Lecture 2: Concentration Bounds, and Tricks

Markov goes to Las Vegas

Remember from last lecture that we saw two types of randomised algorithms: Las Vegas and Monte Carlo.

There are more, of course: *e.g.*, Bellagio algorithms. And others! Mathematicians and computer scientists clearly love gambling – look up Jacob Bernoulli when you have a chance.

Las Vegas algorithms: the algorithm is *always correct*, but the running time is only bounded *in expectation*.

Monte-Carlo algorithms: the algorithm is only *correct with high probability*, but the running time is bounded *with probability one*.

Wouldn't it be nice if there was a way to go from one to the other? Well, as it turns out, there is:

Lemma 0.4.2. *Suppose there exists a Las Vegas algorithm A for some task, with expected running time T. Then there exists a Monte Carlo algorithm A' for the same task with* worst-case *running time $O(T)$ and probability of failure $1/100$.*

Proof. The proof is quite simple, and relies on analyzing the following:

Algorithm 2: Algorithm A'.

```
Require: input $x$
1: Run $A$ on $x$ for at most $100T$ steps
2: if $A$ terminated within $100T$ steps then
3:     return $A$'s output                    ▷ Always correct
4: else
5:     return an arbitrary output             ▷ Very likely wrong
```

It should be quite clear that the above algorithm always runs in time at most $100T + O(1) = O(T)$; and also that whenever we reach Line 3, the output of A' must be correct (because A, once it terminates, is always correct).

When we reach Line 5 because A "timed out," however, we cannot really say anything: maybe what we output is correct, but it's most likely wrong. So we'll just assume it's an incorrect output, and all we need to do to prove the lemma is to prove that A "times out" with probability at most $1/100$.

Importantly, all we can use to do so is what we know about A, which is very little: we only know its expected running time is at most T, and that running times are non-negative. That's not a lot to build on, but that's just enough for *Markov's inequality*:

Theorem 0.5 (Markov's inequality)**.** *Let X be a non-negative random variable with $\mathbb{E}[X] < \infty$. For any $t > 0$, we have*

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}$$

The "non-negative" assumption is crucial. It is definitely not true without!

Applying this with X being the running time of A and $t = 100$ proves the lemma. □

Lecture cue: prove Markov's.

Markov and beyond: Randomised Median

As mentioned in the previous lecture, there exists a very neat, highly non-trivial (deterministic) divide-and-conquer algorithm to find the median of (an array of) n numbers in linear time. You have seen it in previous So we will not analyze it again: instead, we will give a simple (Monte Carlo) randomized algorithm, also linear-time.

The idea of the algorithm is relatively simple, yet surprisingly powerful: given as input an array A of n integers, subsample at random a *smaller* array B of $m \ll n$ integers from A, and use B as some sort of "guide" for what is in A. In particular, we expect, if we are not too unlucky, that finding "approximate medians" of B will give us an approximate idea of what the median of A is, and we can then filter out a lot of the elements of A to end up with a much more manageable task. And we can easily find that in B, since it will have much smaller size!

For simplicity, we will throughout assume n is odd, and that all numbers are distinct. Neither of these assumptions is necessary.

Here is the actual algorithm, only missing the value of m (to be determined shortly):

Algorithm 3: Randomised Median in Worst-Case Linear Time.

```
Require: array $A$ of $n$ distinct integers
 1: Set $\Delta = 4\sqrt{m}$                          ▷ Why? We'll see later. "Chebyshev."
 2: Create an array $B$ containing $m$ elements of $A$ chosen indepen-
    dently and uniformly at random (with replacement)
 3: Sort $B$                                                      ▷ Time $O(m \log m)$
 4: Let $\underline{b}$ and $\overline{b}$ be the $(m/2 - \Delta)$-th and $(m/2 + \Delta)$-th elements of $B$
    ▷ "Approximate medians" of $B$
 5:   ▷ Now we use $\underline{b}$ and $\overline{b}$ as "guides" for the contents of $A$. All 3
    steps below take time $O(n)$.
 6: Copy every $x$ of $A$ with $\underline{b} \leq x \leq \overline{b}$ in a new array $C$
 7: Compute the number $k$ of elements of $A$ smaller than $\underline{b}$
 8: Compute the number $\ell$ of elements of $A$ larger than $\overline{b}$
 9: if $k > \frac{n}{2}$ or $\ell > \frac{n}{2}$ then
10:     return fail                          ▷ The median of $A$ cannot be in $C$
11: else if $|C| > \frac{4n\Delta}{m} + 2$ then
12:     return fail                          ▷ We cannot process $C$ fast enough!
13: else
14:     Sort $C$                                                  ▷ Time $O(m \log m)$
15:     return the $(\frac{n+1}{2} - k)$-th element of $C$.
```

