A_l)$ the nth score of agent A_l at interim k. In other words, we change the notation of the comparison of agent A_{l_1} versus agent A_{l_2}: in the main text a comparison was denoted by $\mathbf{c}_j \in \{1, \dots, J\}^2$ but here we make explicit the agent from which the score has been computed resulting in the following equalities: $e_{n,k}(A_{l_1}) = e^{(j)}_{n,k}(\text{id})$ and $e_{n,k}(A_{l_2}) = e^{(j)}_{N+n,k}(\text{id})$ for $n \leq N$.

We denote the comparisons by $(\mathbf{c}_i)_{i\in\mathbf{I}}$. The set of comparisons can be represented as a graph in which each node represents one of the agents to compare, and there exists an edge from j_1 to j_2 denoted (j_1, j_2) if $(j_1, j_2) \in (\mathbf{c}_i)_{i\in\mathbf{I}}$ is one of the comparisons that corresponds to a true hypothesis. This graph is not necessarily connected. We denote $C(i)$ the connected component to which node l (e.g. agent l) belongs, i.e. for any $l_1, l_2 \in C(l)$ there exists a path going from l_1 to l_2. $C(l)$ cannot be equal to the singleton $\{l\}$, because this would mean that all the comparisons with l are in fact false hypotheses, and then l would not belong to a couple in $\mathbf{I}$.

Then, it follows from the construction of permutation test that jointly on $k \leq K$ and $\mathbf{c}_j = (l_1, l_2) \in C(l)$, we have $T^{(j)}_{N,k}(\text{id}) \overset{d}{=} T^{(j)}_{N,1}(\sigma_{1:k})$ for any $\sigma_1, \dots, \sigma_k \in \mathbf{S}_{2N}$.

Let us illustrate that on an example. Suppose that $N = 2$ and $J = 3$ so that the comparison are (A_1, A_2), (A_1, A_3), (A_2, A_3). Consider the permutation

$$\sigma_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 2 & 4 \end{pmatrix}$$

Because all the scores are i.i.d., we have the joint equality in distribution:

$$\begin{pmatrix} |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_2) - e_{2,1}(A_2)| \\ |e_{1,1}(A_3) + e_{2,1}(A_3) - e_{1,1}(A_2) - e_{2,1}(A_2)| \\ |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_3) - e_{2,1}(A_3)| \end{pmatrix} \overset{d}{=} \begin{pmatrix} |e_{2,1}(A_1) + e_{1,1}(A_2) - e_{1,1}(A_1) - e_{2,1}(A_2)| \\ |e_{2,1}(A_3) + e_{1,1}(A_2) - e_{1,1}(A_3) - e_{2,1}(A_2)| \\ |e_{2,1}(A_1) + e_{1,1}(A_3) - e_{1,1}(A_1) - e_{2,1}(A_3)| \end{pmatrix}$$

and hence,

$$(T^{(j)}_{N,1}(\text{id}))_{1\leq j\leq 3} \overset{d}{=} (T^{(j)}_{N,1}(\sigma_1))_{1\leq j\leq 3}.$$

For $k = 2$, we have for $\sigma_2 = \sigma_1$,

$$\begin{pmatrix} |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_2) - e_{2,1}(A_2)| \\ |e_{1,1}(A_3) + e_{2,1}(A_3) - e_{1,1}(A_2) - e_{2,1}(A_2)| \\ |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_3) - e_{2,1}(A_3)| \\ |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_2) - e_{2,1}(A_2) + e_{1,2}(A_1) + e_{2,2}(A_1) - e_{1,2}(A_2) - e_{2,2}(A_2)| \\ |e_{1,1}(A_3) + e_{2,1}(A_3) - e_{1,1}(A_2) - e_{2,1}(A_2) + e_{1,2}(A_3) + e_{2,2}(A_3) - e_{1,2}(A_2) - e_{2,2}(A_2)| \\ |e_{1,1}(A_1) + e_{2,1}(A_1) - e_{1,1}(A_3) - e_{2,1}(A_3) + e_{1,2}(A_1) + e_{2,2}(A_1) - e_{1,2}(A_3) - e_{2,2}(A_3)| \end{pmatrix}$$

$$\overset{d}{=} \begin{pmatrix} |e_{1,1}(A_1) + e_{2,1}(A_2) - e_{1,1}(A_1) - e_{2,1}(A_2)| \\ |e_{1,1}(A_3) + e_{2,1}(A_2) - e_{1,1}(A_3) - e_{2,1}(A_2)| \\ |e_{1,1}(A_1) + e_{2,1}(A_3) - e_{1,1}(A_1) - e_{2,1}(A_3)| \\ |e_{1,1}(A_1) + e_{2,1}(A_2) - e_{1,1}(A_1) - e_{2,1}(A_2) + e_{1,2}(A_1) + e_{2,2}(A_2) - e_{1,2}(A_1) - e_{2,2}(A_2)| \\ |e_{1,1}(A_3) + e_{2,1}(A_2) - e_{1,1}(A_3) - e_{2,1}(A_2) + e_{1,2}(A_3) + e_{2,2}(A_2) - e_{1,2}(A_3) - e_{2,2}(A_2)| \\ |e_{1,1}(A_1) + e_{2,1}(A_3) - e_{1,1}(A_1) - e_{2,1}(A_3) + e_{1,2}(A_1) + e_{2,2}(A_3) - e_{1,2}(A_1) - e_{2,2}(A_3)| \end{pmatrix}$$

and then, we get jointly

$$(T^{(j)}_{N,k}(\text{id}))_{1\leq j\leq 3, k\leq 2} \overset{d}{=} (T^{(j)}_{N,k}(\sigma_1 \cdot \sigma_2))_{1\leq j\leq 3, k\leq 2}.$$

This reasoning can be generalized to any N, J and K:

$$(T^{(j)}_{N,k}(\text{id}))_{k\leq K, \mathbf{c}_j\in C(l)^2} \overset{d}{=} (T^{(j)}_{N,k}(\sigma_{1:k}))_{k\leq K, \mathbf{c}_j\in C(l)^2}.$$

Then, by construction, the different connected component $C(l)$ are independent of one another and hence,

$$(T_{N,k}^{(j)}(\mathrm{id}))_{k\leq K, j\in\mathbf{I}} \overset{d}{=} (T_{N,k}^{(j)}(\sigma_{1:k}))_{k\leq K, j\in\mathbf{I}}.$$

Lemma 2 follows from taking the maximum on all the comparisons and because the boundaries do not depend on the permutation.

