Organization

The project is done in pairs, on one of the five topics below. Each topic has three **guided tasks**, then a part where **you choose how to go further**. Each topic comes with a **starter notebook**, on the course page: it contains the simulator, the evaluation, the plots, and the cells to complete, marked `... # TODO`. Everything else in the notebook is given and works.

What you do. For each of the three tasks of your topic:

- complete the code, and run it to produce the figure or the table asked for;
- answer the questions of the task in the cell "Answer", in a few sentences.

Then **go further** in one direction of your choice: one of the suggestions of your topic, or your own idea. State the question you ask, and answer it in the same way: code, a figure, a short answer. One well-studied direction is better than several superficial ones.

What a good answer is. Short, precise, and supported by your figure, with the uncertainty: compare means *with their error bars*. For instance, on the bandits of Lab 2, not "UCB is better", but "UCB ends with a regret of 40 ± 5, against 120 ± 20 for ε-greedy: ε-greedy keeps pulling the bad arms at the constant rate ε, while UCB stops pulling an arm once its upper confidence bound falls below the best mean". (The numbers are illustrative.) Use the concepts of the course (bias and variance, exploration, confidence intervals) to explain *why*.

What you hand in. The completed notebook (`.ipynb`), executed, with its outputs. It must run from top to bottom in less than 15 minutes on a laptop. The format of the upload will be given on the course page.

Defense. 10 minutes per pair at session 10: 5 minutes to present your results from the notebook (no slides needed), then 5 minutes of questions, including on your code.

Grading. The three tasks (code, figures, and the quality of the answers), the direction you chose to go further, and the defense.

Rules. Do not modify the simulators, and do not read in your agents the attributes marked "for the evaluation only". Language models may be used to write code, but each member of the pair must be able to explain every line during the defense.

Calendar. Pairs and topic by email by November 4. Session 8 (November 25): Task 1 done, and your questions. Notebook due on December 8; defenses at session 10 (December 9).

Setup. Download the setup archive from the course page again (it contains `seqlearn/projects/`) and replace your folder `seqlearn/`, then download the notebook of your topic. The Colab version of each notebook runs without installation.

1 Recommendation under drift

Context. A news site recommends one of 10 items to each visitor. As in the track `users` of Lab 4, each visitor has a profile over 5 moods, and item a is clicked with probability $\langle u_t, w_a \rangle$, where u_t is the profile and w_a the appeal of the item: a linear bandit of dimension 50 (session 3). The appeals *change*: at 4 unknown rounds out of $T = 20\,000$, about half of the items get new appeals (news become old, new topics appear). Notebook: `project1-drift.ipynb`.

Tasks.

1. **Forgetting is necessary.** Run LinUCB (given) and UCB1 without context on 5 seeds. Expected: the regret curves, and why LinUCB does not recover after a change point.
2. **A sliding window.** Implement LinUCB on the last w observations (a few lines), and compare $w \in \{500, 2000, 5000\}$. Expected: the regret curves, which window is best, and why a too small and a too large window are both bad (bias and variance).
3. **Choosing offline.** Implement the *replay* evaluation from uniform logs (Li et al., 2011; session 5), and compare the replay estimates of the click rates with the online ones, for 20 000 and 200 000 logs. Expected: the table, and why the number of logs matters for an agent that forgets.

To read. Session 3 (LinUCB) and session 5 (learning from logs). Garivier and Moulines, *On upper-confidence bound policies for switching bandit problems*, ALT 2011 (sliding-window UCB). Russac, Vernade and Cappé, *Weighted linear bandits for non-stationary environments*, NeurIPS 2019 (sections 1 and 2). Li, Chu, Langford and Wang, *Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms*, WSDM 2011 (sections 3 and 4).

Going further, for instance. A discounted LinUCB (Russac et al.); linear Thompson sampling with a window; detecting the change points and restarting; the variance of the replay.

2 Bayesian optimization

Context. The hyperparameters of a model are chosen with a small number of trainings. Here, a small neural network is trained in numpy on a classification problem, and the budget is 40 trainings, as if each one took hours. The point $x \in [0, 1]^4$ encodes the learning rate, the weight decay, the number of hidden units and the batch size; each evaluation returns a *noisy* validation error. Bayesian optimization models the error by a Gaussian process (a kernel version of the ridge regression of session 3) and chooses the next point by optimism, as UCB. Notebook: `project2-bayesopt.ipynb`; the Gaussian process is given.

Tasks.

1. **GP-UCB.** Write the loop of Bayesian optimization: fit the Gaussian process, evaluate the point that minimizes $\mu(x) - \beta s(x)$ among random candidates, recommend the point of lowest posterior mean at the end. Expected: the errors of the 40 evaluations of one run, against random search, and why the recommendation uses the posterior mean rather than the best observed error.

2. **Against random search.** Compare on 20 seeds the true error of the recommended point, for random search, GP-UCB, and GP-UCB recommending the best observed point. Expected: the figure, and whether the differences are significant.
3. **Exploration.** Compare $\beta \in \{0, 2, 8, 32\}$. Expected: the figure, and the link with the exploration-exploitation trade-off of the bandits.

To read. Session 2 (optimism) and session 3 (ridge regression). Srinivas, Krause, Kakade and Seeger, *Gaussian process optimization in the bandit setting*, ICML 2010 (sections 1 and 2). Rasmussen and Williams, *Gaussian Processes for Machine Learning*, MIT Press 2006, chapter 2 (free online). Shahriari et al., *Taking the human out of the loop: a review of Bayesian optimization*, Proceedings of the IEEE 2016 (an overview).