First, let's look at the time complexity. Assuming for now that $m = O(n/\log n)$ and $\frac{4n\Delta}{m} = O(n/\log n)$ (they will be!), the total time is dominated by Lines 6 to 8, and so the algorithm runs in (worst-case) time $O(n)$. Good.

Second, let's look at the correctness. Suppose the algorithm reaches Line 15: then it not hard to see that the element returned is at position $k + \frac{n+1}{2} - k = \frac{n+1}{2}$ in A: that is, it indeed returns the median.

So the algorithm always runs in time $O(n)$, and when it does not output fail it correctly outputs the median of A. This only leaves us with the third point: *what is the probability the algorithm returns fail?*

This can only happen because of three things ("bad events"):

Event E_1: Too many elements are smaller than $\underline{b}$: $k > \frac{n}{2}$

Event E_2: Too many elements are larger than $\overline{b}$: $\ell > \frac{n}{2}$

Event E_3: C is too large: $|C| > \frac{4n\Delta}{m} + 2$

Why is that an issue, again?

We want to get an upper bound on the probability *at least one* of these three events occurs. We could try to argue these events are independent (maybe?) and try to bound $\Pr[E_1 \cup E_2 \cup E_3] = 1 - \Pr[\overline{E_1} \cap \overline{E_2} \cap \overline{E_3}]$, and maybe (?) get some reasonable bound as a result. But independence is tricky to reason about, and nobody wants to do that if they do not have to. So instead, we will use the *union bound*:

The union bound sounds basic, but it is truly a fundamental, powerful tool.

Lemma 0.5.1 (Union Bound). *Let $E_1, \dots, E_k, \dots$ be a (possibly countably infinite) family of (possibly dependent) events. Then*

$$\Pr\left[\bigcup_{k=1}^{\infty} E_k\right] \leq \sum_{k=1}^{\infty} \Pr[E_k].$$

This is great! No need to worry about independence: now we immediately have by the union bound that

$$\Pr[E_1 \cup E_2 \cup E_3] \leq \Pr[E_1] + \Pr[E_2] + \Pr[E_3]. \tag{6}$$

By symmetry, one can also convince themselves that $\Pr[E_1] = \Pr[E_2]$, so we only have two things to analyze.

Bounding $\Pr[E_1]$. What is the probability that ℓ, the number of elements of A smaller than $\underline{b}$, exceeds $\frac{n}{2}$? By definition, if it exceeds $\frac{n}{2}$, then $\underline{b}$ is larger (or equal to) the median of A, But $\underline{b}$ is the $(\frac{m}{2} + \Delta)$-th element of B, which means that among the m we picked uniformly at random (with replacement) to create B, at most $\frac{m}{2} - \Delta$ were larger than the median.

Which should be unlikely: when we pick *one* element uniformly at random from A, the probability to get an element larger than the median is exactly $\frac{n-1}{2} \cdot \frac{1}{n} = \frac{1}{2} - \frac{1}{2n}$. So "by linearity of expectation" the expected number of elements larger than the median is $\frac{m}{2} - \frac{m}{2n}$. But $\frac{m}{2} - \Delta$, that's *much* smaller than that! Can we quantify this?

Thankfully yes. Let's call "the expected number of elements larger than the median" X. Then we can write $X = \sum_{i=1}^{m} X_i$, where $X_i \in \{0,1\}$ is the indicator random variable for "the i-th element sampled to go into B was larger than the median of A." That is, all X_is are i.i.d., and Bernoulli random variables with parameter $p := \frac{1}{2} - \frac{1}{2n}$., and so from what we saw about Binomials last week we get that $\operatorname{Var}[X] = mp(1-p) = \frac{m}{4}\left(1 - \frac{1}{n^2}\right) < \frac{m}{4}$.

This means X is a Binomial random variables with parameters m and p: $X \sim \operatorname{Bin}(m, p)$.