D.2 Proof of Lemma 3

Denote by $\mathbf{C}_k$ the (random) value of $\mathbf{C}$ at the beginning of interim k. We have:

$$\begin{aligned}\mathrm{FWE} &= \mathbb{P}\left(\exists j \in \mathbf{I}: \quad H_j \text{ is rejected}\right)\\ &= \mathbb{P}\left(\exists k \leq K: \quad \overline{T}_{N,k}^{(\mathbf{C}_k)}(\mathrm{id}) > B_{N,k}^{(\mathbf{C}_k)}, \underset{j,\mathbf{c}_j\in\mathbf{C}_k}{\arg\max}\, \overline{T}_{N,k}^{(j)}(\mathrm{id}) \in \mathbf{I}\right). \end{aligned} \tag{13}$$

Then, let k_0 correspond to the very first rejection (if any) in the algorithm. Having that the argmax is attained in $\mathbf{I}$,

$$\overline{T}_{N,k_0}^{(\mathbf{C}_{k_0})}(\mathrm{id}) = \max\{T_{N,k_0}^{(j)}(\mathrm{id}), \mathbf{c}_j \in \mathbf{C}_{k_0}\} = \max\{T_{N,k_0}^{(j)}(\mathrm{id}), j \in \mathbf{I}\} = \overline{T}_{N,k_0}^{(\mathbf{I})}(\mathrm{id})$$

Moreover, having that the comparisons indexed by $\mathbf{I}$ are included into $\mathbf{C}_{k_0}$, we have $B_{N,k_0}^{(\mathbf{C}_{k_0})} \geq B_{N,k_0}^{(\mathbf{I})}$. Injecting these two relations in Equation (13), we obtain:

$$\begin{aligned}\mathrm{FWE} &\leq \mathbb{P}\left(\exists k \leq K: \quad \overline{T}_{N,k}^{(\mathbf{I})}(\mathrm{id}) > B_{N,k}^{(\mathbf{I})}, \underset{j\in\mathbf{C}_k}{\arg\max}\, \overline{T}_{N,k}^{(j)}(\mathrm{id}) \in \mathbf{I}\right)\\ &\leq \mathbb{P}\left(\exists k \leq K: \quad \overline{T}_{N,k}^{(\mathbf{I})}(\mathrm{id}) > B_{N,k}^{(\mathbf{I})}\right).\end{aligned}$$

This proves the desired result.

E Asymptotic results for two agents

E.1 Convergence of boundaries and comparing the means

Because there are only two agents and no early stopping, we simplify the notations and denote

$$t_{N,i}(\sigma_i) = \sum_{n=1}^{N} e_{\sigma_i(n),i}(2) - \sum_{n=N+1}^{2N} e_{\sigma_i(n),i}(1),$$

and

$$\begin{aligned}T_{N,k}(\sigma_{1:k}) &= \left|\sum_{i=1}^{k}\left(\sum_{n=1}^{N} e_{\sigma_i(n),i}(2) - \sum_{n=N+1}^{2N} e_{\sigma_i(n),i}(1)\right)\right|\\ &= \left|\sum_{i=1}^{k} t_{N,i}(\sigma_i)\right|,\end{aligned}$$

and

$$B_{N,k} = \inf\left\{b > 0 : \frac{1}{((2N)!)^k} \sum_{\sigma_1,\ldots,\sigma_k\in\mathbf{S}_{2N}^k} \mathbb{1}\{T_{N,k}(\sigma_{1:k}) \geq b\} \leq q_k\right\}.$$

When there is only one interim ($K = 1$), we have the following convergence of the randomization law of $T_{N,1}(\sigma)$.

Proposition 1 (Theorem 17.3.1 in (Lehmann et al., 2005))**.** *Suppose $e_{1,1}(1), \dots, e_{N,1}(1)$ are i.i.d. from P and $e_{1,1}(2), \dots, e_{N,1}(2)$ are i.i.d. from Q and both P and Q have finite variance. Then, we have:*

$$\sup_t \left| \frac{1}{(2N)!} \sum_{\sigma \in \boldsymbol{S}_{2N}} \mathbb{1}\left\{ \frac{1}{\sqrt{N}} T_{N,1}(\sigma) \leq t \right\} - \Phi\left(t/\tau(P,Q)\right) \right| \xrightarrow[N\to\infty]{P} 0$$

where Φ is the normal c.d.f. and $\tau(P,Q)^2 = \sigma_P^2 + \sigma_Q^2 + \frac{(\mu_P - \mu_Q)^2}{2}$.

Using the non-sequential result from proposition 1, we can show the following theorem that controls the asymptotic law of the sequential test.

Theorem 3. *Suppose that P and Q both have a finite second moment. Then, for any $1 \leq k \leq K$, $\frac{1}{\sqrt{N}} B_{N,k} \xrightarrow[N\to\infty]{} b_k$ where the real numbers b_k are defined as follows. Let $W_1, \dots, W_K$ be i.i.d. random variables with law $\mathcal{N}(0,1)$. Then b_1 is the solution of the following equation:*

$$\mathbb{P}\left(|W_1| \geq \frac{b_1}{\tau(P,Q)} \right) = \frac{\alpha}{K},$$

and for any $1 < k \leq K$, b_k is the solution of

$$\mathbb{P}\left(\left| \frac{1}{k} \sum_{j=1}^{k} W_j \right| > \frac{b_l}{\tau(P,Q)}, \quad \forall j < k, \left| \frac{1}{j} \sum_{i=1}^{j} W_i \right| \leq \frac{b_j}{\tau(P,Q)} \right) = \frac{\alpha}{K}.$$

The conditions on Theorem 3 are very weak: finite second moment to be able to use the central limit theorem.