Going further, for instance. The length scale of the kernel (fixed or fitted by maximum likelihood); expected improvement or Thompson sampling instead of GP-UCB; cheap evaluations with fewer epochs and sequential halving (Hyperband, Li et al., JMLR 2018).

3 Exploration in reinforcement learning

Context. In Lab 7, Q-learning with ε-greedy learned nothing on the maze until its values were initialized optimistically. This topic studies two MDPs where random moves rarely find the rewards: *RiverSwim* (6 states; swimming upstream is slow, and the large reward is at the source) and a *slippery maze* (a small reward next to the start, the goal far away). Episodes last 50 steps, $\gamma = 0.95$, and the budget is 1000 episodes. Notebook: `project3-explore.ipynb`; Q-learning and value iteration are given.

Tasks.

1. **ε-greedy fails.** Run Q-learning with $\varepsilon = 0.1$ on both MDPs. Expected: the learning curves, the policy learned, and why, from the structure of each MDP.
2. **Optimism in Q-learning.** Implement a count bonus $c/\sqrt{N(s,a)}$ added to the rewards, and compare it with optimistic initialization. Expected: the learning curves, which MDP is solved, and why the maze is harder for Q-learning.
3. **Optimism in a model.** Complete a model-based agent that estimates the transitions from its counts and plans by value iteration with a bonus (MBIE-EB), and study its parameter β on the maze. Expected: the learning curves, why it learns much faster than Q-learning, and its cost in computation.

To read. Session 7 (value iteration, Q-learning, exploration). Sutton and Barto, *Reinforcement Learning: An Introduction*, 2nd edition, MIT Press 2018, sections 2.6, 6.5 and chapter 8 (free online). Strehl and Littman, *An analysis of model-based interval estimation for Markov decision processes*, JCSS 2008 (sections 1 to 3).

Going further, for instance. Posterior sampling for MDPs (Osband, Russo and Van Roy, *(More) efficient reinforcement learning via posterior sampling*, NeurIPS 2013); schedules of ε; a longer river.

4 Model arena

Context. An arena ranks 20 language models from duels judged by annotators, as Chatbot Arena. The preferences follow a Bradley-Terry model (session 6), perturbed by a term that makes some of them *cyclic*, and 10% of the annotators answer at random. The quality of a model is its *Borda score*, its probability of beating an opponent drawn uniformly. The goal is to find 3 good models (Borda score at least the third best minus 0.02) with few duels. Notebook: `project4-arena.ipynb`; the Bradley-Terry fit is given.

Tasks.

1. **Uniform duels.** Play duels between random pairs, and estimate the top 3 by the win rates and by Bradley-Terry, for budgets from 500 to 10 000 duels, on 100 arenas. Expected: the proportion of good top 3 against the budget, and how many duels are needed.
2. **Choosing the duels.** After half of the budget, focus the duels on the 8 best models so far. Expected: the comparison with uniform duels, its significance, and the link with successive elimination.
3. **When Bradley-Terry is wrong.** Increase the cyclic term, and measure the error of the fitted model on the probabilities and the quality of the rankings. Expected: the figure, and which ranking to trust when no score explains the duels.

To read. Session 6 (pure exploration, Bradley-Terry). Chiang et al., *Chatbot Arena: an open platform for evaluating LLMs by human preference*, ICML 2024 (sections 3 and 4). Kalyanakrishnan, Tewari, Auer and Stone, *PAC subset selection in stochastic multi-armed bandits*, ICML 2012. Bengs, Busa-Fekete, El Mesaoudi-Paul and Hüllermeier, *Preference-based online learning with dueling bandits: a survey*, JMLR 2021 (section 3, on the notions of winner).

Going further, for instance. A stopping rule with confidence intervals (stop as soon as 3 models are surely good); Thompson sampling on the Bradley-Terry scores; other notions of winner (Copeland, von Neumann).

5 Rating the players of a team game

Context. An online game opposes two teams of 5 players, in a league of 200 players. Team A beats team B with probability $\sigma\big(\sum_{i\in A} s_i - \sum_{j\in B} s_j\big)$, where s_i is the hidden strength of player i: the Bradley-Terry model of session 6, for teams. The league moves: the strengths drift, players leave and are replaced by newcomers who improve, and some friends always play in the same team. The publisher rates the players to predict the matches and to form fair teams. Notebook: `project5-league.ipynb`; the season and its evaluation (log-loss of the predictions, Kendall correlation of the ratings, fairness of the matches) are given.

Tasks.

1. **Elo for teams.** Implement Elo where the rating of a team is the sum of the ratings of its players, and choose the step k. Expected: the log-loss and the ranking against k, and why a too small and a too large step are both bad.

2. **Newcomers.** Give a larger step to the players with fewer than 30 matches. Expected: the table, on which players it helps, and why a step per player (an uncertainty per rating) is the principled version.
3. **Matchmaking.** Form balanced teams with the ratings. Expected: the table, how much fairer the matches are, what happens to the ranking, and why the log-loss cannot compare two matchmakers.

To read. Session 6 (Bradley-Terry, Elo, choosing the step). Elo, *The rating of chessplayers, past and present*, 1978 (chapter 1). Glickman, *Parameter estimation in large dynamic paired comparison experiments*, JRSS C 1999 (the Glicko system). Herbrich, Minka and Graepel, *TrueSkill: a Bayesian skill rating system*, NeurIPS 2006.

Going further, for instance. A Gaussian rating per player (Glicko, TrueSkill); Bradley-Terry by maximum likelihood on a window of matches; the parties of friends; matchmaking that favours the matches with uncertain players.