Midterm Project: Video and Writeup

CSE 541: Interactive Learning, Autumn 2026
Instructor: Kevin Jamieson

Due Sunday, November 15, 2026, 11:59 PM Pacific **No late submissions.**

The idea

Pick one tool from the first half of the course, a bound, an algorithm, or a proof technique, and **extend it**: go deeper than lecture went, or follow it into a neighboring problem. Then **explain it to your classmates**. The second part is the point. The course has covered a lot of ground quickly, and the best way to own a piece of it is to teach it. By the end of the quarter the two rounds of projects will have given everyone a tour of a few dozen topics we could not fit into lecture.

The project is **individual**, one per person. Topics for this round are drawn from everything up to linear bandits: supervised and online learning, exponential weights, portfolios, online convex optimization, stochastic and adversarial multi-armed bandits, lower bounds, and linear bandits with experimental design. The final project will draw on the second half of the course.

Deliverables

1. **A 5-minute YouTube video** (± 30 seconds; unlisted is fine). Slides with a voiceover, a whiteboard talk, or an animated explainer are all fine, and production quality is not graded. Clarity and accuracy are. At least one big idea should come across clearly to a classmate who has taken the course so far but has not read your papers. Add captions: YouTube's auto-captions, corrected where they are wrong, are fine.

2. **A 2–3 page writeup** in LaTeX, 11pt font, 1-inch margins, submitted as a PDF on Gradescope. Three pages *including* references; longer writeups will be penalized. It introduces the problem, surveys the relevant papers, states results correctly and with the right assumptions, explains the key idea behind at least one of them, and says what is open or what you would do next.

3. **An Ed post** titled "[Your name]: [Project title]" containing the video link and a two-sentence abstract, in the *Midterm projects* thread. The writeup goes to Gradescope only.

If you want to propose a topic that is not on the list below, or are unsure whether a paper is in scope, ask on Ed.

What a good project looks like

This is a **theory project**: a focused survey of an area that extends something from lecture. You are not expected to prove new theorems. You are expected to read several papers, understand one of them well enough to explain its central idea and where its proof departs from the proof you saw

in class, state the results with their assumptions, and articulate what remains open. Writeups that only paraphrase abstracts do poorly. A small simulation that illustrates the phenomenon you are explaining is welcome but not required, and it does not substitute for understanding the argument.

The project should contain at least one thing you understood deeply enough to re-derive on a whiteboard, and your video should be built around it.

Rubric (100 points)

Video (45)		
The big idea	A classmate can state, after watching, what the main result or method is and why it works.	20
Correctness	Results are stated with the right assumptions; no false claims; honest about limitations.	15
Organization and timing	Setup, idea, evidence, takeaway; within 5 minutes ± 30 s; captions present.	10
Writeup (45)		
Problem and context	The setting is defined precisely and connected to what we did in class.	10
Survey and accuracy	The relevant papers are identified, results are stated correctly, and the key idea of at least one is explained.	20
Extension	What is open or under-explored, why, and what you would try next.	10
Writing	Clear, within 3 pages, references complete.	5
Participation (10)		
Ed post and engagement	Your post is up on time, and by **Friday, November 20** you have left a substantive comment or question on at least two classmates' projects.	10

AI policy

The course AI policy applies. AI may help you find papers, check your understanding, and polish your writing and slides; it may not write your survey or your explanation. If you include a simulation, you must supply the algorithm yourself: AI may turn an algorithm you have fully specified into code, but "implement LinUCB" is not allowed. Add a short note at the end of the writeup naming the tools you used and how. You should be able to explain and defend everything you submit, and you will be asked to in the Ed comments.

Topics

Each entry names a direction and one or two places to start. These are starting points, not reading lists. You are encouraged to propose your own topic, especially one that connects to your research.

Online learning and experts (Lectures 1–4)

- **Second-order and adaptive exponential weights.** Hoeffding gave us $\sqrt{T \log d}$ for any losses. Bounds that scale with the variance of the losses or with the loss of the best expert can be much smaller, and the learning rate can be tuned on the fly. Start with AdaHedge (de Rooij, van Erven, Grünwald & Koolen 2014) and Squint (Koolen & van Erven 2015).