The test performed in AdaStop when comparing two agents corresponds to testing

$$\mathbb{1}\left\{ \exists k \leq K : \frac{1}{\sqrt{N}} T_{N,k}(\mathrm{id}) > \frac{1}{\sqrt{N}} B_{N,k} \right\}$$

and from Theorem 3 and the central-limit theorem, $\frac{1}{\sqrt{N}} T_{N,k}(\mathrm{id})$ converges to $\sum_{j=1}^{k} W_j \sqrt{\sigma_P^2 + \sigma_Q^2}$, and $B_{N,k}/\sqrt{N}$ converges to b_k. Hence the test is asymptotically equivalent to:

$$\mathbb{1}\left\{ \exists k \leq K : \sum_{j=1}^{k} W_j \sqrt{\sigma_P^2 + \sigma_Q^2} > b_k \right\}.$$

Then, in the case in which $\mu_P = \mu_Q$, we have $\tau(P,Q) = \sqrt{\sigma_P^2 + \sigma_Q^2}$ and

$$\begin{aligned}
\mathrm{FWE} &= \mathbb{P}\left(\exists k \leq K : \sum_{j=1}^{k} W_j \sqrt{\sigma_P^2 + \sigma_Q^2} > b_k \right) \\
&= \sum_{k=1}^{K} \mathbb{P}\left(\left| \frac{1}{k} \sum_{j=1}^{k} W_j \right| > \frac{b_l}{\tau(P,Q)}, \quad \forall j < k, \left| \frac{1}{j} \sum_{i=1}^{j} W_i \right| \leq \frac{b_j}{\tau(P,Q)} \right) \\
&= \sum_{k=1}^{K} \frac{\alpha}{K} = \alpha.
\end{aligned}$$

Hence, for the test $H_0: \quad \mu_P = \mu_Q$ versus $H_1: \quad \mu_P \neq \mu_Q$, our test is asymptotically of level α.

E.2 Proof of Theorem 3

For $x \in \mathbb{R}$, we denote:

$$R_{N,k}(x) = \frac{1}{(2N)!} \sum_{\sigma_k \in \mathbf{S}_{2N}} \mathbb{1}\{t_{N,k}(\sigma_k) \leq x\}.$$

$R_{N,k}$ is the c.d.f. of the randomization law of $t_{N,k}(\sigma_k)$, and by Proposition 1, it converges uniformly to a Gaussian c.d.f when N goes to infinity.

Convergence of $B_{N,1}$ By definition of the boundary, we have

$$\frac{1}{\sqrt{N}}B_{N,1} = \frac{1}{\sqrt{N}}\min\left\{b>0 : \frac{1}{(2N)!}\sum_{\sigma_1\in\mathbf{S}_{2N}}\mathbb{1}\{|T_{N,1}(\sigma_1)|>b\}\leq\frac{\alpha}{K}\right\}.$$

This implies

$$\begin{aligned}\frac{1}{(2N)!}\sum_{\sigma_1\in\mathbf{S}_{2N}}\mathbb{1}\{|T_{N,1}(\sigma_1)|\leq B_{N,1}\} &= \widehat{R}_{N,1}\left(\frac{1}{\sqrt{N}}B_{N,1}\right)-\widehat{R}_{N,1}\left(-\frac{1}{\sqrt{N}}B_{N,1}\right)\\ &\geq 1-\frac{\alpha}{K}\end{aligned}$$

and for any $b<B_{N,1}$, we have:

$$\frac{1}{(2N)!}\sum_{\sigma_1\in\mathbf{S}_{2N}}\mathbb{1}\{|T_{N,1}(\sigma_1)|\leq b\} = \widehat{R}_{N,1}\left(\frac{b}{\sqrt{N}}\right)-\widehat{R}_{N,1}\left(-\frac{b}{\sqrt{N}}\right)<1-\frac{\alpha}{K}.$$

Then,

$$\begin{aligned}&\Phi\left(\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)-\Phi\left(-\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)\\ \geq&\widehat{R}_{N,1}\left(\frac{B_{N,1}}{\sqrt{N}}\right)-\widehat{R}_{N,1}\left(-\frac{B_{N,1}}{\sqrt{N}}\right)-\left|\Phi\left(\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)-\widehat{R}_{N,1}\left(\frac{B_{N,1}}{\sqrt{N}}\right)\right|\\ &-\left|\Phi\left(-\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)-\widehat{R}_{N,1}\left(-\frac{B_{N,1}}{\sqrt{N}}\right)\right|\\ \geq&1-\frac{\alpha}{K}-2\sup_t\left|\Phi\left(\frac{t}{\tau(P,Q)}\right)-\widehat{R}_{N,1}(t)\right|.\end{aligned}$$

Hence, by taking N to infinity, we have from Proposition 1:

$$\liminf_{N\to\infty}\Phi\left(\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)-\Phi\left(-\frac{B_{N,1}}{\tau(P,Q)\sqrt{N}}\right)\geq 1-\frac{\alpha}{K}.$$

And for any $\varepsilon>0$, because of the definition of $B_{N,1}$ as a supremum, we have:

$$\limsup_{N\to\infty}\Phi\left(\frac{B_{N,1}+\varepsilon}{\tau(P,Q)\sqrt{N}}\right)-\Phi\left(-\frac{B_{N,1}+\varepsilon}{\tau(P,Q)\sqrt{N}}\right)<1-\frac{\alpha}{K}.$$

By continuity of Φ, this implies that $\frac{1}{\sqrt{N}}B_{N,1}$ converges almost surely and its limit is such that:

$$\Phi\left(\frac{\lim_{N\to\infty}B_{N,1}/\sqrt{N}}{\tau(P,Q)}\right)-\Phi\left(-\frac{\lim_{N\to\infty}B_{N,1}/\sqrt{N}}{\tau(P,Q)}\right)=1-\frac{\alpha}{K}.$$