Why are we interested in the variance? Good question! We want to argue that most of the time X is "not too far from its expectation" $\frac{m}{2}\left(1 - \frac{1}{n}\right)$, and in particular that getting as low as $\frac{m}{2} - \Delta = \frac{m}{2}(1 - \frac{2}{\sqrt{m}})$ is truly a freak event. Unfortunately, using Markov's inequality here will not be enough... we need something stronger.

Do it: try and apply Markov's inequality to X. Why doesn't it work? Then try to apply it to $m - X$: why is the result too weak?

And that's where *Chebyshev's inequality* comes into play: instead of just using the expectation, Chebyshev allows you to leverage additional information you may have about the random variable, specifically its variance, to (usually) get stronger bounds:

Theorem 0.6 (Chebyshev's inequality). *Let X be a random variable with $\mathbb{E}\left[X^2\right] < \infty$. For any $t > 0$, we have*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\operatorname{Var}[X]}{t^2}$$

Another way to look at it: this is saying that a random variable usually may fluctuate around its expectation by give or take a few standard deviations (*i.e.*, a few $\sqrt{\operatorname{Var}}$)... but *more*? That's unlikely.

By the way, this is why we set $\Delta = 4\sqrt{m}$: the standard deviation of X we computed about is $\approx \sqrt{m}/2$, so that's the right order of magnitude.

Compared to Markov's inequality, Chebyshev's:

- Provides *two-sided* bounds (bounds the probability to deviate too far below *and* too far above the expectation) ✓
- Gives a bound that decays quadratically ($\propto 1/t^2$) instead of linearly ($\propto 1/t$), so is better for $t \geq 1$ (Fig. 2) ✓
- Does not require the random variable to be non-negative ✓
- Requires knowing a bound on the variance (if it exists) $\times$

So we want to use Chebyshev's inequality to argue $\Pr\left[\, X \leq \frac{m}{2} - \Delta \,\right]$ is small. Here we go: rewriting $\frac{m}{2} - \Delta = \mathbb{E}[X] - \left(\Delta - \frac{m}{2n}\right)$ and using

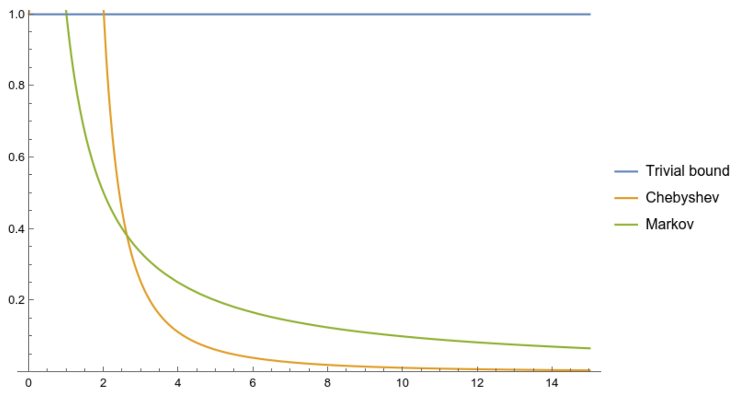

Figure 2: An illustration of the tail bounds $\Pr[X \geq t]$ (as a function of $t \geq 0$) given by Markov and Chebyshev's inequalities, for the specific case of a non-negative random variable $X \geq 0$ with both expectation and variance equal to 1: $\mathbb{E}[X] = \mathrm{Var}[X] = 1$. Note that Chebyshev does not require that $X \geq 0$, this is simply here for the sake of comparison with Markov's inequality, which does.

$m \leq n$ (so $\Delta - \frac{m}{2n} \geq \Delta - \frac{1}{2} \geq \frac{\Delta}{2} = 2\sqrt{m}$), we have

$$\begin{aligned}
\Pr\left[X \leq \frac{m}{2} - \Delta \right] &= \Pr\left[X \leq \mathbb{E}[X] - \left(\Delta - \frac{m}{2n}\right) \right] \\
&\leq \Pr\left[|X - \mathbb{E}[X]| \geq \left(\Delta - \frac{m}{2n}\right) \right] \\
&\leq \frac{\mathrm{Var}[X]}{(\Delta - \frac{m}{2n})^2} && \text{(by Chebyshev)} \\
&\leq \frac{m}{4(\Delta/2)^2} && \text{(Bound on variance)} \\
&= \frac{1}{16} && \text{(Setting of } \Delta)
\end{aligned}$$

This gives us

$$\Pr[E_1], \Pr[E_2] \leq \frac{1}{16}, \tag{7}$$

using Chebyshev's inequality. We only have to bound $\Pr[E_3]$ to conclude.