- **Tracking a moving comparator.** Homework 1 did fixed share. Adaptive regret (Hazan & Seshadhri 2009) and strongly adaptive regret (Daniely, Gonen & Shalev-Shwartz 2015) ask for low regret on *every* interval at once. How is this achieved, and at what cost?
- **Mixability and the aggregating algorithm.** We saw constant regret for the log loss and for the squared loss. Vovk's aggregating algorithm characterizes exactly which losses allow $O(\log d)$ regret and which are stuck at $\sqrt{T}$. Start with Vovk (1998) and Chapter 3 of Cesa-Bianchi & Lugosi, *Prediction, Learning, and Games*.
- **Efficient universal portfolios.** Cover's algorithm has $d \log T$ regret but is exponential-time; online gradient descent is cheap but has $\sqrt{T}$ regret. The gap has been closed from both sides: Kalai & Vempala (2002), online Newton step (Hazan, Agarwal & Kale 2007), Luo, Wei & Zheng (2018), Zimmert, Agarwal & Kale (2022). Map the regret-versus-computation frontier.
- **Follow the perturbed leader.** Exponential weights needs a weight per expert, which fails when the experts are the 2^d paths in a graph or the $d!$ rankings. Kalai & Vempala (2005) add random noise to follow-the-leader instead and get regret for combinatorial experts in polynomial time. Explain why perturbation plays the role of the entropy regularizer.
- **No-regret learning in games.** If both players of a zero-sum game run exponential weights, the average of their plays converges to a Nash equilibrium (Freund & Schapire 1999). What happens for the *last* iterate, for general-sum games, or for correlated equilibria? Daskalakis & Panageas (2019) is one entry point.
- **Calibration.** A forecaster is calibrated if, of the days on which it says "70%", it rains on 70%. Foster & Vohra (1998) showed calibration is achievable against any adversary by an argument that looks a lot like our minimax ones. What is the connection to regret, and what rates are possible?
- **Online-to-batch and back.** We used online algorithms to get i.i.d. guarantees (OGD with averaging). The reverse direction, using online regret as a definition of learnability, leads to sequential Rademacher complexity (Rakhlin, Sridharan & Tewari 2010). What does the online analog of VC dimension look like? (Littlestone dimension is the warm-up.)

Online convex optimization (Lecture 5)

- **Fast rates in OCO.** Strong convexity gives $\log T$ regret for OGD with $\eta_t = 1/(\alpha t)$ (Homework 1), and exp-concavity gives $d \log T$ via online Newton step (Hazan, Agarwal & Kale 2007). Explain where each condition is used and why the log loss of portfolios is the motivating example.
- **Adaptive step sizes.** AdaGrad (Duchi, Hazan & Singer 2011) replaces the single learning rate by a per-coordinate one and has a clean regret bound. Explain the bound, what it buys for sparse gradients, and how it relates to the mirror descent view of Homework 1.
- **Mirror descent beyond the simplex and the ball.** FTRL with the entropy regularizer gave exponential weights and with the quadratic gave OGD. Which regularizers suit other geometries, such as the ℓ_1 ball, the spectrahedron, or the set of doubly stochastic matrices? Shalev-Shwartz's survey (2012) and Hazan's book are the places to start.
- **Bandit convex optimization.** OCO with only the loss value revealed, not the gradient. The one-point gradient estimate of Flaxman, Kalai & McMahan (2005) gives $T^{3/4}$; getting

$\sqrt{T}$ took a decade (Bubeck, Eldan & Lee 2017). Explain the estimator and why the problem is so much harder than linear bandits.

- **Delayed feedback.** In practice the loss of round t arrives rounds later. How do exponential weights and OGD degrade with delay, and what is the right way to handle it? Joulani, György & Szepesvári (2013) is the starting point, and the same question for bandits is a natural extension.

Stochastic bandits (Lectures 5–8)

- **Asymptotically optimal UCB.** Lecture 8 showed that $\sum_i 2 \log T / \Delta_i$ is the Lai–Robbins limit for consistent policies. KL-UCB (Garivier & Cappé 2011; Cappé et al. 2013) attains it with the right constant. Explain what changes in the index and in the analysis compared with the UCB we proved.