Or said differently, let $W\sim\mathcal{N}(0,1)$, then we have the almost sure convergence $\lim_{N\to\infty}\frac{1}{\sqrt{N}}B_{N,1}=b_1$ where b_1 is the real number defined by

$$\mathbb{P}\left(|W|\geq\frac{b_1}{\tau(P,Q)}\right)=\frac{\alpha}{K}.$$

Convergence of $B_{N,k}$ for $k>1$. We proceed by induction. Suppose that $\frac{1}{\sqrt{N}}B_{N,k-1}$ converges to some $b_{k-1}>0$ and that for any $d_1,\ldots,d_{k-1}$, the randomization probability, it holds that:

$$\begin{aligned}\sup_{d_1,\ldots,d_{k-1}}\Bigg|\frac{1}{((2N)!)^{k-1}}\sum_{\sigma_1,\ldots,\sigma_{k-1}\in\mathbf{S}_{2N}}\mathbb{1}\left\{\forall j\leq k-1,\sum_{i=1}^{j}t_{N,i}(\sigma_i)\leq d_j\sqrt{N}\right\}\\ -\mathbb{P}\left(\forall j\leq k-1,\sum_{i=1}^{j}W_i\leq\frac{d_j}{\tau(P,Q)}\right)\Bigg|\xrightarrow[a.s.]{N\to\infty}0,\end{aligned}\quad(14)$$

where $W_1, \ldots, W_{k-1}$ are i.i.d. $\mathcal{N}(0,1)$ random variables. In other words, the randomization law converges uniformly to the joint law described above with the sum of Gaussian random variables.

Then, by uniform convergence and by convergence of the $B_{N,j}$, we have that:

$$\frac{1}{((2N)!)^{k-1}} \sum_{\sigma_1,\ldots,\sigma_{k-1}\in\mathbf{S}_{2N}} \mathbb{1}\left\{T_{N,j}\left(\sigma_{1:j}\right) > B_{N,k-1}, \quad \forall j < k-1,\, T_{N,j}\left(\sigma_{1:j}\right) \leq B_{N,j}\right\}$$

converges to

$$\mathbb{P}\left(\left|\sum_{i=1}^{l} W_i\right| > \frac{b_l}{\tau(P,Q)}, \quad \forall j < l,\, \left|\sum_{i=1}^{j} W_i\right| \leq \frac{b_j}{\tau(P,Q)}\right) = \frac{\alpha}{K}. \tag{15}$$

the last equality follows by construction of $B_{N,j}$ for $j < k$.

We have:

$$B_{N,k} = \min\left\{b > 0 : \frac{1}{((2N)!^k)} \sum_{\sigma_1,\ldots,\sigma_k\in\mathbf{S}_{2N}} \mathbb{1}\left\{\begin{matrix}\left|\sum_{j=0}^{k} t_{N,j}(\sigma_j)\right| \geq b, \\ \forall j<k,\, \left|\sum_{i=0}^{j} t_{N,i}(\sigma_i)\right| \leq B_{N,j}\end{matrix}\right\} + \sum_{i=1}^{k-1} q_i \leq \frac{k\alpha}{K}\right\}.$$

By the induction hypothesis, we have $q_i \xrightarrow[n\to\infty]{} \alpha/K$ for any $i < k$.

Let $W_1, \ldots, W_k$ be i.i.d. $\mathcal{N}(0,1)$ random variables. We show the following lemma that proves part of the step k of the induction hypothesis, proved in Section E.3.

Lemma 4. *Suppose Equation* (15) *is true. Then,*

$$\sup_{d_1,\ldots,d_k} \left| \frac{1}{((2N)!)^k} \sum_{\sigma_1,\ldots,\sigma_k\in\boldsymbol{S}_{2N}} \mathbb{1}\left\{\forall j \leq k,\, \sum_{i=1}^{j} t_{N,i}\left(\sigma_i\right) \leq d_j\sqrt{N}\right\} - \mathbb{P}\left(\forall j \leq k,\, \sum_{i=1}^{j} W_i \leq \frac{d_j}{\tau(P,Q)}\right)\right| \xrightarrow[a.s.]{N\to\infty} 0,$$

Then, what remains is to prove the convergence of $B_{N,k}$. Let us denote:

$$\Psi_k(d_k) = \mathbb{P}\left(\left|\sum_{i=1}^{k} W_i\right| > \frac{d_k}{\tau(P,Q)}, \forall j \leq k-1,\, \left|\sum_{i=1}^{j} W_i\right| \leq \frac{b_j}{\tau(P,Q)}\right).$$

From Lemma 4, we have that

$$\left|\Psi_k\left(\frac{B_{N,k}}{\sqrt{N}}\right) - \frac{1}{((2N)!)^k} \sum_{\sigma_1,\ldots,\sigma_k\in\mathbf{S}_{2N}} \mathbb{1}\left\{T_{N,k}\left(\sigma_{1:k}\right) > B_{N,k}, \quad \forall j < k,\, T_{N,j}\left(\sigma_{1:j}\right) \leq B_{N,j}\right\}\right|$$

converges to 0 as N goes to infinity. Hence,

$$\limsup_{N\to\infty} \Psi_k\left(\frac{B_{N,k}}{\sqrt{N}}\right) \leq \alpha/K.$$

Then, similarly to the case $k = 1$, we also have for any $\varepsilon > 0$:

$$\liminf_{N\to\infty} \Psi_k\left(\frac{B_{N,k} - \varepsilon}{\sqrt{N}}\right) \geq \alpha/K$$

and by continuity of Ψ_k (which is a consequence of the continuity of the joint c.d.f. of Gaussian random variables) we conclude that $B_{N,k}/\sqrt{N}$ converges almost surely to b_k.

E.3 Proof of Lemma 4

In this proof, let $E_{\sigma_{1:k}}(x)$ denote the expectation of the randomization law defined for some function $f : \mathbf{S}_{2N}^k \to \mathbb{R}$ by:

$$E_{\sigma_{1:k}}[f(\sigma_{1:k})] = \frac{1}{((2N)!)^k} \sum_{\sigma_1,\ldots,\sigma_k \in \mathbf{S}_{2N}} f(\sigma_{1:k}).$$

Please note that this expectation is still random and should be differentiated from the usual expectation $\mathbb{E}$.

