

Self-Evaluation Test

Instructors: *Alex Andoni, Edo Liberty*

This is a self-evaluation test for you to confirm that you have the sufficient background for the class, and identify potential parts to brush up before the class. You are expected to understand and solve all problems on this test. You do not have to turn in solutions.

The main prerequisite for the class is mathematical maturity, i.e., being able to follow and write rigorous mathematical proofs, of both combinatorial and analytical flavors, at the level of COMS 4231 (Analysis of Algorithms). In terms of concrete topics, you should be comfortable with: linear algebra (eigenvalues/eigenvectors, matrix norms, SVD); probability (linearity of expectation, variance, concentration bounds); and basic algorithmic notions (order notation for runtime, correctness proofs, NP-hardness). No machine learning or coding background is assumed. As a result, the problems below are partitioned by category.

Questions marked with $\star$ are harder/challenge questions — they are at the level of what is covered in the class.

1 Math

1.1 Norms

For a vector $x \in \mathbb{R}^n$, and a real $p \in [1, \infty]$, the p -norm of x is defined as $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$.¹ Prove the following:

- $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2$, using Cauchy–Schwarz if needed. For which vectors x is each of the inequalities tight?
- when $p = \log_2 n$, $\|x\|_p$ is a 2 approximation to $\|x\|_\infty$ (i.e., $\|x\|_\infty \leq \|x\|_p \leq 2\|x\|_\infty$).

$\star$ In the spirit of above, how can you relate the 10-norm of x to the 5-norm of x ?

1.2 Order notation: runtime of Merge-Sort

Merge-Sort is an algorithm for sorting n numbers, which proceeds as follows. Suppose the input is an array $A[1 \dots n]$ of length n (which, for simplicity, assume is a power of 2). We partition the array into two, $A[1 \dots n/2]$ and $A[n/2 + 1 \dots n]$. We sort each part recursively and then merge the 2 (sorted) parts. It is known that the latter merge operation can be accomplished in time $O(n)$.² The base case of the Merge-Sort is when $n = 1$, in which case we do nothing (the array is already sorted).

Here, you are to determine whether the following statement/proof is correct or not (why or why not).

¹For $p = \infty$, $\|x\|_\infty$ is formally defined to be $\max_{i \in [n]} |x_i|$, which can be seen as the limit as $p \rightarrow \infty$.

²Remember that $O(f(n))$ is the set of functions $g : \mathbb{N} \rightarrow \mathbb{N}$ such that there exist $c > 0$ and $n_0 \in \mathbb{N}$ such that, for $n \geq n_0$, $g(n) \leq c \cdot f(n)$.

Claim 1. *The runtime of the above MergeSort algorithm is $O(n)$.*

Proof. Let $T(n)$ be the runtime of Merge-Sort on an array of length n . Then, by the above description we have that:

$$T(n) = 2 \cdot T(n/2) + O(n),$$

corresponding to the two recursive calls and the merge operation respectively.

We prove the claim that $T(n) = O(n)$ by induction. Indeed, we have that:

$$T(n) = 2 \cdot T(n/2) + O(n) = 2 \cdot O(n/2) + O(n) = O(n).$$

This completes the proof of the claim. □

2 Linear algebra

2.1 Quadratic forms

Let A be a n -by- n symmetric matrix whose maximum eigenvalue is 42. What is the minimum and maximum possible value for the quadratic form $x^T A x$, where $x \in \mathbb{R}^n$ are unit-norm vectors? What about $x^T A^2 x$?

2.2 Matrix norms and the SVD

Let $A \in \mathbb{R}^{n \times d}$ be a matrix of rank r , with singular values $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$. Recall the singular value decomposition (SVD): $A = \sum_{i=1}^r \sigma_i u_i v_i^T$, where $u_1, \dots, u_r \in \mathbb{R}^n$ are orthonormal and $v_1, \dots, v_r \in \mathbb{R}^d$ are orthonormal. Two standard matrix norms are the spectral (operator) norm $\|A\|_2 = \max_{x \in \mathbb{R}^d: \|x\|_2=1} \|Ax\|_2$, and the Frobenius norm $\|A\|_F = \left(\sum_{i,j} A_{ij}^2 \right)^{1/2}$.