Bounding $\Pr[E_3]$. This is the last piece:[4] we want to bound the probability that C is "too large," that is the probability $|C|$ exceeds $4m$. Since all we have seen so far relies either on computing the expectation of the quantity of interest, its variance, or both, it would seem reasonable to start with computing $\mathbb{E}[|C|]$ and maybe use Markov's inequality: unfortunately, the quantity $\mathbb{E}[|C|]$ is quite tricky to compute.

[4] For now at last: there will be more down the line.

Instead, we will take an alternative path: let $R_{\underline{b}}$ denote the rank of $\underline{b}$ in A (and similarly $R_{\overline{b}}$ for $\overline{b}$). We have already proven that

$$\Pr\left[R_{\underline{b}} > \frac{n}{2} \right], \Pr\left[R_{\overline{b}} < \frac{n}{2} \right] \leq \frac{1}{16}$$

(this is Eq. (7)). Now, we want to prove that $|C| = R_{\overline{b}} - R_{\underline{b}} + 2 \leq \frac{4n\Delta}{m} + 2$ with high probability: so it'd *suffice* to show that

$$\Pr\left[R_{\underline{b}} < \frac{n}{2} - 2\frac{n\Delta}{m} \right], \Pr\left[R_{\overline{b}} > \frac{n}{2} + 2\frac{n\Delta}{m} \right] \leq \frac{1}{32}$$

as this would imply the result. One may ask: *why these particular values?* The reason is that since $\underline{b}$ is defined as the $(\frac{m}{2} - \Delta)$-th element in B, if the uniform random sampling was "representative" enough we expect to have an $\frac{1}{2} - \frac{\Delta}{m}$ fraction of the original array A on its left: so having much less than that, say a fraction $\frac{1}{2} - 2\frac{\Delta}{m}$, "should" be unlikely.

As before, we will only focus on bounding $\Pr\left[\, R_{\underline{b}} < \frac{n}{2} - 2\frac{n\Delta}{m} \,\right]$, as the case of $R_{\overline{b}}$ is similar (by symmetry).

To proceed, let's consider the set S of the $s := \frac{n}{2} - 2\frac{n\Delta}{m}$ smallest elements of A – let's call them "tail elements". The key observation is that if *fewer* than $\frac{m}{2} - \Delta$ elements of S end up in B, then $\underline{b}$, being the $(\frac{m}{2} - \Delta)$-th element of B, is not a tail element: and so must have $R_{\underline{b}} \geq s$. Based on that, we want to show that the probability to have *at least* $\frac{m}{2} - \Delta$ of S in B is small.

Now this looks familiar: the probability to pick an element of S when choosing one from A uniform at random is $\frac{s}{n} = \frac{1}{2} - \frac{2\Delta}{m}$. The number of elements from S in B (call it Y) is then a Binomial random variable with parameters m and $\frac{s}{n}$. We have $\mathbb{E}[Y] = \frac{ms}{n} = \frac{m}{2} - 2\Delta$, $\operatorname{Var}[Y] = \frac{ms}{n}(1 - \frac{s}{n}) \leq \frac{ms}{n} \leq \frac{m}{2}$; and we want to bound

$$\begin{aligned}\Pr\left[\, Y \geq \frac{m}{2} - \Delta \,\right] &= \Pr[\, Y \geq \mathbb{E}[Y] + \Delta \,] \\ &\leq \Pr[\, |Y - \mathbb{E}[Y]| \geq \Delta \,] \\ &\leq \frac{\operatorname{Var}[Y]}{\Delta^2} && \text{(Chebyshev)} \\ &= \frac{m}{2 \cdot 16m} && (\text{as } \Delta = 4\sqrt{m}) \\ &= \frac{1}{32}\end{aligned}$$

To summarize, we've just shown that

$$\Pr\left[\, R_{\underline{b}} < \frac{n}{2} - 2\frac{n\Delta}{m} \,\right] \leq \Pr\left[\, Y \geq \frac{m}{2} - \Delta \,\right] \leq \frac{1}{32}$$

We can similarly get $\Pr\left[\, R_{\overline{b}} > \frac{n}{2} + 2\frac{n\Delta}{m} \,\right] \leq \frac{1}{32}$, and so, "by a union bound,"