First, let us first handle the convergence of step k. We have:

$$\begin{aligned}
&\frac{1}{(2N)!} \sum_{\sigma_k \in \mathbf{S}_{2N}} \mathbb{1}\left\{ \sum_{j=1}^{k} t_{N,j}\left(\sigma_j\right) \leq d_k \sqrt{N} \right\} \\
&\quad = \frac{1}{(2N)!} \sum_{\sigma_k \in \mathbf{S}_{2N}} \mathbb{1}\left\{ \frac{1}{\sqrt{N}} t_{N,k}\left(\sigma_k\right) \leq d_k - \frac{1}{\sqrt{N}} \sum_{j=1}^{k-1} t_{N,j}\left(\sigma_j\right) \right\} \\
&\quad = \widehat{R}_{n,k}\left(d_k - \frac{1}{\sqrt{N}} \sum_{j=1}^{k-1} t_{N,j}\left(\sigma_j\right) \right).
\end{aligned}$$

Because the convergence in proposition 1 is uniform, we have:

$$\begin{aligned}
&\left| \widehat{R}_{n,k}\left(d_k - \frac{1}{\sqrt{N}} \sum_{j=1}^{k-1} t_{N,j}\left(\sigma_j\right) \right) - \Phi\left(\frac{1}{\tau(P,Q)} \left(d_k - \frac{1}{\sqrt{N}} \sum_{j=1}^{k-1} t_{N,j}\left(\sigma_j\right) \right) \right) \right| \\
&\leq \sup_t \left| \widehat{R}_n(t) - \Phi\left(\frac{t}{\tau(P,Q)} \right) \right| \xrightarrow[n\to\infty]{} 0.
\end{aligned}$$

Then, using this convergence we have that, when N goes to infinity,

$$E_{\sigma_{1:k}}\left[\mathbb{1}\left\{ \forall j < k, \sum_{i=1}^{j} t_{N,i}\left(\sigma_i\right) \leq d_j \sqrt{N} \right\} \right]$$

converges uniformly on $d_1, \ldots, d_k$ to:

$$\begin{aligned}
&E_{\sigma_{1:k}}\left[\mathbb{1}\left\{ \forall j < k-1, \sum_{i=1}^{j} t_{N,i}\left(\sigma_i\right) \leq d_j \sqrt{N} \right\} \mathbb{P}\left(W_k \leq \frac{1}{\tau(P,Q)} \left(d_k - \frac{1}{\sqrt{N}} \sum_{i=1}^{k-1} t_{N,i}\left(\sigma_i\right) \right) \right) \right] \\
&= \mathbb{E}\left[E_{\sigma_{1:k-1}}\left[\mathbb{1}\left\{ \begin{array}{c} \forall j<k-2, \sum_{i=1}^{j} t_{N,i}(\sigma_i) \leq d_j\sqrt{N}, \\ \frac{1}{\sqrt{N}} \sum_{i=1}^{k-1} t_{N,j}(\sigma_i) \leq \min(d_k - \tau(P,Q) W_k, d_{k-1}) \end{array} \right\} \right] \right].
\end{aligned} \tag{16}$$

Then, using the induction hypothesis, Equation (16) converges to Equation (14), hence in the limit we have

$$\begin{aligned}
\mathbb{E}\left[\mathbb{1}\left\{ \begin{array}{c} \forall j<k-2, \sum_{i=1}^{j} W_i \leq \frac{d_j}{\tau(P,Q)}, \\ \frac{1}{\sqrt{N}} \sum_{i=1}^{k-1} W_i \leq \frac{1}{\tau(P,Q)} \min(d_k - W_k, d_{k-1}) \end{array} \right\} \right] &= \mathbb{E}\left[\mathbb{1}\left\{ \forall j < k, \sum_{i=1}^{j} W_i \leq \frac{d_j}{\tau(P,Q)} \right\} \right] \\
&= \mathbb{P}\left(\forall j \leq k, \sum_{i=1}^{j} W_i \leq \frac{d_j}{\tau(P,Q)} \right).
\end{aligned}$$

F On early accept in AdaStop

Let $\mathbf{C} \subset \{\mathbf{c}_1, \ldots, \mathbf{c}_J\}$ be a subset of the set of comparisons that we want to do. Let us denote:

$$\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_1^k) := \max\left(T_{N,k}^{(j)}(\sigma_1^k), \quad \mathbf{c}_j \in \mathbf{C} \right) \quad \text{and} \quad \underline{T}_{N,k}^{(\mathbf{C})}(\sigma_1^k) := \min\left(T_{N,k}^{(j)}(\sigma_1^k), \quad \mathbf{c}_j \in \mathbf{C} \right),$$

and

$$\overline{B}_{N,k}^{(\mathbf{C})} := \inf\left\{ b > 0 : \frac{1}{m_k} \sum_{\sigma \in \widehat{\mathcal{S}}_{N,k}} \mathbb{1}\{\overline{T}_{N,k}^{(\mathbf{C})}(\sigma_1^k) \geq b\} \leq \overline{q}_k \right\}, \tag{17}$$

and

$$\underline{B}_{N,k}^{(\mathbf{C})} := \sup\left\{ b > 0 : \frac{1}{m_k} \sum_{\sigma \in \widehat{\mathcal{S}}_{N,k}} \mathbb{1}\{\underline{T}_{N,k}^{(\mathbf{C})}(\sigma_1^k) \leq b\} \leq \underline{q}_k \right\} \tag{18}$$

where the $\underline{q}_i = \lfloor \frac{\alpha i}{2K} \binom{2N}{N} \rfloor / (\frac{1}{2}\binom{2N}{N}) - \underline{q}_{i-1}$ and $\overline{q}_i = \lfloor \frac{\beta i}{2K} \binom{2N}{N} \rfloor / (\frac{1}{2}\binom{2N}{N}) - \overline{q}_{i-1}$ and where $\widehat{\mathcal{S}}_{N,k}$ is defined in Equation (10). By construction of $\widehat{\mathcal{S}}_{N,k}$, for each $\sigma_1^k \in \widehat{\mathcal{S}}_{N,k}$, we have the following property

$$\forall m < k, \quad \overline{T}_{N,m}^{(\mathbf{C})}(\sigma_1^k) \leq \overline{B}_{N,m}^{(\mathbf{C})} \text{ and } \underline{T}_{N,m}^{(\mathbf{C})}(\sigma_1^k) \geq \underline{B}_{N,m}^{(\mathbf{C})}.$$

In AdaStop, modify the decision step (lines 9 to 13 in Algorithm 3) to Algorithm 4.

Algorithm 4: Early accept.