- **Thompson sampling, two ways.** Frequentist regret bounds (Agrawal & Goyal 2012, 2013; Kaufmann, Korda & Munos 2012) versus the information-theoretic Bayesian analysis of Russo & Van Roy (2014, 2016), which also yields information-directed sampling. Pick one lens and explain it.

- **Minimax-optimal algorithms and best of both worlds.** MOSS (Audibert & Bubeck 2009) removes the $\log T$ from $\sqrt{nT \log T}$. Tsallis-INF (Zimmert & Seldin 2019) is simultaneously optimal for stochastic and adversarial bandits with a single algorithm: FTRL with the Tsallis entropy. Explain why that regularizer and not the Shannon entropy of EXP3.

- **Best-arm identification, fixed confidence.** We did successive elimination. LUCB (Kalyanakrishnan et al. 2012), lil'UCB (Jamieson et al. 2014), and Track-and-Stop (Garivier & Kaufmann 2016) get closer to the $\sum_i \Delta_i^{-2} \log(1/\delta)$ lower bound, the last one exactly. What is the role of the anytime confidence bound, and of the lower bound itself, in these algorithms?

- **Best-arm identification, fixed budget.** Given exactly T pulls, output an arm and minimize the probability of being wrong: successive rejects (Audibert, Bubeck & Munos 2010), sequential halving (Karnin, Koren & Somekh 2013), and a surprising lower bound (Carpentier & Locatelli 2016) showing that the fixed-budget problem is genuinely harder than fixed confidence.

- **Confidence bounds that hold for all time.** The union bound over time in UCB costs a $\log T$; the law of the iterated logarithm says $\sqrt{\log \log t}$ is the truth. Jamieson et al. (2014) and Howard, Ramdas, McAuliffe & Sekhon (2021) build such *confidence sequences* with martingales; they are also the right tool for stopping an A/B test early without lying (Johari, Koning, Pekelis & Walsh 2017).

- **Non-stationary bandits.** Means that drift or switch. Sliding-window and discounted UCB (Garivier & Moulines 2011), EXP3.S (Auer et al. 2002), and algorithms that detect changes without knowing how many there are (Auer, Gajane & Ortner 2019). This is the bandit analog of Homework 1's dynamic regret.

- **Heavy-tailed rewards.** Everything we proved assumed sub-Gaussian noise. Bubeck, Cesa-Bianchi & Lugosi (2013) show that with only a finite $(1+\epsilon)$-th moment, robust mean estimators (median of means, truncation) recover $\log T$ regret. Explain what breaks and what is recovered.

- **Dueling bandits.** Only pairwise comparisons are observed ("which of these two is better?"). Yue, Broder, Kleinberg & Joachims (2012) and RUCB (Zoghi et al. 2014). This is the feedback model behind learning from human preferences, and a good project connects the two.
- **Combinatorial semi-bandits.** The arms are subsets $x \in \{0,1\}^d$ (a route, a slate of k ads), the reward is $\langle \theta, x \rangle$, and you observe the coordinates of θ that you included. Chen, Wang & Yuan (2013) and Kveton et al. (2015). How much does seeing the pieces help compared with bandit feedback on the sum?
- **Batched and limited-adaptivity bandits.** Clinical trials and A/B platforms update their policy a handful of times, not every round. How many batches are needed to match fully adaptive regret? Perchet, Rigollet, Chassang & Snowberg (2016) and Gao, Han, Ren & Zhou (2019).
- **Bandits with knapsacks.** Each pull consumes resources and you stop when a budget runs out. Badanidiyuru, Kleinberg & Slivkins (2013) started the area; it has a very different flavor from regret minimization.
- **Continuum-armed and Lipschitz bandits.** Homework 2 does the $T^{2/3}$ discretization. The zooming algorithm (Kleinberg, Slivkins & Upfal 2008) and $\mathcal{X}$-armed bandits (Bubeck, Munos, Stoltz & Szepesvári 2011) adapt the discretization to where the function is nearly optimal and do much better on benign instances.
- **Lower bounds beyond the two we proved.** The change-of-measure template also gives lower bounds for best-arm identification with many answers, for linear bandits (Lattimore & Szepesvári 2017), and for non-stationary settings. Pick one and show how the construction changes.