Do it! That's good practice.

$$\begin{aligned}\Pr[\, E_3 \,] &= \Pr\left[\, |C| > \frac{4n\Delta}{m} + 2 \,\right] \\ &\leq \Pr\left[\, R_{\underline{b}} < \frac{n}{2} - 2\frac{n\Delta}{m} \,\right] + \Pr\left[\, R_{\overline{b}} > \frac{n}{2} + 2\frac{n\Delta}{m} \,\right] \\ &\leq \frac{1}{16}\,. \end{aligned} \tag{8}$$

Putting it together. The probability that the algorithm fails is bounded, from Eq. (6), by

$$\Pr[\, E_1 \cup E_2 \cup E_3 \,] \leq \Pr[\, E_1 \,] + \Pr[\, E_2 \,] + \Pr[\, E_3 \,] \leq \frac{1}{16} + \frac{1}{16} + \frac{1}{16} = \frac{3}{16}\,.$$

When it doesn't fail, we have seen that it is correct; and it *always* runs in time at most $O(n)$. So... we're done! *Almost.* We haven't chosen the value of m yet!

So what do we need? For our running time, we need both $O(m \log m)$ and $O(\frac{n\Delta}{m} \log \frac{n\Delta}{m}) = O(\frac{n}{\sqrt{m}} \log \frac{n}{\sqrt{m}})$ to be $O(n)$.

There are many ways to do so, but one aesthetically pleasing choice is to make both equal:

$$m = \frac{n}{\sqrt{m}}$$

which leads to setting $\boxed{m = n^{2/3}}$. To conclude:

Theorem 0.7. *Randomised Median (Algorithm 3) is a linear-time Monte Carlo algorithm with failure probability at most* $3/16$.

But can we bring down this failure probability?

This is all very good, but, when you think about it, a failure probability of $3/16 \approx 19\%$ might be too much for many applications. Can we somehow bring this down to 1%? 0.01%? δ, for any $\delta \in (0, 1]$ of our choosing?

The obvious natural approach would be to go back to our analysis, see what the bottlenecks were, and modify the parameters to achieve smaller error probability. This would work *here*, but it may not *always* work, and honestly it is also very inconvenient. We went through a lot of trouble to establish Theorem 0.7, it would be nice not to have to start all over again!

($\star\star$) Go through the argument and see what happens to the probability of failure when you choose a larger Δ, for instance $m^{3/4}$ or $\sqrt{n}$.

Fortunately, it *is* possible: there is a way to take our algorithm (and the guarantees we proved for it), and amplify its success probability *in a blackbox way*. Of course, there is a cost: we will need to run the algorithm several times – the price is more computation time, and more random bits.

Random bits are not always cheap: they are a resource, like time, and memory.

Here is the idea: given the input array A run the algorithm (Algorithm 3) T times on A, using fresh (independent) random bits each time. If at any point the algorithm does not return fail, then return the median it outputs. If this never happens, return fail.

Since we have a Monte Carlo algorithm, whenever we return something else than fail this is guarantee to be correct, and we have the median. So what is the probability to output fail now? Well, we need *all* T independent runs to fail. And they are all independent, so the probability that they all fail is at most

$$\left(\frac{3}{16}\right)^T$$

Solving for this to be less than δ, we get that taking

$$T = \left\lceil \frac{\log(1/\delta)}{\log \frac{16}{3}} \right\rceil = O(\log(1/\delta))$$

suffices. This gives the following:

Corollary 0.7.1. *For any* $\delta \in (0, 1]$*, the Repeated Randomised Median described above is a Monte Carlo algorithm with failure probability at most* δ *and worst-case time complexity* $O(n \log(1/\delta))$.

This simple "trick" is your first example of *probability amplification.*

Viva Las Vegas!

In light of Corollary 0.7.1, it is natural to wonder: why stopping there? Can we convert any Monte Carlo algorithm into a Las Vegas algorithm, providing a converse to Lemma 0.4.2?

The answer is *not always* (not for every Monte Carlo algorithm), but in this particular case yes. The key observation is that Algorithm 3 is not *any* Monte Carlo algorithm: it never "fail silently." That is, when the Algorithm 3 fails, it tells us so! This is a very valuable feature.