1 **if** $\overline{T}_{N,k}^{(C)}(\mathrm{id}) > \overline{B}_{N,k}^{(C)}$ **then**
2 Reject $H_{j_{\max}}$ where $\mathbf{c}_{j_{\max}} = \arg\max \left(T_{N,k}^{(j)}(\mathrm{id}), \quad \mathbf{c}_j \in \mathbf{C} \right)$.
3 Update $\mathbf{C} = \mathbf{C} \setminus \{\mathbf{c}_{j_{\max}}\}$
4 **else if** $\underline{T}_{n,k}^{(C)}(\mathrm{id}) < \underline{B}_{N,k}^{(C)}$ **then**
5 Accept $H_{j_{\min}}$ where $\mathbf{c}_{j_{\min}} = \arg\min \left(T_{N,k}^{(j)}(\mathrm{id}), \quad \mathbf{c}_j \in \mathbf{C} \right)$.
6 Update $\mathbf{C} = \mathbf{C} \setminus \{\mathbf{c}_{j_{\min}}\}$

The resulting algorithm has a small probability to accept a decision early, and as a consequence it may be unnecessary to compute the score of some of the agents in the subsequent steps.

As an illustration of the performance of early accept, if we run AdaStop with early parameter $\beta = 0.01$ (used to define the boundary in Equation (18)) on the Walker2D-v3 experiment from Section 5.3, the experiment stops at interim 2 and 10 scores have been collected for each agent. This has to be compared with the fact that, in Section 5.3, we showed that without early accept, AdaStop uses 30 scores for DDPG and TRPO. In this example, early stopping saves a lot of computations and results in a significant speed-up without affecting the final decisions.

G Agent comparison on Atari environments

In this section, we discuss the approach of (Agarwal et al., 2021) on the problem of comparing RL agents on Atari environments. The methodology from Agarwal et al. (2021) prescribes to give confidence intervals around some measure of location (typically mean or interquartile mean, IQM) of the agents score. We demonstrate this approach in the left and middle sub-figures of Figure 6. However, it is not clear how to draw a conclusion from such graphs (see (Cumming & Finch, 2005) for a discussion on doing inference with confidence intervals). Moreover, there is no definite criterion on how to construct the confidence interval. For example, this approach gives rise to three very different confidence interval plots in Figure 6: the plots on the left and in the middle are the ones presented in (Agarwal et al., 2021) and the plot on the right is a modified version of the plot in the middle, where we added the error due to performing multiple comparisons.

In the leftmost plot of Figure 6, it is easy to draw conclusions but on the other hand the theoretical interpretation is not clear because the fact that different games have different laws for their scores is not taken into account. The middle and rightmost plots in Figure 6 are a naive approach to the problem that assumes that all the games are very different from one another, and as such the confidence intervals are too large to draw conclusions. We think that there should be an intermediate approach that considers a cluster

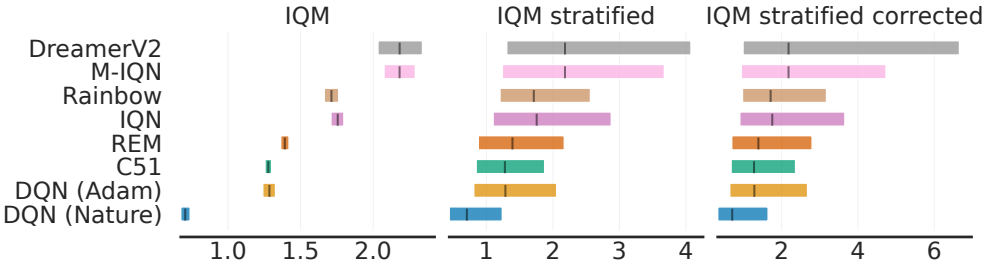


Figure 6: 95% confidence intervals on the IQM of the "Human Normalized Score" of a set of algorithms with various methods using the results from (Agarwal et al., 2021).

of similar games (e.g. all the easy games, all the maze-like games, all the difficult games, etc.) and treat these games as having all the same law. This approach would produce smaller (and more interpretable) confidence intervals compared to the naive approach, providing a middle-ground between the first plot and second plot.

H Implementation details and additional plots

H.1 Complementary experiment for Section 5.1

We consider 10 agents which score distributions are listed in Fig. 7, where the first column indicates the labels of the agents as they are used in Fig. 8.

Similarly to the setup of cases 1 and 2 (see Section 5.1), we execute AdaStop with $K = 5$, $N = 5$, $\alpha = 0.05$. As indicated in Section 4, we do not enumerate all the permutations in the permutation test as this would be too expensive. Instead we use 10^4 randomly selected permutations to compute our test statistics at each interim.

In contrast to the experiments reported in the main text, we use early accept (with $\beta = 0.01$) to avoid situations where all agents are compared with the maximum number of scores, *i.e.* NK scores. This may occur when each agent has a similar distribution to at least one other agent.

We show the performance of AdaStop for multiple agents comparison in Fig. 8, which corresponds to the output of one execution of AdaStop. The table (left part of Fig. 8) summarizes the decisions of the algorithm for every pair of comparisons. The violin plots (right part of Fig. 8) reflect empirically measured distributions in the comparison. From this figure, we can see that almost all agents are clustered according to the mean of this distribution, except for *MG3, Mt5 and MG3 that are assigned to two different groups at once. Interestingly, except for *MG3, these clusters are correctly formed. Moreover, similarly to the two previous cases, we have executed AdaStop $M = 5\ 000$ times to measure the FWE of the test. Empirically, we measured a FWE of 0.0178 for the test comparing the distributions and a FWE of 0.0472 for the test comparing the means: both are below 0.05. This example illustrates the fact that AdaStop can be efficiently used to compare the score of several agents simultaneously.