- Express $\|A\|_2$ and $\|A\|_F$ in terms of the singular values of A . (Hint: $\|A\|_F^2 = \text{tr}(A^T A)$.)
- Prove that $\|A\|_2 \leq \|A\|_F \leq \sqrt{r} \|A\|_2$. When is each inequality tight?
- ★ Fix $k < r$. What is $\min_{B \in \mathbb{R}^{n \times d}: \text{rank}(B) \leq k} \|A - B\|_F$, and which B achieves it? What about the same question for the spectral norm $\|A - B\|_2$?

2.3 Linear systems and least squares

Suppose A is a n -by- n real matrix and $b \in \mathbb{R}^n$. For the unknown vector $x \in \mathbb{R}^n$, describe what are the possible sets of solutions for the following:

- $Ax = b$;
- ★ $\text{argmin}_{x \in \mathbb{R}^n} \|Ax - b\|_2$;
- ★ let $f(x) = \|Ax - b\|_2^2$. Compute the gradient $\nabla f(x)$. What equation must a minimizer x^* of f satisfy, and how does this relate to the previous item?

To give a better idea of what's asked, consider the " $Ax = b$ " case when $n = 1$ (i.e., just a simple equation). Here's the answer for this case:

The possible sets of solutions are: 1) there may not be a solution (empty set), 2) may be exactly one solution, or 3) any $x \in \mathbb{R}$ is a solution.

The above 3 cases correspond to when 1) $A = 0$, $b \neq 0$, 2) $A \neq 0$, 3) $A = b = 0$.

3 Probability

3.1 Mean estimation

Let p be a probability distribution over $[0, 1]$. The goal is to estimate the mean μ of the distribution p , up to an additive error ϵ , where $\epsilon \in (0, 1)$ is given in advance. How many samples suffice to output some $\hat{\mu}$ such that $\Pr[|\hat{\mu} - \mu| \leq \epsilon] \geq 0.9$? More generally, how many samples suffice for $\Pr[|\hat{\mu} - \mu| \leq \epsilon] \geq 1 - \delta$, for a given $\delta \in (0, 1)$?

3.2 Random projections

Let $x, y \in \mathbb{R}^d$ be fixed vectors, and let $g = (g_1, \dots, g_d)$ be a random vector whose coordinates are i.i.d. standard Gaussians, $g_i \sim N(0, 1)$ (recall that $\mathbb{E}[g_i] = 0$, $\mathbb{E}[g_i^2] = 1$, and $\mathbb{E}[g_i^4] = 3$).

- Compute $\mathbb{E}[\langle g, x \rangle]$ and $\mathbb{E}[\langle g, x \rangle^2]$. More generally, compute $\mathbb{E}[\langle g, x \rangle \cdot \langle g, y \rangle]$.
- ★ Take k independent copies $g^{(1)}, \dots, g^{(k)}$ of g , and output $Z = \frac{1}{k} \sum_{j=1}^k \langle g^{(j)}, x \rangle^2$ as an estimate of $\|x\|_2^2$. How large should k be so that $\Pr[(1 - \epsilon)\|x\|_2^2 \leq Z \leq (1 + \epsilon)\|x\|_2^2] \geq 0.9$?

3.3 Hashing

A function $h : U \rightarrow [m]$, where U is a discrete set (e.g., integers of $O(\log n)$ bits), and $m \in \mathbb{N}$, is called a hash function with m buckets. (One can see this as assigning one of the m buckets to each possible item from U .) Most common hash function is a random hash function: each $h(i)$, $i \in U$, is chosen iid at random from $[m]$.

Below, consider a set $S \subset U$ of size n .

- Let h be a random hash function with $m = n$. What is the expected size of a bucket, i.e., $|h^{-1}(i) \cap S|$ for any $i \in [m]$?
- A collision is an event where distinct $x, y \in S$ satisfy $h(x) = h(y)$. What is the expected number of collisions among elements of S under h ? (To give an example, if one has precisely three distinct items $x, y, z \in S$ falling into the same bucket, they generate 3 collisions: (x, y) , (y, z) , and (z, x)).