Consider the following algorithm:

Algorithm 4: Randomised Median in Expected Linear Time.

```
Require: array $A$ of $n$ distinct integers
1: repeat
2:     Run Algorithm 3 on $A$ (with fresh random bits)
3:     Let $y$ be the output
4: until $y \neq \mathsf{fail}$
5: return $y$
```

Correctness is immediate: whenever this new algorithm stops, the y it outputs is the median of A. But *does it ever stop*? And if so, what is its expected running time?

Let $\tau(n) = O(n)$ be the (worst-case) running time of Algorithm 3, and K be the (random) number of loop iterations before Algorithm 4 terminates. Clearly, the (random) running time of Algorithm 4 is (at most) $K \cdot \tau(n)$. What can we say about K?

Some vocabulary: K as defined here is a *geometric random variable* with parameter $p \geq 13/16$.

1. The probability that $K \geq 1$ is 1: we always run the loop at least once.
2. The probability that $K \geq 2$ is at most $3/16$: to go to the second iteration of the loop, the first call to Algorithm 3 must have failed.
3. The probability that $K \geq 3$ is at most $(3/16)^2$: to go to the third iteration of the loop, the first two calls to Algorithm 3 must have failed (and they are independent).
4. The probability that $K \geq k$ is at most $(3/16)^{k-1}$: to go to the k-th iteration of the loop, the first $k-1$ calls to Algorithm 3 must have failed (and they are independent).

This is particularly useful, since from what we say in the first chapter we can write

$$\mathbb{E}[K] = \sum_{k=1}^{\infty} \Pr[K \geq k]$$

and here this becomes

$$\mathbb{E}[K] \leq \sum_{k=1}^{\infty} \left(\frac{3}{16}\right)^{k-1} = \frac{16}{13} \leq 1.231$$

This means that the expected running time of our Las Vegas algorithm, Algorithm 4, is at most $1.231 \cdot \tau(n) = O(n)$!

Corollary 0.7.2. *For any $\delta \in (0,1]$, the Indefinitely Repeated Randomised Median (Algorithm 4) is a Las Vegas algorithm with expected time complexity $O(n)$.*

More generally, we can prove the following:

Theorem 0.8. *Let A be a Monte Carlo algorithm with worst-case running time $T(n)$ and constant failure probability $p \in (0,1)$, with the following extra guarantee: one can detect whether the output of A is incorrect in time $O(1)$. Then there exists a* Las Vegas *algorithm A' for the same task with expected running time $O(T(n))$ (where the hidden constant in the $O(\cdot)$ depends on p).*

Proof. Your turn! □

And to conclude, something totally different!

Probability amplification by Majority Vote

The Randomised Median algorithm we saw (Algorithm 3) was quite nice, as far as Monte Carlo algorithms go: whenever it failed, *it told us so.* But that's usually not the case. Consider for instance the following scenario: someone implemented a very useful thing, say a data structure with its API, and gives you access. You cannot see the code or the implementation to check it's correct: all you can do is query that data structure D, and on input element x this query $\mathcal{Q}$ to D is supposed to output

$$\mathcal{Q}(x) = \begin{cases} \textsf{yes} & \text{if } x \in D \\ \textsf{no} & \text{if } x \notin D \end{cases}$$

Unfortunately, the implementation is *not* correct, or something is wrong: for whatever reason, each query behaves somewhat randomly, and is only correct with probability 60%. And when it's wrong, of course, you don't know it!

This sounds ridiculous? Wait until you hear about hashing and Bloom filters later in the course.

Can we use this data structure access in a blackbox way to obtain better guarantees, and have queries that are correct with probability 99% instead? Probability $1-\delta$?

The answer is, again, *yes*. And the probability amplification technique to use here is very intuitive: a simple *majority vote*. Here's what we will do, where $T = T(\delta)$ is an integer to be determined shortly:

Algorithm 5: More reliable data structure via majority vote.

```
Require: blackbox access to $D$ via $\mathcal{Q}$; input $x$
1: for $t = 1, 2, \dots, T$ do
2:     $y_t \leftarrow \mathcal{Q}(x) \in \{\textsf{yes}, \textsf{no}\}$
3: return majority$(y_1, \dots, y_T)$   ▷ yes if at least half of the $y_t$'s are yes
```

Let us analyze this. For any fixed x, we know that each $y_t \in \{0,1\}$ is the correct answer with probability at least $6/10$, and are independent (we assume that the random errors are independent, at least). So if we define

$$Y = \sum_{t=1}^{T} \mathbb{1}_{\{y_t \text{ is correct}\}}$$

we have a sum of independent Bernoulli random variables. And since we take a majority vote, the only way for our output to be incorrect is to have *more than half* of the T answers being incorrect, that is, to have $Y < \frac{1}{2}T$.