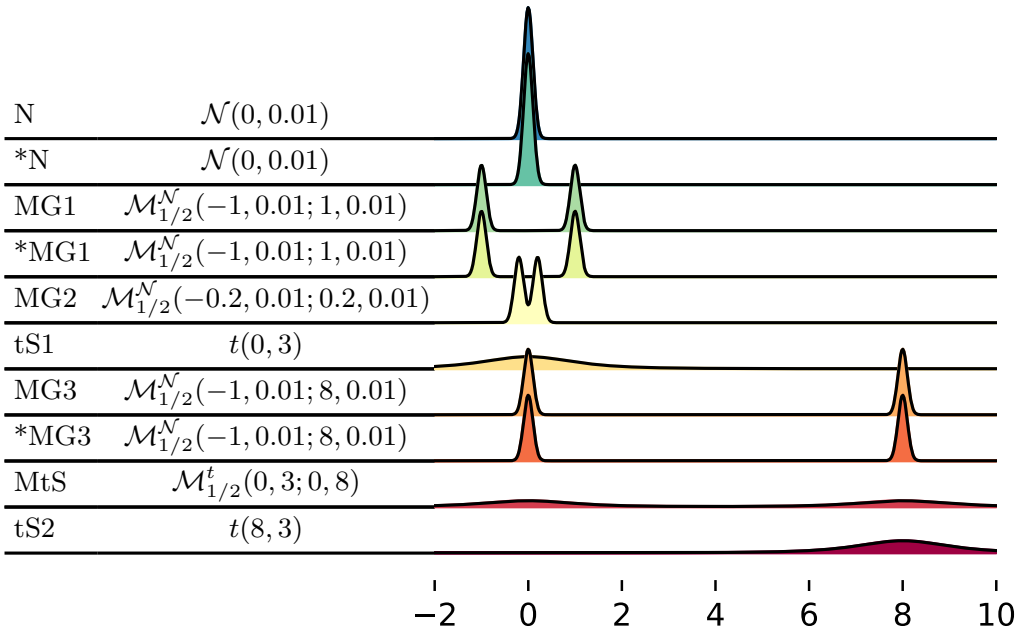


Figure 7: Toy example 3, with an illustration of the involved distributions.

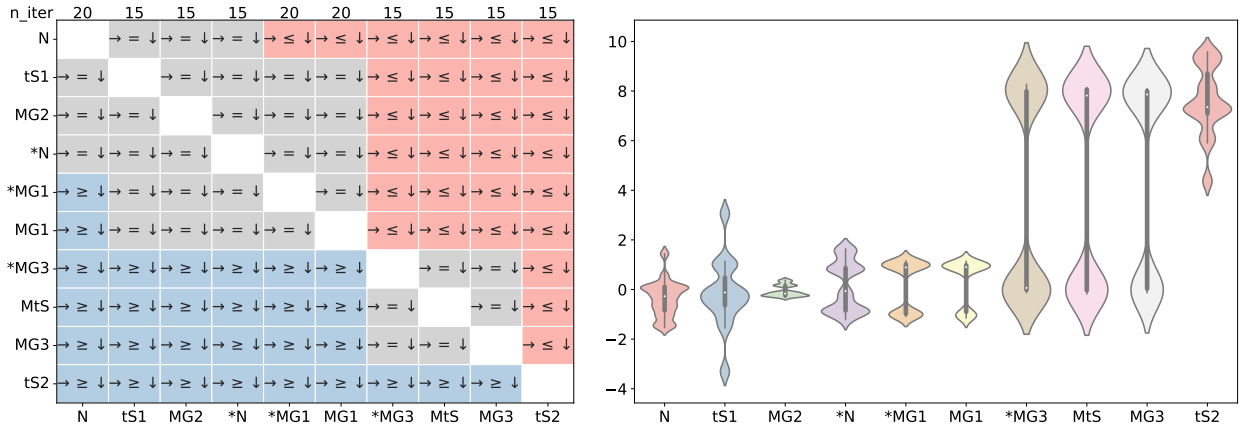


Figure 8: AdaStop decision table (left) and measured empirical distributions (right).

H.2 MuJoCo Experiments

In this section, we detail the experimental setup of the MuJoCo experiments, as well as adding new plots.

Hyperparameters. Table 3 lists the hyperparameters used for each Deep RL agent on the MuJoCo benchmark. Each agent is trained during 10^6 interactions with its environment in the cases of HalfCheetah-v3, Hopper-v3, and Walker2d-v3. It is trained during 2.10^6 interactions in the cases of Ant-v3 and Humanoid-v3. In all cases, a training episode is made of no more than 10^3 interactions.

Scores. According to algorithm 3, after each agent is trained, the resulting policy performs 50 evaluations episodes. The score of the agent is the mean performance of these 50 runs.

Learning curves. Fig. 10 presents sample efficiency curves for all algorithms in each environment. The shaded areas represent 95% bootstrapped confidence intervals, computed using `rliable` (Agarwal et al.,

	DDPG	TRPO	PPO	SAC
γ	0.99	0.99	0.99	0.99
Learning Rate	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}
Batch Size	128	64	64	256
Buffer Size	10^6	1024	2048	10^6
Value Loss	MSE	MSE	AVEC (Flet-Berliac et al., 2021)	MSE
Use gSDE	No	No	No	Yes
Entropy Coef.	-	0	0	`auto`
GAE λ	-	0.95	0.95	-
Advantage Norm.	-	Yes	Yes	-
Target Smoothing	0.005	-	-	0.005
Learning Starts	10^4	-	-	10^4
Policy Frequency	32	-	-	-
Exploration Noise	0.1	-	-	-
Noise Clip	0.5	-	-	-
Max KL	-	10^{-2}	-	-
Line Search Steps	-	10	-	-
CG Steps	-	100	-	-
CG Damping	-	10^{-2}	-	-
CG Tolerance	-	10^{-10}	-	-
LR Schedule	-	-	Linear to 0	-
Clip ϵ	-	-	0.2	-
PPO Epochs	-	-	10	-
Value Coef.	-	-	0.5	-
Train Freq.	-	-	-	1 step
Gradient Steps	-	-	-	1

Table 3: Hyperparameters used for the MuJoCo experiments.

2021). Note that each curve may be an aggregation of a different number of scores. The number of scores can be found in the bottom right table of Fig. 9.

Additional comparison plots. Fig. 9 expands upon the comparisons given in the main text (in Fig. 5) by also plotting the score distributions of each agent using boxplots.