A family $\mathcal{H}$ of hash functions h is k -wise independent if for any distinct $a_1, \dots, a_k \in U$ and (not necessarily distinct) $q_1, \dots, q_k \in [m]$, we have that

$$\Pr_{h \in \mathcal{H}}[h(a_1) = q_1 \wedge \dots \wedge h(a_k) = q_k] = 1/m^k.$$

Note that the aforementioned random hash function is $|U|$ -wise independent (or, to be more precise, the family $\mathcal{H}$ of all functions $h : U \rightarrow [m]$ is $|U|$ -wise independent).

- ★ How do the estimates from above change if our hash function h is chosen from some 2-wise independent hash family $\mathcal{H}_2$?

4 Algorithms

4.1 Matrix products

Let $Q, K, V \in \mathbb{R}^{n \times d}$ be three matrices, with $d \ll n$ (think $n = 10^5$ and $d = 100$). Assume that multiplying an a -by- b matrix with a b -by- c matrix takes $O(abc)$ time (the schoolbook algorithm).

- What is the runtime of computing the n -by- d matrix $(QK^T)V$ by first forming QK^T ? What is the runtime of computing $Q(K^TV)$ instead? (Why are the two results equal?) Which order should one use?
- ★ Now suppose we need to compute $\text{softmax}(QK^T)V$, where softmax is applied to each row of the n -by- n matrix $M = QK^T$: row i of M , $(M_{i1}, \dots, M_{in})$, is replaced by $(e^{M_{i1}}, \dots, e^{M_{in}}) / \sum_{j=1}^n e^{M_{ij}}$. Does the faster order from the previous item still apply? Why or why not? (This is the attention mechanism of transformers; computing it fast is a topic of the class.)

4.2 Sampling from a discrete distribution

You are given n non-negative weights $w_1, \dots, w_n$ (e.g., $w_i = e^{s_i}$ for some scores s_i , as in the last layer of a language model), with $W = \sum_{i=1}^n w_i > 0$. The task is to draw a random index $I \in [n]$ with $\Pr[I = i] = w_i/W$, repeatedly (many times, from the same weights). Assume that you can draw a uniformly random real number in $[0, 1)$ in $O(1)$ time, and that arithmetic on real numbers takes $O(1)$ time.

- Give an algorithm that, after $O(n)$ preprocessing, draws each sample in $O(\log n)$ time. Prove that it outputs I with the correct distribution, and analyze its runtime.
- ★ Suppose that between samples the weights change: an update sets $w_i \leftarrow w'_i$ for a single index i (and, correspondingly, W changes). Can you support both updates and samples in $O(\log n)$ time each?

4.3 NP-hardness

Let P be an optimization problem (say, a minimization problem) that is NP-hard, and assume $P \neq \text{NP}$. For each of the following statements, say whether it follows from these assumptions, and briefly why or why not.

1. There is no polynomial-time algorithm that solves P exactly on all inputs.
2. There is no polynomial-time algorithm that solves P exactly on some restricted class of inputs (e.g., inputs with a special structure).
3. There is no polynomial-time algorithm that, on every input, outputs a solution whose cost is at most twice the optimum.
4. Every algorithm that solves P exactly on all inputs takes time $2^{\Omega(n)}$ on some inputs of size n .

(Several problems in the LLM pipeline are NP-hard — for example, choosing an optimal vocabulary for a tokenizer — and are handled in practice by heuristics or approximation algorithms. The distinctions above will be a recurring theme in the class.)

4.4 ★ Alice tells Bob

Suppose Alice holds a number n and wants to communicate it to Bob *approximately* using minimal possible communication. In particular, Bob should be able to output some n' satisfying $n \leq n' \leq 2n$. How many bits does Alice need to send to Bob to achieve this goal? (Answer up to $O(\cdot)$ is enough.)

Does the answer change if it's a “dialogue” — Alice and Bob sequentially communicate to each other a number of bits (think of it as if Bob can “ask questions”) — and the “total communication” is the total number of bits exchanged ?