It's even a Binomial r.v. with parameters T and $p \geq 6/10$, but we will not need to be that precise.

But $\mathbb{E}[Y] \geq \frac{6}{10}T$, so to be wrong we need Y to be more than $\frac{1}{10}T$ away from its expectation. This should ring a bell: *we can use Chebyshev's inequality for that!*

We *could*, but that will not be good enough (that won't give a good enough bound). We can do better! Enters the *Chernoff bound*:

Try it: Chebyshev should get you something like $\Pr\left[Y < \frac{1}{2}T\right] \leq \frac{24}{T}$.

Theorem 0.9 (Chernoff bound). *Let $X_1, \dots, X_n$ be* independent *random variables taking value in $[0,1]$, and let $P := \sum_{i=1}^n \mathbb{E}[X_i]$ For any $\gamma \in (0,1]$ we have*

$$\Pr\left[\sum_{i=1}^n X_i > (1+\gamma)P\right] < \exp(-\gamma^2 P/3) \tag{9}$$

$$\Pr\left[\sum_{i=1}^n X_i < (1-\gamma)P\right] < \exp(-\gamma^2 P/2) \tag{10}$$

We can apply it with "$n = T$, $P = \frac{6}{10}T$, and $\gamma = \frac{1}{6}$" (the last one to have $(1-\gamma)\cdot\frac{6}{10}T = \frac{1}{2}T$), and that immediately gives us

$$\Pr\left[Y < \frac{1}{2}T\right] \leq e^{-\frac{1}{120}T}$$

which decays *exponentially* with T. In particular, for large T this is much, much better than what Chebyshev would give: In any case:

Sure, the constant $1/120$ in there is not great, but we could do better by sweating a bit more.

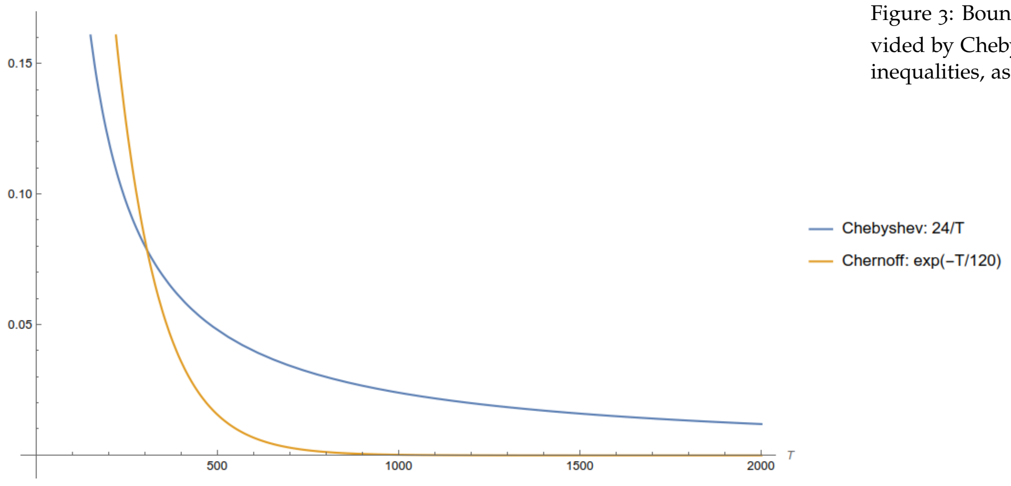

Figure 3: Bounds on $\Pr\left[Y < \frac{1}{2}T\right]$ provided by Chebyshev's and Chernoff's inequalities, as a function of T.

we wanted a failure probability less than δ? Then is suffices to solve for (integer) T:

$$e^{-\frac{1}{120}T} \leq \delta$$

giving $T \geq \lceil 120 \ln(1/\delta) \rceil$. So with taking $T(\delta) = O(\log(1/\delta))$ in Algorithm 5 suffices to amplify the success probability of each query on x from 60% to $1 - \delta$, at the (small?) cost of $O(\log(1/\delta))$ more queries to the data structure D.