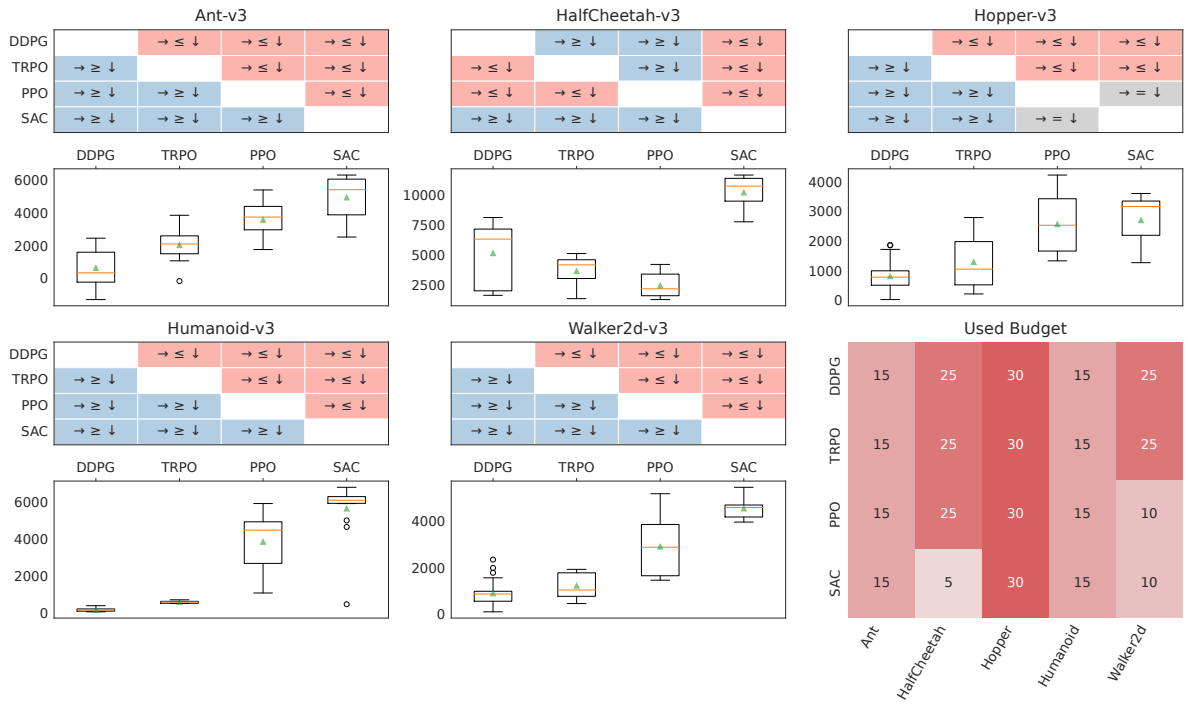


Figure 9: AdaStop decision tables (top) and score distributions (bottom) for each MuJoCo environment, and the budget used to make these decisions (bottom right). The medians are represented by green triangles and the means by horizontal orange lines.

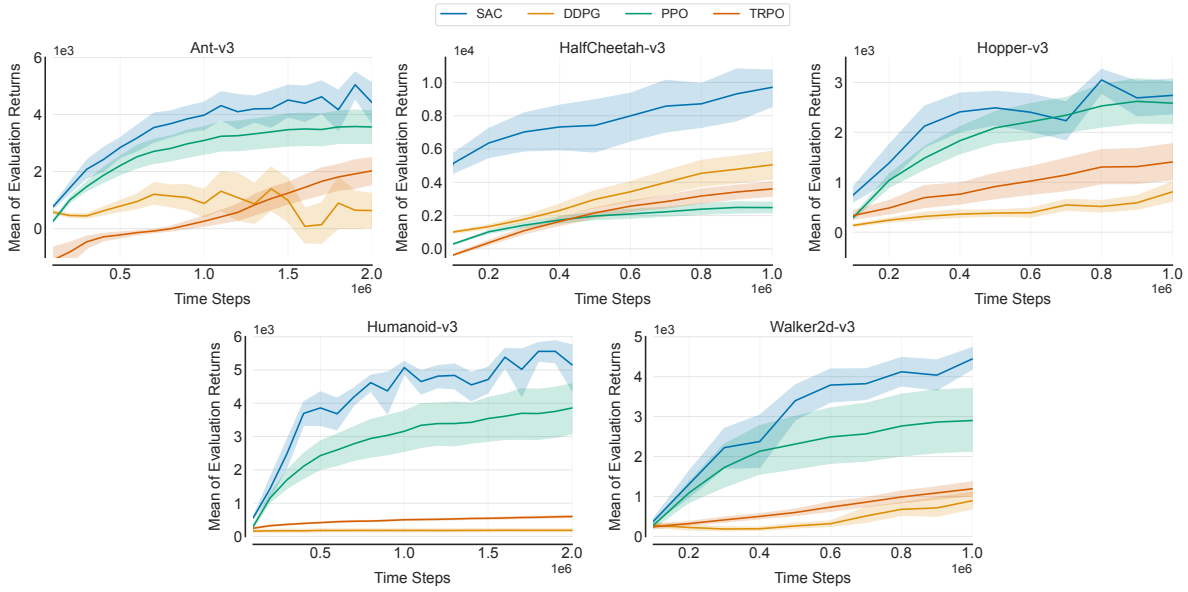


Figure 10: Mean of scores with 95% stratified bootstrap confidence intervals. Note that the curves in one figure may use a different number of scores, depending on when AdaStop made the decisions.

H.3 Additional plot for Section 5.2

To illustrate Section 5.2, Figure 11 represents the kernel density estimation of the distribution of scores from Colas et al. (2019).

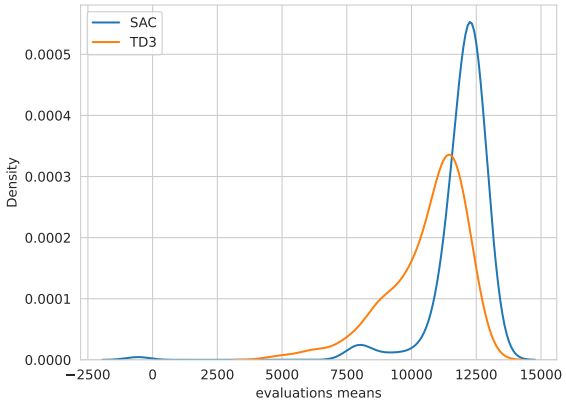


Figure 11: Distributions of scores for a SAC agent and a TD3 agent on HalfCheetah obtained with 192 independent scores. Each score is obtained on an agent that has been trained during 2.10^6 interactions.