Adversarial bandits (Lecture 8)

- **High-probability and implicit-exploration EXP3.** Our EXP3 bound is in expectation; with constant probability the actual regret can be far worse. EXP3.P (Auer et al. 2002) and EXP3-IX (Neu 2015) fix this with biased estimators. Explain the bias-variance trade-off in the loss estimate.
- **Adaptive adversaries and policy regret.** If the adversary reacts to your past plays, the regret we defined is the wrong quantity (Arora, Dekel & Tewari 2012). What can be guaranteed, and against which adversaries?
- **Adversarial linear bandits.** EXP3 for arms with features, where the loss is $\langle \ell_t, x \rangle$ and you see only your own loss. Bubeck, Cesa-Bianchi & Kakade (2012) get $\sqrt{dT \log n}$ by sampling from an exploration distribution that is, once again, a design. Explain the estimator and where Kiefer–Wolfowitz enters.
- **Contextual bandits with experts: EXP4.** n experts each recommend one of k arms every round, and you compete with the best expert while observing only your own reward (Auer et al. 2002). This is coming in the second half of the course, so a project here should go deeper: its $\sqrt{kT \log n}$ bound, its computational cost, and what came after.

Linear bandits and experimental design (Lecture 9 onward)

- **Pure exploration in linear bandits.** Homework 2 shows on the Soare instance that pulling good arms is the wrong strategy. Soare, Lazaric & Munos (2014), the transductive designs

and RAGE algorithm of Fiez, Jain, Jamieson & Ratliff (2019), and Degenne, Ménard, Shang & Valko (2020). What is the right complexity measure and how do the algorithms track it?

- **The end of optimism.** Lattimore & Szepesvári (2017) give the asymptotic instance-dependent lower bound for linear bandits and show that optimistic algorithms such as LinUCB cannot achieve it, for the same reason as the Soare instance. Explain the lower bound and the algorithm that does achieve it.
- **Computing optimal designs.** Kiefer–Wolfowitz says D- and G-optimality coincide; computing the design is its own subject: Frank–Wolfe and its $O(d \log d)$ support bound, Khachiyan's (1996) rounding, and the minimum-volume-ellipsoid view in Todd's book (2016). Explain the geometry and how the algorithms compare.
- **Misspecified linear bandits.** The reward is only approximately linear in the features. Lattimore, Szepesvári & Weisz (2020) show that an ϵ-misspecification costs $\epsilon\sqrt{d}\,T$ regret and that this is unavoidable. Why $\sqrt{d}$, and what does it say about using learned features?
- **Sparse linear bandits.** d is large but θ^* has $s \ll d$ nonzeros. Abbasi-Yadkori, Pál & Szepesvári (2012) and Hao, Lattimore & Wang (2020): the regret depends on s, but the exploration has to be designed differently, and in some regimes nothing beats $T^{2/3}$.
- **Generalized linear and logistic bandits.** Binary rewards with $\Pr(y = 1) = \sigma(\langle x, \theta^* \rangle)$. Filippi, Cappé, Garivier & Szepesvári (2010) and Faury, Abeille, Calauzènes & Fercoq (2020). How do least squares and the confidence ellipsoid generalize, and why does the nonlinearity make exploration harder?
- **Kernelized bandits and Bayesian optimization.** Replace $\langle x, \theta^* \rangle$ by a function in a reproducing kernel Hilbert space, as in Homework 2's Sobolev example. GP-UCB (Srinivas, Krause, Kakade & Seeger 2010) and its frequentist analysis (Chowdhury & Gopalan 2017). What plays the role of d?
- **Experimental design for active learning.** Choosing which points to label in regression or deep learning is an experimental design problem. Survey how G/D-optimal designs and leverage scores are used for batch active learning, and where the linear theory stops applying.

Final notes

- The best projects connect to something you already care about. If your research has a sequential decision in it, this is the time to look at it through the course's lens.
- A project should contain a derivation you re-did or a proof idea you can explain on a whiteboard. Purely conceptual overviews score poorly.
- Ask on Ed. Scoping questions and "is this paper too hard" are welcome.