We have used Markov's and Chebyshev's inequalities and the Chernoff bound, and this allowed us to analyse and amplify the success probability of our randomised algorithms. In the next lectures, we will see a related technique, also based on the Chernoff (or, looking ahead, Hoeffding) bound: a generalisation of the "majority vote" trick for when the output can take more than only 2 values, called the *median trick.*

Concentration inequalities: a summary

We summarize the "concentration bounds" used in the chapter so far, along with some others (more advanced) that will come in handy in the next chapters. These will be sufficient in many or most settings. There are, of course, *many* others, and many refinements or variants of the bounds we present here. If you are interested, see *e.g.*, Chapter 2 of [5] or [6] for a much more comprehensive and insightful coverage.

They are called that way because they quantify how "concentrated" (in terms of probability) a random variable is around its expectation.

[5] Roman Vershynin. *High-dimensional probability*, volume 47 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, Cambridge, 2018

[6] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities*. Oxford University Press, Oxford, 2013. A nonasymptotic theory of independence

We start with the mother of all concentration inequalities, Markov's inequality:

Theorem 0.10 (Markov's inequality). *Let X be a non-negative random variable with $\mathbb{E}[X] < \infty$. For any $t > 0$, we have*

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}$$

Applying this to $(X - \mathbb{E}[X])^2$, we get

Theorem 0.11 (Chebyshev's inequality). *Let X be a random variable with $\mathbb{E}\left[X^2\right] < \infty$. For any $t > 0$, we have*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\mathrm{Var}[X]}{t^2}$$

By applying Markov's inequality to the moment-generating function (MGF) of $\sum_{i=1}^n X_i$ in various ways, one can also obtain the following statements:

Theorem 0.12 (Hoeffding bound). *Let $X_1, \dots, X_n$ be independent random variables, where X_i takes values in $[a_i, b_i]$. For any $t \geq 0$, we have*

$$\Pr\left[\sum_{i=1}^n X_i > \sum_{i=1}^n \mathbb{E}[X_i] + t\right] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right) \tag{11}$$

$$\Pr\left[\sum_{i=1}^n X_i < \sum_{i=1}^n \mathbb{E}[X_i] - t\right] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right) \tag{12}$$

Corollary 0.12.1 (Hoeffding bound)**.** *Let $X_1, \dots, X_n$ be i.i.d. random variables taking value in $[0,1]$, with mean μ. For any $\gamma \in (0,1]$ we have*

$$\Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| > \gamma\right] \leq 2\exp(-2\gamma^2 n) \tag{13}$$

Theorem 0.13 (Chernoff bound)**.** *Let $X_1, \dots, X_n$ be independent random variables taking value in $[0,1]$, and let $P := \sum_{i=1}^{n} \mathbb{E}[X_i]$ For any $\gamma \in (0,1]$ we have*

$$\Pr\left[\sum_{i=1}^{n} X_i > (1+\gamma)P\right] < \exp(-\gamma^2 P/3) \tag{14}$$

$$\Pr\left[\sum_{i=1}^{n} X_i < (1-\gamma)P\right] < \exp(-\gamma^2 P/2) \tag{15}$$

In particular, if $X_1, \dots, X_n$ are i.i.d. with mean μ, then for any $\gamma \in (0,1]$ we have

$$\Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| > \gamma\mu\right] \leq 2\exp(-\gamma^2 n\mu/3) \tag{16}$$

As a rule of thumb, the "multiplicative" (Chernoff) from Theorem 0.13 is preferable to the "additive" bound (Hoeffding) from Corollary 0.12.1 whenever $\mu := P/n \ll 1$. In case one only has an upper or lower bound on the quantity $P = \sum_{i=1}^{n} \mathbb{E}[X_i]$, the following version of the Chernoff bound can come in handy:

Theorem 0.14 (Chernoff bound (upper and lower bound version))**.** *In the setting of Theorem 0.13, suppose that $P_L \leq P \leq P_H$. Then for any $\gamma \in (0,1]$, we have*

$$\Pr\left[\sum_{i=1}^{n} X_i > (1+\gamma)P_H\right] < \exp(-\gamma^2 P_H/3) \tag{17}$$

$$\Pr\left[\sum_{i=1}^{n} X_i < (1-\gamma)P_L\right] < \exp(-\gamma^2 P_L/2) \tag{18}$$